ARL-TR-9901 • Apr 2024

DEVCOM ARMY RESEARCH LABORATORY

On Agility and Control of High-Dimensional Dynamical Systems

by Charles Edelson, Mulugeta Haile, Gregory Barber, and Mark Bundy

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-9901 ● APR 2024

On Agility and Control of High-Dimensional Dynamical Systems

by Charles Edelson
Indiana University Bloomington

Mulugeta Haile, Gregory Barber and Mark Bundy
DEVCOM Army Research Laboratory

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
April 2024	Technical Report	February 1, 2023	February 1, 2024

4. TITLE AND SUBTITLE

On Agility and Control of High-Dimensional Dynamical Systems

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)

Charles Edelson, Mulugeta Haile, Gregory Barber, and Mark Bundy

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

DEVCOM Army Research Laboratory
ATTN: FCDD-RLR-EV
Aberdeen Proving Ground, MD 21005

8. PERFORMING ORGANIZATION REPORT NUMBER

ARL-TR-9901

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES

ORCIDs: Charles Edelson, 0000-0002-6737-5224; Mulugeta Haile, 0000-0002-3038-9155; Gregory Barber, 0000-0002-6919-5083

14. ABSTRACT

Modern robotics systems are incapable of replicating the agile behaviors of biological organisms moving through complex natural environments. We investigate the origin of this performance discrepancy by reviewing current literature in the fields of soft robotics control and biological sensorimotor control–agile behavior. We discuss the use of model predictive control and decentralized control for the agile control of soft and compliant robotics systems before reviewing theories of how biological organisms perform sensorimotor and reflexive control. Finally, we finish our discussion with a review of current metrics of behavioral agility and discuss potential new metrics that could be used for computational design and optimization of agile systems.

15. SUBJECT TERMS sensorimotor, high-dimensional dynamics, complex systems, artificial intelligence, Mechanical Sciences, Military Information Sciences

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	36

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Mulugeta Haile	410-278-5289

STANDARD FORM 298 (REV. 5/2020)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

1. Introduction

When comparing the graceful motions of a fish swimming through its natural environment to an unmanned underwater vehicle (UUV), it quickly becomes evident that the fish is both more agile and dynamic than the UUV. While the structural differences between a fish and a standard UUV are certainly an important aspect of this increased mobility, the fish's behavioral flexibility and sensorimotor control plays an equally important role. The end result is that the co-evolution of body and brain have allowed the fish to move in ways that current robotic systems cannot replicate. Understanding this performance discrepancy has generated significant interest in determining how biological systems are able to perform such dynamic and precise control and how these behaviors can be replicated in synthetic systems. Insights into these questions could provide both new theoretical techniques and routes for application in engineered systems.

Correspondingly, there is an overwhelming body of work addressing these topics and related research questions. The purpose of this report is to provide a foundation for researchers interested in pursuing projects within this domain. Specifically, this report provides a background review of agile control in biological and robotic systems before discussing potential methods for the computational design of agile systems. The report is divided into two main sections comprised of control problems in soft robotics systems and agile control in biological and robotic systems. Section 2 provides a background on the control of soft robotic systems and how this problem is different from traditional rigid robot control. Section 3 continues to build on this foundation by reviewing work on sensorimotor and agile control in biological organisms before looking at metrics of agility in robotic systems. Finally, the report ends with a summary of the important points, suggested additional readings, and potential future research directions.

2. Control Problems in Soft Robotics

The application of control theory to robotics systems is, in no small part, responsible for the widespread success and adoption of robotic systems in modern society. Semi-autonomous systems require the development of a control unit and algorithms to function, resulting in a tight coupling between engineered system design and control theory. Improvements in control theory enabled the design of new robotic systems and, similarly, attempts to solve problems in industry led to advances in con-

trol theory.* This has resulted in a vast body of work on the applications of control theory to robotic systems. However, despite the fact that soft and compliant robotic systems are in no way a new invention,[3] much of the historic theory on the control of robotic systems is concerned with the control of traditional rigid robotic systems. Thus, to make this report more concise and approachable, this review will focus on model predictive control and decentralized control, as these two techniques have been applied successfully to a variety of compliant robotic systems. Both of these methods lend themselves well to soft robotic systems, with recent studies exploring their theoretical, computational, and experimental properties. Section 2.1 summarizes and discusses a selection of studies applying these techniques and provides additional resources to form a more complete picture of the current state of this field. It begins with an overview of what makes the control of soft robots difficult before moving into subsections on model predictive control and decentralized control. Each subsection begins with a statement of the control problem and overview of the method being discussed before reviewing recent studies within that domain.

2.1 Challenges with Soft Robot Control

2.1.1 Motivating Example

For the purpose of this report, a soft robot is any system *not* composed of rigid limbs and rotational actuators. Traditional rigid robot designs are most often controlled using optimal control algorithms. These are algorithms developed to solve optimal control problems, which are the subset of control problems that attempt to find a control trajectory that extremizes an auxiliary cost function.† For example, minimizing the amount of energy a robot uses while performing a task or maximizing the return for an investment scheme are both examples of optimal control problems. While classical methods of optimal control lend themselves well to traditional rigid robot designs, the nonlinear and high degrees of freedom (DoFs) of soft robots makes their control difficult both computationally and theoretically. Consider, for example, the difference between a single rigid robotic arm with an actuator and a soft tentacle-like robotic arm with the same actuator (Fig. 1).

*A prime example of the former is the development of dynamic programming, a result of mathematical research by Richard Bellman at the Rand Corporation, which eventually enabled computers to solve optimal control tasks directly.[1] An important example of the latter is the mathematical theory of proportional–integral–derivative (PID) controllers. While early PID controllers were first developed and applied at the turn of the 20th century, the mathematical theory of their operation lagged by almost two decades.[2]

†See Kirk[4] for an introduction to classical optimal control.

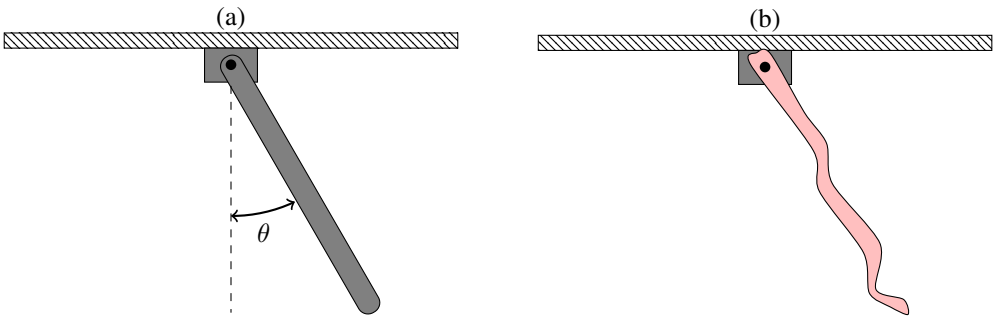

Fig. 1 Two simple implementations of a robotic arm: (a) A rigid robot arm attached to an actuator and (b) a soft, flexible robotic arm attached to the same actuation. While the rigid robot's dynamics can be represented compactly with a single DoF (Eq. 2), the soft robot does not lend itself to any low complexity representation.

Using the state space description of control systems, a dynamics function, $f(\vec{x}(t), \vec{u}(t))$, that updates the state of our robot, $\vec{x}(t)$, given control inputs, $\vec{u}(t)$, can be defined as follows:

$$\frac{d\vec{x}}{dt} = f(\vec{x}(t), \vec{u}(t)) \; . \tag{1}$$

For the rigid robot this can be explicitly written down in terms of a single rotational DoF, θ, as follows:

$$\frac{d}{dt}\begin{pmatrix}\theta \\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}\dot{\theta} \\ \frac{1}{I}[\tau_a - mgl\sin\theta]\end{pmatrix}, \tag{2}$$

where τ_a is the applied actuator torque and I is the robot arm's moment of inertia. While this system of equations is nonlinear, it is approachable computationally and amenable to a dynamic programming solution. Additionally, for small values of θ, the system is approximately linear and can be controlled exactly for specific cost functions. In contrast, no such simple solution exists for the tentacle robot. It could be potentially modeled as a spring-mass system or, alternatively, a continuum mechanics approach could be used to model it as a massive string. However, either approach would require a substantial increase in the number of DoFs compared to our rigid robot (in fact, the continuum system has infinite DoFs by definition). Additionally, these complex systems rarely have analytical solutions, requiring numerical approximations of the underlying dynamics. All of these concerns combined make computational modeling and control of these systems from first principle extremely difficult. To top it all off, this is one of the simplest soft robots you could imagine. Things continue to get worse as you add complexity into the design and

control requirements of more realistic soft robotic systems.

2.1.2 Solutions: Data and Local Control

Modern work in soft robot control attempts to solve these problems using a variety of different approaches. Two approaches that consistently appear in the literature are data-driven modeling approaches and simplifications of control to local, instead of global, rules.[5–11] The former solution avoids the difficulty associated with formulating analytical models of these complex systems, while the latter approach instead breaks the problem down into simpler “independent" control problems. Both methods have their merits and drawbacks, which are discussed in Sections 2.2 and 2.3. As a final note, while this section focuses on control in the context of soft robots, many of the methods discussed here can also be applied to hard robotic systems with high DoFs. For example, the brittle star robot[12–14] and the snake robot[15] are all hard robots, but their high DoFs and nonlinear dynamics make their control difficult with traditional optimal control methods.

2.2 Model Predictive Control

2.2.1 Model-Based Control

A model-based control is any control that relies on having a model of the system’s dynamics (alternatively known as the “plant’s dynamics" in model-based control and design literature). The most popular example of a model-based control is classical optimal control. One of the key steps in solving classical optimal control problems is specifying the dynamics of the system, $f(\vec{x}(t), \vec{u}(t))$. These dynamics serve as the model of the system. While the model is often specified analytically, this is not strictly required. For instance, in the earlier example of the soft tentacle robot, it was established that a purely analytical model of its behavior would be difficult to construct. This issue could be avoided instead by constructing a statistical model using data collected from a physical actualization of the robot. The new statistical model could then be used as the model of the system’s dynamics for a model-based control.

While many model-based controls allow the use of data-derived models, this alone does not make a particular model-based control useful for soft robot control. As outlined in Section 2.1, further limitations are often necessary, such as a focus on local dynamics. One example of a model-based control that takes this approach and successfully applies it to soft robotics systems is model predictive control, which limits

its solutions to finite time horizons. This effectively limits the dynamics the control algorithm considers to those local in time. Section 2.2.2 describes this method and reviews its application to soft robotic control. For a general review of model-based controls in soft robotic systems, see Della Santina et al.[7]

2.2.2 Model Predictive Control

Model predictive control is a specific model-based control, and, as such, provides an algorithm for controlling a dynamical system given a model of its dynamics. Consider a system whose dynamics are model by a function $f(\vec{x}(t), \vec{u}(t))$ and Eq. 1, where $\vec{x}(t)$ and $\vec{u}(t)$ are the state and control input of the system at time t, respectively. Additionally, there is some system cost function, $P(\vec{x}(t), \vec{u}(t))$, that takes the form

$$P(\vec{x}(t), \vec{u}(t)) = \int_{t_0}^{t_f} g(\vec{x}(t), \vec{u}(t))dt + h(\vec{x}(t_f)\vec{u}(t_f)) \tag{3}$$

over the time interval $[t_0, t_f]$. In other words, $P(\vec{x}(t), \vec{u}(t)$ gives the cost of operating the system through trajectory $\vec{x}(t)$ with controls $\vec{u}(t)$ over the time interval $[t_0, t_f]$. Given this, the optimal control trajectory is the pair $\vec{x}^*(t)$ and $\vec{u}^*(t)$ that minimizes the cost $P(\vec{x}(t), \vec{u}(t))$ over the control interval $[t_0, t_f]$. Mathematically this is given by the following relationship:

$$P(\vec{x}^*(t), \vec{u}^*(t)) = \min_{\vec{u}(t)} P(\vec{x}(t), \vec{u}(t)) \ . \tag{4}$$

Finding this optimal trajectory and its accompanying control trajectory is the goal of optimal control theory and can be accomplished with an optimal control algorithm. However, it is important to note that $\vec{x}(t)$ and $\vec{u}(t)$ may be elements of some constrained set and t_f may be large or even unbounded. This, in turn, means Eq. 4 can be a constrained minimization problem over infinite time, a difficult class of problems to solve in general.

Model predictive control attempts to simplify this problem by considering instead only “small" finite time horizons. First, it introduces two new time intervals, Δt and δt with $\Delta t > \delta t$. Then the original time interval, $[t_0, t_f]$, is discretized into steps of size δt, resulting in a sequence of time points $\{t_0, t_1, t_2, ..., t_f\}$ where $t_i = t_0 + i\delta t$. Next, to find $\vec{u}^*(t)$, a finite time optimal control problem is iteratively solved at each

new time point. Starting at t_0 the $\vec{u}'(t)$ is identified, which minimizes

$$P_f(\vec{x}(t), \vec{u}(t); t_0, \Delta t) = \int_{t_0}^{t_0+\Delta t} g(\vec{x}(t), \vec{u}(t))dt + h(\vec{x}(t_f), \vec{u}(t_f)) \ .$$

Then $\vec{u}^*(t)$ is set to $\vec{u}'(t)$ for the time interval $[t_0, t_1]$, and the process is repeated starting at t_1 with initial state $\vec{x}(t_1)$ given by

$$\begin{aligned} \vec{x}(t_1) &= \vec{x}(t_0) + \int_{t_0}^{t_1} f(\vec{x}'(t), \vec{u}'(t))dt \\ &= \vec{x}(t_0) + \int_{t_0}^{t_1} f(\vec{x}^*(t), \vec{u}^*(t))dt \ . \end{aligned}$$

Thus, to find $\vec{u}^*(t)$ during the interval $[t_i, t_{i+1}]$, $\vec{u}'(t)$ is found such that

$$P_f(\vec{x}'(t), \vec{u}'(t); t_i, \Delta t) = \min_{\vec{u}'(t)} P_f(\vec{x}(t), \vec{u}(t); t_i, \Delta t) \tag{5}$$

with initial state given by

$$\vec{x}(t_i) = \vec{x}(t_0) + \int_{t_0}^{t_i} f(\vec{x}^*(t), \vec{u}^*(t))dt \ , \tag{6}$$

and is taken as $\vec{u}^*(t)$ for $[t_i, t_{i+1}]$. An important note is that Δt and δt are parameters of the model predictive control algorithm and need to be chosen for each specific control problem.

2.2.3 Advantages and Limitations of Model Predictive Control

Model predictive control breaks an optimal control problem into a set of smaller local-in-time optimal control problems that can be solved in series via Eqs. 5 and 6. This has the benefit that the control trajectory can be computed online, as you only need the solution for the next δt time interval, rather than the entire problem time. This online computation also means model predictive control based controllers can be left unsupervised for large stretches of time. Their explicit inclusion of constraints in the iterative process implies it is unlikely for a controller to attempt to move outside the permitted regions of state space. Additionally, it reduces a possibly infinite time problem into a series of k step processes that can be solved using dynamic programming. These appealing characteristics are responsible for model predictive control's early adoption into industrial control pipelines, particularly in petrochemical and process industries. Furthermore, while there are simpler meth-

ods of process control, such as PID controllers, model predictive control schemes produce higher quality results when the systems dynamics are well modeled.

Of course, model predictive control is not without limitations. The most important of them is that, as its name implies, the process relies on having a high-quality model of the system's dynamics. When this condition is not met, the control trajectories produced using model predictive control cannot be trusted. Furthermore, while model predictive control solves a series of finite time horizon optimal control problems to produce its optimal trajectory, it never directly solves the original control problem presented in Eq. 4. This introduces the possibility that the trajectory constructed using model predictive control is not optimal for solving the original infinite time horizon optimal control problem. Luckily, industry's widespread adoption of model predictive control provided a substantial impetus to determine when model predictive control solutions converge to the full optimal control solutions. For a review of research into the stability and optimality of model predictive control trajectories, as well as a review of the early history of model predictive control, see Mayne et al.[16] For a more detailed review of industrial model predictive control, see Qin and Badgwell[17] or Holkar and Waghmare.[18] Finally, Schwenzer et al.[19] provide a more recent review of the state of model predictive control with emphasis on practical applications and recent advances in theory.

2.2.4 Model Predictive Control for Robot Control

As mentioned in Section 2.2.3, model predictive control is an older control method and one of the earliest widely applied model-based control methods. Despite this, it was not commonly used for the control or robotics systems until recently. This dearth of applications stems from the difficulty of accurately modeling the nonlinear and high-dimensional dynamics inherent to all but the simplest of robots. Modern advances in computers have greatly expanded the scope of problems that can be modeled using data-intensive techniques like neural networks and Gaussian processes. This allows accurate modeling of these previously unapproachable systems and, in turn, makes model-based controllers feasible. A number of recent publications applying model predictive control to robotic systems are reviewed in this section. Special attention has been given to applications on nonlinear systems, as well as compliant robotic systems.

Erez et al. used model predictive control to control a simulated humanoid robot

competing in a series of Defense Advanced Research Projects Agency challenges.[20] Lenz et al. designed a fast real-time model predictive control framework taking advantage of modern deep learning tools, which they named DeepMPC, to control a robot cutting fruit.[21] Neurnert et al. designed a method for fast computation of nonlinear model predictive control trajectories using linearization and a custom iterative Linear Gaussian Quadratic control algorithm before demonstrating its use on a hexacopter and ball bot with a variety of control tasks.[22] Gillespi et al. used a neural network to learn the nonlinear dynamics of a pneumatic soft robot, which became the model used in a model predictive control implementation to control the robot's actuation.[23] Finally, Bruder et al. used Koopman operator theory and extended dynamic mode decomposition to design a model predictive control algorithm for the control of a pneumatic soft tentacle robot.[24]

While at first glance each of these papers appears to take very different approaches to solving their control problems, they share important similarities. First and foremost, each study has a step where they model–approximate the dynamics of their system. Second, each project formulates a novel cost function for their task. In fact, a large part of Erez et al.'s success is in the design of modular and hierarchical cost functions.[20] In fact, when these two properties are considered in tandem, a clear trend applying model predictive control to a new problem materializes: model the system dynamics, design a control cost function, and apply the model and cost function in a model predictive control framework to solve for control trajectories.

In contrast to their similarities, what sets each approach apart is their trade-off between generalizability and implementation cost. Both Lenz et al. and Gillespi et al. use a neural network to effectively model the nonlinearities of their systems and, while this approach is theoretically extremely general, its implementation required training on a data set produced specifically by the target system.[21,23] This limits the actual generalizability of their solutions, as a network produced to control a specific actuator or robot cannot be directly applied to controlling a different one. Meanwhile, Neurnert et al. still requires a pre-computed model of the systems dynamics, but can be implemented immediately given this model and is shown to quickly and accurately complete non-trivial control tasks.[22] Bruder et al.'s approach is particularly interesting, as it is extremely general while simultaneously requiring very little training data and results in a linear control problem.[24] However, there is a subtle trade-off between the accuracy of the method and correctly choosing a

set of representation functions for the observables. Selecting too large of a set of observables hampers the implementation speed and complexity, but a smaller set of observables may result in inabilities to model important dynamics of the system.

2.3 Decentralized Control

2.3.1 Decentralized Systems

While model predictive control describes a particular algorithm for constructing controls for classical control problems, decentralized control instead focuses on the control of decentralized systems. A decentralized system is a system where the individual DoFs are treated independently and the global behavior of the system is emergent from local interactions. For example, a traffic grid is a good example of a decentralized system. To understand how traffic flows through the entire system, you need to look at how each individual car interacts with other nearby cars and traffic lights. In this example, the cars are the individual DoFs and the states of the traffic lights are what control flow. With this in mind, it is possible to see that the entire system could be controlled in a decentralized manner by designing control rules for each individual traffic light, given the actions of the nearby cars and traffic lights.

Many systems can be represented and constructed as either a centralized or decentralized systems. If all the traffic lights in our traffic grid example were simultaneously controlled by a single central controller, our traffic grid would instead be a centralized system (Fig. 2). So what is the benefit of representing and controlling the traffic grid as a decentralized system? Surely a centralized controller would always find more optimal controls than a decentralized system. The answer is that many physical systems have too high of a complexity or number of DoFs to control centrally. While it is entirely possible to centrally control all the traffic lights for a small town, simultaneously controlling all the traffic lights in a major city like Miami would be computationally infeasible. Thus, to control such a system the representation of its dynamics must be simplified somehow, and modeling it as a decentralized system is an effective way of accomplishing this. Referring back to the earlier example, it is conceptually simpler to control Miami's traffic grid with a number of independent controllers that are each responsible for a small neighborhood than a single central controller that needs to know the state of the entire system.

Since decentralized control schemes naturally break global problems into individual local control problems, they are well-suited for use on soft robotics systems. Section 2.3.2 discusses how modeling compliant robots as decentralized systems and designing decentralized control rules can mitigate some of the traditional difficulties of controlling soft robotics systems. It begins with a discussion of general decentralized control before discussing specific examples in the control of compliant robots.

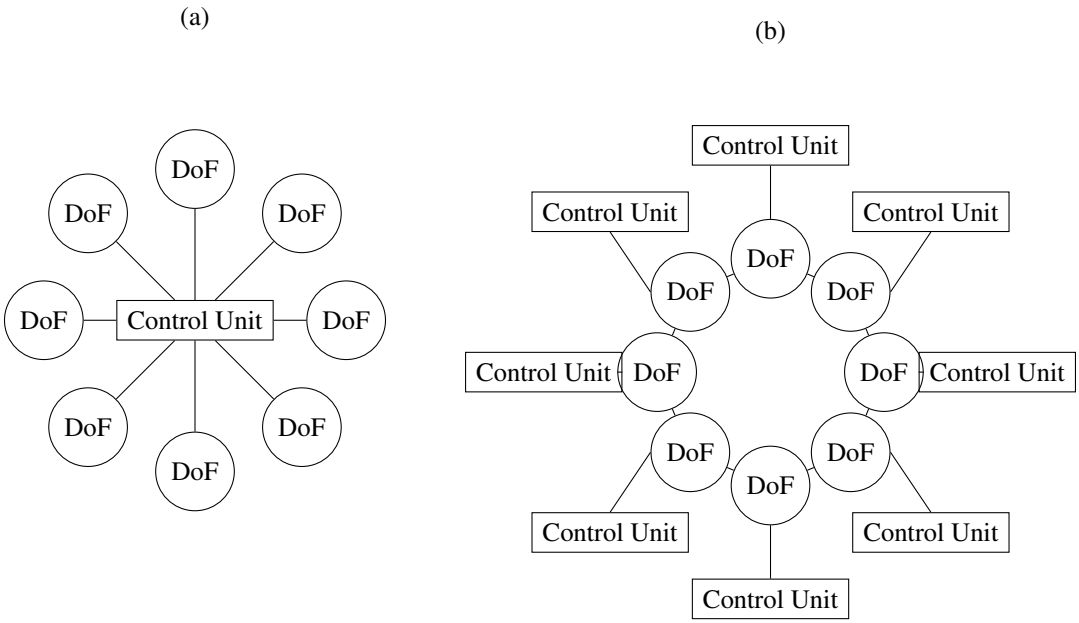

Fig. 2 Schematic of the difference between (a) central and (b) decentralized control schemes. For the centralized control problem all DoFs connect directly to the central control unit. In contrast, DoFs for the decentralized control unit connect to one local control unit. Additionally, each DoF is connected to their neighbors.

2.3.2 Decentralized Control

As discussed in the previous section, treating a complex system as decentralized can significantly simplify the representation of its dynamics and, by extension, its control. However, it does come at the cost of a paradigm shift in how system control is thought about. Recall that a state space representation of a system is often used in classical optimal control. That is, the system is represented as a trajectory of states, $\vec{x}(t)$, whose state is updated by a dynamics function, $f(\vec{x}(t), \vec{u}(t))$, with control inputs, $\vec{u}(t)$, according to Eq. 1. While it is possible to maintain this representation in decentralized control, it is counterproductive to the goal of simplification. Instead,

the system is represented by considering the state of the ith DoF, $\vec{x}_i(t)$. To make this difference more concrete, in the previous traffic example $\vec{x}(t)$ would be the state of *every* car in the traffic grid, while $\vec{x}_i(t)$ is the state of the ith car. Furthermore, when considering how the ith car updates its state, it must depend on other nearby cars, traffic lights, and relevant sensory–control signals. Generalizing this idea, the new dynamics equation for the ith DoF with interacting neighbors, N, and control inputs, $\vec{u}_i(t)$, becomes

$$\frac{d\vec{x}_i}{dt} = f(\vec{x}_i(t), \{\vec{x}_k(t)\}_{k\in N}, \vec{u}_i(t)) \ , \tag{7}$$

where $\{\vec{x}_k(t)\}_{k\in N}$ are the states of neighboring DoFs. By adopting the formalism represented by Eq. 7, the system is transformed into a decentralized system where individual DoFs are updated according to their interactions with their neighbors and controls. Additionally, note that while they are being referred to as neighbors, there is no strict requirement that interacting DoFs be physically proximate. The distance scale and strength of interactions between DoFs are important characteristics of a decentralized system, and have substantial effects on the dynamics and control of the system.[25]

Next, it is important to consider the control cost function and how to appropriately update it for a decentralized system. Recall that classical control problems have cost functions that take the form described by Eq. 3. That is, they are composed of a term for the system running cost, $g(\vec{x}(t), \vec{u}(t))$, which can be something like the energy needed for running at robot per unit time, and a term for the terminal cost, $h(\vec{x}(t_f), \vec{u}(t_f))$, which could be something like a penalty for a robot based on how far away its final state is from a desired final state. Each of these functions depends on the full system state and control and, in this regard, Eq. 3 is a global cost function. As mentioned before, it can often be infeasible to compute global functions of a large decentralized system. To avoid these costly computations, the global cost function is reformatted into a local cost function as follows:

$$P(\vec{x}_i(t), \vec{u}_i(t)) = \int_{t_0}^{t_f} g(\vec{x}_i(t), \vec{u}_i(t))dt + h(\vec{x}_i(t_f), \vec{u}_i(t_f)) \ . \tag{8}$$

In contrast to Eq. 7, the local cost equation (i.e., Eq. 8) does not consider the states of the ith degree of freedom's neighbors. There are circumstances where it might be desirable to include this information in the cost. As an example, in systems where

DoFs repel each other, a close neighbor may result in a higher local running cost. In these cases, Eq. 8 can be updated to consider neighbor states, or a composite cost function can be constructed from the costs of individual DoFs.

Decentralized control is a vast and well-studied topic due to its importance to modern computer systems and infrastructure design (the motivation for the earlier example). Depending on the specifics of a decentralized system, different approaches can be used to handle Eq. 7 and design controls to optimize the local and/or global states of the system. However, the mathematical details of the general case of decentralized control are far beyond the scope of this report. For a more comprehensive review of the mathematics of decentralized control, see Bakule.[26] Alternatively, see van Schuppen et al.[27] for both a mathematical overview and practical examples. For general information about the status of decentralized control see Antonelli[28] or Bakule.[29]

2.3.3 Decentralized Control in Soft Robotic Systems

Soft robotic systems' large number of DoFs is a significant challenge when designing feasible control trajectories. As with the Miami traffic grid example, it is often unrealistic to have a single central controller with direct connections to every individual DoF in a complex and compliant robotic system. And, as with the traffic grid example, decentralized control provides an avenue to ameliorate this issue. Instead of having a single central controller, a number of local controllers are distributed throughout the robotic system and used to locally control individual, or small groups of, DoFs. Controlling the entire system then becomes a problem in designing local controls that, when interacting, produce an emergent global control.

This approach is implemented in a variety of way throughout the literature. Lal et al. used cellular automata to design a decentralized control scheme for a brittle star robot.[12] Additionally, Lal et al. used a genetic algorithm to design decentralized primitive directional control signals for the same brittle star robot.[13] Umedachi et al. designed a decentralized control method for controlling a soft slime mold robot capable of deforming to squeeze between obstacles and through tight channels.[30] Watanabe et al. designed an analytical nonlinear control scheme for directed movement in a brittle star robot.[14] As a final example, Butler et al. designed and proved the consistency of a cellular-automata-based decentralized control scheme for a reconfigurable soft robot.[31]

When considered as a group, the aforementioned studies reveal a common algorithm for developing a decentralized control scheme for a soft robotic system. Their first step was identifying individual DoFs. In the brittle star robot each individual arm link was treated as a DoF, while the DoFs in the slime mold robot were discretized nodes along the boundary of the robot. The next step was determining which DoFs interacted and what information each DoF has about their neighbors. Capturing all relevant interactions had substantial effects on the quality of the controls that could be constructed. For example, a variety of different neighborhood designs were tested in Lal et al. with with drastic variance in their control success.[12] Umedachi et al. had both local and global interactions between their DoFs, but each individual DoF only had access to local sensory information.[30] Despite this, it was possible to design a control that effectively directed the robot and appeared to behave biologically. The final step was the design of a local decentralized control for each DoF. While simple in statement, this final part of the process is both the most important and most difficult. Both first principle and data-driven approaches were employed, with each method enjoying success across a variety of different robot designs. While analytic control schemes were simpler to implement, they were less general and more difficult to design. In contrast, data-driven methods allowed for generality, but required the design of either high-quality simulation or an actual robot for the production of training data. In either case, the result was tailored to the specifics of the study robot, making the methods the more important contribution of each study.

2.4 Soft Robotics Control Summary and Conclusion

Section 2 has attempted to provide background on the unique challenges of designing controllers for soft robotic systems and review a number of methods that have been successfully applied to the control of compliant robotic systems. While model predictive control and decentralized control appear to be unrelated methods at first glance, closer inspection reveals they possess a number of similarities. Primarily, both methods attempt to simply control a more manageable problem by looking for local solutions. The solutions are local in time in model predictive control, while they are local to each DoF in decentralized control. Additionally, both methods allow the inclusion of data-driven models, allowing separation of dynamics and control.

This report has been interested primarily in the theoretical control of soft robotic

system and, as such, has ignored a major aspect of the difficulties of controlling soft robots; that is, the the actual implementation of control units and actuators in compliant materials. To an engineer, this is arguably a more important feature than knowing a controller can theoretically be implemented for a given robot. Thankfully, other reviews have spent time looking at the current state of the design of controllable soft robots. For further details about this, see Iida and Laschi,[32] Ren et al.,[33] Fang et al.,[34] and Stella and Hughes.[35]

3. Agile Control in Biological and Robotic Systems

One of the hallmarks of biological systems is their ability to adapt to changes in the environment. Even organisms traditionally considered extremely simple, such as insects, are able to integrate sensory information quickly and design rapid responses to the world around them. In particular, when this behavior occurs on a short timescale compared to the typical behavioral timescales of the organism, we refer to it as an agile behavior. These types of agile responses, such as a fly nimbly avoiding a flyswatter, can be found all over the animal kingdom, and even in some plants, protists, and bacteria. Despite this, replicating agile behaviors in synthetics systems is a difficult problem with a long history of research in both biological and engineering sciences.[36] Even the most advanced implementations for soft robot control reviewed in the Section 2 appear sluggish and uncoordinated when compared to living systems. For example, Octobot (i.e., the first full autonomous soft robot) is a engineering marvel that paved the way for a new generation of autonomous soft robots.[37] However, when its motions are compared to an actual octopus (the inspiration for its design), it appears slow and uncoordinated. Section 3.1 attempts to shed light on the performance discrepancy between biological and robot systems by reviewing the two related topics of sensorimotor control and reflexive behaviors. Both topics have been widely studied in the biological sciences, and a number of groups have tried to incorporate ideas from these fields into their designs of robotic systems. Section 3 also includes a review of current definitions of agility in motile systems before discussing the design of metrics for agility and their potential applications for robot design.

3.1 Sensorimotor Control

3.1.1 Sensorimotor Systems

The sensorimotor system of an organism is the set of biological components that enable it to sense and interact with the external environment. Traditionally, interacting has been understood to mean movement relative to the environment. However, no significant difference is made in most literature between translational movements and movements of an organism's body. In fact, a large number of studies focusing on the sensorimotor system limit their attention to simple "stationary" motor tasks, such as pointing and eye movement. While most use of the term *sensorimotor system* is in the context of biological organisms, it can also be used to refer to the analogous set of components in a robotic system. For example, the sensorimotor system of a simple quadcopter would be the motors, props, sensors, and control unit.

The sensorimotor system of any given organism is a complex and highly interconnected network of biological components. For example, the sensorimotor system in humans is comprised of all the sensory organs (eyes, ears, mouth, noise, mechanoreceptors, etc.), the central nervous system, the peripheral nervous system, the skeleton, and all the musculature. Understanding how all of these systems couple and interact is a devilishly complex problem. To make things worse, sensorimotor systems change throughout ontogeny and can be adapted both physically and through behavioral mechanisms. Altogether, this makes understanding the sensorimotor systems of even simple organisms a daunting task. With this in mind, this report does not attempt to review sensory-motor coupling in the general sense, but rather focuses on the specific problem of sensorimotor control. For the curious, here are a number of interesting papers studying and reviewing the sensorimotor systems of a variety of organims: Diamond et al.[38] look at the whisker sensorimotor system of mammals, Blitz and Nusbaum[39] look at neural circuit flexiblity in a decapod sensorimotor system, Huston and Jayaraman[40] review sensory integration in insects, and Cohen and Denham[41] discuss whole organism sensorimotor modeling of the nematode (*Caenorhabditis elegans*). While not comprehensive, these works hopefully provide a glimpse of the breadth of problems and approaches that fall within the category of sensorimotor systems research.

3.1.2 The Sensorimotor Control Problem

Understanding how the sensorimotor system is controlled to create coordinated motion is an important question in the biological sciences. When viewed through the lens of control theory, this particular problem is referred to as sensorimotor control. As in classical optimal control theory, sensorimotor control theory takes the approach that an organism can be represented as a dynamical system with a state space representation, $\vec{x}(t)$, and corresponding dynamics equation, f. However, while classical optimal control describes this relationship with Eq. 1, sensorimotor control problems require a more abstract approach. First, both the state of an organism and its environment can affect the organism's dynamics. This means that the dynamics model must be, in general, nonautonomous and time dependent, updating Eq. 1 to

$$\frac{d\vec{x}}{dt} = f(\vec{x}(t), \vec{u}(t), t) \ . \tag{9}$$

The next major change is the functional relationship between $\vec{x}(t)$ and the control trajectory, $\vec{u}(t)$. Until now it has been assumed that there exists an explicit dependence of $\vec{u}(t)$ on $\vec{x}(t)$, meaning the algorithm for constructing control trajectories has direct access to the system's state. However, biological systems rarely, if ever, have direct access to their exact state variables. Instead, they must rely on sensory information about their current state and environment for designing their control trajectories. Generally, we can say that an organism has access to some observable, $\vec{y}(t)$, and that this observable has some functional relationship with the organsim's state. That is, there exists some function g such that

$$\vec{y}(t) = g(\vec{x}(t), t) \ , \tag{10}$$

where the time dependence again represents the possible influence of the environment on the sensorimotor system's performance. For example, an organism moving through a three-dimensional world never actually knows it exact position relative to any fixed coordinate system. However, the scene it sees is affected by its position and orientation. In this regard, the visual stream of the organism is an observable that tells the organism something about its current state in the world. Of important note, the dimension of a system's state and the dimension of its observable do not need to match. Continuting with the same example, the dimension of the visual signal (a function of the number of independent photoreceptors an organism has) is not typically the same as the dimension of its state (the position of the organism

in space in this context). For the specific case where g is invertible (i.e., there is a single unique value of x that produces each value of g and vice versa) it is said that the dynamical system is observable, which, as mentioned previously, has been the assumption of the report until now. With this in mind, the control trajectories must instead be a function of the observables, meaning

$$\vec{u}(t) = h(\vec{y}(t), t) \tag{11}$$

for some h that is determined as a result of the optimal control solution. The importance of this distinction is that sensorimotor control trajectories are only implicitly dependent on $\vec{x}(t)$ via g. The current organism state can be used only for the design of a control when g is both known *and* invertible.

The final difference is the addition of stochasticity into our dynamical system. Nonobservable systems have multiple states that correspond to a given observable, $\vec{y}(t)$. Furthermore, many natural environments exhibit chaotic behaviors, making them difficult to model deterministically. One approach for dealing with these difficulties is treating $\vec{x}(t)$ as a random variable. While this approach adds significant complexity to the mathematics of our dynamical system, it allows us to model the uncertain and stochastic behaviors of real biological systems. Unfortunately, reviewing stochastic control in any generality is beyond the scope of this report. For a brief introduction to the topic, see either Kappen[42] or Lu and Zang.[43]

3.1.3 Sensorimotor Control of Biological Systems

As was established earlier, a full review of sensorimotor control theory and its applications to biological systems is substantially beyond the scope of this report. This is partially because many high-quality reviews of the problem already exist. For example, Franklin and Wolpert provide a detailed review of five major modes of computation in sensorimotor control: Bayesian decision theory, optimal feedback control, predictive control or forward models, impedance control, and learning.[44] Similarly, Körding and Wolpert provide a brief overview of the application of Bayesian decision theory to human sensorimotor.[45] Todorov provides a general review of the various optimal control laws derived to explain results from experiments in sensorimotor control, with an emphasis on open-loop and closed-loop control.[46] In combination, these three studies provide an overview of human sensorimotor control research while providing a strong foundation for further research

and specialization.

In addition to the previously mentioned reviews, a few original studies stood out as particularly interesting and/or applicable to the agile control of soft-bodied systems. The first is Todorov and Jordan, where the idea of using stochastic optimal feedback control as a model for biological sensorimotor control was first introduced.[47] The paper outlines how a linear-quadratic-Gaussian control problem can be used to accurately model human motor coordination. The second is Mitrovic et al., which introduces the use of limb impedance control as a method of dealing with internal sources of motor noise.[48] While their results are specific to human muscular control, the idea of impedance-based controls is more general and applicable across any mechanical system with muscle-like properties.[49–51]

Given the current state of sensorimotor control as outlined by the cited literature, it becomes clear that uncertainty management is a driving force in biological sensorimotor systems. Nearly every method previously mentioned is related to managing either external uncertainty, as in the case of Bayesian decision theory, or internal uncertainty, as in impedance control and forward modeling. This is a defining difference between the current discussion of sensorimotor control and our earlier discussion of soft robotics control. In Section 2, proper modeling of a high-dimensional system and simplification of its dynamics was the major focus of the works discussed. While this problem still exists for sensorimotor systems, it is compounded by not having access to the raw state variables for control design and instead using noisy sensory signals. This limitation, coupled with the finite control precision of biological organisms, forces naturally evolved sensorimotor control systems to be robust to both internal and external noise. This, in turn, implies that any computational method for constructing sensorimotor control trajectories should share similar theoretical properties, leading to the focus on uncertainty management in the literature.

3.2 Reflexive Behaviors and Measures of Agility

3.2.1 Agile Motion in Biological Systems

As has been emphasized throughout this review, one of the most striking differences between current robotic systems and biological organisms is how “clumsy" robotics systems are compared to their biological counterparts. Living animals often appear to move effortlessly through complex and dynamic environments. For example,

consider the difference between a hawk flying through a forest canopy in pursuit of prey and a drone navigating this same environment. While the hawk is able to boldly navigate its natural environment, the drone must proceed with extreme caution to avoid getting enmeshed within the network of branches and vines that make up the canopy. The hawk's ability to precisely navigate this complex environment is a prime example of the agile behaviors that were discussed in the introduction of this section. Such behaviors are currently beyond the ability of all but the most specialized robot designs. Even then, the agility of these specialized designs is often constrained to one type of environment.[52–54] In contrast, living organisms are often capable of behaving agilely in a wide range of natural environment. The next few sections of this report will finally and directly attempt to understand the origin of the performance differences observed between biological and robotic systems before reviewing current methods to both quantify and bridge this gap. The majority of the focus will be devoted to the topics of reflexive behaviors and agility, with special interest in metric of agility in robotic systems.

3.2.2 Reflexive Behaviors

One particular aspect of agile behavior in biological systems is agile sensorimotor control. This is an organism's ability to quickly adjust sensorimotor trajectories in response to important external stimuli. The hawk in the previous example needs to make hundreds of micro-adjustments to its pose while navigating the forest canopy to avoid obstacles in its path. Many of those adjustments are made on the fly since the hawk does not know what path its prey will take before the pursuit begins. How does the hawk decide which adjustments to make when presented with a wealth of sensory information while simultaneously focusing on the task of catching its prey? One way the hawk accomplishes this task, beyond the methods discussed in Section 3.1.3 on sensorimotor control, is by offloading some of the primitive response loops to its peripheral nervous system. These independent fixed-response patterns are commonly referred to as reflexes. Since they are independent of the central nervous system, reflexes are often much quicker than typical behavioral responses. This makes them important for time-sensitive response tasks, such as how the hawk should reorient its wing after brushing against a tree branch. Additionally, the reflexes reduce the burden on the central processing system of the hawk, allowing it to focus more of its attention on the squirrel it is pursuing.

Every animal with a well-developed nervous system appears to have reflexes that

serve a variety of important purposes. For example, motor reflexes in humans are important for stabilizing locomotion trajectories.[55] However, despite the importance and wide use of reflexes in biological systems, it is unclear how to implement these same behaviors in robotic systems. Even the reflexes of simple organisms hide a surprising complexity. For example, Coulston and Pakzad attempted to model the bending reflex of the leech (*Hirudo medicinalis*) using an artificial neural network.[56] While a leech is much more primitive than a human, copying this easily described reflex still posed a significant challenge. Accurately replicating this behavior in a soft robot would require a whole suit of pressure sensors and a dedicated computational pathway, all to make the soft robot simply bend away from being poked. How would it then be possible to replicate something as complex as human locomotion? One suggestion is to look at how reflexes develop in biological systems. Marques et al. developed a model of limb motion that relied on an anti-Hebbian update rule to show how spontaneous motor activity can be coordinated into recognizable behavioral reflexes.[57] It is already a commonly held postulate that biological neural networks update in Hebbian fashion, so perhaps it is reasonable to think reflexes develop in a similar way.[58–60] This is a particularly tempting line of thought when compared to the alternative, which is to custom design each individual reflex. As an example of this, Kamegawa et al. designed a three-dimensional reflex for navigating over irregularities along a prespecified pathway in a snake-like search and rescue robot.[15] Although their final product works well for their designed task, their method cannot be adapted to tackle new reflexes without additional thought and planning.

Finally, it is worth noting that decentralization plays an important role in reflexive behavior. Reflexes work by decentralizing a behavioral computation to a local piece of the peripheral nervous system. Since decentralization is already a popular way of dealing with the complexity of soft robot control, it seems reasonable that reflexive behaviors should pair nicely with this control paradigm. However, there exists a major difference in how decentralization is managed in reflexes and in soft robot control. The nervous systems of most organisms have a natural hierarchy, with the brain acting as a central processor. This is in contrast to the forms of decentralized control discussed earlier in this report, where all control units have equal computational powers and roles. So, while reflexes work to decentralize the computations of sensorimotor systems, a central processor still exists within the organism and continues to manage control trajectory design in a centralized way. While different

than the earlier examples of decentralization in soft robots, this semi-decentralized design is very attractive as it allows the development of “cheap" central controllers and controls. In this paradigm, the central controller works in broad strokes, while the peripheral controllers deal with the specific details of implementing the desired trajectory.

3.2.3 Measures of Agility

While reflexive behaviors are certainly an important part of an organism’s agility, they are not the only piece of the puzzle. Some amount of agility clearly stems from morphology, and another part is related to the sensory systems. And, of course, the central nervous system of the organism must play an important role in agility. As an example of how all these systems work together to create agile behaviors, Daley looked at how Guinea fowl adjusted to unseen potholes while running.[61] They found that the birds used several control strategies to recover from the unexpected perturbations: independent limb control, feed-forward regulation of leg-cycling rates, load-dependent muscular actuation, and multistep recovery reflexes. This diversity of strategies for dealing with a common obstacle hints at the complexity required to develop truly agile systems.

With this complexity in mind, determining which features of an organism contribute to its agility is a difficult open problem and an important step that must be completed before designing agile robotic systems. One way to approach this problem is to define agility in a qualitative sense and then develop metrics to measure agility in both biological and robotic systems. Along this line, Sheppard and Young attempt to define agility in a 2006 review of human agility as “a rapid whole-body movement with change of velocity or direction in response to a stimulus.”[62] While this definition is proposed in the context of human athlete performance, its applies equally well to the movement of Guinea fowl over an unexpected obstacle. In this way, it captures the essence of agility: a *rapid* response to an external stimulus. In other words, being able to achieve large accelerations is not sufficient to be deemed an agile system; you need to generate these accelerations in response to an external stimulus.

While this definition provides a qualitative description of agility, it still remains to develop metrics of agility that properly capture its quantitative behavior. Given this definition, a direct attempt to measure a system’s agility would be to provide

an external response stimulus (e.g., the sudden appearance of a pothole) and then measure the time to maximum response as well as the maximum response itself. However, it is unclear how to do this in general. When is the stimulus provided for the Guinea fowl, what is the maximum response, and when does it occur? Although this may seem like a pathological example, it is more the norm than the exception for agile behaviors, which often take the form of continuous full body responses to changing sensory signals. This implies an important corollary: metrics of agility must be task specific. While it might be possible to construct composite measures of agility by averaging over specific metrics, ultimately a stimulus and response must be identified within the framework of the qualitative definition described in the previous paragraph.

Two solutions come to mind for dealing with the aforementioned issue; they are simplifying to proxy metrics or relying on high-dimensional state data. As an example of the first approach, Eckert and Ijspeert create a set of “simple" and “easy" benchmarks for terrestrial legged robot agility.[63] Based on dog agility tasks, the metrics they propose are all measures of a system’s ability to perform simple tasks, such as turning in a tight circle or jumping. While these tasks do not directly measure a response to a specific external stimulus, they establish upper bounds for potential responses. In this way, they serve as proxies for agility, with the most agile systems being those that score best across all measured tasks. The alternative to this metric-based approach would be to measure the total state of the system and how it changes in response to a specific stimuli. While this seems like an ambitious idea, recent advances in pose estimation and object tracking using deep learning make this computationally intense approach feasible. For example, the pose estimation network developed by Mathis et al. could be used to measure gait over the length of a video, and changes in cyclical behavior could be measured as metrics for agility.[64]

3.3 Agile Behavior Summary and Conclusion

Section 3 discussed a few of the unique adaptations that allow biological systems to perform agile behaviors and how these methods could be potentially replicated in robotic systems. A review of sensorimotor control was provided, with a focus on rts that developed axiomatic theories of sensorimotor control, before beginning a discussion of reflexive behaviors and how they can help simplify sensorimotor control by distributing control computations across the peripheral nervous system. Finally, the problem of defining agility in motile systems was considered and two methods

of constructing metrics for agile systems were reviewed. Across all these topics, a key idea was the management of sensory information in a world full of uncertainty. All of the adaptations discussed appear to help mitigate uncertainty in either state estimation or control implementation, further suggesting that management of uncertainty is a driving principle in biological systems.

4. Conclusion

This report has attempted to provide a foundation for researchers interested in studying questions related to the dynamic and precise control of biological systems and how to replicate these behaviors in synthetic systems. To this end, a review was provided of important background material in the realms of soft robot control, sensorimotor control, and agile control. Additionally, further readings for specialized topics were suggested when appropriate. Beyond the basic foundation this report provides to the reader, it also discussed a number of potential future research directions. Most importantly, Section 3.2 regarding metrics of agility suggests two potential approaches for measuring agility in motile systems that have been tested only on an extremely limited basis. Expanding this research to additional tasks and morphologies would go a long way in advancing the current state of the art in the agile control of synthetic systems.

5. References

1. Bellman R, Lee E. History and development of dynamic programming. IEEE Control Systems Magazine. 1984;4(4):24–28.

2. Bennett S. The past of PID controllers. IFAC Proceedings Volumes. 2000;33(4):1–11.

3. Tauber F, Desmulliez M, Piccin O, Stokes AA. Perspective for soft robotics: the field's past and future. Bioinspir Biomim. 2023;18(3):035001.

4. Kirk DE. Optimal control theory: an introduction. Courier Corporation; 2004.

5. Lee C, Kim M, Kim YJ, Hong N, Ryu S, Kim HJ, Kim S. Soft robot review. Int J Control Autom. 2017;15:3–15.

6. Das A, Nabi M. A review on soft robotics: modeling, control and applications in human-robot interaction. In: 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS); p. 306–311.

7. Della Santina C, Duriez C, Rus D. Model-based control of soft robots: a survey of the state of the art and open challenges. IEEE Control Systems Magazine. 2023;43(3):30–65.

8. Kim D, Kim SH, Kim T, Kang BB, Lee M, Park W, Ku S, Kim D, Kwon J, Lee H, Bae J, Park YL, Cho KJ, Jo S. Review of machine learning methods in soft robotics. PLoS One. 2021;16(2):e0246102.

9. Schegg P, Duriez C. Review on generic methods for mechanical modeling, simulation and control of soft robots. PLoS One. 2022;17(1):e0251059.

10. Wang J, Chortos A. Control strategies for soft robot systems. Adv Intell Syst. 2022;4(5):2100165.

11. Zhang J, Fang Q, Xiang P, Sun D, Xue Y, Jin R, Qiu K, Xiong R, Wang Y, Lu H. A survey on design, actuation, modeling, and control of continuum robot. Cyborg Bionic Syst. 2022;2022:9754697.

12. Lal SP, Yamada K, Endo S. Studies on motion control of a modular robot using cellular automata. In: AI 2006: Advances in Artificial Intelligence: 19th Australian Joint Conference on Artificial Intelligence; Hobart, Australia; 2006 Dec. Proceedings 19; p. 689–698.

13. Lal SP, Yamada K, Endo S. Emergent motion characteristics of a modular robot through genetic algorithm. In: Advanced Intelligent Computing Theories and Applications with Aspects of Artificial Intelligence: 4th International Conference on Intelligent Computing, ICIC 2008; Shanghai, China; 2008 Sep. Proceedings 4; p. 225–234.

14. Watanabe W, Kano T, Suzuki S, Ishiguro A. A decentralized control scheme for orchestrating versatile arm movements in ophiuroid omnidirectional locomotion. J R Soc Interface. 2012;9(66):102–109.

15. Kamegawa T, Akiyama T, Suzuki Y, Kishutani T, Gofuku A. Three-dimensional reflexive behavior by a snake robot with full circumference pressure sensors. In: 2020 IEEE/SICE International Symposium on System Integration (SII); p. 897–902.

16. Mayne DQ, Rawlings JB, Rao CV, Scokaert PO. Constrained model predictive control: stability and optimality. Automatica. 2000;36(6):789–814.

17. Qin SJ, Badgwell TA. An overview of industrial model predictive control technology. In: Aiche symposium series; Vol. 93; p. 232–256.

18. Holkar K, Waghmare LM. An overview of model predictive control. Int J Control Autom. 2010;3(4):47–63.

19. Schwenzer M, Ay M, Bergs T, Abel D. Review on model predictive control: an engineering perspective. Int J Adv Manuf Technol. 2021;117(5-6):1327–1349.

20. Erez T, Lowrey K, Tassa Y, Kumar V, Kolev S, Todorov E. An integrated system for real-time model predictive control of humanoid robots. In: 2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids); p. 292–299.

21. Lenz I, Knepper R, Saxena A. DeepMPC: Learning deep latent features for model predictive control. In: Robotics: science and systems XI; robotics; 2015.

22. Neunert M, de Crousaz C, Furrer F, Kamel M, Farshidian F, Siegwart R, Buchli J. Fast nonlinear model predictive control for unified trajectory optimization and tracking. In: 2016 IEEE International Conference on Robotics and Automation (ICRA); IEEE; 2016. p. 1398–1404.

23. Gillespie MT, Best CM, Townsend EC, Wingate D, Killpack MD. Learning nonlinear dynamic models of soft robots for model predictive control with neural networks. In: 2018 IEEE International Conference on Soft Robotics (RoboSoft); IEEE; 2018. p. 39–45.

24. Bruder D, Gillespie B, Remy CD, Vasudevan R. Modeling and control of soft robots using the Koopman operator and model predictive control. arXiv. 2019 Feb 7. https://arxiv.org/abs/1902.02827.

25. Battiston F, Amico E, Barrat A, Bianconi G, Ferraz de Arruda G, Franceschiello B, Iacopini I, Kéfi S, Latora V, Moreno Y, Murray M, Pexioto T, Vaccarino F, Petri G. The physics of higher-order interactions in complex systems. Nature Physics. 2021;17(10):1093–1098.

26. Bakule L. Decentralized control: an overview. Annu Rev Control. 2008;32(1):87–98.

27. van Schuppen JH, Boutin O, Kempker PL, Komenda J, Masopust T, Pambakian N, Ran AC. Control of distributed systems: tutorial and overview. Eur J Control. 2011;17(5-6):579–602.

28. Antonelli G. Interconnected dynamic systems: an overview on distributed control. IEEE Control Systems Magazine. 2013;33(1):76–88.

29. Bakule L. Decentralized control: status and outlook. Annu Rev Control. 2014;38(1):71–80.

30. Umedachi T, Takeda K, Nakagaki T, Kobayashi R, Ishiguro A. Fully decentralized control of a soft-bodied robot inspired by true slime mold. Biol Cybern. 2010;102:261–269.

31. Butler Z, Kotay K, Rus D, Tomita K. Cellular automata for decentralized control of self-reconfigurable robots. In: Proceedings of the ICRA 2001 Workshop on Modular Robots; p. 21–26.

32. Iida F, Laschi C. Soft robotics: challenges and perspectives. Procedia Comput Sci. 2011;7:99–102.

33. Ren L, Li B, Wei G, Wang K, Song Z, Wei Y, Ren L, Liu Q. Biology and bioinspiration of soft robotics: actuation, sensing, and system integration. iScience. 2021;24(9).

34. Fang J, Zhuang Y, Liu K, Chen Z, Liu Z, Kong T, Xu J, Qi C. A shift from efficiency to adaptability: recent progress in biomimetic interactive soft robotics in wet environments. Adv Sci. 2022;9(8):2104347.

35. Stella F, Hughes J. The science of soft robot design: a review of motivations, methods and enabling technologies. Front Robot AI. 2023;9:1059026.

36. Bhushan B. Biomimetics. Philos Trans Math Phys Eng Sci. 2009;367(1893):1443–1444.

37. Wehner M, Truby RL, Fitzgerald DJ, Mosadegh B, Whitesides GM, Lewis JA, Wood RJ. An integrated design and fabrication strategy for entirely soft, autonomous robots. Nature. 2016;536(7617):451–455.

38. Diamond ME, Von Heimendahl M, Knutsen PM, Kleinfeld D, Ahissar E. 'Where' and 'what' in the whisker sensorimotor system. Nat Rev Neurosci. 2008;9(8):601–612.

39. Blitz DM, Nusbaum MP. Neural circuit flexibility in a small sensorimotor system. Curr Opin Neurobiol. 2011;21(4):544–552.

40. Huston SJ, Jayaraman V. Studying sensorimotor integration in insects. Curr Opin Neurobiol. 2011;21(4):527–534.

41. Cohen N, Denham JE. Whole animal modeling: piecing together nematode locomotion. Curr Opin Syst Biol. 2019;13:150–160.

42. Kappen HJ. An introduction to stochastic control theory, path integrals and reinforcement learning. In: AIP conference proceedings; Vol. 887; p. 149–181.

43. Lu Q, Zhang X. Presented at: LIASFMA (Sino-French International Associated Laboratory for Applied Mathematics) Autumn School "Control and Inverse Problems of Partial Differential Equations" at Zhejiang University, 2016 Oct 17–22, Hangzhou, China.

44. Franklin D, Wolpert D. Computational mechanisms of sensorimotor control. Neuron. 2011;72(3):425–442.

45. Körding KP, Wolpert DM. Bayesian decision theory in sensorimotor control. Trends Cogn Sci. 2006;10(7):319–326.

46. Todorov E. Optimality principles in sensorimotor control. Nat Neurosci. 2004;7(9):907–915.

47. Todorov E, Jordan MI. Optimal feedback control as a theory of motor coordination. Nat Neurosci. 2002;5(11):1226–1235.

48. Mitrovic D, Klanke S, Osu R, Kawato M, Vijayakumar S. A computational model of limb impedance control based on principles of internal model uncertainty. PLoS One. 2010;5(10):e13601.

49. Hogan N. Impedance control: an approach to manipulation: Part ii—implementation. Transaction of the ASME. 1985;107.

50. Song P, Yu Y, Zhang X. Impedance control of robots: an overview. In: 2017 2nd International Conference on Cybernetics, Robotics and Control (CRC); p. 51–55.

51. Abu-Dakka FJ, Saveriano M. Variable impedance control and learning—a review. Front Robot AI. 2020;7:590681.

52. Delcomyn F. Biologically inspired robots. In: Bioinspiration and Robotics Walking and Climbing Robots. IntechOpen. 2007.

53. Kim S, Laschi C, Trimmer B. Soft robotics: a bioinspired evolution in robotics. Trends Biotechnol. 2013;31(5):287–294.

54. Zimmerman S, Abdelkefi A. Review of marine animals and bioinspired robotic vehicles: classifications and characteristics. Prog Aerosp Sci. 2017;93:95–119.

55. Zehr EP, Stein RB. What functions do reflexes serve during human locomotion? Prog Neurobiol. 1999;58(2):185–205.

56. Coulston C, Pakzad S. A biological-based neural network model of leech reflexive behaviors. In: Proceedings IEEE International Conference on Developing and Managing Intelligent System Projects; 1993; p. 32–39.

57. Marques HG, Bharadwaj A, Iida F. From spontaneous motor activity to coordinated behaviour: a developmental model. PLoS Comput Biol. 2014;10(7):e1003653.

58. Hebb DO. The first stage of perception: growth of the assembly. J Organ Behav. 1949;4(60):78–60.

59. Gerstner W, Kistler WM. Mathematical formulations of Hebbian learning. Biol Cybern. 2002;87(5):404–415.

60. Munakata Y, Pfaffly J. Hebbian learning and development. Dev Sci. 2004;7(2):141–148.

61. Daley MA. Understanding the agility of running birds: sensorimotor and mechanical factors in avian bipedal locomotion. Integr Comp Biol. 2018;58(5):884–893.

62. Sheppard JM, Young WB. Agility literature review: classifications, training and testing. J Sports Sci. 2006;24(9):919–932.

63. Eckert P, Ijspeert AJ. Benchmarking agility for multilegged terrestrial robots. IEEE Transactions on Robotics. 2019;35(2):529–535.

64. Mathis A, Mamidanna P, Abe T, Cury K, Murthy V, Mathis M, Bethge M. Markerless tracking of user-defined anatomical features with deep learning. In: CSF Conference: Hand, Brain and Technology: the Somatosensory System; 2018; p. 97.