On the Specification Complexity of Linguistic Control Procedures

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

Abstract

In this paper we combine the ideas of trigger based hybrid systems with that of motion description languages in order to show how continuous machines should interpret and operate on computer generated symbolic (linguistic) inputs in order to make the input alphabet meaningful from a control theoretic point of view. We furthermore focus our attention on so called quantized input-output machines, i.e. machines where the admissible control signals, U, as well as the measurement set, Y, are both finite, which allows us to define a complexity measure of a given task as the minimum number of bits needed to code a word over the input alphabet that achieves the task. We also show how this measure can help us understand why feedback control is to prefer over purely open-loop control in some specific situations in terms of the specification complexity.

1 Introduction

The use of computers for controlling mechanical devices, such as robots or machine tools, has brought to the fore the need for a systematic study of the interactions between the symbolic computer programs and the continuous device dynamics. Typically, this interaction is defined by the way the continuous machine operates on the outputs from a computer generated *motion control program*. Such a motion control program generates strings of symbolic inputs to a real-time system that controls the device based on sensory information about internal states as well as the environment it is operating in. In order to understand the interactions between these two heterogeneous components, different *hybrid architectures*, serving as abstractions between continuous and discrete control, have been suggested. In [3] a general model for such a *trigger based hybrid system* that is driven by symbolic input strings 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 motion control program and the evolution of the scalar p triggers the reading of that string. 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 [2, 11]. By combining these two ideas we will construct interpreter mechanisms for generating meaningful control commands from symbolic inputs, and we argue that by letting the symbolic inputs correspond directly to

continuous control signals, the resulting interaction between the symbolic and the continuous can be given a clear control theoretic interpretation.

The fact that the symbolic inputs are computer generated also implies that they have to be drawn from a finite alphabet, thus stressing the point that we have to control our continuous devices using a quantized set of control inputs. This situation also arises in a number of applications where there is a smallest detectable change associated with the system, which calls for a natural quantization of the control inputs. For analogous reasons we furthermore assume that the observations that we make, i.e. that are fed back to the motion control unit, take on values in a finite set as well. The problem at hand is thus to control a finitely labeled transition system using quantized inputs.

In [6, 7] a *complexity measure* was defined as the minimum number of bits needed to be transmitted from the computer to the device in order to make the system achieve a given task. However, in that work, the system was evolving on a finite automaton, i.e. the state space was finite, which significantly simplifies the nature of the problem. The main contribution in this paper is thus a modification of those results for continuous dynamical systems, which calls for a hybrid model for capturing the interactions between the symbolic inputs and the continuous device dynamics.

We believe that the information theoretic approach to control theory proposed in this paper, that captures the complexity of the task, the system dynamics, and the input alphabet in a unified way, has a number of potential applications. For instance, in teleoperated robotics (see for example [1]), the control signals are transmitted over communication channels in which the presence of channel noise makes it preferable to transmit instructions that are as short as possible. A related problem arises in the area of minimum attention control, where an attention functional is defined as a measure of the control variability. (See for example [9].) The problem then becomes that of minimizing the cost functional under the additional constraint that the servomechanism should perform in a satisfactory way. It can also be argued that this way of imposing complexity constraints on control procedures has implications for decentralized or embedded control strategies, where the idea is to minimize the communication between different control modules, or micro-controllers, at the same time as sufficient information must be available in order for the overall system to meet its specifications, as suggested in [5].

The outline of this paper is as follows: In Section 2 we define the basic model of a *quantized input-output system* and show how this can operate on symbolic inputs as dictated by the trigger based hybrid system model. We then, in Section 3, define the *specification complexity* associated with the model, and continue, in Section 4, with two results (Theorem 4.1 and Corollary 4.1) that tell us how to design control procedures with low specification complexity in the particular case when the system is driven between desired boundary points.

2 Quantized Input-Output Machines

The primary objects of study in this paper are so called *motion description languages* (MDLs). However, before we can define what we mean by a MDL, some comments about formal languages must be made. Given a finite set, or alphabet, A, by $A^\star$ we understand the set of all strings of finite length over A. We let $a \in A$ denote an element in A, and use boldface $\mathbf{a} \in A^\star$ to denote words over A. There is furthermore a naturally defined binary operation on $A^\star$, namely the concatenation of strings, i.e. if $\mathbf{a}_1, \mathbf{a}_2 \in A^\star$ then $\mathbf{a}_1 \cdot \mathbf{a}_2 \in A^\star$. 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 thee free monoid over a finite alphabet. (See for example [8] for an introduction to this subject.)

Now, consider the finite sets A, B. We will let A^B denote the set of mappings from B to A, and if $\phi \in A^B$ and $B_s \subseteq B$ we use $\phi(B_s)$ for $\{a \in A \mid a = \phi(b) \text{ for some } b \in B_s\}$. We also let $B \backslash B_s$ denote the set $\{b \in B \mid b \notin B_s\}$. Furthermore, we use $card(A)$ for the cardinality of A.

The concept of a *motion alphabet* has been proposed recently in the literature as a finite set of symbols representing different control actions that, when applied to a specific machine,

define segments of motion [2, 10, 11]. A MDL is thus given by a set of strings that represent such 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 language is defined relative to the physical device that is to be controlled.

Definition 2.1 (Quantized Input-Output Machine) *A quantized input-output machine is given by* $M = (U, X, Y, F, H)$*, where* U *is a finite set of admissible inputs,* Y *is a finite set of outputs,* $X \subset \mathbb{R}^n$ *is the state space of the system,* $F : X \times U \to TX$ *defines the system evolution, and* $H : X \to Y$ *is a measurable output function. The evolution of the machine is given by* $\dot{x} = F(x, u),\ y = H(x)$.

Now, since we want each letter in the motion alphabet to correspond directly to a control command that generates a particular motion segment on a given machine, we choose, in contrast to the definitions in [2, 11], to define the motion description language as

Definition 2.2 (Motion Description Language) *Given a quantized input-output machine* M*. Relative to* M *we let a motion description language be given by a subset of the free monoid over the set* $U \times U^{Y \times U} \times \{0,1\}^Y$*. In other words, we let the letters in the motion alphabet be triples of the form* (u, k, ξ)*, where* $u \in U,\ k : Y \times U \to U$*, and* $\xi : Y \to \{0,1\}$*. If, at time* t_0*,* M *receives the input string* $(u_1, k_1, \xi_1), \ldots, (u_p, k_p, \xi_q)$*, then* x *evolves according to*

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

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, and we can now define trajectories on these types of systems. Given an input alphabet $\Sigma \subset U \times U^{Y \times U} \times \{0,1\}^Y$ and an input symbol $\sigma = (u, k, \xi) \in \Sigma$. By the flow we understand

$$\phi_M(x_0, \sigma, t) = x_0 + \int_0^t F(x(s), k(y(s), u))ds,$$

where the subscript M refers to the particular machine that Σ is defined relative to.

If there exists a finite time $T \ge 0$ such that $\xi(H(\phi_M(x_0, \sigma, T))) = 1$ then we let the interrupt time be given by

$$\tau_M(\sigma, x_0) = \min\{t \ge 0 \mid \xi(H(\phi_M(x_0, \sigma, t))) = 1\}.$$

If no such finite time T exists then we say that

$$\tau_M(\sigma, x_0) = \tau_\infty,$$

where τ_∞ is a given distinguishable symbol. We let the final point on the trajectory generated by σ be

$$\chi_M(\sigma, x_0) = \phi_M(x_0, \sigma, \tau_M(\sigma, x_0))$$

if $\tau_M(\sigma, x_0) \neq \tau_\infty$ and use the notation $\chi_M(\sigma, x_0) = \chi_\infty$ otherwise. We furthermore let $\chi_M(\sigma, \chi_\infty) = \chi_\infty$, $\forall \sigma \in \Sigma$.

Now, let us define the integral curve $\Phi_M(x_0, \sigma)$ as

$$\Phi_M(x_0, \sigma) = \bigcup_{t \in [0, \tau_M(\sigma, x_0))} \phi_M(x_0, \sigma, t)$$

if $\chi_M(\sigma, x_0) \neq \chi_\infty$, and use the notation $\Phi_M(x_0, \sigma) = \Phi_\infty$ otherwise.

It is possible to combine a collection of such integral curves to obtain the integral curve generated by a word over Σ instead of just a letter. Let $\boldsymbol{\sigma} \in \Sigma^\star$ and assume that $\boldsymbol{\sigma}$ has length q. (Denoted by $|\boldsymbol{\sigma}| = q$.) In other words, $\boldsymbol{\sigma}$ is a concatenation of q input symbols. We can then define

$$\Phi_M(x_0, \boldsymbol{\sigma}) = \bigcup_{i=1}^{q} \Phi_M(\chi_M^i, \sigma_i),$$

where the initial points can be defined iteratively as

$$\begin{aligned} \chi_M^1 &= x_0 \\ \chi_M^{p+1} &= \chi_M(\sigma_p, \chi_M^p), \ p = 1, \ldots, q-1, \end{aligned}$$

and

$$\boldsymbol{\sigma} = \sigma_1 \cdot \ldots \cdot \sigma_q.$$

If $\Phi_\infty \notin \Phi_M(x_0, \boldsymbol{\sigma})$ we say that $\boldsymbol{\sigma}$ is a *terminating input sequence*, or in other words that it generates a trajectory of finite time duration. We denote this by $\boldsymbol{\sigma} \in \mathcal{T}_M(x_0)$. If $\boldsymbol{\sigma}$ is terminating then we let the final point on the trajectory be given by $\bar{\chi}_M(x_0, \boldsymbol{\sigma}) = \chi_M^{q+1}$, where χ_M^{q+1} is defined above.

Based on these initial definitions we now have a model for the way linguistic inputs affect continuous dynamical systems, as well as a description of how these symbolic inputs generate integral curves.

3 Specification Complexity

Two observations should be made already at this point: The first is that if the input alphabet is $\Sigma = U \times U^{Y \times U} \times \{0,1\}^Y$, then the size of Σ depends exponentially on the size of Y. In other words, if we choose a large output set Y, i.e. equip our machine with many sensors, or sensors with high resolution, then the input symbols will be drawn from a large alphabet. An increase in knowledge of the system and the environment (large Y) could thus potentially result in an increase in the number of bits we need to transmit 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 [4].) The second observation to be made is that we have imposed no probability distribution over our input alphabet, i.e. we assume that all symbols are equally likely be used.

Based on these observations we can note that the optimal code length of a symbol over a uniformly distributed finite alphabet is given by the entropy of the alphabet, which is equal to $\log_2(card(\Sigma))$. (See for example [4] for an accessible account of this classic result.) For words over Σ we can thus capture the dependency on the size of the input alphabet as well as the number of symbols used if we, in a manner analogous to [6], define the complexity of a control procedure as the description length of the shortest input sequence that drives the system between given boundary points.

Definition 3.1 (Description Length) *Let S be a finite set. The description length of a word $\mathbf{s} \in S^\star$ is given by*

$$\mathcal{L}(\mathbf{s}, S) = |\mathbf{s}| \log_2(card(S)).$$

Now, since we want to include the complexity of the task in the formulation we want to be able to parameterize the performance of the system in a direct way. In this paper we do this by assuming that we start at a given point x_0 and try to drive the system to the open ball

$$\mathcal{B}(x_f, \varepsilon) = \{x \in X \mid \|x - x_f\| < \varepsilon\}.$$

Thus ε becomes the parameter that describes what level of accuracy we want our input sequence to achieve.

Definition 3.2 (Specification Complexity) *By the specification complexity of the task of driving the machine M between x_0 and the set $\mathcal{B}(x_f, \varepsilon)$ we understand*

$$\mathcal{C}(M, x_0, x_f, \varepsilon) = \min_{\boldsymbol{\sigma} \in \mathcal{T}_M(x_0) \mid \bar{\chi}_M(x_0, \boldsymbol{\sigma}) \in \mathcal{B}(x_f, \varepsilon)} \{\mathcal{L}(\boldsymbol{\sigma}, \Sigma)\}.$$

In other words, if we let $\boldsymbol{\sigma}$ be the word with the shortest description length over a motion alphabet Σ that drives the machine M from x_0 to $\mathcal{B}(x_f, \varepsilon)$ then

$$\mathcal{C}(M, x_0, x_f, \varepsilon) = \mathcal{L}(\boldsymbol{\sigma}, \Sigma).$$

Remark 3.1 *Some additional comments about the code lengths of the control inputs should be made. If we assume that we have been able to establish a probability distribution over Σ we can form the entropy*

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

where σ_i occurs with probability p_i. It is now possible to use an optimal coding scheme, such as the Huffman code, 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 [13] then tells us that

$$\mathcal{H}(\Sigma) \leq l^\star(\Sigma) < \mathcal{H}(\Sigma) + 1.$$

However, in this paper we assume that $p_i = p_j$ for all i, j and leave the problem of optimal input codes to the future.

We now have a possibility of describing how complicated it is to perform a certain task on a certain machine, given the input and output sets. In the next section we will use these tools for proving some results that illustrate some particular cases where feedback is to prefer over open-loop control from a complexity point of view.

4 Complexity Results

This way of thinking about control signals in terms of their complexity, or their information theoretic content, was introduced in [6]. The driving motivation behind that work was a desire to see how the availability of sensory information affects the specification complexity when driving a finite state machine between given boundary points. One key observation that enabled us to answer this question was that when we give everyday instructions for navigation between given locations we tend to divide the instructions into one long open-loop segment followed by a closed-loop fine tuning part. That instructions tend to have this structure was verified empirically by a statistical examination of the type of instructions that were generated by *MapQuest* [12], which is a widely used program that provides travel directions online. An example of such a statement could be: `Follow the interstate until you get to Exit Y, then make`

`a right turn and follow the signs to X`. These types of instructions thus tell us that we can expect to get short instructions if we use open-loop control on parts of the state space, complimented by closed-loop control around well-defined landmarks.

In this section we generate results analogous to those in [6], but defined on quantized input-output machines instead of finite state machines. For this program to be successful, we first need to define some key concepts:

Definition 4.1 (Open-Loop Reachability) *A set $X_f \subset X$ is open-loop reachable from $x_0 \in X$ if there exists a terminating input symbol $(u, k_U, \xi) \in \Sigma$ such that*

$$\chi_M(x_0, (u, k_U, \xi)) \in X_f,$$

where $k_U(y, u) = u, \ \forall y \in Y, u \in U$.

Definition 4.2 (Control Invariant Closed-Loop Reachability) *A point $x_f \in X_f \subset X$ is said to be control invariantly closed-loop reachable in X_f with precision ε if there exists a terminating input symbol of the form $(u, k_Y, \xi) \in \Sigma$ such that, for all $x_0 \in X_f$,*

$$\begin{aligned} &\Phi_M(x_0, (u, k_Y, \xi)) \subset X_f \\ &\chi_M(x_0, (u, k_Y, \xi)) \in \mathcal{B}(x_f, \varepsilon), \end{aligned}$$

where k_Y belongs to the set $\{k \in U^{Y \times U} \mid k(y, u_i) = k(y, u_j), \ \forall y \in Y, u_i, u_j \in U\}$.

These two concepts are illustrated in Figure 1.

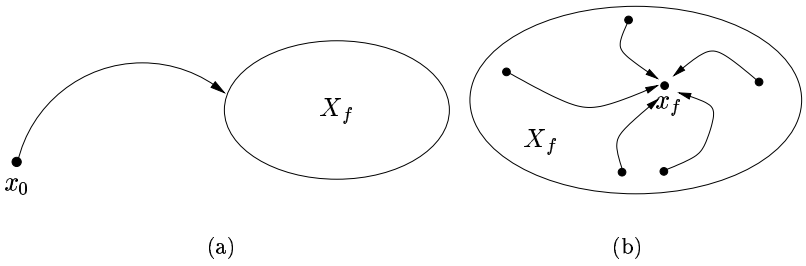

Figure 1: In the left figure, X_f is open-loop reachable from x_0. In the right figure, x_f is control-invariantly closed-loop reachable in X_f.

In order to compare the complexities when controlling a machine where sensory information is available to a machine where only open-loop signals can be used, we need a characterization of what we mean by pure open-loop control:

Definition 4.3 (Open-Loop Motion Description Languages and Machines) *Let U and Y be finite sets. By an open-loop motion description language we understand any subset of the free monoid over the set $\Sigma_{ol} = U \times \{k_U\} \times \{1\}$. We furthermore let $M_{ol} = (U, X, Y, F, H)$ denote the quantized input-output machine that operates on inputs $\boldsymbol{\sigma}_{ol} \in \Sigma_{ol}^{\star}$ as follows:*

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

where T_i now denotes the time at which the output changes from $H(x(T_{i-1}))$ to any other value in Y. ($T_0 = t_0$.)

We thus have a construction where only open-loop control is used and where a new symbol is read whenever the outputs change values. Analogous to finite state machines we can now define the *distance*, $dist(M_{ol}, x_0, x_f, \varepsilon)$, between x_0 and $\mathcal{B}(x_f, \varepsilon)$ as the length of the shortest word $\boldsymbol{\sigma}_{ol} \in \Sigma_{ol}^{\star}$ such that $\bar{\chi}_{M_{ol}}(x_0, \boldsymbol{\sigma}_{ol}) \in \mathcal{B}(x_f, \varepsilon)$. This directly gives us that

$$\mathcal{C}(M_{ol}, x_0, x_f, \varepsilon) = dist(M_{ol}, x_0, x_f, \varepsilon) \log_2(card(U)).$$

We are now ready to formulate our main complexity theorem:

Theorem 4.1 *Given the machines M and M_{ol} defined on the same sets U, Y, X and functions F, H, where we assume that $card(U) \geq 2$. Let $x_f \in X_f$ be control invariantly closed-loop reachable in X_f with precision ε. If $H(X_f) \cap H(X \backslash X_f) = \emptyset$, $H(X_f) \cap H(\mathcal{B}(x_f, \varepsilon)) = \emptyset$, and X_f is open-loop reachable from $x_0 \in X \backslash X_f$ then we can construct a motion alphabet Σ relative to M such that*

$$\frac{\mathcal{C}(M, x_0, x_f, \varepsilon)}{\mathcal{C}(M_{ol}, x_0, x_f, \varepsilon)} \leq \frac{4card(H(X_f))}{dist(M_{ol}, x_0, x_f, \varepsilon)}.$$

Proof: When controlling M the idea is to only use a reduced number of observations, denoted by $Y_{X_f} = \pi_{X_f}(Y)$, where

$$\pi_{X_f}(y) = \begin{cases} y & \text{if } y \in H(X_f) \\ e & \text{if } y \in H(X \backslash X_f), \end{cases}$$

where $e \notin Y$. That this projection is well-defined follows directly from the fact that $H(X_f) \cap H(X \backslash X_f) = \emptyset$ and we thus only care about the observations made on the reduced subset X_f, i.e. on the part of X where feedback can be utilized effectively

We can now let $\Sigma = \{u_0\} \times U^{Y_{X_f} \times \{u_0\}} \times \{0,1\}^{Y_{X_f}}$, where u_0 is a distinct element in U. Using this input alphabet it is possible to drive M between x_0 and $\mathcal{B}(x_f, \varepsilon)$ using only one instruction, namely $(u_0, k^{\star}, \xi^{\star})$, where $\xi^{\star}(y) = 1$ if and only if $y \in \mathcal{B}(x_f, \varepsilon)$, and where

$$k^{\star}(y, u_0) = \begin{cases} u_{ol} & \text{if } y = e \\ k_{cl}(y) & \text{if } y \neq e. \end{cases}$$

Here $u_{ol} \in U$ is the open-loop controller that drives M from x_0 to X_f, and k_{cl} is the closed-loop controller that takes M from any point in X_f to $\mathcal{B}(x_f, \varepsilon)$. The idea is thus to start off with an open-loop segment, followed by a high precision feedback part that directs us toward the desired state, as shown in Figure 2.

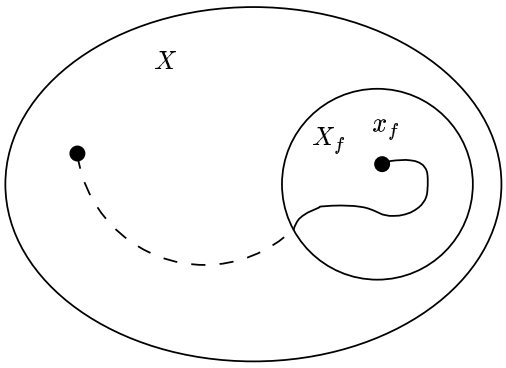

Figure 2: The single instruction that combines open-loop and closed-loop control in Theorem 4.1 is depicted.

The size of the input set Σ is

$$\begin{aligned} card(\Sigma) &= card(U)^{card(Y_{X_f})} 2^{card(Y_{X_f})} \\ &= (2card(U))^{1+card(H(X_f))} \\ &\leq (2card(U))^{2card(H(X_f))} \\ &\leq card(U)^{4card(H(X_f))}. \end{aligned}$$

Thus

$$\mathcal{C}(M, x_0, x_f, \varepsilon) \leq 4card(H(X_f)) \log_2(card(U)).$$

For M_{ol} we have already seen that

$$\mathcal{C}(M_0, x_0, x_f, \varepsilon) = dist(M_{ol}, x_0, x_f, \varepsilon) \log_2(card(U))$$

and the theorem follows. ∎

However, goals are seldom final goals. More often they tend to be intermediary goals in a grander scheme. This is for instance the case when mobile robots are navigating using landmarks. The theory that we have developed so far does not acknowledge this fact, and in this paragraph we modify the theory so that we can take this natural extension into account, i.e. where a number of goal states are visited by the machine.

It is clear that the premises on which Theorem 4.1 are based are too restrictive to capture the desired *chained* structure that intermediary goals give rise to. Instead we need to extend the trajectories through a chain of goals states. This can be achieved by assuming that we work with a scenario where feedback can be exploited effectively around different states, i.e. the intermediate goals. We also assume that the sets on which the observers are defined are open-loop reachable from each other. In this case we can use open-loop control for driving the system between these sets, as seen in Figure 3. For the sake of completeness, we explicitly state this chained extension of the Theorem 4.1 as a corollary:

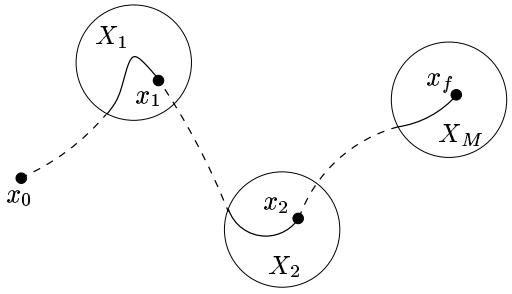

Figure 3: A chained version of the main theorem. The dashed lines correspond to open loop trajectories, while feedback is used for generating the solid-line trajectories.

Corollary 4.1 *Assume that* $card(U) \geq 2$ *and let the sets* $X_1, \ldots, X_M$ *be disjoint sets with* $card(H(X_i)) \leq C$, $i = 1, \ldots, M$. *Furthermore assume that* $H(X_i) \cap H(X \backslash X_i) = \emptyset$ *for all* i *and that* $H(X_i) \cap H(X_j) = \emptyset$, $i \neq j$. *Let* $x_f \in X_M$ *be control-invariantly reachable in* X_M *with precision* ε *and let* X_1 *be open-loop reachable from* x_0. *Assume that there exists intermediary goals* $x_i \in X_i$, $i = 1, \ldots, M-1$ *such that* $\mathcal{B}(x_i, \varepsilon)$ *is control-invariantly reachable in* X_i, X_{i+1} *is open-loop reachable from any point in* $\mathcal{B}(x_i, \varepsilon)$, *and* $H(X_i) \cap H(\mathcal{B}(x_i, \varepsilon)) = \emptyset$. *Then there exists an input alphabet* Σ, *defined relative to* M, *such that*

$$\frac{\mathcal{C}(M, x_0, x_f, \varepsilon)}{\mathcal{C}(M_{ol}, x_0, x_f, \varepsilon)} \leq \frac{4MC}{dist(M_{ol}, x_0, x_f, \varepsilon)}.$$

The proof of this corollary is just a straight forward modification of the proof of Theorem 4.1.

One conclusion to be drawn from Corollary 4.1 is that the increase in description length, caused by the summation over many intermediate goals, can be counteracted 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.

5 Conclusions

In this paper we propose a framework for modeling the way linguistic inputs affect continuous devices by combining the already established ideas of trigger based hybrid systems with that of motion description languages. We show how continuous machines should operate on the symbolic inputs in order to give the input alphabet a meaningful control theoretic interpretation.

Since we focus on finite input and output sets we can capitalize on the finitely describable aspects of the control signals, which furthermore enables us to follow the development in [6], but for continuous dynamical systems, and use information theoretic tools when analyzing the complexity of a given task. We define such a complexity measure as the number of bits needed to code a word over the input alphabet and show how this measure can help us understand when feedback control is to prefer over purely open-loop control in terms of the specification complexity.

Natural extensions of these results would be to investigate how the choice of Y and U would affect the complexity, which would provide a unified treatment of actuator and sensor selection, as well as control design for a number of applications such as mobile robot control. The search for such a "fundamental theorem of robotics" still lies in the future, but we believe that the theory outlined in this paper holds one of the keys toward enabling such an endeavor.

References

[1] K. Brady and T.J. Tarn. Internet-Based Remote Teleoperation. In *IEEE International Conference on Robotics and Automation*, Leuven, Belgium, 1998.

[2] 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.

[3] 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.

[4] T.M. Cover and J.A. Thomas. *Elements of Information Theory*, John Wiley & Sons, Inc., New York, 1991.

[5] M. Egerstedt. Linguistic Control of Mobile Robots. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui, Hawaii, Oct. 2001.

[6] M. Egerstedt and R.W. Brockett. Feedback Can Reduce the Specification Complexity of Motor Programs. *IEEE Conference on Decision and Control*, pp. 1651–1656, Orlando, FL, Dec. 2001.

[7] M. Egerstedt. Some Complexity Aspects of the Control of Mobile Robots. *American Control Conference*, Anchorage, Alaska, May, 2002.

[8] J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation, 2nd Ed.*, Addison-Wesley, New York, 2001.

[9] D. Hristu and K. A. Morgansen. Limited Communication Control. *Systems and Control Letters*, Vol. 43, No. 7, pp. 193–205, July 1999.

[10] D. Hristu and S. Andersson. Directed Graphs and Motion Description Languages for Robot Navigation and Control. *Proceedings of the IEEE Conference on Robotics and Automation*, May. 2002.

[11] 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.

[12] MapQuest: `http://www.mapquest.com/`

[13] C.E. Shannon. A Mathematical Theory of Communication. *Bell Systems Technical Journal*, Vol. 27, pp. 379–423, 1948.