A SENSORIMOTOR BASIS OF SPEECH COMMUNICATION

BY

JACOB BRYAN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Doctoral Committee:

Professor Stephen E. Levinson, Chair
Professor Mark Hasegawa-Johnson
Associate Professor Ryan K. Shosted
Assistant Professor Lav R. Varshney

ABSTRACT

This dissertation presents the development of sensorimotor primitives as a means of constructing a language-agnostic model of speech communication. Insights from major theories in speech science and linguistics are used to develop a conceptual framework for sensorimotor primitives in the context of control and information theory. Within this conceptual framework, sensorimotor primitives are defined as a system transformation that simplifies the interface to some high dimensional and/or nonlinear system. In the context of feedback control, sensorimotor primitives take the form of a feedback transformation. In the context of communication, sensorimotor primitives are represented as a channel encoder and decoder pair. Using a high fidelity simulation of articulatory speech synthesis, these realizations of sensorimotor primitives are respectively applied to feedback control of the articulators, and communication via the acoustic speech signal. Experimental results demonstrate the construction of a model of speech communication that is capable of both transmitting and receiving information, and imitating simple utterances.

To Ann, my partner and teammate in all things worth doing.

ACKNOWLEDGMENTS

Completing my doctoral research and writing this thesis has been among the most challenging and rewarding experiences of my life, and I could not have done it without the love and support of my family and friends. Foremost among them is my wife, Ann Bryan, who was always there to encourage me, help me talk through a problem, or break me out of my writer's block. I look forward to returning the favor as she begins writing her own doctoral thesis.

I would also like to acknowledge my fellow graduate students from the Language Acquisition and Robotics group, both past and present. Luke Wendt, Felix Wang, and Yuchen He have provided stimulating discussion and helpful feedback on my research throughout my time in the research group. I would especially like to acknowledge W. Jacob Wagner, who is responsible for introducing me to the concept of sensorimotor primitives for hierarchical control and whose efforts were critical in working with the Praat articulatory synthesis model.

Lastly, I would like to thank my graduate advisor, Dr. Stephen E. Levinson, and my committee members, Dr. Mark Hasegawa-Johnson, Dr. Ryan Shosted, and Dr. Lav Varshney, who provided valuable insights into my research and challenged me to become a more competent and confident researcher. In particular, I would like to express my deep gratitude to Dr. Levinson for his support, guidance, and encouragement throughout my graduate career. His guidance has been fundamental in shaping my ability bring rigorous mathematics to abstract problem spaces.

The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the author. Approved for Public Release; Distribution Unlimited. Public Release Case Number 19-0854.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Speech communication is defined as the transmission of information via production and reception of the speech signal. Although the speech signal is heavily studied, the robustness achieved in human speech communication is not fully understood. This is evidenced by the fact that automatic speech recognition systems are still outperformed by most humans. In contrast with traditional research of speech and language, this thesis considers the study of speech communication from a constructive approach by modeling speech as a communication system using an articulatory model of speech production. With this approach, the problem of speech communication can be naturally addressed by utilizing the principles of both control and information theory. Throughout this thesis, we will develop the conceptual framework of sensorimotor primitives as the building blocks of the speech communication system. Using primitives to represent the transmission of information through speech, we hope to gain deeper insight into the relationship between speech, language, and intelligence.

This chapter provides background, motivation, and basic principles of the research presented throughout this thesis. We will first provide motivation for the study of articulatory speech synthesis and its importance in the development of *speech primitives* as a basis for communication and language. We will consider major theories of speech communication in order to place this sensorimotor basis of speech into the greater context of speech science and linguistics. The general notion of sensorimotor primitives will then be introduced in the context of these prominent theories.

1.1 Motivation

The development and control of an articulatory model of speech synthesis is most directly motivated by the need for synthesized speech that sounds more natural than what is achievable using traditional concatenative methods [1]. In contrast with concatenative methods of speech synthesis, articulatory methods are able to leverage the dynamics of speech production as a means of addressing phenomena such as co-articulation. Given that the spectral features of a phoneme can be significantly different depending on what it precedes or follows, this approach eliminates the need for an exhaustive dictionary of multiphonic units. Additionally, even the most state-of-the-art methods of concatenative speech synthesis still lack the natural flow of human speech. Although concatenative methods do offer the benefit of computation simplicity, the improvement garnered by increased computational power appears to yield diminishing returns [2]. With advances in computing over recent decades, synthesizing speech through a biologically faithful model of the vocal tract is now possible on a general purpose computer. By focusing on articulatory speech production, this thesis represents a return to first principles in speech signal processing.

At a fundamental level, this work is motivated by questions about the nature of human language and the optimization principles that guide its development. Speech and language are both unique to humans and fundamentally tied to the notion of human intelligence [3]. For the majority of humans, language and language acquisition are closely tied to the speech signal, and as a consequence, speech has an intrinsic influence on the structure of languages. Furthermore, the notion of primitive units of control has been actively researched in the context of robotics; however, the relation between such primitive units and language is often neglected. In order to address this, we seek to introduce a general framework for defining the notion of these primitive units of control in terms of linguistic structure. With this in mind, we now consider the significance of language in the study of artificial intelligence and robotics. From there we will provide more in-depth reasoning as to why articulatory speech synthesis is of critical importance to this investigation of language, communication, and control.

1.1.1 Language Acquisition and Intelligence

At present, humans are the most comprehensive example of intelligence that is available for study. One approach to developing a system with intelligence comparable to humans would be to attempt to directly model the human brain. In practice, however, direct modeling of the human brain is prohibitive due to the complexity of the human nervous system and the combinatoric challenge of modeling the connections between neurons. Historically, the challenges of modeling the human brain have been circumvented by utilizing the notion of functional equivalence in the development of intelligent systems [4][5]. Under this paradigm, we need not concern ourselves with the actual mechanisms of the human brain but instead focus on developing a mathematical and computational model that exhibits equivalent behavior. With this notion of functional equivalence, we shift our focus to developing the fundamental mathematical structures of intelligence by drawing inspiration from human intelligence.

Although there are many important factors that contribute to human intelligence, language is among the most fundamental. The use of language is ubiquitous in humans and has been shown to be tied directly to the human ability to problem solve. Evidence for the significance of language in human intelligence appears in studies of people who, for reasons other than neurological impairments, never fully acquired use of language and syntax. For individuals lacking full linguistic ability, cognitive tasks such as planning or abstraction are difficult or impossible [6][7]. In addition, as children acquire language their understanding of the surrounding environment reinforces and is reinforced by the process of language acquisition [8]. With such observations in mind, we posit that language and the process by which it is acquired are the critical elements that enable the general purpose intelligence found in human cognition.

While linguistic structure has been thoroughly studied in the form of formal grammars, such mathematical structures give limited insight into natural languages used by humans [3]. This is likely due to the fact that such mathematical considerations of language do not provide any insight into the driving principles that lead to the emergence of language; they only detail the rules that natural language appears to follow. In essence, such a top-down analysis of language neglects the more far reaching role that language plays in

cognition in general. Rather than focus on the structure of language as it exists, we instead focus our attention on the acquisition process. In particular, we seek to identify the necessary ingredients for a system to automatically acquire language in a manner similar to humans.

1.1.2 Language and Embodiment

In Alan Turing's 1950 paper [5], he presents the notion of functional equivalence as the primary paradigm for the development of an intelligent machine. Under that paradigm he presented his "imitation game," which is now commonly known as the Turing test. Since this publication, the Turing test has strongly influenced artificial intelligence research leading to a disproportionate focus on abstracting specific human behaviors. In this same publication, however, Turing suggested that this approach may not be successful and presented an alternative approach to developing an intelligent machine. Interestingly, Turing's alternative touches on the significance of language and language acquisition as fundamental to the development of an intelligent machine. The following excerpt is taken from the second to last paragraph of the 1950 paper:

> We may hope that machines will eventually compete with men in all purely intellectual fields. But which are the best ones to start with? Even this is a difficult decision. Many people think that a very abstract activity, like the playing of chess, would be best. It can also be maintained that it is best to provide the machine with the best sense organs that money can buy, and then teach it to understand and speak English. This process could follow the normal teaching of a child. Things would be pointed out and named, etc. Again I do not know what the right answer is, but I think both approaches should be tried.

Significantly, Turing's alternative suggests that a machine would need some form of embodiment (sensors and actuators) in order to learn and understand language. Since the time of Turing's writing, experimental studies have strengthened the hypothesis that language is strongly related with sensory perception. In particular, the structure of one's native language has been shown to directly influence something as simple as the perception of color.

Specifically, a color that has a definitive name in a person's native language is more readily distinguished from other similar colors [9]. This is further supported by experimental results that show that prelingual children perceive color using a different brain region than adults [10]. In addition, one's native language has also been shown to influence more abstract cognitive processes, such as predicting another person's behavior [11]. Furthermore, in children, the ability to categorize objects and abstract concepts is reinforced by learning the words for describing/naming them [8]. These results lend credence to Turing's intuition that motor function and sensory experience are fundamental and requisite for the acquisition of language.

1.1.3 Linguistic Structure in an Analog World

Linguistic structure may be thought of as a symbolic framework within which a set of symbols are constructed and related to one another. This structure is then used to represent information in a manner that can be communicated from one agent to another. From a more information theoretic perspective, language serves as a coding scheme that provides redundancies that reduce the probability of error (i.e. misinterpretation) when communicating over an unreliable channel. A central problem with this notion of language is that linguistic structure is inherently symbolic in nature while the world in which we live is, as far as we know, analog in nature. In order to learn language, we must first address the problem of mapping from the analog environment to a symbolic representation. Moreover, the high variability between human languages and the manner in which languages evolve over time suggest that such a mapping should be both adaptive and self-organizing in nature [3].

Although there has been extensive research into the problem of learning a symbolic structure within an analog environment, most existing methods require some *a priori* knowledge of the structure of the data. For example, machine learning techniques have been used to automatically characterize the probabilistic structure of a set of data in order to automatically generate meaningful insights about the mechanics by which the data were generated [12][13][14]. In addition, the problem of one-shot learning such that a machine might be able to generalize from a single experience (in much the same way people do) has also seen significant advances [15][16]. While these ad-

vances are promising, they each require some level of prior knowledge of the structure to be learned. Although it is possible that the human brain might be hardwired with some level of prior knowledge about the structure of the environment, this hardwiring is still the result of genetic adaptation that has simply been learned on an evolutionary time-scale. With this in mind, we will consider the problem of learning both the symbolic structure and the correspondence between the symbols and the environment as starting from a blank slate. Given the inherent feedback loop between categorization and language acquisition [8], it seems likely that the mathematics of control theory play a large part in this process. In particular, we will focus our attention on this problem of mapping the continuous sensory input and motor outputs onto a symbolic and linguistic framework.

1.1.4 Speech Articulation as a Starting Point

The fundamental premise behind this work is that language is intrinsically tied to the speech signal. This is not to say that languages cannot take other forms; in fact, sign language has been observed to emerge in communities of people born with hearing impairments [17]. Rather, we posit that the evolution of language occurred through utilization of the speech signal for communication. As such, the speech signal and the speech production process are integral to the study of language acquisition. Furthermore, the speech production process has a strong influence on the structure of the speech signal and consequently on the structure of natural language.

At a fundamental level, if we consider language as a coding scheme for the transmission of information, the dynamics of the vocal tract impose fundamental restrictions on the structure of this code in much the same way that bandwidth limitations impose restrictions on coding schemes used for wireless communication. In particular, the dynamics of the vocal tract limit the speed at which speech sounds may be produced and the speed at which speech can transition from one sound to another. In addition, the momentum of the articulators causes the production of one phoneme to influence the production of the following phoneme. These observations are further reinforced by the fact that production models have proven useful as a structural basis for speech analysis [18].

It is with all of this in mind that we focus our attention on the problem of automatically learning to communicate through speech produced by an anatomically faithful model of the human vocal apparatus. Fundamental to developing a means of speech communication is the task of articulator control and the task of encoding information onto the speech signal. In order to accomplish this, we will develop the notion of sensorimotor primitives as a general framework for addressing both challenges. We begin this development by first considering speech communication from an engineering perspective.

1.2 Speech Communication

In order to appropriately set up the speech communication problem, we will first consider three prominent theories that each address the question of how humans transmit information using the speech signal. Specifically, we will review the acoustic theory of speech production [19], the motor theory of speech perception [20], and the more recent exemplar theory of speech [21]. By relating these theories to the feedback communication channel framework shown in Figure 1.1, we will draw the necessary insights to develop a generalized notion of speech primitives.

Figure 1.1: A communication channel interpretation of speech production and perception.

In the system diagram of Figure 1.1, we denote the information-bearing message as s_n, the motor inputs to the vocal apparatus as u_t, the proprioceptive output of the vocal tract as x_t, and the acoustic signal as y_t. In general, we will use the subscript n to indicate a discrete sequence while the subscript t indicates a continuous sequence. In the following discussion, we will focus our attention on the implications of each theory on the nature of the encoder and decoder in this diagram. Using these insights, we will then introduce the notion of speech primitives in the context of this feedback communication system.

1.2.1 The Acoustic Theory of Speech Production

We begin with the acoustic theory of speech production [19], which posits that the acoustic speech signal is composed of segments that are isomorphic to an alphabet of symbols. Historically, phonemes have been a favorite candidate for such a dictionary of symbols. Under this framework, the transmitted message is fully represented by the frequency content of the waveform. This implies that speech production is simply a goal-oriented activity in which the speaker strives to achieve a given set of acoustic targets. As a consequence, the articulatory process used to produce an acoustic signal is effectively unimportant to the listener.

Notably, the mapping of natural speech segments onto phonemes is often ambiguous. Furthermore, mistakes commonly occur in the speech production process, which leads to missing or incorrect phones occurring in the speech signal. In order to deal with these ambiguities and errors, it is argued that the listener first infers the utterance that the speaker *intended* to produce before mapping the intended utterance onto a symbol sequence. In essence, the listener performs error correction directly on the signal using some internal model of speech prior to decoding the message.

Interpreting it in terms of a system diagram, we consider the structure of the encoder and decoder that this acoustic theory implies. We have that the encoder maps the message onto a sequence of speech units, $\tilde{y}_n$, that is then produced by the vocal apparatus. The message is then decoded in two stages: first by estimating the intended speech sequence $\hat{y_n}$, followed by decoding the message $\hat{s}_n$ alphabetically from $\hat{y_n}$. A system diagram of the transmitter structure is represented in Figure 1.2.

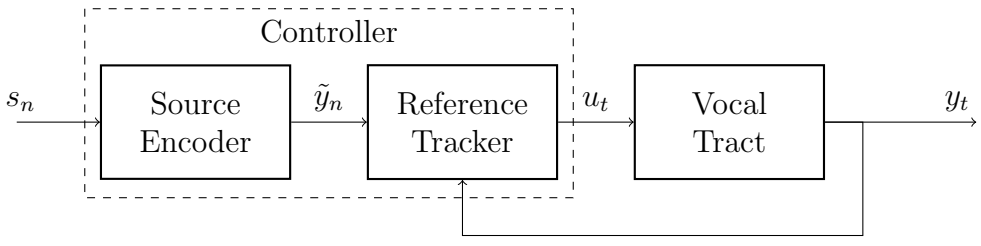


Figure 1.2: A communication system interpretation of the acoustic theory of speech production.

Given that this theory is built on the hypothesis that the speech signal is alphabetic, then the dynamics of how it was produced are unimportant. In

essence, we need only to model the signal itself in order to perform recognition. This ultimately led to the well known source-filter model of the speech signal shown in Figure 1.3, wherein the speech production process is abstracted away in terms of a time varying linear signal model. By casting the speech signal in terms of a signal source and a linear time varying filter, the computational and mathematical challenges of modeling the dynamics of the vocal tract are bypassed. Historically, much of the research in speech recognition and speech synthesis has implicitly and explicitly made use of the source filter model [3][22][23].

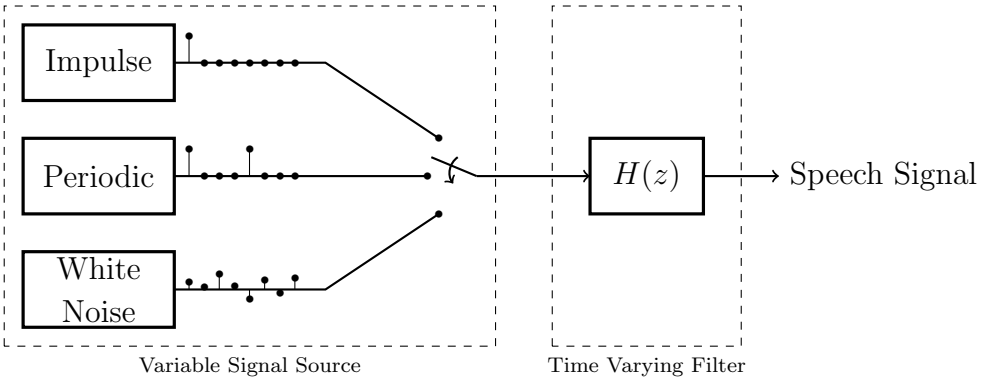


Figure 1.3: The source filter model of speech.

Although the source filter model of speech enables the use of powerful mathematical tools for speech processing, such as linear prediction, it does not naturally address the higher order structure of the speech signal. As a result, it fails to fully capture phenomena that exist in natural speech, such as coarticulation and phoneme substitution. Some of these phenomena can be dealt with by using multiphonic utterances (e.g. diphones, triphones, words, phrases, etc.) as the building blocks of the speech signal rather than single phones or phonemes. This approach can be used to address the context sensitivity of phonemes and it enables the traditional mathematics of speech signal processing such as the hidden Markov model [3]. However, it also combinatorially increases the amount of training data necessary for both recognition and synthesis. Although this combinatoric explosion can typically be dealt with using modern computing technology, it does not seem likely that the human brain has access to such levels of computing power. In short, a comprehensive theory of speech communication should solve these ambiguities using the constraints of the speech production process.

1.2.2 The Motor Theory of Speech Perception

Converse to the acoustic theory of speech production is the motor theory of speech perception. As previously noted, mapping the speech signal onto a sequence of phonemes is often ambiguous. Nevertheless, humans are capable of robustly performing speech recognition and phoneme identification despite this ambiguity. In order to address this, the motor theory of speech perception proposes that humans utilize the acoustic signal in order to infer a set of invariant neuromotor commands [20][24]. These neuromotor commands would then map isomorphically onto an alphabet of articulatory phonemes that represent the symbol sequence of the transmitted message. Under this paradigm, speech is then perceived by inferring the articulatory gestures of a speaker based on the observed acoustic signal.

In order to represent the motor theory of speech perception, it is best to interpret the encoder as a control system. Specifically, we represent speech production in terms of a reference tracking control system where the reference signal corresponds to articulator positions or some abstraction of them. Figure 1.4 is a system diagram of this interpretation of the motor theoretic notion of speech communication. Here the message is encoded as a sequence of reference targets $\tilde{x}_n$ that the articulators are driven toward using a reference tracking control. Speech is then recognized by estimating this target sequence using an acoustic observer that mirrors this controller architecture. Under this paradigm, $\tilde{x}_n$ represents the set of desired articulatory gestures that are alphabetic in that each gesture maps to a symbol that is to be transmitted. Speech communication is then reduced to finding a means of encoding the message s_n as a set of target gestures on $\tilde{x}_n$ which can be estimated by the acoustic observer. It should be noted that this theory requires that the articulatory gestures be fully observable (in the control theoretic sense) based on the acoustic signal y_t. In addition, this theory suggests that an internal model of the speech production process is essential to the robustness of human speech perception.

Although the motor theory of speech perception presents a potentially elegant explanation of the robustness of human speech perception, the implementation of a speech recognition system based on this concept has encountered significant hurdles. The foremost obstacle to directly applying such an approach comes from the fact that acoustic to articulatory inver-

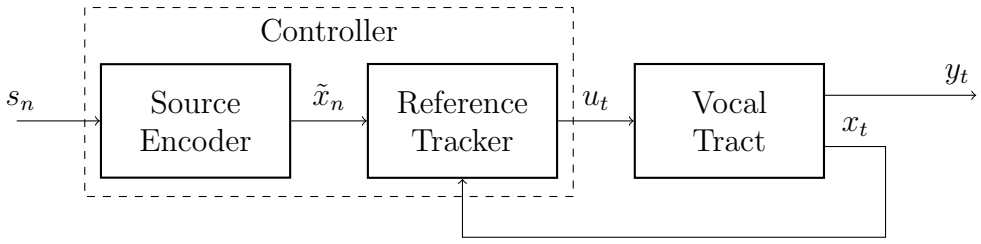


Figure 1.4: Speech production model based on motor theory of speech perception.

sion is a poorly constrained optimization problem. Specifically, there exist many possible solutions to finding a vocal tract area function that produces a given speech sound. This ambiguity can be addressed by imposing additional constraints on the problem, such as minimizing the energy expended or minimizing the total movement of the articulators, though it is unclear what set of constraints correspond to those implemented within the brain [25]. Despite such challenges, gesture-based articulatory models have been demonstrated to account for phenomena such as coarticulation and phoneme substitution in isolated word recognition [26]. Additionally, the use of articulatory models has been shown to yield improvements in speech recognition wherein both audio and articulator data are available during classification and/or during training [27]. Furthermore, the use of an underlying articulatory model has been shown to naturally account for the inherent asynchrony between audio and video features when performing audio visual speech recognition [28].

Recently, studies of speech production have shown that there exists ambiguity in speech articulation as it maps to the acoustic waveform, most notably in the English /r/ [29]. This result suggests that human speech perception is not performed solely by inferring a set of neuromotor commands; rather, the acoustic targets themselves play a fundamental role [30]. Although such a result suggests that a purely motor theoretic model of speech perception is unlikely, the use of articulatory models in automatic speech recognition has been demonstrated to improve recognition performance. Interestingly, an experiment in speech perception in infants with and without a bite-block suggests that motor function does play role in human speech perception well before the subjects had learned to speak [31]. Furthermore, a recent survey of functional magnetic resonance imaging (fMRI) and transcranial magnetic stimulation (TMS) studies of speech perception indicates that the motor cor-

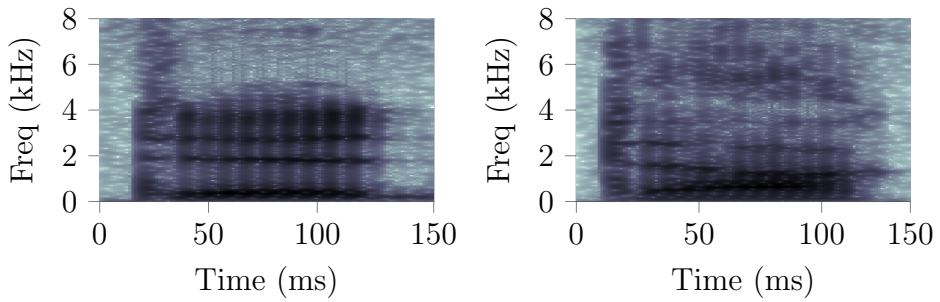
tex is relevant to speech perception and comprehension [32]. Although the motor theory is unlikely to fully explain human speech perception, these lines of evidence suggest that motor control does play an important role.

1.2.3 Exemplar Theory of Speech

Both the motor and acoustic theories of speech fundamentally assume that speech is composed of some set of information-bearing units that are used to construct words and larger utterances; however, this assumption is based on little more than human intuition. As is a common theme in artificial intelligence research, such intuitions of the internal functions of the human brain are often misleading. Exemplar theory essentially rejects intuition that speech is composed of alphabetic units and instead proposes a more empirically motivated theory of how speech is stored in the brain for the purpose of speech recognition and communication. In this section, we will briefly outline the evidence for the rejection of phonemes in favor of the idea that speech is recognized using exemplar utterances that are stored in memory [21].

As stated in Section 1.2.2, phonemes generally lack a clean definition in terms of spectral characteristics. As such, determining phoneme boundaries or phoneme categorization based on the acoustic spectrum is often poorly defined. This problem is especially prominent when one examines phonemes that are co-articulated. One such example is the /d/ phoneme when pronounced in the diphones /di/ and /du/. Perceptually, these two utterances begin with the same phoneme; however, their spectral features shown in Figure 1.5 are quite different. Much like the study of the English /r/ phoneme illustrates a shortcoming in the motor theory of speech perception, this spectral variation in the /d/ phoneme seems to illustrate a similar shortfall in the acoustic theory of speech production. One possible, and arguably likely, reason for this problem is that the assumption of the existence of phonemes is fundamentally incorrect.

If phonemes are an incorrect characterization of speech, why has speech research been so strongly influenced by intuition that speech is segmental? In short, this intuition likely stems from the ubiquitous training in reading and writing an alphabetic orthography, particularly in western cultures and especially among academics. While training in an alphabetic orthography



(a) /di/ (b) /du/

Figure 1.5: A comparison of the spectrograms of [di] and [du], illustrating the high degree of variance in the definition of the [d] phoneme.

is a powerful tool for logical processing (such as arithmetic), it also biases our intuition about how our cognitive processes actually work. In essence, humans generally learn to speak long before we learn an alphabetic written language, so the assumption that speech is alphabetic requires more justification than a strong intuitive feeling [21]. To further bolster the case against alphabetic speech, it has been shown that training in an alphabetic orthography directly influences ones sense of the segmental nature of speech. This was shown by demonstrating that Chinese individuals who were literate in a non-alphabetic orthography were unable to perform tasks that demonstrated basic phoneme awareness that is ubiquitous in individuals who were trained in an alphabetic orthography [33].

The alternative to the phonetic theory of speech as argued by Port [21] is that speech is stored in the brain as a trajectory through some high dimensional feature space. This rich representation of speech is then used for speech recognition by performing something akin to nearest neighbor classification. In effect, he argues that we store utterances in the brain to form a distribution in this space of trajectories and perform recognition based on where a new utterance falls in this distribution. The support for such an idea largely comes from observations showing that human listeners perform better at recognition tasks when hearing the utterance spoken by the same speaker each time [34]. In essence, the listener is using features that are specific to that speaker when performing recognition, even though those features have no bearing on the meaning conveyed by the utterance.

If speech is not represented as small alphabetic units in the brain, then

we must consider some alternative means by which it is represented. The alternative posited by exemplar theory is that speech is presented as a trajectory through some feature space and speech recognition is performed by matching an utterance up to some exemplar trajectory or distribution of exemplar trajectories. Although the case against the existence of phonemes is generally compelling, I argue that it is necessary to consider speech production and perception in terms of a communication system. As such, the notion of speech representation suggested in exemplar theory only indirectly addresses how information is encoded in the speech signal. This fundamental issue touched upon by Port [21] as an open question that is not yet explained through exemplar theory:

> After all, if memory is very detailed and rich, why would languages need to have phonologies? Why do they all appear to build words from a large, but still limited, number of component fragments (features, segments, onsets, codas, etc.)...

Claude Shannon's theory of communication necessitates that information is fundamentally represented as discrete units [35]. As a result, a communication signal must, at some level, be representable as a finite set of discrete elements for communication to occur. This would necessitate that the speech signal be composed of some set of information theoretic units. This notion is reminiscent of distinctive feature theory that emerged from the Prague school of linguistics around the same time of Shannon's original publication. In its most mature form, the theory effectively argues that the speech signal is broken down into a set of binary features that differentiate between speech sounds [36][37][38][39]. It is important to note that the discretization of speech sounds need not be universal; rather, it only need be consistent between transmitter and receiver. In the case of two people communicating through speech, all that matters is that the speaker and listener agree on this discretization in order for a message to be reliably transmitted between them. The observations made in exemplar theory suggest somewhat strongly that any discretization of the speech signal is effectively localized between agents that regularly interact.

1.3 Speech Primitives as a Unifying Framework

Each of the prominent theories of speech communication carry important insights that can be used to develop a more universal framework. Ultimately, all of these theories are built upon assumptions about how and where information is encoded in the speech signal, whether it be acoustic or articulatory. In this work, we strive to construct a model of speech that is free from such assumptions. Specifically, we seek to model the feature space in which information is encoded in the speech signal. Furthermore, we seek a method of automatically identifying this feature space that does not incorporate the biases of human intuition.

In order to discover such a feature space, we consider the implications of both the motor and acoustic theories of speech. Specifically, we will explore the notion of sensorimotor primitives as a set of operators that map between some space of primitive features and the speech signal in a manner that fundamentally incorporates both the mechanics of speech production and acoustic features of the signal itself. We can do this by representing speech communication in terms of the system diagram shown in Figure 1.6, where the information-bearing message is encoded on the primitive features denoted by f_t. Our goal is then to develop a model of the primitive controller and observer that allows us to encode information onto the speech signal, where both articulatory and acoustic feedback play a central role. It is expected that such a development will naturally lead to a representation of speech that will directly address the question of how humans encode information within the signal.

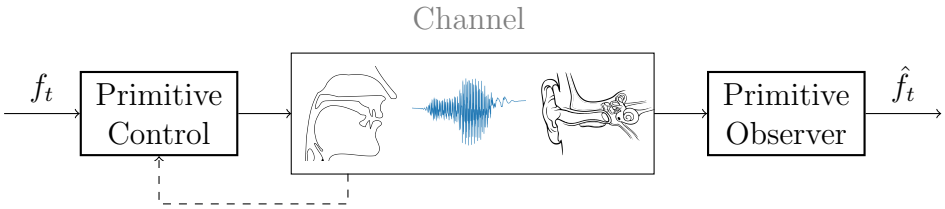


Figure 1.6: A system-level diagram showing how speech primitives fit into the speech communication channel.

1.4 Contributions of this Thesis

In this thesis, we develop the concept of speech primitives in terms of a model of speech communication via control of the human vocal apparatus. Importantly, we utilize the concept of sensorimotor primitives in developing this model of speech and restrict the development of speech primitives to unsupervised learning methods. As such, we will construct two models of primitives based on linear feedback control and the decomposition of nonlinear operators. We will then show their representational power as both an analytic and generative representation of speech. Finally, we will consider the application of this framework to address the general question of encoding information onto the speech signal and postulate future directions for this research.

The remainder of this thesis is organized as follows. Chapter 2 lays out the theoretical framework by which we will construct the notion of sensorimotor primitives. Chapter 3 provides a review of relevant models of speech synthesis and their relation to this work. Chapter 4 is an overview of the Praat model of the vocal apparatus that is used for the experiments in this thesis. Chapter 5 applies the notion of sensorimotor primitives to linear feedback control, and develops an unsupervised method by which they can be learned. Chapter 6 details experimental results in applying linear speech primitives to the Praat vocal tract model. Chapter 7 develops the application of sensorimotor primitives to nonlinear communication channels through the decomposition of nonlinear operators. Chapter 8 details a final set of experimental results in applying nonlinear speech primitives to the Praat model of the vocal tract. Finally in Chapter 9, the implications of this work are discussed in relation to speech communication and language acquisition in general.

CHAPTER 2

PRIMITIVES: A FUNCTIONAL DEFINITION

In this chapter we provide a more in depth treatment of the notion of primitives as it pertains to this thesis. In the spirit of Norbert Wiener's *Cybernetics* [40], we will approach the process of speech communication in a holistic manner by considering the speech signal in the context of control and communication. Specifically, we will draw upon insights that have been developed in the context of control theory and robotics and extend those ideas in order to apply them within the domain of articulatory speech production. We begin by defining sensorimotor primitives as a means of simplifying the interface to a complicated system that may be high dimensional or nonlinear. We then apply this framework to both the domains of linear feedback control systems and nonlinear communication channels and relate these system implementations of sensorimotor primitives to articulatory control and acoustic communication respectively.

2.1 On the Concept of Primitives

In the context of robotics and control, the notion of movement primitives has many working definitions, which tend to differ depending on the application around which they were developed [41]. Central to most of these definitions is some notion of breaking down the space of control actions into components such that any general action can be characterized as a composition of these elements. This decomposition may be applied over the space of control inputs to achieve some form of coordinative control, or it may be applied over the time domain in order to find a set of sequential gestures. Implicit in these notions is the goal of simplifying a complicated control problem. We take this implicit goal to be the guiding principle in our development of sensorimotor primitives; however, we will consider a more generalized approach to defining

primitives as a means of problem simplification.

Among the various definitions of primitives, the manner by which they are learned varies rather greatly depending on the focus of the research. In some cases, primitives are assumed to be known *a priori* and are essentially designed or selected by a human designer [42][16]. In other cases, they are learned through guided training, such as a teacher guiding the learner through the swing of a tennis racket [43][44]. In still others, primitives may be constructed through fully unsupervised means such as Todorov's development of sensorimotor primitives [45][46][47]. In the case of human speech, we note that humans acquire speech in a manner that is largely unguided, i.e. much of the learning is conducted through unguided exploration. In addition, we noted in Chapter 1 that we seek a model of speech that is as free from the bias of human intuition as possible. As such, we will constrain our development of speech primitives to methods that are fundamentally unsupervised.

2.1.1 A Holistic Perspective

We begin our development of sensorimotor primitives by first modeling how an agent interacts with its environment. In general, an agent interfaces with the environment by receiving sensory inputs that are dependent on the state of the environment and producing actuator outputs that influence the state of the environment. This relationship is graphically shown in Figure 2.1, where we emphasize that the generation of the speech signal is fundamentally a means of influencing the state of the environment in a manner that other agents can detect and interpret. Fundamental to this formulation is the idea that the agent has an internal state that is influenced by the sensory inputs it receives. As we proceed with developing speech primitives, we will build them up from this fundamental notion of mapping from sensory input to motor control through some internal state space.

Relating back to Figure 1.6, the primitive features are effectively a representation of the internal state space, the *primitive controller* maps from internal state to motor control, and the *primitive observer* maps from sensory input to internal state. As we proceed to develop a means of learning these primitive mappings, it is important to emphasize that all three of these

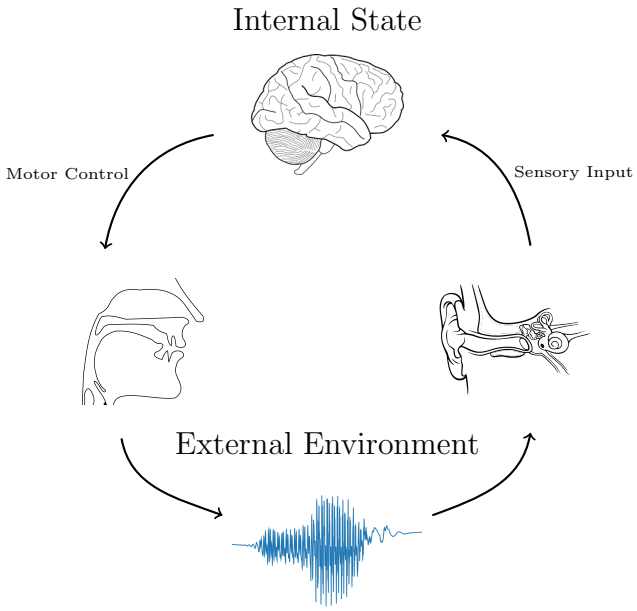


Figure 2.1: A diagram of an agent interacting with the environment through sensor inputs and motor actuation.

components are fundamentally tied together. In particular, selection of primitive feature space is influenced by the operators that map in and out of it. If we were to consider only the motor control (as in the motor theory of speech perception) or only the sensory input (as in the acoustic theory of speech production), the resulting feature space would not fully represent the agent's interaction with the environment. As we develop our notion of speech primitives, we seek a sensorimotor representation such that the sensory input and motor output are fundamentally built in.

2.1.2 Primitives in Speech

It should be noted that movement primitives have been studied in the context of speech production. The first instance of primitives being directly used for speech production can be found in the Task Dynamic model which will be detailed in Chapter 3; however, the notion of primitives in this case is in essence fully defined by human designers. As such, the development of primitives that are used in the Task Dynamic model does not bear much relevance to our goal of developing primitives in an unsupervised manner.

More significantly, Ramanarayanan, et al. developed a data driven approach to the notion of movement primitives of the vocal tract [48][49]. In their formulation, convolutional nonnegative matrix factorization was applied under sparseness constraints (cNMFsc) in order to conduct a form of segmentation of speech production data. Interestingly, this approach was applied to audio visual data consisting of real-time MRI video and recorded audio as well as data collected from articulatory synthesis software. The resulting set of movement primitives was then shown to correspond to linguistically meaningful gestures that represented a means of segmenting the audiovisual speech data into something analogous to phonetic units. While these results may be interesting, the use of cNMFsc imposes a great deal of a priori restrictions on the nature of the movement primitives that are derived from the data. In particular, this approach requires that the duration of the primitives and number of primitives also be chosen *a priori*. Lastly, the decomposition method of cNMFsc does not appear to have any principled justification other than the observation that nonnegative matrix factorization typically decomposes data into perceptually relevant components. In this work, however, we seek to develop a notion of primitives from the first principles of control and communication theory.

2.2 A Control Theoretic Perspective

We begin our development of speech primitives by drawing insight from the notion of *sensorimotor primitives* as developed by Todorov [45][46][47]. In his development, *sensorimotor primitives* are defined in terms of a feedback transformation and the control policy implemented in that space. We define the dynamical system with internal state z_t, control input u_t and observable output y_t. The dynamics of the system are then defined in Equation 2.1 where the function f defines the system dynamics and the function g defines the sensory output mapping.

$$\begin{aligned} z_{t+1} &= f(z_t, u_t) \\ y_t &= g(z_t, u_t) \end{aligned} \tag{2.1}$$

In the case of the vocal tract, the overall dimensionality of z_t is expected to be large and cannot be directly observed. In order to interface with this

system, we seek a feedback transformation that utilizes the system dynamics in order to reach some useful representation. We first consider the state estimator $\hat{z}_t(y_{t-p}, \ldots, y_t, u_{t-p}, \ldots, u_t)$, or simply $\hat{z}_t$, which is a function of the past inputs and outputs of the system. We then define the following feedback transformation, where h_t is the observable state and v_t is the control input of a new dynamic system shown in Equation 2.2, that is coupled to the original system from Equation 2.1.

$$h_{t+1} = \tilde{f}\,(h_t, v_t) \tag{2.2}$$

$$\begin{aligned} h_t &= \mathcal{T}(\hat{z}_t) \\ u_t &= \tilde{\mathcal{G}}(\hat{z}_t, v_t) \quad = \quad \mathcal{G}(h_t, v_t) \end{aligned} \tag{2.3}$$

Sensorimotor primitives are then defined as the feedback transformation shown in Equation 2.3, where typically the transformation performed by the operators $\mathcal{T}$ and $\mathcal{G}$ is chosen such that the task of controlling the system is simplified in some manner (e.g. linearization and/or dimension reduction). Given such a feedback transformation, we can then define $\mathbf{h}$ to be the *primitive feature space*, where the *primitive controller* represented by $\mathcal{G}$ and the primitive observer is represented by $\mathcal{T}$. We can now formulate communication by selecting control inputs v_t that drive toward some target state $\tilde{h}$ that can be observed by an external agent. This formulation is graphically represented in Figure 2.2 where the high level controller is labeled as an *encoder* in order to emphasize that this formulation is fundamentally one of communication.

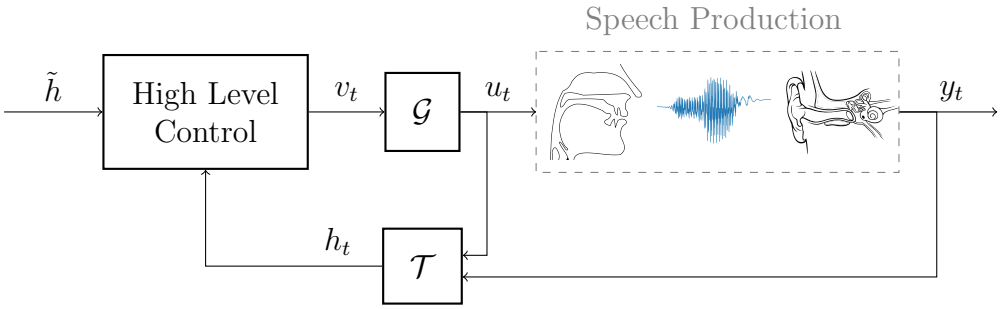


Figure 2.2: A schematic representation of speech production based on sensorimotor primitives as an interface to the vocal tract.

This concept of feedback transformation as primitives is particularly pow-

erful in that it lends itself to being modularized such that disjoint sensory channels can be incorporated through a cascade of feedback transforms. This is particularly important in the context of vocal tract control, given that proprioceptive feedback and acoustic feedback both play an important role. In Chapter 5, we will develop a linear implementation of these sensorimotor primitives and demonstrate their efficacy in controlling the vocal tract geometry. Conversely, employing linear sensorimotor primitives to control the acoustic signal is, as will be shown in Chapter 6, doomed to failure due to the highly nonlinear relationship between motor inputs and acoustic features. In order to complete our mapping from the speech signal through some primitive feature space, we must employ a nonlinear approach. Furthermore, the feedback control framework assumes knowledge of the control inputs in order to estimate the corresponding internal state. Given that this assumption is never true in the context of speech perception, we must take a step back and recast our problem from the perspective of communication theory.

2.3 A Communication Theoretic Perspective

When considering the individual agent controlling the vocal tract, it is reasonable to assume that the agent has knowledge of both the sensory input and motor outputs. In the case of two agents interacting through the environment, one speaking and one listening, the listener only has knowledge of the speaker's effect on its sensory inputs. As such, we must find a means of learning the mapping directly from sensory input onto the internal primitive features. Relating back to Figure 2.1, we can learn this mapping through an agent's own interaction with the environment. Specifically, we consider the communication channel laid out in Figure 2.3 where the primitive controller and primitive observer can be thought of as learning an encoder and decoder for a nonlinear communication channel.

By considering the speech communication channel in this feed-forward manner, we can employ nonlinear methods of estimating the encoder $\mathcal{E}$ and decoder $\mathcal{D}$ in tandem. In particular, Chapter 7 will detail an adaptation of a general form of neural network architecture that was developed as a means of accomplishing this goal.

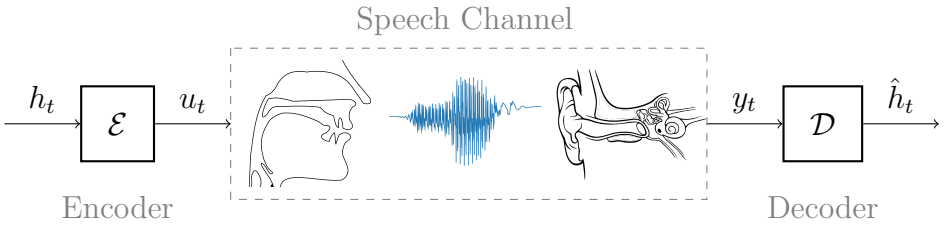


Figure 2.3: Notional depiction of the speech communication channel that maps from motor control commands to acoustic features.

2.4 Conclusion

In this chapter, we have developed a general framework of sensorimotor primitives as a means of interfacing between an agent and the environment. We then applied this framework to both feedback control and communication as a means of defining specific notions of sensorimotor primitives that can be used to model speech communication. In both cases, the common thread is the notion of learning a transform that maps from an agent's internal state to the state of the external environment through sensory input to actuator control. In this thesis, we will consider the problem of vocal tract control with proprioceptive feedback under the feedback transformation construction. In addition, we will apply the channel coding framework laid out in Section 2.3 to the problem of communication through the acoustic speech signal.

Our development of speech primitives is fundamentally built around characterizing the dynamics of an agent's integration with the environment. As such, it is important that our primitives be learned using a model of the vocal tract that is as anatomically faithful as possible. In the following two chapters, we will review several significant articulatory models of speech production and the articulatory model developed by Boersma, which is used extensively in this thesis.

CHAPTER 3

SIMULATING THE VOCAL TRACT

The central goal of this research is to develop a model of communication via control of the vocal tract. As a first step toward this goal, this chapter details the articulatory synthesizer used to implement this speech communication channel and places it in the context of broader articulatory synthesis research. Given the rich history of articulatory speech synthesis, focus is given to existing models of speech production that share a control architecture that is relevant to our development of speech primitives, namely, the Task Dynamic model [42] and the DIVA model (Directions Into Velocities of Articulators) [50]. This chapter begins with a review of the anatomy of the human vocal apparatus, giving special focus to the key anatomical structures used in speech production. With this basic understanding of the anatomy of speech production, the Task-Dynamic model and DIVA are reviewed. The review of each of these models includes a special focus on their models of vocal tract articulation and their relation to historically prominent articulatory models, specifically the three models developed by Coker and Fujimura [51], Mermelstein [52], and Maeda [53].

3.1 Anatomy of the Vocal Apparatus

Given that many simulations of human speech production only model key portions of the corresponding anatomy, we begin by drawing a distinction between the vocal tract and vocal apparatus. Throughout this chapter, we will refer to the *vocal tract* as the set of articulators and tubes that extend from just above the glottis to lips and nostrils; the *vocal apparatus* is the larger system of all articulators and tubes that are involved in speech production and extends from lungs to lips and nose. In essence, the vocal apparatus encompasses the entire system of anatomical structures involved in speech

production while the vocal tract encompasses only a subset of that system. The anatomy of speech production can be broken down into three major sections of the vocal apparatus: the lungs, the larynx, and the vocal tract. Figure 3.1 shows a cross-sectional view of the vocal apparatus labeled with the key actuators for reference.

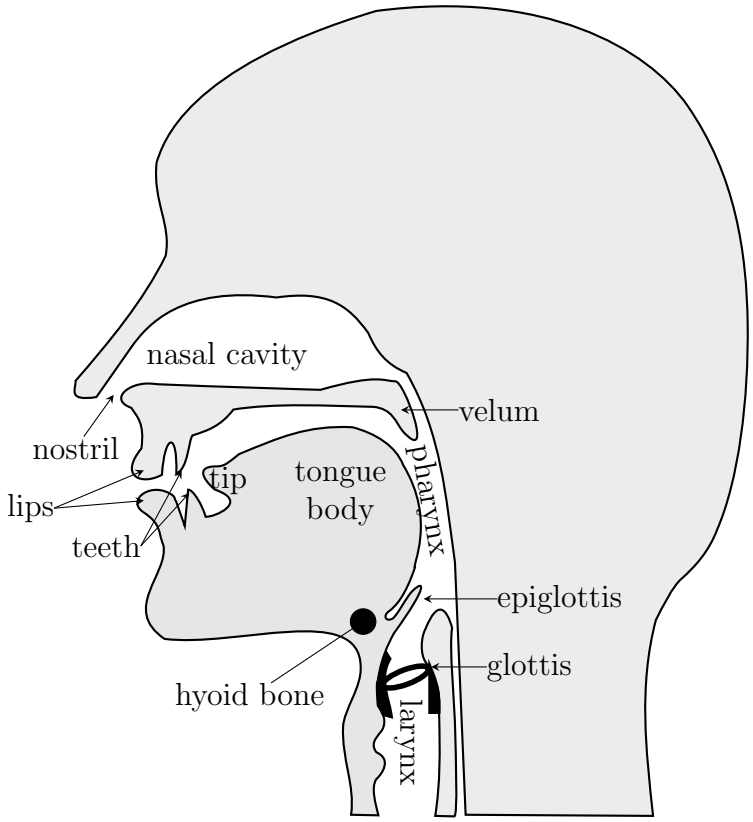


Figure 3.1: Midsagittal view of the vocal tract anatomy.

At a high level, the speech signal is produced by the vocal apparatus by forcing air out through the lungs, generating vibrations (typically in the larynx), and shaping the spectrum of those vibrations using the vocal tract. The source-filter model of speech is derived from this segmentation of the vocal apparatus by assuming that all of these three components are independent of one another and abstracting them as a power source (lungs), a signal generator (larynx), and a time varying filter (vocal tract) as depicted in Figure 3.2. Although the source filter model is a relatively rough approximation of the speech production process, it serves as a strong mathematical simplification that has been useful in many speech signal processing applications such as speech coding and recognition [19][22]. Additionally, the source-filter

model is a useful abstraction when discussing the process of speech production because it clearly highlights the primary function of each component of the vocal apparatus.

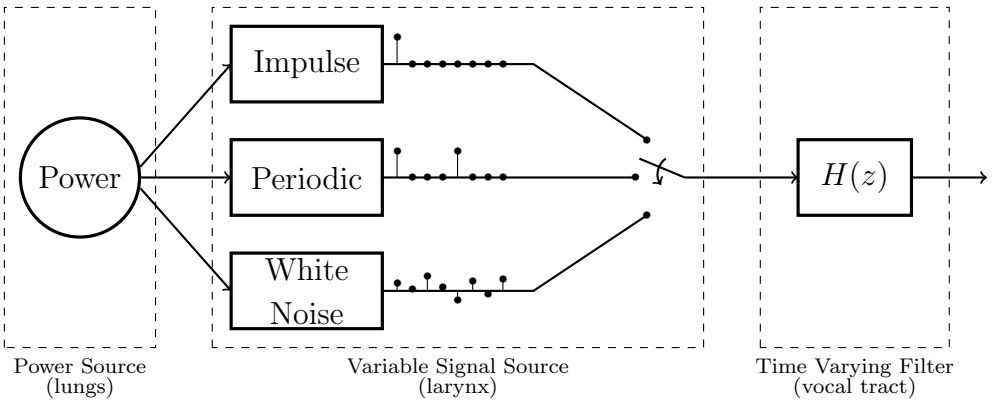


Figure 3.2: The source filter model of speech.

3.1.1 The Power Source: Lungs

The primary function of the lungs is to metabolize oxygen into the bloodstream through breathing. Breathing is performed by actuating the diaphragm, ribcage, and abdominal muscles to change the volume of the chest cavity and drive the pressure within the lungs above or below atmospheric pressure. When inhaling, the chest cavity expands and the pressure in the lungs drops below atmospheric pressure, causing air to rush into the lungs so long as a path exists. When exhaling, the chest cavity contracts such that the pressure in the lungs rises just above atmospheric pressure, forcing air out of the lungs. It is during exhaling that speech is typically produced.

During speech, the lungs primarily function as a power supply where the energy of the speech signal is derived from the pressure differential from the interior of the lungs to the outside of the lips and nostrils. When a person speaks, air is expelled from the lungs at near constant pressure at just above atmospheric pressure, and the muscles of the ribcage and diaphragm are controlled to maintain this near constant pressure throughout the duration of the utterance [22]. The pressure generated by the lungs corresponds to the energy available to the speech signal and, as such, the lungs represent a primary means of volume control for the speech signal.

Given that the lungs serve a relatively simple function in speech production and they have little interaction with the rest of the vocal apparatus, the lungs are often simplified as a constant pressure source in most articulatory synthesizers. This simplification allows for computational resources to be focused on the articulations of the vocal tract and aerodynamics of the vocal folds, both of which have a more direct relationship with the spectrum of the signal. In contrast, the energy supplied by the lungs plays a significant role in the volume and prosody of speech. Although these aspects of speech production are typically considered unimportant in speech recognition, they are still information-bearing features of the speech signal. As such, a more faithful simulation of the lungs is an important element for our model of speech primitives for communication.

3.1.2 The Signal Source: Larynx

The larynx is a system of muscles, cartilage, and ligaments that are used to control the vocal folds which are illustrated in Figure 3.3. The vocal folds consist of two masses that stretch across the larynx from front to back. The slit-like opening between the vocal folds, called the glottis, is used to modulate a pressure wave onto the air passing through it based on the size of the opening and tension in the vocal folds. The glottal opening and vocal fold tension are controlled by the muscles attached to the thyroid and arytenoid cartilages, as well as the hyoid bone which is shown in Figure 3.1. In the context of speech production, the larynx has three key modes of operation: voiced speech, unvoiced speech, and ejectives.

In voiced speech, the vocal folds are to used to generate a semi-periodic pressure wave in the air passing through the larynx, which is generally referred to as the *glottal wave*. By adjusting the spacing and tension of the vocal folds, they can be made to oscillate as air passes through the glottis. This oscillation of the vocal folds is due to Bernoulli's principle [22], which roughly states that as the velocity of airflow increases, the local pressure decreases. When air passes through a constriction of the glottis, the pressure within the glottis drops causing the vocal folds to be drawn together until the glottis closes. With enough tension in the vocal folds and pressure building behind the closed glottis, the vocal folds will spring open and release a puff

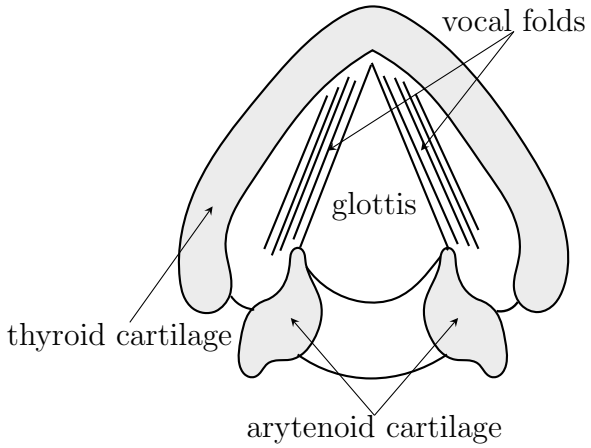


Figure 3.3: Diagram of the vocal folds as viewed from the top down.

of air. This cycle is then repeated such that the output of the glottis consists of a periodic train of puffs of air. The period of this oscillation is the *pitch period* or *fundamental frequency* of the speech signal. The glottal wave produced by this process provides the excitation that is used to produce vowel sounds and some voiced consonants.

In unvoiced speech, the vocal folds do not oscillate but they are less open and more tense than during breathing. In this state, the air flow within and just after the glottis becomes turbulent and produces high frequency noise. Turbulent flow through the glottis is known as *aspiration* and is used in whispered speech or certain phonemes such as "h" as in "happy." Aspiration can also be performed during voicing (the folds vibrate and produce turbulent flow) in order to create a 'breathy' sounding voice.

Lastly, the glottis can be actuated in order to produce a single puff of air, rather than a periodic sequence of puffs, which is used in the production of ejective speech sounds. To do this, the glottis is fully closed allowing pressure from the lungs to build up before the glottis is opened, releasing a single puff of air. In this case, the glottis is specifically forced closed and open again by the muscles of the larynx, not the Bernoulli effect.

In terms of the source-filter model of speech, the three production modes of the larynx are abstracted as signal sources at the input of the time varying filter. Voiced speech is represented by a periodic impulse train, unvoiced speech is represented by Gaussian white noise, and ejective speech is represented by a single impulse. It should be noted that the vocal folds can be made to move in other interesting ways that do not fit neatly into the categories of voiced,

unvoiced, and ejective. In these cases, the vocal folds produce prominent harmonic vibrations that elicit artifacts in the glottal waveform. Incorporating these nuanced behaviors into the source-filter representation of speech would require the inclusion of a complicated set of signal sources. This increased complexity would quickly diminish the inherent value of the source filter model of speech, namely its simplicity. Alternatively, a model of speech synthesis that directly simulates aerodynamic and myoelastic behavior of the air and tissue within the larynx can more elegantly represent a wide range of pressure waves produced in the larynx. With advances in modern computing, the computational cost of a direct simulation of the larynx no longer outweighs the benefits, especially in the context of unsupervised learning of speech primitives for communication.

3.1.3 The Vocal Tract

The vocal tract consists of the oral cavity from the larynx to the lips and nasal cavity from velum to nostrils. The nasal cavity is coupled to the oral cavity by opening the velum. The length of the vocal tract, from glottis to lips, is approximately 17cm on average for adult males and approximately 14cm for females. The shape of the vocal tract varies based on the movement of the jaw, tongue, lips, and other actuating muscles. Actuation of the vocal tract serves two primary purposes in speech production: spectral shaping of the speech signal, and sound generation through turbulent flow and ejectives.

The vocal tract shape, typically represented by the cross sectional *area function* measured along the length of the vocal tract, determines the acoustic resonance frequencies of the vocal tract. These resonance frequencies are known as *formants* and are widely considered to be critical information-bearing features in the speech signal, particularly in western languages. The term formant is typically used to refer to the spectral contribution of one such resonant frequency and is represented in terms of both center frequency and bandwidth. When the vocal tract is excited with a glottal waveform, the resultant speech waveform will have spectral peaks corresponding roughly to the formant frequencies of the vocal tract. The spectral distribution of the speech signal is naturally abstracted as a linear filter where the pole placement determines the formants. The actuators that play a critical role

in control of the vocal tract area function, and consequently the formants, are indicated in Figure 3.1. Notably, the tongue plays a dominant role in controlling the area function of the vocal tract because it extends along such a large portion of the vocal tract and is very dexterous. As a result, accurately modeling the behavior of the tongue is critical in articulatory synthesis.

The nasal cavity is largely immobile and as such can be considered to have a constant area function. The only actuator that directly affects the nasal cavity and its interactions with the speech signal is the velum, which opens and closes a pathway to the oral cavity. When the velum is open, the airflow through the pharynx and oral cavity is allowed to flow through the nasal cavity, thus altering the resonance behavior of the vocal tract. If flow through the mouth is blocked by the lips or tongue, sound is propagated through the nasal cavity to produce nasal sounds. Since the nasal cavity has a large volume, the acoustic wave produced through nasal sounds is strongly dominated by lower formant frequencies. If the velum is open during other speech sounds, those sounds are *nasalized* in that airflow through the nasal passage reduces pressure elsewhere in the tract and the resonances of the nasal cavity dissipate energy to form an *anti-resonance* - much like a zero in a linear filter.

Additionally, the vocal tract can be used as a sound source through either turbulent flow or an impulsive release of pressure. If a constriction of the vocal tract is juxtaposed to a relatively unconstricted segment, the change in air flow velocity will result in turbulent flow and thus generate high frequency noise. Alternatively, a constriction in the vocal tract momentarily closes off air flow such that pressure can be built up and quickly released, generating an ejective sound. In both cases, the shape of the vocal tract still affects the spectral shape of the noise and the glottis may or may not generate noise simultaneously.

As with lungs and larynx, abstracting the vocal tract as a time varying filter sacrifices nuance in order to reduce the computational cost of the model. While this trade-off is acceptable for many applications, the fidelity of the speech production model is of utmost importance in our exploration of speech primitives. As such, a direct simulation of the vocal tract is better suited to this work. Given the large number of actuators that drive the area function of the vocal tract, such a model must necessarily deal with the problem of coordinating various actuators to achieve a specific goal. The two prominent

models of speech production that achieve this are known as the Task Dynamic model and the DIVA model. In the following sections, these models will be examined with special focus given to the control of the vocal tract geometry and techniques used to synthesize the resultant acoustic waveform.

3.2 The Task Dynamic Model

The Task Dynamic model is an articulatory speech synthesis model that incorporates a representation of the control hierarchy of human speech production [42]. In addition to modeling a set of articulators and their effect on the vocal tract area function, this model incorporates a set of "coordinative structures" into the articulator control. These coordinative structures are represented as movement primitives which drive multiple articulator variables to produce a desired vocal gesture. A set of these gestural primitives can then be used to drive a model of the vocal tract to produce a desired utterance.

3.2.1 The CASY Model

The vocal tract model used in the Task Dynamic model was originally the articulatory synthesizer (ASY) which was further developed into the configurable articulatory synthesizer (CASY) [54]. The geometry used in ASY and CASY is largely based on the models developed by Coker and Fujimura [51] and Mermelstein [52]. In particular, CASY effectively incorporates the useful elements of both the Coker and Mermelstein models as a means of striking a balance between the benefits and drawbacks to each approach. In order to better understand this trade space and its role in the design of CASY, we will begin by briefly outlining and contrasting the Coker and Mermelstein models.

The model developed by Coker and Fujimura, commonly referred to as the Coker Model, is among the first computational models of the human vocal tract in which the area function is used to synthesize speech. The value of the area function is based on a geometric representation of the vocal tract articulators. The relationship between area function and articulators was derived from existing X-ray video of a sagittal view of the vocal tract

during speech production. Since this representation is only two dimensional, the area function is determined by assuming that any given cross sectional area of the vocal tract is rotationally symmetric. In order to represent the dynamics of the vocal tract, the movement of each articulator is modeled as an overdamped (or critically damped) mass spring system defined according to a time constant that is specific to each variable. The choice of articulators and their effect on the area function are largely dependent on the observation that the spectrum of the speech signal is strongly related to constrictions of the vocal tract. In addition, the control variables of the vocal tract were designed to be as independent as possible, in order to reduce the complexity of controlling the vocal tract. As a result, the behavior of these articulators is not biologically representative; however, this does not inhibit Coker model's ability to produce a wide range of phonemes.

The vocal tract model developed by Mermelstein also utilizes a geometric representation of the vocal tract. As with the Coker model, the set of variables used to describe this model was chosen based on careful analysis of existing X-ray data containing a sagittal view of the vocal tract during speech production. In contrast, however, Mermelstein's model is designed around a set of variables that are intended to directly correspond to articulators in the human vocal tract [52]. Given the more anatomically relevant control inputs, the Mermelstein model is more useful in the study of human speech production. Specifically, the Mermelstein model can be used to reasonably approximate the trajectories through which humans drive their articulators as a means of gaining insight into the control strategies used in human speech.

The ASY vocal tract model represents a synthesis of the Coker and Mermelstein models in order to balance the costs and benefits of using articulators that are either independent or anatomically representative. This model was further improved with the development of CASY, which allowed for the geometry of the vocal tract to be more easily configured based on variation between speakers. In addition, the Coker, Mermelstein, and ASY models all represent the tongue body as the arc of a circle whose center and radius are variable and treated as control inputs. The CASY model improved on this by representing the tongue body as a conic arc, which allows for a diverse range of the tongue geometries and transitions between them. As a result of these improvements, the CASY model is better able to represent the geometry and

dynamics of the human vocal tract [54].

In each of these vocal tract models, speech is synthesized based on the area function. In the original development of the Coker model and in its typical use, sound was generated using an analog resonance synthesizer to generate audio based on the formants produced by the model. The formants were determined by applying the discrete form of the Webster Horn equation for a hard walled lossless vocal tract and implementing an iterative search to find the resonant frequencies. Speech synthesis using both the Mermelstein and CASY models is performed by computing the acoustic transfer function corresponding to the area function based on the assumption that the walls of the vocal tract are rigid. Once the transfer function of the vocal tract is known, the resulting speech signal can be generated by applying an excitation signal to the appropriate location in the vocal tract. Specifically, a semiperiodic glottal wave is assumed to be generated in the location of glottis and the white noise source for fricatives is assumed to be anterior to the point of most constriction.

3.2.2 Gestural Movement Primitives

The key contribution of the Task Dynamic model is the focus on gestural patterning in speech. Gestures in this case refer to a sequence of acoustically relevant vocal tract configurations. In order to accomplish this, the model implements a set of *gestural primitives* that each drive a subset of articulators in a coordinated fashion in order to produce an acoustically relevant constriction. Each constriction that is controlled by a gestural primitive is represented by a *vocal tract variable*, and the trajectory of a vocal tract variable is used to represent a specific gesture. Each gestural primitive applies closed loop control to the articulators that are relevant to that constriction in order to produce the cross-sectional area specified by the value of its vocal tract variable. The vocal tract variables, denoted by z, are implemented as a mass spring model defined in Equation 3.1, where B is a diagonal matrix of damping coefficients, K is a diagonal matrix of stiffness coefficients, and z_0 is the equilibrium of the variable. The movement of the articulators is then determined by the kinematic relationship shown in Equation 3.2 where ϕ is defined to be the vector of articulator variables, and J is the

Jacobian matrix defined by $J_{ij} = \frac{\partial z_i}{\partial \phi_j}$. The acceleration of the articulators is then given by Equation 3.3 where the weighted pseudoinverse is defined by $J^* = W^{-1}J^\top(JW^{-1}J^\top)^{-1}$. In this model, the elements of the diagonal weighting matrix W serve as "gating variables" that define the set of articulator variables that are driven by a given tract variable.

$$\ddot{z} = -B\dot{z} - K(z - z_0) \tag{3.1}$$

$$\begin{aligned} z &= z(\phi) \\ \dot{z} &= J\dot{\phi} \\ \ddot{z} &= J\ddot{\phi} + \dot{J}\dot{\phi} \end{aligned} \tag{3.2}$$

$$\ddot{\phi}_A = J^* \left(M^{-1} \left[-BJ\dot{\phi} - K\Delta z(\phi) \right] \right) - J^*\dot{J}\dot{\phi} + (I_n - J^*J)\,\ddot{\phi}_d \tag{3.3}$$

Using this model, an utterance can be represented as a set of gestures in the space of vocal tract variables. Because of the closed loop control structure implemented in the gestural primitives, a specific utterance is dependent on the control targets rather than the specific movements of the articulators. This robust form of gesture representation results in a model that is tolerant to mechanical disturbances to the articulators and elegantly represents compensatory articulation and coarticulation. Finally, we note that this approach carries a great deal of similarity to more recent developments in robotics and control of systems with large numbers of degrees of freedom, namely the development of "dynamic movement primitives" [43][44]. Dynamic movement primitives are defined in terms of a point attractor toward which some primitive variable (analogous to the vocal tract variable in Task Dynamic model) is driven by the controller. The controller can then be defined as a simple mass spring system with an equilibrium point that can be steered by some high level control action.

In the case of dynamic movement primitives, the structure of these primitives is defined more generally and inferred based on imitation learning (e.g. the instructor moves the robotic actuator through a given movement or task like swinging a tennis racquet). This is somewhat more general than the gestural primitives used in the Task Dynamic model, which were largely defined based on human analysis rather than using some form of statistical inference.

Regardless, either supervised learning or the a priori development of a control structure requires information that is unlikely to be available to a human while they learn to speak. As such, our exploration of speech primitives for communication is strongly focused on primitives that are learned without supervision through exploration of the articulatory space.

Although the use of feedback control and gestural scores results in a robust representation of speech, the role of acoustic feedback is not addressed in this model. As a result, the controller lacks a direct means of estimating the receiver's ability to estimate the gestures used for a particular utterance. When coupled with the inherent shortcomings of the motor theory of speech perception noted in Chapter 1, this notion of a gestural score is not sufficient for representing the process of speech communication.

3.3 The DIVA Model

DIVA (directions into velocities of articulators) presently represents the most advanced model of articulatory speech production [55][50]. The model structure used in DIVA is notable in that it implements articulator control using artificial neural networks for the purpose of imitating speech phenomena observed in humans, primarily for the purpose of studying speech pathologies. The control structure utilizes both somatosensory and acoustic feedback to drive the articulators of the vocal tract in a hierarchical control scheme. Interestingly, early versions of this model did not actually simulate the speech production process, but it utilized a "Speech Recognition System" that acted as an expert system to map vocal tract shapes to speech sounds. As DIVA was further developed, a modified version of Maeda's vocal tract simulation [53][56] was implemented and used to produce the resultant acoustic signals.

3.3.1 The Maeda Vocal Tract Model

Much like other geometric models of the vocal tract, the Maeda model utilized a two-dimensional representation where the area function at a given point along the vocal tract is determined by the corresponding midsagittal cross-section. Unlike the rotational symmetry found in other models, the cross sectional area is determined by the formula shown in Equation 3.4, where

α_x and β_x vary along the length of the vocal tract and y is the width of the midsagittal cross-section. The actual values of α and β were determined in an *ad hoc* manner.

$$A = \alpha_x y^{\beta_x} \tag{3.4}$$

Simulation of the acoustic waveform produced by the vocal tract is accomplished in a manner very similar to the method used in both the Mermelstein and CASY models. Specifically, the transfer function of the vocal tract is determined by modeling the area function as a lossy transmission line. The sound source is then implemented in the same manner as Mermelstein's model by injecting a glottal wave or white noise at the appropriate points along the vocal tract. Once again, this model simulates the glottal wave as a function of time by assuming a constant pressure input from the lungs, rather than directly modeling the dynamics that drive the oscillation of the vocal folds.

Much like the Mermelstein model, the Maeda model is controlled through a set of articulators that correspond to actual articulators that exist in the vocal tract. The choice of articulators was determined based on cineradiographic and labiofilm data using statistical analysis rather than the more traditional engineering analysis used to fit the Coker and Mermelstein models to existing data. Maeda makes a particular note that the mapping from the space of articulators to the area function can be treated as a linear transformation and, as such, can be determined via factor analysis. Solving for this transformation, however, is complicated by the existence of multiple solutions, most of which are arbitrary and do not correspond to the physical anatomy of the vocal tract articulators. In order to solve this problem, Maeda utilizes "arbitrary component analysis", wherein the most important articulators (such as the jaw) are determined a priori and their influence is subtracted from the correlation matrix. The resultant correlation matrix can then be further decomposed using PCA to capture the effects of less significant articulator parameters.

It is significant that the Maeda model was designed to capture the fact that the vocal tract has excess degrees of freedom relative to the resultant speech signal. As a result, it is well suited for the characterization of compensatory articulation due to the imposed orthogonality of the articulators. It is this characteristic in particular that makes the Maeda model well suited for the

goals of the DIVA model of speech production, specifically the study of speech pathologies.

3.3.2 Neural Model of Control

In many ways, DIVA builds directly from the design of the Task Dynamic model. The control is implemented in a hierarchy such that a low level controller drives the vocal tract toward 'orosenory' targets (much like the Task Dynamic vocal tract variables) and a high level controller adjusts those targets based on acoustic feedback. Acoustic targets are extended by representing them as target *regions* rather than the point attractor structure used by the Task Dynamic model. Additionally, the use of neural networks to implement the controller allows for a flexible nonlinear mapping that is able to represent much more complex behavior than can be represented by a linear feedback controller. Lastly, the controller parameters are learned through babbling wherein random articulator movements are super-imposed on an oscillatory jaw movement, which is intended to mimic vocal play seen in infants.

Research involving DIVA has increasingly focused on the mapping between components of the model and brain regions by empirically relating behavioral, neurophysiological, lesion, and neuroanatomical data to subsets of the model's neural nets [50]. As such, the implicit goal in the development of DIVA has been imitation of human behavior as a means of predictively modeling speech pathologies. In contrast, the development of speech primitives presented in this thesis is intended to be a more general framework for modeling speech communication as opposed to direct imitation of human behavior.

3.4 Conclusion

In this chapter, we have provided an overview of basic vocal tract anatomy and its nuanced relationship to the structure of the acoustic speech signal. In addition, we have reviewed several prominent models of articulatory speech production. Although each of the models lends important insights to our development of speech primitives, they incorporate significant trade-offs due to limitations in computational power available when they were originally

developed. Given that our development of speech primitives is heavily influenced by the structure of the speech production model, we seek a speech production model that is more anatomically faithful than those outlined in this chapter. The following chapter will introduce a model that does suit the needs of this research; namely, the Praat model and its Python adaption we call the Pyraat library.

CHAPTER 4

THE PRAAT MODEL AND PYRAAT LIBRARY

The Pyraat vocal tract model is an implementation of the vocal tract model developed by Paul Boersma for his Praat software package. Boersma's vocal tract model is significantly more advanced than the articulatory models used in both the Task Dynamic model and DIVA, and is capable of simulating speech phenomena that had not been simulated by any preceding speech production models. Pyraat is a stand-alone python library, developed by the author and W. Jacob Wagner, that implements Boersma's Praat vocal tract model. By implementing the Pyraat library in Python, algorithms for articulatory control of the Praat vocal tract model can be rapidly developed.

This chapter will focus on the Praat vocal tract geometry, the simulation of the acoustic signal, and the adaptation of the Praat software for the purpose of implementing feedback control. In addition, we will also provide an overview of the simulation of the acoustic signal, and discuss how the model software has been modified for the research presented in this thesis. Because of these significant improvements over its predecessors, the Praat model is capable of simulating speech phenomena that had not been previously simulated in any other vocal tract model. Given that this model is far more anatomically accurate than its predecessors, it is presently the best choice for our development of speech primitives.

4.1 The Praat Model

The development of Praat was motivated by an investigation of the principles of functional phonology and as such is inherently focused on the transmission of information via the speech signal. As such, the mechanisms used to generate and receive the speech signal must be faithfully represented by the simulator. As a result, the Praat model of articulatory speech synthesis seeks

to faithfully model the entire vocal apparatus, from lungs through the lips and nostrils. In contrast with models that simplify the vocal tract as having rigid walls, the Praat model simulates all the walls of the vocal tract as a set of mass spring systems in the same manner as the classical two-mass model of the glottis [22]. In addition, the Praat model simulates the acoustic wave by directly simulating the pressure inside the vocal apparatus, which allows the internal air pressure to affect mass-spring walls of the vocal tract and the area function. These significant improvements in the anatomical faithfulness of this model come at a much higher computational cost; however, this cost is no longer prohibitive in light of modern computing systems.

4.1.1 Vocal Tract Geometry and Control

Unlike most vocal tract simulations, the Praat model represents the entire vocal apparatus from lungs to lips. The lungs through the larynx are modeled as a set of tubes that progressively branch out in order to model the branching of the bronchi down to the alveoli in the lungs. The articulators of the vocal tract and the vocal tract geometry are directly based on the Mermelstein model and depicted in Figure 4.1. Control of the articulators is implemented through inputs that represent the muscles that are used to drive the articulators. The muscle inputs drive the articulators by changing the equilibrium length of the muscles and thus influencing the equilibrium point of the articulators. The behavior of these muscle inputs is effectively equivalent to the muscle control strategy proposed by the equilibrium point hypothesis for motor control [57][58]. Praat does not include any explicit coordinative structures like those found in DIVA or the Task-Dynamic model; rather, control of the vocal tract is performed by manually defining the trajectories of the muscle input parameters. In addition, the lengths of the tubes are allowed to vary within the simulation, much like the model due to Maeda which included extension of the lips.

The vocal apparatus as modeled by Praat consists of a total of 78 tube segments, with the lower respiratory system (lungs through bronchi) consisting of 29 segments, the trachea and glottis consisting of 8 segments, the oral cavity consisting of 27 segments, and the nasal cavity consisting of 14 segments. Each tube segment is made up of a mass spring system where the

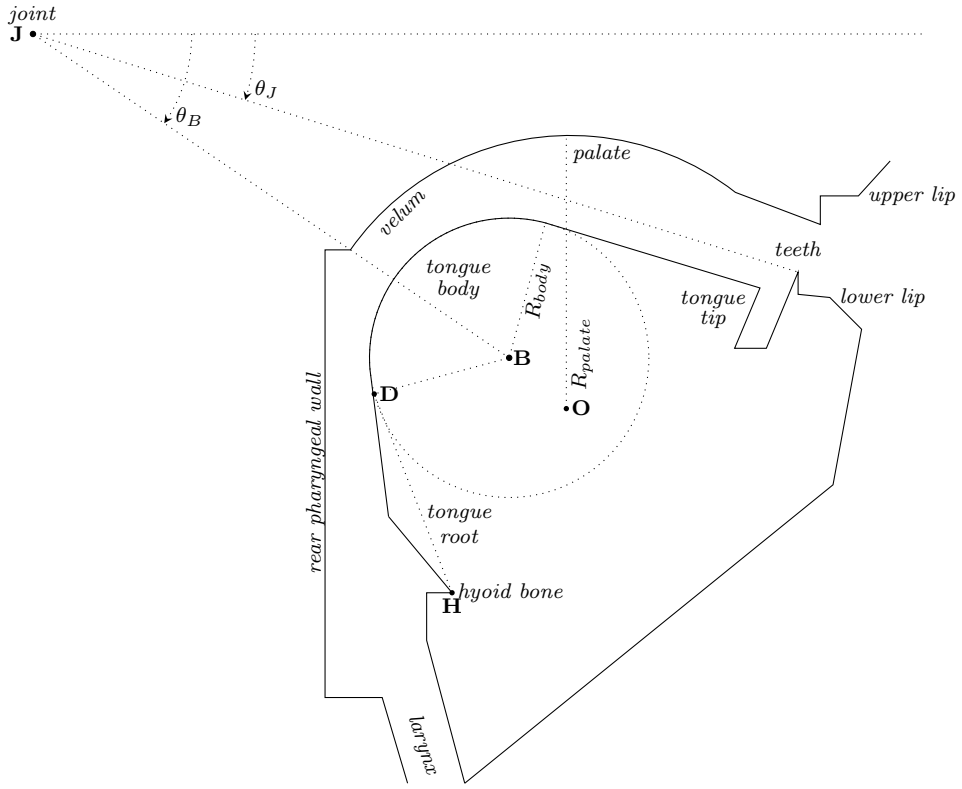


Figure 4.1: Geometry of the Praat vocal tract model where J is the joint about which the jaw hinges, θ_J is the jaw angle, B is the center of the tongue body, θ_B and R_{body} are the angle and radius of the tongue body center, R_{palate} is the radius of the palate, and D is used to determine the tongue root shape.

Table 4.1: Table of the input parameters and their effect on the vocal apparatus.

Muscle	**Function**
Lungs	Compression of the lungs, simplification of diaphragm and abdominal muscles
Interarytenoid	Constriction of the larynx
Cricothryroid	Tension of the vocal chords
Thyroarytenoid	Combination of aryepiglottic sphincter muscles
Posterior Cricoarytenoid	Opening of glottis
Lateral Cricoarytenoid	Opening of glottis
Stylohyoid	Upward movement of the hyoid bone
Sternohyoid	Downward movement of the hyoid bone
Thyropharyngeus	Constriction of the ventricular folds
Sphincter	Backward movement of the hyoid bone, forward movement of rear pharyngeal wall
Hyoglossus	Downward movement of the tongue body
Styloglossus	Upward movement of the tongue body
Genioglossus	Forward movement of the tongue body
Upper Tongue	Upward curling of the tongue tip
Lower Tongue	Downward curling of the tongue tip
Transverse Tongue	Thickening of the tongue
Vertical Tongue	Thinning of the tongue
Risoris	Spreading of the lips
Orbicularis Oris	Rounding of the lips
Levator Palatini	Opening or closing of the velo-pharyngeal port
Masseter	Closing of the jaw
Mylohyoid	Opening of the jaw
Lateral Pterygoid	Horizontal jaw position
Buccinator	Oral wall tension

muscle inputs outlined in Table 4.1 determine the equilibrium length of the various tube segments. In addition, each pair of adjacent tube sections is coupled together by a coupling spring force. Finally, contact between walls of the vocal tract is modeled via a collision force wherein the interacting walls of the vocal tract are compressed. In order to model such collisions while maintaining a smooth area function, the collisions are performed through a "zipper" process which does not attempt to simulate any actual asymmetric behavior of the walls.

Because the vocal tract walls are modeled as mass spring systems, the Praat vocal tract model is capable of simulating speech phenomena not found in other articulatory synthesizers. Since the entirety of the vocal tract is modeled in the same way, it is possible to achieve the same type of oscillation found in the glottis (during voicing) at any point in the vocal tract, provided that the tube segments are subjected to the correct conditions. As a result, the model can produce clicks at various points in the vocal tract and even inflate the vocal tract in what Boersma refers to as "ballooning." In addition, the model is capable of producing trills, though manually setting up the correct conditions and sequence of muscle inputs is prohibitively challenging. It should be noted, however, that achieving such fidelity comes with a significant increase in both the number of control inputs and the overall computational cost.

Although the Praat model of the vocal apparatus is more biologically faithful than others, it does make some concessions for the sake of simplifying control of the articulators through the set of muscle inputs. First, the muscles' effects on the articulators are explicitly additive, which greatly simplifies the more complex interactions found in the muscles of the human vocal tract. Second, some of the muscles involved in speech are grouped together in order to reduce the number of control inputs. Among these grouped muscles are the lungs, some of the muscles in the larynx, and the various muscles of the tongue. Control of the tongue is an especially significant simplification in this model in that it does not directly attempt to capture the behaviors of the muscles; instead, the control inputs for the tongue correspond directly to the articulators themselves. This concession is a significant departure from the design philosophy of the Praat model, but it captures the movement of the tongue without the need for modeling the complex interactions of its muscle structures.

4.1.2 Aerodynamics

Because the walls of the vocal tract are deformable, the set of equations that govern the geometry of the Praat vocal apparatus is coupled to the equations that govern the aerodynamics within the vocal tract. This myoelastic-aerodynamic coupling allows for a higher fidelity simulation at the cost of increased computational complexity. Specifically, the set of aerodynamic differential equations is simplified substantially by the assumption of a rigid vocal tract geometry. Notably, all other models of articulatory synthesis that have been presented in this chapter have employed this "rigid vocal tract" assumption.

In order to properly deal with the variable vocal tract walls within this simulation, the treatment of continuity of mass must first be dealt with. In Boersma's development of this model, he derives the continuity of mass equation directly in the form shown in Equation 4.1, where A is the area function, x is the position along the length of the tube, and v is the particle velocity along the tube. If the area function is assumed constant at any instant in time, then it could be eliminated from this equation and the resultant simulation could be greatly simplified. This equation is particularly important when we consider the role of the lungs in this model as the pressure source for the air moving through the vocal tract. Specifically, if we assume that the vocal tract is filled with an incompressible fluid (i.e. ρ is constant), then a change in area A produces an opposing change in volume flow vA. That is to say that a decrease in the area of the vocal tract will force air out of the vocal tract and an increase in area function will pull air in.

$$\frac{\partial(\rho A)}{\partial t} + \frac{\partial(\rho v A)}{\partial x} = 0 \tag{4.1}$$

Other important aspects of this aerodynamic simulation include some important but less novel considerations. First, the Bernoulli effect is accounted for along all tubes in the vocal tract in the same manner as the traditional two mass glottal model. Friction is captured by accounting for viscous resistance of the air along the vocal tract walls, which is significant in the case of sections of the vocal apparatus are modeled as multiple parallel tubes (i.e. lungs, bronchi, nasal cavity). Turbulence is not directly modeled as an aerodynamic process; instead, the Reynolds number is used to determine the amplitude of a white noise source applied to the pressure function along

the vocal tract. Importantly, the same set of aerodynamics equations is applied across each tube segment throughout the entire vocal apparatus, which results in a more consistent model of speech production.

4.1.3 Numerical Simulation

The numerical simulation of the acoustic and myoelastic equations is performed in discrete time steps. The differential equations governing the myoelastic and aerodynamic equations are numerically approximated using first order approximations, finite difference methods, and a modified version of the Lax-Wendroff method. Justification for each of these numerical methods based on accuracy, stability, and frequency response is provided in Chapter 4 of Boersma's "Functional Phonology" [59], to which the interested reader is referred . As a result of these numerical methods, the sampling period must be smaller than the time needed for sound to travel the length of any single tube segment in order guarantee numerical stability. Given that the smallest tube is 0.7mm and the speed of sound is approximately 350m/s within the vocal tract, the simulation must be computed at a minimum sampling rate of approximately 500kHz. At the time of its development, simulating 1 second of speech data with Praat took approximately 1,000 seconds, but this is reduced to the range of 2-3 seconds on a modern desktop computer.

4.2 Export to Python: Pyraat

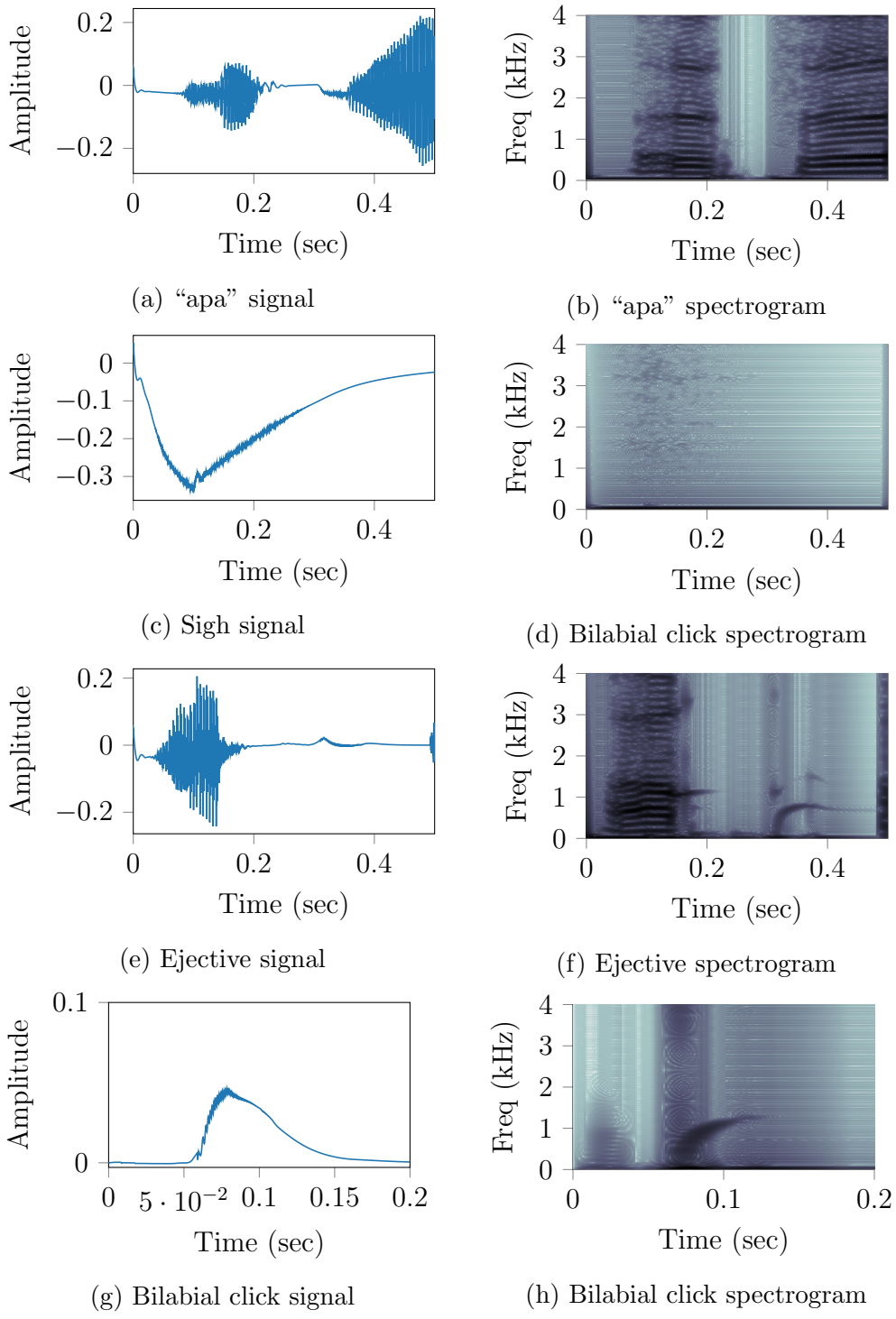
The Praat software includes the vocal tract simulation outlined in this section; however, the interface to this vocal tract model is relatively limited. In particular, the vocal tract control inputs can only be controlled via a simple open loop control script where the control inputs linearly interpolate between a fixed set of targets. Although this might be suitable for producing a desired utterance repeatably, it does not allow for closed loop control of the vocal tract. In order to implement a closed loop controller on the Praat model, it has been extracted from the open source Praat software and implemented as a Python library called Pyraat. The Python library is written in C++, the native programming language that Praat was originally written in, and exported to Python using Boost to expose the C++ functions to Python.

Using this library, the Praat vocal tract model can be controlled using feedback control implemented in a high level scripting language with negligible increase in run-time.

The Pyraat library interfaces with the Praat model using the same set of muscle inputs defined in Table 4.1. In addition, the library provides the resultant acoustic waveform, as well as the area function and pressure at each tube segment sampled at the same rate as the acoustic waveform. As in the original Praat software, the sampling rate of the acoustic waveform is variable and can be defined based on the requirements of each specific application. Since the minimum sampling rate for simulating the aerodynamic-myoelastic equations is much higher than the necessary sampling rate for audio signal processing, the acoustic waveform is captured by subsampling the pressure function at the lips and nose. Example acoustic waveforms and their corresponding spectrograms are shown in Figure 4.2.

4.3 Conclusion

It should be noted that the Praat model was developed for the purpose of investigating the concept of functional phonology, which argues that phonological structure is the result of optimizing the reliability of speech communication [59]. This notion of phonology is central to the research presented in this thesis; however, our investigation of this concept is notably different from that used by Boersma. Specifically, we seek to develop a notion of speech primitives that are fundamentally unsupervised. Conversely, Boersma's treatment of functional phonology is fundamentally built around imitation of human speech and generally presupposes the existence of phonemes as the fundamental units of the speech signal. Even so, the common thread between these two lines of research is the need for a highly accurate articulatory model of speech synthesis. As such, the development of speech primitives in the remaining chapters of this thesis will make heavy use of the Pyraat library as a means of interfacing our speech primitives with a vocal tract model.



(a) “apa” signal

(b) “apa” spectrogram

(c) Sigh signal

(d) Bilabial click spectrogram

(e) Ejective signal

(f) Ejective spectrogram

(g) Bilabial click signal

(h) Bilabial click spectrogram

Figure 4.2: Example utterances simulated using the Pyraat library.

CHAPTER 5

PRIMITIVES FOR LINEAR FEEDBACK CONTROL

The concept of sensorimotor primitives was broadly introduced in Chapter 2 and this chapter further develops the framework in terms of feedback control. In this chapter, we formally develop the concept of sensorimotor primitives as it applies to feedback control and tailor it for specific application to the Pyraat speech synthesizer. In keeping with the notion of primitives as a means of simplifying the problem of interfacing with the environment, we define sensorimotor primitives in terms of a generalized feedback transformation. Under this definition, we consider the linear case of sensorimotor primitives and how they can be learned in an unsupervised manner, with special consideration for high dimensional systems. Finally, we consider some practical considerations when estimating these linear sensorimotor primitives from very large sets of data.

5.1 A Formal Definition

Rather than motivate the notion of primitives using a specific control problem, such as control of the human vocal tract, we will first consider a more general treatment of the concept. The use of primitives for robotics and control is primarily motivated by reducing the complexity of a given control task. This is typically achieved by imposing some form of hierarchy in which a simple high level controller uses simplified actions to steer a low level controller that in turn drives the system of interest, as illustrated by Figure 5.1.

Throughout this chapter, we will consider the development of primitives in the context of this control hierarchy. The primary insight provided by Todorov [45] is that the low level controller in Figure 5.1 should be treated as a feedback transformation that serves to provide a "simplified" interface

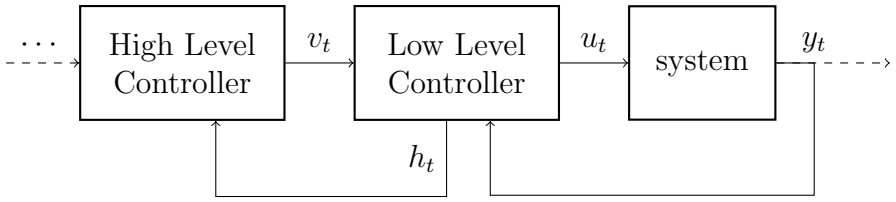


Figure 5.1: A general hierarchical control structure with two feedback transformation stages in the control hierarchy.

for the high level controller. In the case of high dimensional systems with redundant control inputs (such as those seen in biological motor control), a simplified interface would reduce the dimensionality of the system through coordination of redundant elements. We now proceed with developing the formal mathematical framework for achieving this goal through unsupervised learning methods.

5.1.1 Primitives as a Feedback Transformation

Consider the general dynamical system defined by Equation 5.1, where z_t is the internal state of the system, y_t is the observed output, u_t is the low level control input, and f and g represent the dynamics and sensory mappings respectively.

$$\begin{aligned} z_{t+1} &= f(z_t, u_t) \\ y_t &= g(z_t, u_t) \end{aligned} \tag{5.1}$$

We seek a low level controller input u_t that can be steered by some high level control signal v_t. Let h_t be the internal state of the low level controller, let $\overleftarrow{X}_t^p$ denote the past p observations of the inputs and outputs, as shown in Equation 5.2, and let $\overrightarrow{X}_t^f$ be the future of length f, as shown in Equation 5.3.

$$\overleftarrow{X}_t^p = \begin{bmatrix} \overleftarrow{Y}_t^p \\ \overleftarrow{U}_t^p \end{bmatrix}, \quad \overleftarrow{Y}_t^p = \begin{bmatrix} y_{t-p} \\ \vdots \\ y_{t-1} \end{bmatrix}, \quad \overleftarrow{U}_t^f = \begin{bmatrix} u_{t-p} \\ \vdots \\ u_{t-1} \end{bmatrix} \tag{5.2}$$

$$\overrightarrow{X}_t^f = \begin{bmatrix} \overrightarrow{Y}_t^f \\ \overrightarrow{U}_t^f \end{bmatrix}, \quad \overrightarrow{Y}_t^f = \begin{bmatrix} y_t \\ \vdots \\ y_{t+f} \end{bmatrix}, \quad \overrightarrow{U}_t^f = \begin{bmatrix} u_t \\ \vdots \\ u_{t+f} \end{bmatrix} \tag{5.3}$$

The fundamental step in this formulation is then to fit a low level controller to the conditional probability model in Equation 5.4.

$$P\left(\overrightarrow{X}_t^f \middle| \overleftarrow{X}_t^p\right) = \sum_h P\left(\overrightarrow{X}_t^f \middle| h_t\right) P\left(h_t \middle| \overleftarrow{X}_t^p\right) \tag{5.4}$$

With this conditional probability structure, the internal state of the low level controller can be used to compactly represent the history of the system and determine the control actions of both the high and low level controller. Specifically, we can estimate the internal state by computing $\hat{h}_t = \mathbb{E}\left(h|\overleftarrow{X}_t^p\right)$, where $\mathbb{E}$ denotes the expectation operator. The high level control action is then determined by $v_t = \mathcal{Q}(\hat{h})$, where $\mathcal{Q}$ is some task-specific control policy. The low level control actions can then be determined by $u_t = \mathbb{E}\left(u|v_t\right)$.

A sensorimotor primitive is then defined to be a feedback transformation that satisfies this probabilistic model [45]. We then define the general feedback transformation as shown in Equation 5.5. In this formulation, $\mathcal{T}$ and $\mathcal{G}$ are deterministic operators that represent the dynamics of the system and η_t and ϵ_t are uncorrelated zero-mean random variables. The control input u_t is then determined at each time step by selecting the corresponding row from the future history produced by $\mathcal{G}$. Under this formulation, we will refer to h as the internal state of the low level controller and v is now the control input that is driven by a high level control policy.

$$\begin{aligned} h_t &= \mathcal{T}\left(\overleftarrow{X}_t^p\right) + \eta_t \\ \overrightarrow{X}_t^f &= \mathcal{G}\left(v_t\right) + \epsilon_t \end{aligned} \tag{5.5}$$

Using the general model structure shown in Equation 5.5, we have that the conditional probability structure of Equation 5.4 is satisfied. In addition, the problem of control may be separated into two steps. The first is the problem of unsupervised learning of $\mathcal{T}$ and $\mathcal{G}$, which is the focus of this chapter. The second is the problem of learning a set of task-specific control policies of the form $v_t = \mathcal{Q}(h_t)$, which is central to the problem of speech communication and will be dealt with in Chapter 6.

5.1.2 Gestures vs. Coordinative Control

The notions of gestures and coordinative control are both common themes in the study of primitives and, as such, it is important to address how these concepts relate to this model of sensorimotor primitives. Coordinative structures are critical to the notion of dimensionality reduction when dealing with systems with many degrees of freedom. Gestures, on the other hand, can be thought of as trajectories through the space of control actions and are central to tasks such as movement segmentation. Under the framework of sensorimotor primitives, both of these concepts are elegantly represented in terms of the operators $\mathcal{T}$, $\mathcal{G}$, and $\mathcal{Q}$.

The notion of coordinative control is naturally represented in terms of the operator $\mathcal{G}$. Let the low level control input be defined as $u_t = [u_t^1, \ldots, u_t^n]^\top$ and let the high level control input be $v_t = [v_t^1, \ldots, v_t^n]^\top$, where v_t^i and u_t^j are scalar elements of each vector. Without loss of generality, suppose $\mathcal{G}$ maps $v_t \to u_t$ for some fixed value of h_t. Then we have that a change in any single v_t^i will result in a change to a subset of u_t^j values. The structure of this mapping then forms the basis of coordinative control in that a single dimension of the high level controller maps to multiple dimensions of the low level controller.

Gestures, on the other hand, are neatly represented in terms of the high level control policy $\mathcal{Q}$. Let $\tilde{h}_t$ be a control target in $\mathbb{H} = \text{span}(h_t)$ and let $\mathcal{Q}$ define the policy $v_t = \mathcal{Q}(h_t)$ such that $h_t \to \tilde{h}$ as $t \to \infty$. For a given starting point h_0, we have that $\mathcal{Q}$ defines a trajectory through the state space $\mathbb{H}$ to the point $\tilde{h}$. An important contrast in such an approach to gestures as compared to other gesture based primitives [43][44] is that the trajectory is defined in the space of coordinated control inputs rather than considering individual trajectories for each control input. A key advantage to this approach is that the gestures learned at this higher level of control are both more intuitive for human understanding and naturally deal with the issue of time-aligning various low level trajectories.

5.2 The Linear Assumption: Dynamic Factor Analysis

We now proceed with the development of a method of estimating the operators of our feedback transformation. It should be noted that although the

basic formulation is similar to that developed by Todorov [46], the estimation method presented here is largely due to collaboration with W. Jacob Wagner [60]. Let us assume that the system under analysis is linear and represented by the linear state space model shown in Equation 5.6. Note that in this case, we have that u_t is determined by a linear function of the internal state of the system and the high level control input is given by v_t. We can then reformulate the dynamics of the system as the predictive model shown in Equation 5.7.

$$\begin{aligned} z_{t+1} &= Az_t + Bu_t + Fv_t \\ y_t &= Cz_t + Du_t + Gv_t \\ u_t &= Ez_t + Hv_t \end{aligned} \tag{5.6}$$

$$\overrightarrow{X}_t^f = \mathcal{G}\mathcal{T}\overleftarrow{X}_t^p + \mathcal{E}\overrightarrow{V}_t^f \tag{5.7}$$

The past and future history are denoted by $\overrightarrow{X}_t^f$ and $\overleftarrow{X}_t^p$, as previously defined in Equations 5.2 and 5.3, and $\overrightarrow{V}_t^f$ is the vector of future high level control inputs. The matrices $\mathcal{G}$, $\mathcal{T}$ and $\mathcal{E}$ can then be derived from the linear system model to find the definitions shown in Equation 5.8, where $\bar{A} = (A + BE)$, $\bar{C} = (C + DE)$, $\bar{F} = F(DH + G)^\dagger$, and $\dagger$ denotes the pseudoinverse operator.

$$\tilde{\mathcal{G}} = \begin{bmatrix} \bar{C} \\ \bar{C}\bar{A} \\ \vdots \\ \bar{C}\bar{A}^f \\ E \\ \vdots \\ E\bar{A}^f \end{bmatrix} \tag{5.8a}$$

$$\tilde{\mathcal{T}} = \begin{bmatrix} \bar{F},\ \left(A - \bar{F}\bar{C}\right)\bar{F},\ \ldots \left(A - \bar{F}\bar{C}\right)^f \bar{F},\ B,\ \ldots \left(A - \bar{F}\bar{C}\right)^f B \end{bmatrix} \tag{5.8b}$$

$$\mathcal{E} = \begin{bmatrix} G & 0 & \cdots & 0 \\ \bar{C}F & G & & \vdots \\ \vdots & & \ddots & 0 \\ \bar{C}\bar{A}^{f-1}F & \cdots & \bar{C}F & G \\ H & 0 & \cdots & 0 \\ EF & H & & \vdots \\ \vdots & & \ddots & 0 \\ E\bar{A}^{f-1}F & \cdots & EF & H \end{bmatrix} \tag{5.8c}$$

Under this definition, we have that the operators $\mathcal{T}$ and $\mathcal{G}$ predictively model the future values of y and u with a prediction error given by $\mathcal{E}\overrightarrow{V}_t^f$. We then seek to estimate matrices $\hat{\mathcal{T}}$ and $\hat{\mathcal{G}}$ such that the error is minimized. The matrices that satisfy this condition can then be used to form the linear feedback transform given by Equation 5.9.

$$\begin{aligned} h_t &= \hat{\mathcal{T}}\overleftarrow{X}_t^p \\ \overrightarrow{X}_t^f &= \hat{\mathcal{G}}\left(h_t + v_t\right) \end{aligned} \tag{5.9}$$

Under this framework, the feedback transform represented by $\hat{\mathcal{T}}$ and $\hat{\mathcal{G}}$ forms a linear predictive model that represents the dynamics of the system. Additionally, the rows of $\hat{\mathcal{G}}$ that map to the predicted $\overrightarrow{U}_t^f$ represent a coordinative structure that maps the high level control inputs v_t onto the low level control actions.

5.2.1 Considerations in Model Selection

The predictive model defined by Equation 5.7 represents a generalization of linear prediction analysis as applied to vector signals. It is important to note that traditional linear prediction analysis of the speech signal requires the assumption that the signal is stationary over the window of time being analyzed. The stationarity assumption is equally important in this notion of vector linear prediction, but instead this model assumes that the dynamics of the vocal apparatus are stationary. Given that the simulated vocal tract used in this thesis models the entirety of the speech production process with a fixed set of difference equations, this stationarity assumption is not violated.

In addition, the selection of past and future observation lengths, p and f

respectively, plays an important role in the behavior of this model. As with linear prediction, we have that the prediction error tends asymptotically toward zero as $p \to \infty$; however, this property does not hold when finite training data is available. In general, p must be chosen large enough such that the error is minimized but small enough such that the model is not overfit to the training data. A method of selecting p with finite training data is dealt with in Section 5.2.2. The selection of f, the other hand, minimizes the prediction error when $f = 1$, but for $f > 1$ the predicted control actions tend toward the mean of u. As a result, the selection of $f > 1$ can be used as a means of smoothing the mapping from high level to low level control actions.

5.2.2 Parameter Inference

In general, there are two approaches to learning the operator matrices $\mathcal{T}$ and $\mathcal{G}$ in an unsupervised manner. The most commonly used method for this form of system identification problem is to compute the maximum likelihood (ML) estimate of the parameters using the Expectation-Maximization (EM) algorithm. Unfortunately, computing the ML estimate via the EM algorithm comes with a high computational cost that grows exponentially with the dimension of the system. An alternative is to compute the minimum mean square error (MMSE) estimate using the Subspace Method proposed by Kapetanios for application to high dimensional data sets [61]. Due to the high dimensionality of the Praat model, the EM algorithm is poorly suited for parameter inference in this system so we focus our attention on the Subspace Method.

Let $\mathcal{F} = \mathcal{G}\mathcal{T}$ be the composition of the two linear operators, then we have that the future inputs and outputs can be estimated by Equation 5.10. The linear operators $\mathcal{G}$ and $\mathcal{T}$ can then be found by matrix decomposition of $\mathcal{F}$ in order to get Equation 5.9.

$$\overrightarrow{X}_t^f = \mathcal{F}\overleftarrow{X}_t^p \tag{5.10}$$

The goal is then to estimate $\mathcal{F}$ by minimizing $\|\overrightarrow{\mathbf{X}}_t^f - \mathcal{F}\overleftarrow{\mathbf{X}}_t^p\|^2$ for the observed input/output history of the system. First, we compile T input-output data points to form the matrices $\overleftarrow{\mathbf{X}}^p = \left[\overleftarrow{X}_p^p, \overleftarrow{X}_{p+1}^p, \ldots, \overleftarrow{X}_{T-f}^p\right]$ and

$\overrightarrow{\mathbf{X}}^f = \left[\overrightarrow{X}^f_p, \overrightarrow{X}^f_{p+1}, \ldots, \overrightarrow{X}^f_{T-f}\right]$. By applying the Projection Theorem, we have that the MMSE estimate is given by $\hat{\mathcal{F}} = \overrightarrow{\mathbf{X}}^f \left(\overleftarrow{\mathbf{X}}^p\right)^\dagger$.

From this estimate, we can then estimate $\hat{\mathcal{T}}$ and $\hat{\mathcal{G}}$ using singular value decomposition (SVD) of $\hat{\mathcal{F}} = USV^\top$, as shown in Equations 5.11a and 5.11b.

$$\hat{\mathcal{T}} = S_k^{\frac{1}{2}} V_k^\top \tag{5.11a}$$

$$\hat{\mathcal{G}} = U_k S_k^{\frac{1}{2}} \tag{5.11b}$$

Here we define S_k to be the $k \times k$ diagonal submatrix of S, with U_k and V_k being the first k columns of U and V. Provided that k has been selected correctly and the system under analysis is both linear and stable, we have that the range space of $\hat{\mathcal{G}}$ will coincide with the original state-space z in the limit as $T \to \infty$ [61]. It should be noted the solution for $\mathcal{F} = \mathcal{G}\mathcal{T}$ is not unique. Specifically, there exists an affine ambiguity such that for any nonsingular $k \times k$ matrix C, we have $\tilde{\mathcal{G}} = \mathcal{G}C$ and $\tilde{\mathcal{T}} = C^{-1}\mathcal{T}$ forms a space of valid solution. Although the choice of C does not affect the consistency of this estimate, it does provide a means by which we can impose restrictions on the structure of the state space $\mathbb{H}$. For example, it may be useful to resolve this ambiguity by choosing C such that the dimensions of $\mathbb{H}$ are orthogonal.

Lastly, in order for $\hat{\mathcal{F}}$ to be a consistent estimate we must select a value of p such that $\ln(T)^\alpha < p < T^{\frac{1}{3}}$, where α is dependent on the stability of the state transition matrix A [61]. Alternatively, p can be chosen based on the Akaike information criterion (AIC), for which a choice of $p = 2p_{AIC}$ satisfies this requirement almost surely [62][63]. Let the observation history of the input-output of the system be represented by the matrix $\mathbf{X}$ defined in Equation 5.12, where q is the total number of observations per column. The value for p_{AIC} is then computed using Equation 5.13, where λ_i is the i^{th} eigenvalue of the covariance of $\mathbf{X}$ [64].

$$\mathbf{X} = \begin{bmatrix} u_0 & u_q & u_{2q} & \cdots & u_{T-q} \\ y_0 & y_q & y_{2q} & \cdots & y_{T-q} \\ u_1 & u_{q+1} & u_{2q+1} & \cdots & u_{T-q+1} \\ \vdots & & & & \vdots \\ y_{q-1} & \cdots & & \cdots & y_T \end{bmatrix} \tag{5.12}$$

$$p_{AIC} = \arg\min_{p} -2\log\left(\frac{\prod_{i=p+1}^{k}\lambda_i^{1/(q-p)}}{\frac{1}{q-p}\sum_{i=p+1}^{k}\lambda_i}\right)^{(q-p)T} + 2p(2k-p) \qquad (5.13)$$

5.2.3 Considerations for Large Training Data

We now consider the critical challenge of estimating linear primitive operators from a large set of high dimensional data. Given that the vocal tract is controlled by a set of 26 muscle inputs and the area function is represented by 78 tube segment diameters, a history of the input-output space of the system can quickly consume massive amounts of memory. The critical step in estimating the primitive operators is computing the MMSE estimate of the prediction matrix $\mathcal{F}$, as defined in Equation 5.10. This computation requires a matrix product of an $f \cdot k \times T$ matrix with the pseudo-inverse of a $p \cdot k \times T$ matrix, where T is the total number of data points divided by $f + p$, and k is the dimension of the input-output space ($k = 115$ in the case of the Praat vocal tract model). Since these matrices grow linearly with T, computational memory is very quickly consumed.

In order to deal with the problem of memory allocation, we seek a method of estimating $\mathcal{F}$ through batch updates that do not consume unbounded amounts of memory. One way of accomplishing this is to incrementally estimate $\mathcal{F}$ through a set of rank one updates. More precisely, we can compute a set of intermediate matrices of constant dimension that can be used to compute an MMSE estimate of $\mathcal{F}$.

For simplicity of notation, we'll denote the set of past histories by $\overleftarrow{X}^{p}_{n}$ and $\overrightarrow{X}^{f}_{n}$ based on the definitions given in Equations 5.2 and 5.3 with the addition of the subscript n to indicate that each was constructed from a total of n I/O histories of length $p + f$. Additionally, $\overleftarrow{X}^{p}_{n}(i)$ is used to denote the i^{th} row of $\overleftarrow{X}^{p}_{n}$ and $\overrightarrow{X}^{f}_{n}(i)$ is defined similarly.

We then define the n^{th} update of $\mathcal{F}$, denoted by $\mathcal{F}_n$. It is then possible to rewrite the estimate of $\mathcal{F}_n$ as the product of the variables Φ_n and Ψ_n as shown in Equation 5.14. Importantly, the dimensions of Φ_n and Ψ_n are

proportional to $f \times p$ and $p \times p$ respectively and do not change with n as the value of n only affects the inner dimensions of their respective matrix products. In addition these matrices may be updated incrementally without the need to store all data points in memory simultaneously.

$$\begin{aligned} \mathcal{F}_n &= \left(\overrightarrow{X}^f_n \cdot \overleftarrow{X}^{p\top}_n\right)\left(\overleftarrow{X}^p_n \cdot \overleftarrow{X}^{p\top}_n\right)^{-1} \\ \mathcal{F}_n &= \Phi_n \Psi_n^{-1} \end{aligned} \tag{5.14}$$

Importantly, the data must be normalized to be zero mean and unit variance prior to computing the estimate of $\mathcal{F}_n$; however, the mean and variance must also be estimated from the available data. One solution would be to compute the mean and variance from all the available data and normalize prior to constructing Φ_n and Ψ_n. The drawback of this approach is that it requires storing all the data and parsing it at least twice in order to estimate $\mathcal{F}$. Instead, we seek a method of normalizing the data after constructing Φ_n and Ψ_n. Given that the variance is normalized by a multiplicative operation, this normalization may be achieved by element-wise division of $\mathcal{F}_n$. Normalizing the mean value of the data points is somewhat more nuanced.

Let μ_n be the mean estimated from the first n data points as defined by Equation 5.15. Then we define $\Delta\mu_{n+1}$ to be the change to the estimated mean induced by the addition of a new data point, as shown in Equation 5.16.

$$\mu_n = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{5.15}$$

$$\Delta\mu_{n+1} = \mu_{n+1} - \mu_n = \frac{x_{n+1} - \mu_n}{n+1} \tag{5.16}$$

Next, we use the superscript to denote a tiling operation such that μ^f_n is the mean vector tiled f times so that it has the same dimension as $\overrightarrow{X}^f_n(i)$. We can then define the normalized values of Φ_n and Ψ_n recursively as shown in Equations 5.17 and 5.18, where S^p_n is the summation defined recursively in Equation 5.19 and S^f_n is defined similarly.

$$\begin{aligned} \Phi_{n+1} = \Phi_n &+ \left(x^f_{n+1} - \mu^f_{n+1}\right)\left(x^p_{n+1} - \mu^p_{n+1}\right)^\top \\ &- \left(\Delta\mu^f_{n+1}\right) S^{p\top}_n - S^f_n \left(\Delta\mu^f_{n+1}\right)^\top \\ &+ n\left(\Delta\mu^f_{n+1}\right)\left(\Delta\mu^p_{n+1}\right)^\top \end{aligned} \tag{5.17}$$

$$\begin{aligned}\Psi_{n+1} &= \Psi_n &+ \left(x^p_{n+1} - \mu^p_{n+1}\right)\left(x^p_{n+1} - \mu^p_{n+1}\right)^\top \\ & & - \left(\Delta\mu^p_{n+1}\right) S_n^{p\top} - S^p_n \left(\Delta\mu^p_{n+1}\right)^\top \\ & & + n\left(\Delta\mu^p_{n+1}\right)\left(\Delta\mu^p_{n+1}\right)^\top\end{aligned} \tag{5.18}$$

$$S^p_{n+1} = \sum_{i=1}^{n+1}\left(X^p_i - \mu^p_{n+1}\right) = S^p_n + n\Delta\mu^p_{n+1} + \left(X^p_{n+1} - \mu^p_{n+1}\right) \tag{5.19}$$

Lastly, normalizing the variance of the data only requires that a running estimate of the variance be kept as each new datum is collected. The estimated variance is updated using Equation 5.20. The final estimates of Φ and Ψ can then be normalized by computing the element-wise division shown in Equation 5.21, where $\otimes$ denotes the outer product and the superscript denotes the tiling operation applied to the standard deviation estimated from the n^{th} variance estimate.

$$\sigma^2_{n+1} = \sigma^2_n + \frac{\left(x_{n+1} - \mu_{n+1}\right)^2 + n\Delta\mu^2_{n+1} - 2\Delta\mu_{n+1}S_n - \sigma^2_n}{n+1} \tag{5.20}$$

$$\overline{\Phi}_n = \frac{\Phi_n}{\sigma^f_n \otimes \sigma^p_n} \qquad \overline{\Psi}_n = \frac{\Psi_n}{\sigma^p_n \otimes \sigma^p_n} \tag{5.21}$$

After this final normalization, the n^{th} estimate of the prediction operator $\mathcal{F}$ can be computed as $\mathcal{F}_n = \overline{\Phi}_n \overline{\Psi}_n^{-1}$. As a final note, this incremental method of computing the MMSE estimate of $\mathcal{F}$ consumes a finite and constant amount of memory; however, the values stored in the summation S_n may diverge since the variance of $S_n = n\sigma^2$ according to the central limit theorem [65]. Fortunately, the divergence of these values occurs linearly with the number of data points and does not risk overflow unless exceedingly large amounts of data are used. Even so, S_n is only ever multiplied by $\Delta_m u_{n+1}$ so it is possible to simply store $\frac{S_n}{n+1}$ and effectively eliminate any risk of storing a divergent variable in memory.

5.2.4 Exploration and Model Structure

Estimation of the primitive model parameters requires a history of observations of the system's inputs and outputs. The production of this history is typically referred to as exploration, which consists of driving the inputs of the system with some random sequence of inputs u_t. In the case of the linear feedback transform employed in this model of sensorimotor primitives, the selection of this exploratory control sequence plays a significant role. In the case that u_t is random and uncorrelated with finite variance, we have that the resultant feedback transform will predictively model the mechanics of the system. Importantly, the feedback transformation imposes a closed loop controller that drives the system based on these observed mechanics. In effect, the default control policy determined by $\mathcal{G}$ for $v_t = h_t$ drives the system toward a point corresponding to the average value of u_t and y_t. Alternatively, if some structure were imposed on the control signal used for exploration, the feedback transformation would characterize this structure as part of the system's mechanics. In essence, by structuring the excitation signal u_t, the default control policy is biased in a manner that matches this structure.

It is also possible to utilize a hierarchy of feedback transforms as depicted in Figure 5.2 where each transform is denoted by $(\mathcal{G}_i, \mathcal{T}_i)$. Each feedback transform can then be learned sequentially through random excitation of each new high level control input. In the case that the original system has multiple output channels, it is possible to map these output channels to different levels of the hierarchy. Furthermore, this approach can be tailored to a particular application by imposing differently structured excitation at each level of the hierarchy.

5.3 Conclusion

In this chapter, we developed a notion of sensorimotor primitives in terms of a feedback transformation. Furthermore, we developed a means of estimating the primitive operators under a linear system assumption and developed a means of dealing with high dimensional systems and large data sets. Importantly, this development of linear sensorimotor primitives is particularly well suited to the problem of coordinative control of a high dimensional simulation of the vocal tract. In the following chapter, we will consider experimental

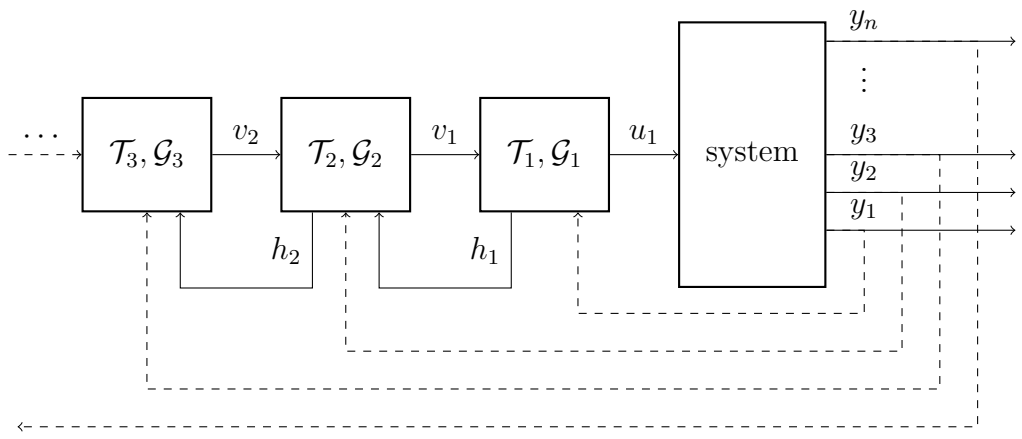


Figure 5.2: The notional implementation of a hierarchy of feedback transformations.

results in applying linear sensorimotor primitives to the Praat vocal tract simulation for the purpose of controlling the vocal tract geometry. We will also explore the problems of learning a high level control policy and estimating the internal state without access to all sensory channels or low level control inputs.

CHAPTER 6

LINEAR FEEDBACK CONTROL

In this chapter, we explore the application of linear sensorimotor primitives to the vocal tract and the development of control policies that interface with the resultant primitive operators. Bearing in mind that the primary focus of this work is to develop a communication model of speech, we focus our attention on the issue of controlling the vocal tract to produce some desired acoustic output. As we will see, the use linear sensorimotor primitives is not, on its own, sufficient for speech communication; however, valuable insights are gained in examining the successes and shortcomings of this approach.

6.1 Linear Articulatory Primitives

We consider the acoustic and articulatory sensory information as separate outputs from the vocal apparatus. Equation 6.1 then represents the model of the system where a_t represents the articulatory sensory output and s_t represents the acoustic sensory output (a for articulatory, s for sound). In general, we will assume that both a_t and s_t are feature vectors that sufficiently represent their respective sensory information. Throughout this chapter, we will explore the hierarchical application of sensorimotor primitives to each of these sensory channels. Specifically, the area function is driven by low level *articulatory primitives* while the acoustic output is driven by high level *acoustic primitives* that interface with the articulatory primitives. Given that the acoustic signal is largely determined by the vocal tract area function, it follows that the proprioceptive sensory channel be the lowest level of the primitive hierarchy.

$$\begin{aligned} z_{t+1} &= f(z_t, u_t) \\ a_t &= g_a(z_t, u_t) \\ s_t &= g_s(z_t, u_t) \end{aligned} \tag{6.1}$$

Let $\overleftarrow{A}_t^p$ be the history vector of a_t and let $\overleftarrow{U}_t^p$ be the history vector of u. Furthermore, let the articulatory input and output history be $\overleftarrow{X}_t^p = \left[\overleftarrow{A}_t^p, \overleftarrow{U}_t^p\right]^\top$. Then we have the feedback transformation corresponding to sensory output, represented by the vocal apparatus area function a_t, defined by Equation 6.2.

$$\begin{aligned} h_t &= \mathcal{T}_a \overleftarrow{X}_t^p \\ u_t &= \mathcal{G}_a\left(v_t\right) \end{aligned} \tag{6.2}$$

In this formulation, the subscript a is used to indicate parameters that correspond to the articulatory feedback transformation. We then seek to learn the set of model parameters $\mathcal{T}_a$ and $\mathcal{G}_a$ such that the vocal tract can be controlled by the high level control input v_t. As a first step in this process, we must first determine a set of articulatory features to include in the sensory output signal a_t.

6.1.1 Articulatory Feature Extraction

The feedback transformation matrices $\mathcal{T}_a$ and $\mathcal{G}_a$ inherently capture a notion of coordinative control by finding correlations between the input and output dimensions of the system. In the case of choosing articulatory features for vocal control, it is important to choose features that are relevant to speech production. It is also important to note that $\mathcal{G}_a$ is effectively a projection matrix that performs linear feature extraction in a manner analogous to principle component analysis. As such, the articulatory features should be chosen such that they carry acoustically relevant information but one should be careful not to present features that have been heavily reduced.

For this experiment, the articulatory features consist of the vocal tract area function, the area function over the lungs, and the cross-sectional area of the velar opening. The area function consists of the cross sectional area of each tube segment starting from the glottis up through the lips and including the opening of the velum. Given that the area functions of the nasal cavity

and bronchial tubes do not change with time (excluding any vibrations of the walls), they are excluded from this set of features.

6.1.2 Random Excitation

The muscle inputs to the vocal tract are excited by applying a random walk over each muscle input with independent increments and independent random time steps. Let $m_i(t)$ be the i^{th} muscle input at time t and let t_0 be the start time for the random walk such that $m_i(t_0) = 0$. We define a sequence of target times t_k by Equation 6.3a where τ_k is a nonnegative random variable. At each target time, the i^{th} muscle target is determined by Equation 6.3b, where δ_k is a random variable. The function m_i is then defined by linear interpolation between the sequence of targets defined by $\{m_i(t_0), m_i(t_1), \ldots\}$. This process is then repeated across all muscle inputs such that the sample paths of all muscle inputs are mutually independent. By defining this random walk, changes in muscle commands are intentionally unsynchronized and can evolve with a bounded random velocity.

$$t_k = t_{k-1} + \tau_k \ , \qquad \tau_k \in [\tau_{min}, \tau_{max}] \tag{6.3a}$$

$$m_i(t_k) = m_i(t_{k-1}) + \delta_k \ , \qquad \delta_k \in [-\delta_{max}, \delta_{max}] \tag{6.3b}$$

Using the random walk defined by Equation 6.3, a random excitation was generated and applied to the Praat vocal tract model for 500 minutes. The random walk parameters for this experiment are defined to be $\delta_{max} = 0.3$ and $(\tau_{min}, \tau_{max}) = (0.02, 0.2)$, where τ is in seconds. The data collected from the simulation is then sampled at a sampling period of 10ms, which is the Nyquist rate corresponding to τ_{min}. The resultant muscle commands from the first 5 seconds of data are shown in Figure 6.1, with the trace corresponding to the *lung* input highlighted. The mean and relative standard deviation of each articulator are shown in Figure 6.2. Figure 6.3 shows a plot of the mean and relative standard deviation of the output features based on the random sequence of muscle inputs. The mean and variance of the articulator inputs are shown as a rough measure of whether or not enough input samples are taken since the mean and standard deviation will converge to $\frac{1}{2}$ and $\frac{1}{\sqrt{12}}$ respectively as the sample size increases.

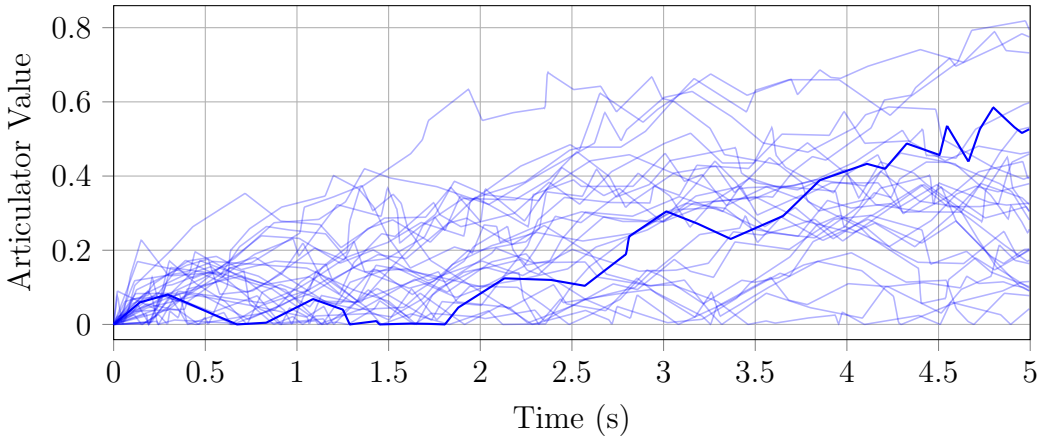


Figure 6.1: The first 5 seconds of each articulator input with the *lung* input is highlighted.

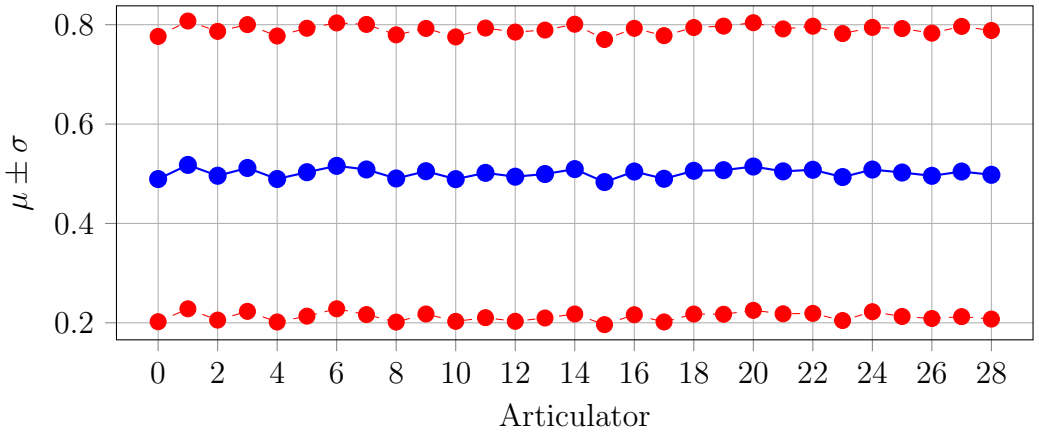
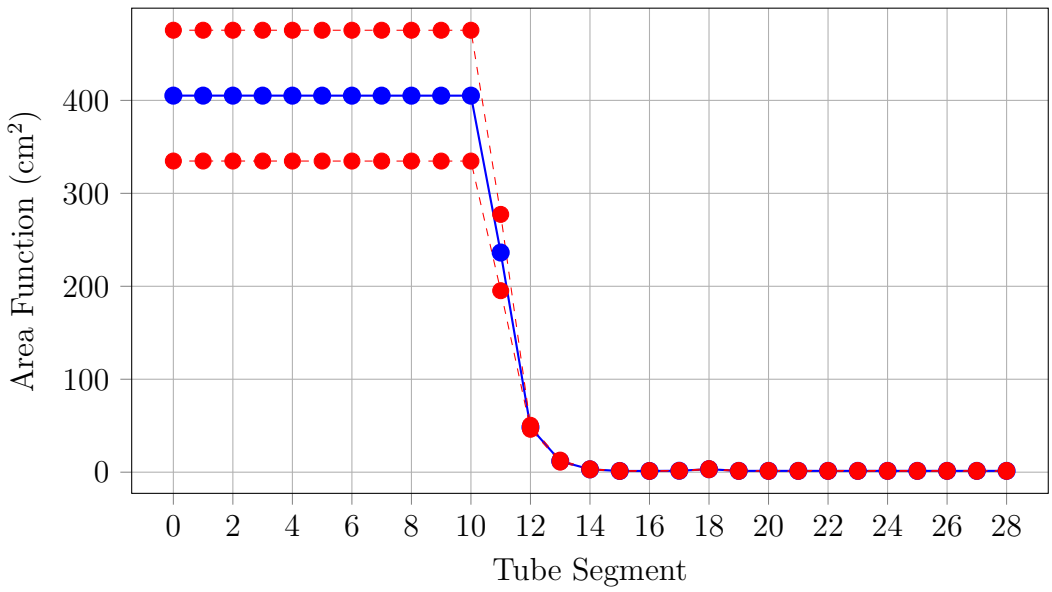
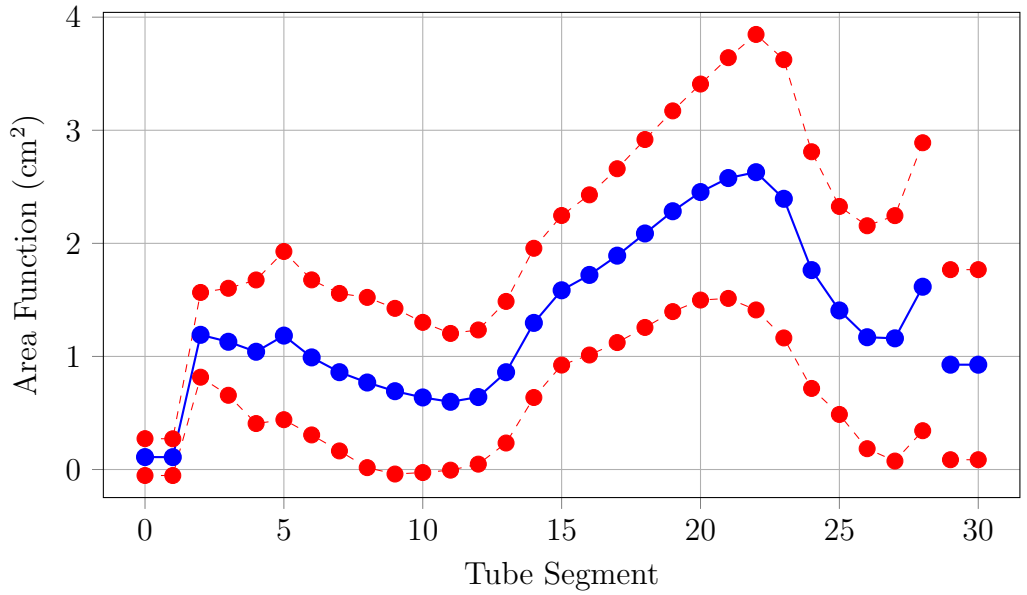


Figure 6.2: The average value of each articulator input with a margin of one standard deviation.



(a) Lungs through trachea



(b) Glottis through lips followed by velar opening

Figure 6.3: The cross-sectional area of the lungs through the trachea, and of the glottis through the lips followed by the velar opening.

6.1.3 Parameter Inference

The articulatory primitive operators are estimated based on the input and output data from the vocal tract under random excitation. Given that the space of possible inputs is quite large, a very large number of samples is needed in order to properly estimate the primitive operators. Fortunately, the method outlined in the previous chapter allows an MMSE estimate of the primitive operators based on an arbitrarily large data set; the only practical limitation is computation time. As such, the primitives are estimated based on a total of 500 minutes of vocal tract data, with input output pairs captured at a frequency of 1 kHz. In total, the primitives are estimated from 30 million input-output vectors.

The history and prediction length were chosen to be $p = 20$ms and $f = 10$ms. It should be noted that the necessary history length is given by $2p_{AIC} \approx 12$ms, which is given by the minimum value indicated in Figure 6.4. Conversely, a longer history results in a smoother estimate of the internal state. As such, p is selected to be as large as possible without significantly impeding computation time. The dimension of h is chosen to be 10; the selection of this parameter essentially determines the primitive controllers resolution.

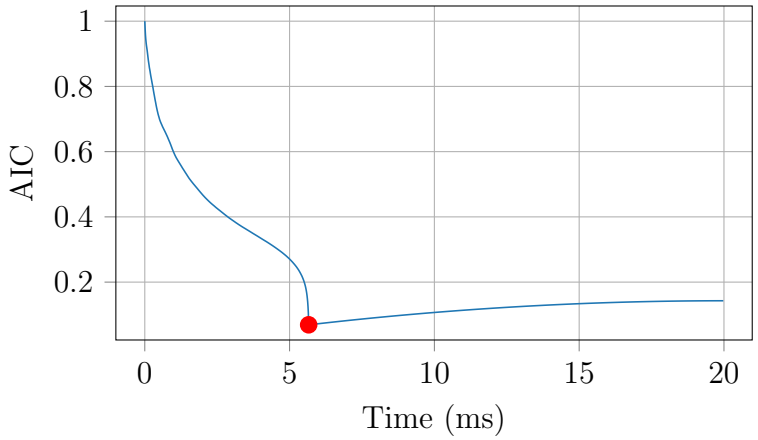


Figure 6.4: The computed AIC value as a function of time computed from a 100s data sample. The minimum value is indicated with a red dot.

The process of estimating the sensorimotor primitives is performed iteratively as the input-output data are generated. In order to judge the stability of the parameter estimates, we iteratively compute the prediction operator $\mathcal{F}_n$ using n minutes worth of simulated data. The relative change in $\mathcal{F}_n$,

defined in Equation 6.4, is then computed as each new batch of data is generated. Additionally, the mean and variance of the elements of $\mathcal{F}_n$ were computed as a means of monitoring the relative change in the matrix over time. Plots of these measurements are shown in Figures 6.5 and 6.6, illustrating the convergence of the prediction operator toward a fixed point value. Notably, the relative change in $\mathcal{F}$ is still rather noisy even after 400 minutes of data have been collected. Arguably, a much larger set of data would lead to a more consistent estimate but the time needed to simulate such data is prohibitively long. Even in the case of this data set, the total computation time was approximately 5 days when run on a standard desktop computer.

$$\Delta\mathcal{F}_n = \frac{\|\mathcal{F}_n - \mathcal{F}_{n-1}\|}{\|\mathcal{F}_n + \mathcal{F}_{n-1}\|} \tag{6.4}$$

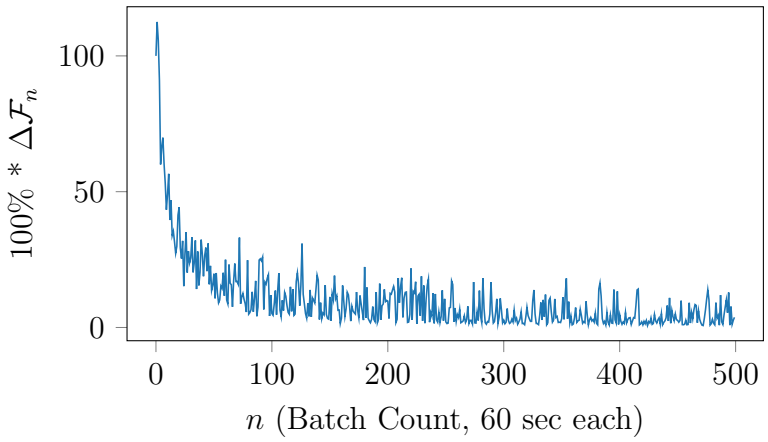
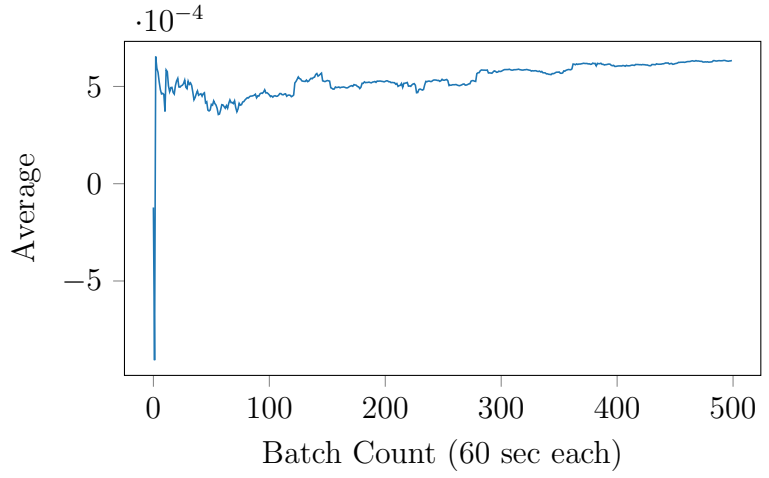
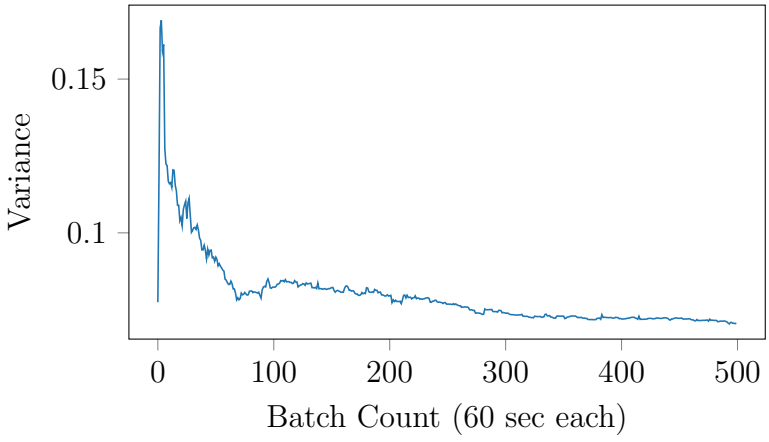


Figure 6.5: Percent change in prediction operator as a function of total data size.

The resultant feedback transform corresponding to the first three dimensions of h is visualized in Figures 6.7 and 6.8. Each parameter in these figures is labeled with a value of k, which defines the corresponding dimension of h. The input parameters can be interpreted as a projection matrix for a linear transform, where the time variation along each feature can be thought of as capturing a distribution of frequencies. The output parameters, on the other hand are more readily interpreted as indicating correlation between articulator inputs and sensory outputs. Although the structures of the input and output matrices are interesting to examine, the interpretation of these results is largely subjective. Instead, we will examine the behavior of the



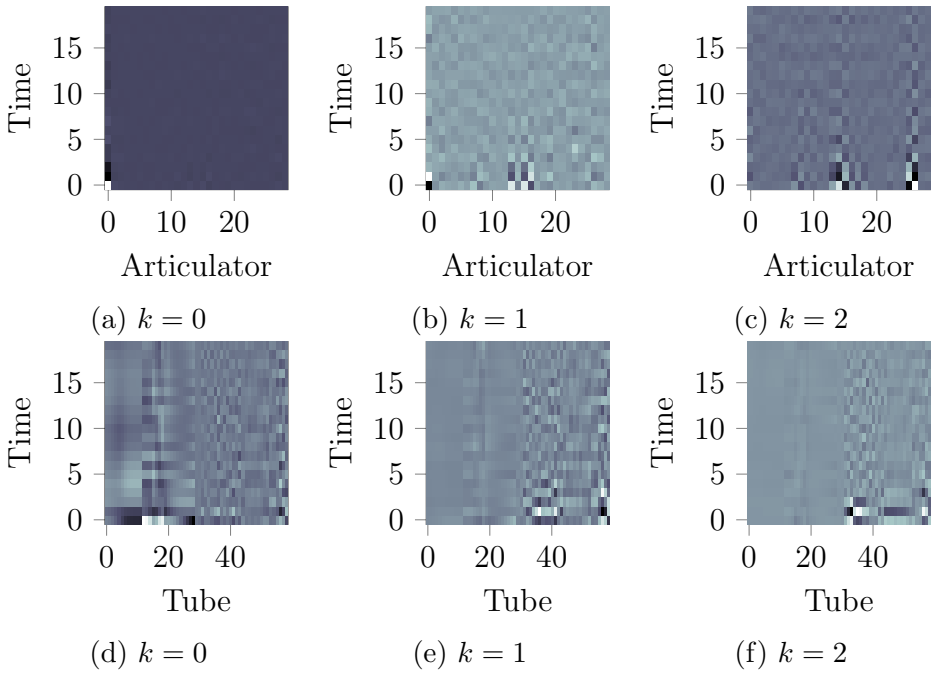
(a)



(b)

Figure 6.6: Statistics computed for the elements of the prediction operator $\mathcal{F}_n$.

vocal tract under the application of this feedback transformation.



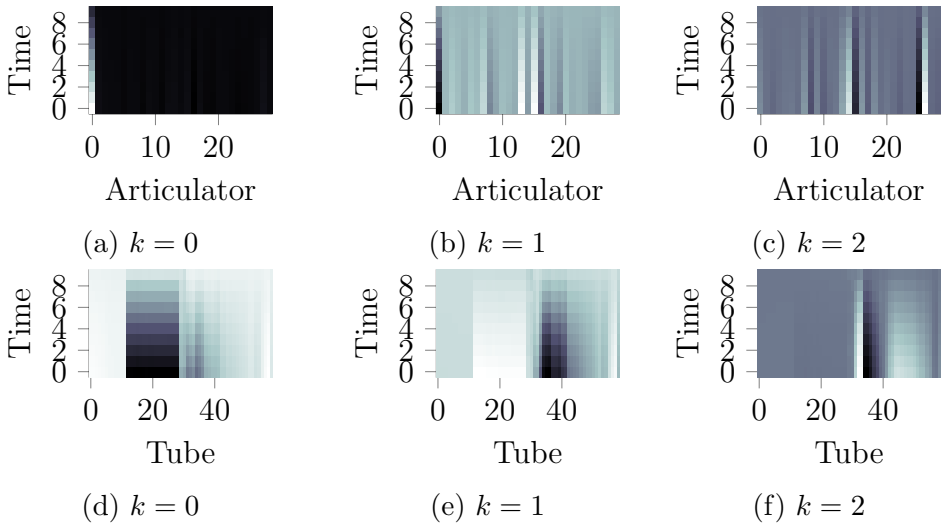
(a) $k = 0$ (b) $k = 1$ (c) $k = 2$

(d) $k = 0$ (e) $k = 1$ (f) $k = 2$

Figure 6.7: Input features extracted from $\mathcal{T}$.

6.2 Articulatory Feedback Control

With an articulatory primitive model in hand, we now consider the problem of feedback control. In order to implement this feedback control, we first must consider the nature of the vocal tract model being employed. In particular, the Praat model is designed such that it adheres to the equilibrium point hypothesis [59]. In terms of feedback control, this effectively means that any given state in the system will act as an equilibrium point so long as the control input is held constant. As a result, controller design is greatly simplified in that state feedback is not needed to stabilize the system. This means that we simply need to learn a feedback controller that is able to steer an already stable system.



(a) $k = 0$ (b) $k = 1$ (c) $k = 2$

(d) $k = 0$ (e) $k = 1$ (f) $k = 2$

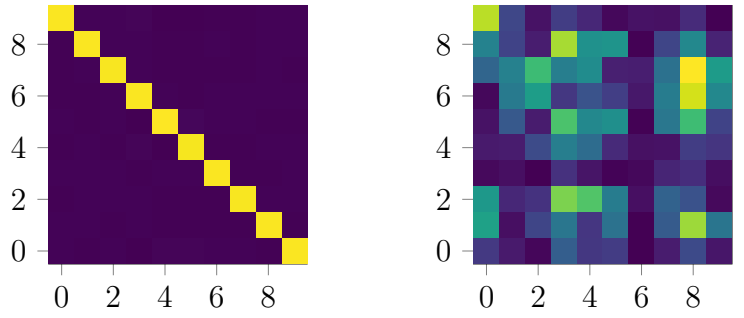
Figure 6.8: Output features extracted from $\mathcal{G}$.

6.2.1 Linear State Feedback Control

A first approach at feedback control is to apply the standard method of linear quadratic regulation. In order to implement this control method, we consider the articulatory primitive model in terms of a simple linear system as shown in Equation 6.5, where h_t is the internal primitive state and v_t is the control input, A represents the state evolution matrix, and B is the control input matrix. Notably, the values of A and B are unknown and must be estimated before any control can be applied. We do this in a manner similar to the process used to estimate the primitive operators, namely by driving the input with a random walk and finding the MMSE operators that best fit the observed state evolution. Given that the primitive operators give us access to the internal state, this estimation is straightforward. The resulting A and B matrices are shown in Figure 6.9. As a form of sanity check, we see that the state matrix A is estimated as the identity matrix, which gives some confidence in the quality of these estimates.

$$h_{t+1} = Ah_t + Bv_t \tag{6.5}$$

State feedback can then be defined by the rule shown in Equations 6.6 and 6.7, where K is the feedback gain and $\tilde{h}$ is the desired state. The feedback gain matrix K can then be determined based on the algebraic Riccati equation (ARE) shown in Equation 6.8, where R and Q represent the vari-



(a) State transition matrix (b) Control input matrix

Figure 6.9: A simple visualization of the results of performing system identification on the primitive controlled vocal tract.

ance of the input and state matrices.

$$v_t = -K\left(h_t - \tilde{h}\right) \tag{6.6}$$

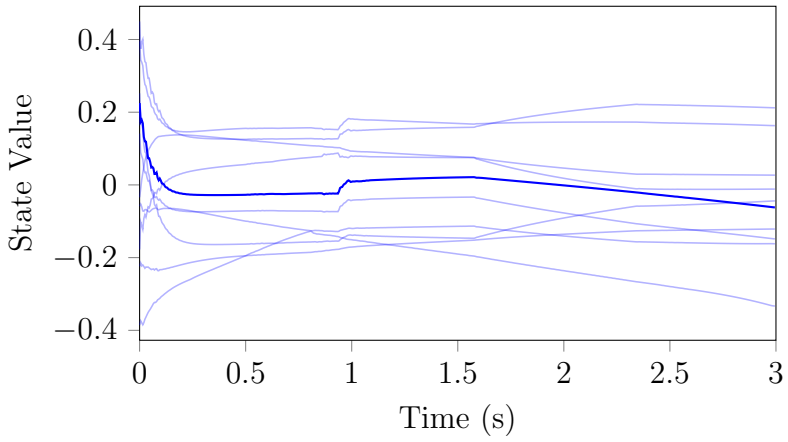
$$K = \left(R + B^\top PB\right)^{-1} B^\top PA \tag{6.7}$$

$$A^\top PA - P - \left(A^\top PB\right)\left(R + B^\top PB\right)^{-1}\left(B^\top PA\right) + Q = 0 \tag{6.8}$$

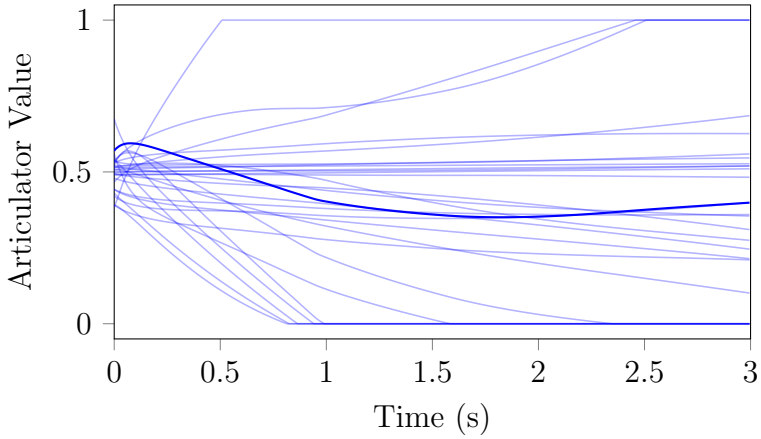
This feedback rule is applied to the primitive model and all dimensions of the internal state toward zero. The resulting state evolution and articulator inputs are shown in Figure 6.10. As we can see, the feedback controller largely fails at driving the states toward zero. This result is hardly surprising as the assumption that the vocal tract is a linear system does not actually hold. In particular, we see that the controller does begin to drive the states toward zero but begins to fail as soon as the articulators begin to hit their boundaries of their input range of $[0, 1]$. Although this result fails to give a method of controlling the vocal tract toward a desired state, it clearly shows that linear state feedback is insufficient.

6.2.2 Reinforcement Learning

An alternative approach to learning a control policy for the articulatory primitive model is through reinforcement learning. We can formulate the problem of control in terms of Q-learning by defining a reward function that



(a) Internal state of the primitive controller



(b) Articulator inputs to the

Figure 6.10: Traces of the state evolution and articulator inputs under LQR feedback control.

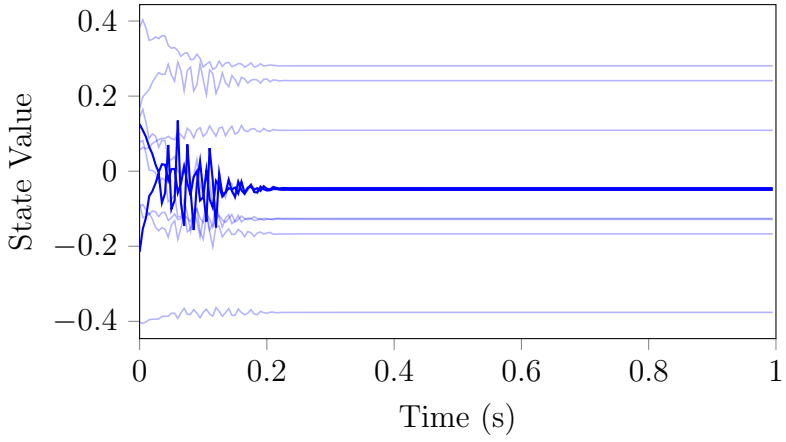
provides positive reward at the desired state and null or negative reward at undesired states. The policy learner then accumulates a 'value' associated with all state-action pairs. Since the states and actions of the primitive model are continuous, Q-learning requires that the Q-function be parameterized through some form of function approximation. For the sake of simplicity, we will consider Q-learning over a quantized version of the state action space.

Q-learning was applied to two dimensions of the state and action spaces respectively (i.e. two-dimensional state space, two-dimensional action space). The unused dimensions of the sensorimotor primitive inputs are held constant at a value of zero, while the first two dimensions are driven by the controller. The discrete action space is then defined by $\{-0.1, 0, 0.1\}$ and the primitive control input is incremented by the selected action value. Each dimension of the primitive state space is quantized into 11 bins evenly spaced over the range of $[-1, 1]$.

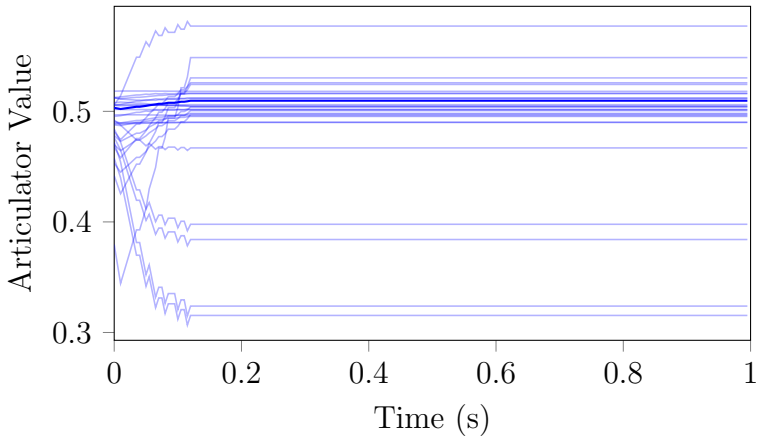
The choice of implementing this controller over only two dimensions stems from the fact that the space of state-action pairs grows exponentially with each new dimension. As such, the time needed to learn a control policy grows exponentially with each new dimension. The choice of two dimensions is sufficient to show that a policy can be learned to simultaneously drive two states to a desired point in the state space. Figure 6.11 illustrates the state evolution and articulator inputs determined by a policy learned after 100 seconds of training, with a controller operating at a sampling period of 5ms. As can be seen, the controller is capable of driving the two internal states toward the bin centered at zero, and no further change in control input is applied once the states reach the target.

This discretized Q-learning based controller demonstrates that a nonlinear control policy can be learned for driving the state space of the primitive controller. Unfortunately, dimensionality is a nontrivial hurdle for this approach. In this simplistic implementation, the space of state-action pairs is quantized into 33^n values (where n is the primitive dimension). As such, the number of training iterations needed to train the policy grows prohibitively large for $n > 3$. In addition the Q-function is only trained to drive the primitive state toward a single point. This means a new controller must be learned for each target state, which further compounds the high computational cost.

Lastly, this approach suffers from a subtle problem in the nature of the primitive states. Specifically, it is not necessarily possible to tell whether



(a) Internal state of the primitive controller



(b) Articulator inputs to the Pyraat simulator

Figure 6.11: Traces of the state evolution and articulator inputs under Q-learning based control policy.

a particular state is reachable. This is primarily due to the fact that the states are not independent of one another and as such, driving one state to a particular value may prohibit another state from reaching some other range of values. This is shown in Figure 6.12, where the control policy is learned for driving three primitive states toward $(0, 0, 0)$. As we can see, the controller is able to drive each state to the desired target before one state is ejected from the goal and the policy is unable to drive it back. A key lesson learned here is that we must first discover the set of valid target states before we can learn a control policy.

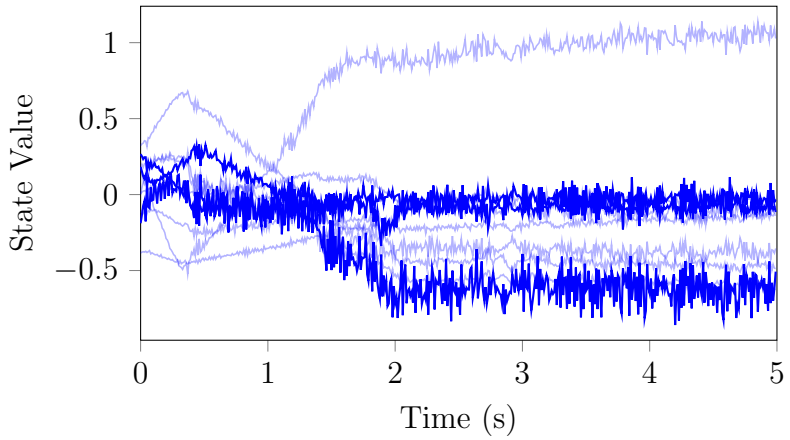


Figure 6.12: Traces of the state evolution under 3 state Q-learning control policy.

6.3 Linear Acoustic Primitives

Having demonstrated some efficacy in the use of linear sensorimotor primitives for articulatory control, we will now briefly explore their application to controlling the acoustic output of the vocal tract. To do this, we will employ mel frequency cepstral coefficients (MFCCs) as an approximation of the acoustic features extracted by the cochlea. The choice of MFCCs is primarily motivated by the fact that MFCCs are inspired by the frequency response observed in the cochlea. In addition, MFCCs have been extensively used in speech recognition technologies as they have been shown to be highly effective for the purpose of classifying speech sounds [66][67]. With this reasoning, we will employ MFCC features as a means of extracting the frequency content

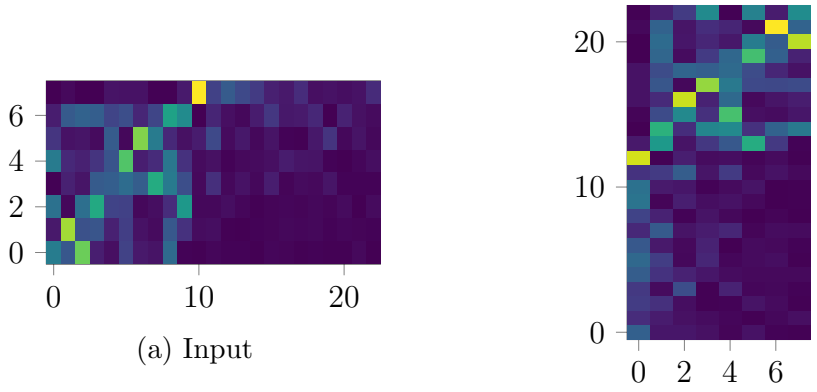
of the acoustic signal produced by the vocal tract.

6.3.1 Parameter Estimation

We implement sensorimotor primitives by defining the system inputs as the articulatory primitive inputs and the outputs as the MFCC features. The primitive model can then be trained using the same parameter estimation procedure outlined in Chapter 5. Training was performed using 500 minutes of training data randomly generated by randomly exciting the articulatory primitives as the control inputs. We discover, however, that the resulting primitive model does not sufficiently represent the system for the purpose of communication. To visualize how this approach fails, we can look at a simple example.

We consider the acoustic primitive operators in the case of a history and prediction length set to $p = f = 1$, which is chosen simply because the primitive operators can be easily visualized. The input controls to the system are the inputs to the articulatory primitives described earlier in this chapter and the outputs are 13 MFCC feature vectors. The estimated input and output operators for the acoustic primitives are shown in Figure 6.13. Since the history lengths are set to unity, it is easy to interpret all dimensions of the operators simultaneously. In particular, we see that the MFCC components of operators primarily contribute to the zeroth state of the new primitive space. As a result, these acoustic primitives make extremely poor acoustic observers if they do not have access to the control inputs of the vocal tract. When this is attempted, the estimated state is restricted to a very small neighborhood around the origin.

The failure of this model is likely due to two primary factors. First is that the linear assumption of the sensorimotor primitive estimation process is strongly violated by the relationship between the primitive inputs and MFCC features. Second is that there appears to be only a very small subset of states in the primitive state space that correspond to sound production. As such, the random stimulation largely produces MFCC features that correspond to little or no noise at all. This poses a challenge that calls for nonlinear methods that can more readily deal with the restricted distribution of sound producing states.



(a) Input

(b) Output

Figure 6.13: Input and output operators of the linear acoustic primitive model. In each matrix visualization, the first 10 columns/rows represent the articulatory primitive control action and the remaining columns/rows represent MFCC values.

6.4 Considerations for Further Development

Having applied linear sensorimotor primitives to the Praat model, several interesting lessons have been learned. As was previously noted, the mapping from vocal tract shape onto acoustic features is generally nonlinear and so the utility of linear primitives was expected to be limited. In addition, there were other observations that play into their failure to represent the acoustic communication channel. First is the fact that most of the vocal tract states (as represented by the articulatory primitives) do not produce any audible sound. Related to this issue is the problem of learning a nonlinear control policy that requires *a priori* knowledge of the desired state. In this section, we will consider each of these problems and their implications for this approach to developing a control theoretic model of speech communication.

6.4.1 The Audible State Space

We define the *audible subspace* as the subset of the articulatory primitive state space which coincides with the production of audible acoustic waveform. As noted previously, during random excitation of the vocal tract, sound is only produced during a comparatively small number of simulations. Given the poor performance of the linear acoustic primitives, this raises the question as to the size of the audible subspace relative to all possible states of the vocal

tract. In addition, some primitive states can only be reached dynamically and thus the system can only pass through them as part of some trajectory.

In order to get a sense of the relative size of the audible subspace, we can consider the total amount of time in which the vocal apparatus produced sound during random excitation. During the training process used to estimate the acoustic primitives discussed in Section 6.3, we tallied up the total time in which the energy of the acoustic waveform crossed a threshold of 10^{-3}. The choice of threshold is selected based on perception of the author while playing the utterance over a set of computer speakers. From the 500 minutes of utterances, approximately 17 minutes and 21 seconds were above the threshold; approximately 3.5%. Given the relative size of the audible subspace, it is unsurprising that the linear sensorimotor primitives are primarily influenced by non-acoustic features.

6.4.2 Control Policy Representation and Learning

Although Q-learning was demonstrated as a basic means of learning a control policy to drive the articulatory primitives, there are a couple of important ways in which this approach could be improved. Foremost is the observation that the discretized formulation of Q-learning that was applied in Section 6.2.2 is a relatively crude implementation of reinforcement learning. In general, we would expect more efficient policy learning if the control policy were represented using a parameterized continuous function.

The larger issue in this approach, however, is that reinforcement learning generally requires the definition of some form of goal state. Given that communication is the fundamental goal of this work, the system would first need to discover the set of goal states before the control policies could be learned. From a communication perspective, the goal states would need to be some set of states that can be distinguished from one another by an observer. Given that linear sensorimotor primitives were shown to make poor acoustic observers, we still need to divine a (probably nonlinear) observer to match with this control policy.

How, then, should one proceed with learning an observer and control policy? In order to address this question, we must revisit our approach to defining speech primitives in order to extend our formulation to the trans-

mission and reception of the acoustic signal. As we will show in the next two chapters, we can solve the problem of the acoustic observer and the high level control policy simultaneously by framing the problem as nonlinear communication channel and applying our sensorimotor primitive framework.

CHAPTER 7

NONLINEAR PRIMITIVES FOR COMMUNICATION

In this chapter, we reconsider our development of speech primitives in order to address the problem of controlling the acoustic signal. As shown in Chapter 6, linear sensorimotor primitives fall short of our needs in constructing a speech communication channel. A fundamental flaw in this approach is ultimately in the construction of an acoustic observer. Although it seems natural to assume that one might be able to construct an observer using an architecture analogous to the controller, the linear systems approach is insufficient for accomplishing this. In order to develop an observer, then, we reformulate the problem as a communication channel rather than a control system.

In this chapter, we formally define the approach to learning primitive operators for transmitting information over a nonlinear communication channel. We specifically consider the path from the inputs to the vocal tract through the output features produced by the cochlea as a channel through which we wish to transmit information. The task is then to learn a pair of operators that map from some space of primitive features through the channel and back onto the primitive features. In essence, we seek to develop a pair of nonlinear encoder and decoder operators that map through a nonlinear channel. Throughout this chapter, we will develop the *inverse channel encoder* (ICE) as a means of learning these nonlinear operators simultaneously by extending the notion of the autoencoder neural network. We then illustrate the behavior of such a mapping by considering a simplified example using written digits as a channel.

7.1 The Autoencoder

We begin our development of the ICE by briefly outlining the traditional *autoencoder neural network* from the perspective of operator theory. Generally

speaking, the autoencoder is implemented in the form of a neural network for the purpose of unsupervised feature learning. Rather than present the concept strictly as a neural network architecture, we will first consider the generalization of the autoencoder as a form of operator decomposition. This generalization is significant in that it more naturally leads to an intuitive development of the ICE architecture.

7.1.1 An Operational Definition

Suppose we have some set of data $X = \{x_0, x_1, \ldots x_N\}$, where each x_i is a vector in some high dimensional space $\mathbb{X}$. Then we define a latent space $\mathbb{H}$ such that $\mathbb{H}$ is constrained in a manner that is useful to the application of interest, e.g. low dimensionality or sparsity are commonly used. An autoencoder is then defined by two operators: the encoder $\mathcal{E} : \mathbb{X} \to \mathbb{H}$ that maps the data onto the latent space, and the decoder $\mathcal{D} : \mathbb{H} \to \mathbb{X}$ that maps the latent space back onto the original data space. The goal is then to learn operators such that $x \approx \hat{x}_i = \mathcal{D} \circ \mathcal{E}(x_i)\ \forall i$. More formally, we seek operators such that $d(X, \hat{X})$ according to some distance metric d. The optimal encoder and decoder operators are then defined by the solution to the nonconvex optimization shown in Equation 7.1.

$$\begin{aligned} \mathcal{D}^*, \mathcal{E}^* = \arg\min_{\mathcal{D},\mathcal{E}} & \, d(X, \hat{X}) \\ \text{s.t.} \quad & \mathcal{E} : \mathbb{X} \to \mathbb{H} \\ \text{and} \quad & \mathcal{D} : \mathbb{H} \to \mathbb{X} \end{aligned} \tag{7.1}$$

In Figure 7.1, we visualize $\mathcal{E}$ and $\mathcal{D}$ as a nonlinear decomposition of the identity operator, denoted by $\mathcal{I}$. This decomposition is conceptually analogous to linear decomposition such as principle component analysis or independent component analysis. Once the operators $\mathcal{E}$ and $\mathcal{D}$ have been learned, the latent space $\mathbb{H}$ serves as a representation of the data that is both *analytic* and *generative*.

It is important to note, however, that unless the topology of $\mathbb{H}$ is somehow constrained, the mapping from the latent space onto the data space may be poorly behaved. In particular, the topology of $\mathbb{H}$ may contain discontinuities or be highly nonconvex in the unconstrained case. As as result, this basic form of the autoencoder is typically not used as a generative model, since

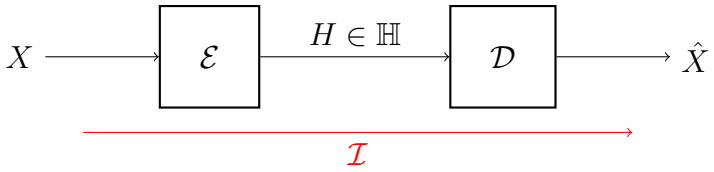


Figure 7.1: The notional form of the Channel Autoencoder.

much of the space in $\mathbb{H}$ does not map into points that bear any meaning in the data space $\mathbb{X}$.

7.1.2 Approximation Using Neural Networks

Given that the optimization shown in Equation 7.1 is in general neither convex nor are the solutions unique, it cannot be solved analytically. Instead, a sub-optimal $\mathcal{E}$ and $\mathcal{D}$ can be learned through using a special architecture of neural network. Although feed-forward neural networks are generally well suited for function approximation problems, they are especially powerful tools for this particular application because their layered structure allows for a straightforward means of function decomposition. Specifically, the network is trained to approximate the identity operator $\mathcal{I} : \mathbb{X} \to \mathbb{X}$ with a special "constraint layer" placed at the middle of the network. Once the network is trained, the identity operator can be easily decomposed by 'cutting' the network at the constraint layer and using each half of the neural net to approximate both $\mathcal{E}$ and $\mathcal{D}$. An example of such a network architecture is shown in Figure 7.2, where the network maps from a four-dimensional (4D) space through a two-dimensional space and then approximates the original 4D input. The dimension of latent space and the width and depth of the network are entirely problem-specific design choices, but it is important that symmetry is maintained between the architecture of $\mathcal{D}$ and $\mathcal{E}$ so that the two networks have equivalent representational power as function approximators.

Interestingly, the general structure of this neural network architecture has been employed for nonlinear manifold learning and unsupervised feature learning since as early as 1987 [68]. With the more modern advances in neural network research, autoencoders have been employed using much deeper and often more advanced architectures with many applications in high dimensional domains, such as image processing and motion learning

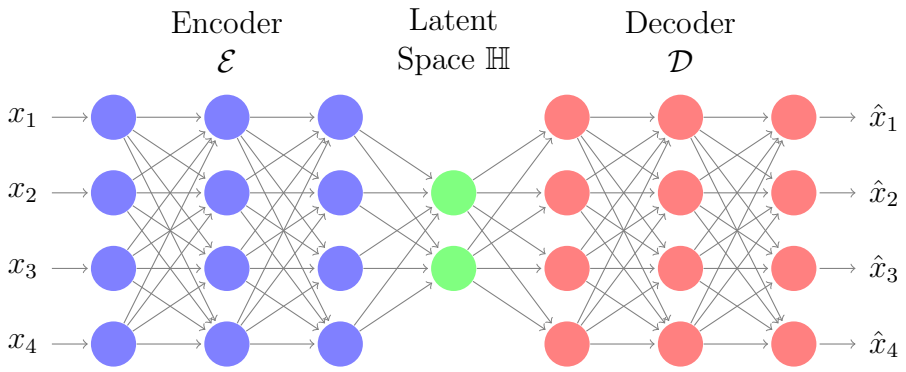


Figure 7.2: Example autoencoder network mapping from $x \in \mathbb{R}^4$ through $h \in \mathbb{R}^2$.

[69][70]. As stated previously, this general form of the autoencoder, though well suited for unsupervised feature learning, performs rather poorly as a generative model. It should be noted, however, that a generative model is of central importance in forming a model of primitive operators for communication. In the following section, we will consider extensions to the autoencoder architecture through specific choices in the nature of the constraint layer as a means of controlling the topology of the latent space.

7.2 Constraining the Latent Topology

In the case of a traditional autoencoder, the data with which the network was trained may map through any arbitrary subset of $\mathbb{H}$. As a result, the points in $\mathbb{H}$ that do not correspond to the training data may map to arbitrary points in $\mathbb{X}$ that have no meaningful relationship with the training data. A consequence of this behavior is that the decoder $\mathcal{D}$ is not well suited as a generative model of the larger population of data from which the training data is taken. In order to solve this problem, we can impose a constraint on the topology of $\mathbb{H}$ during the training process. The variational autoencoder (VAE) solves this problem by imposing an additional constraint on the distribution of points in $\mathbb{H}$ while the network is trained. This VAE architecture can then be further extended into the Channel Autoencoder (ChAE), which directly uses a fixed channel model as the constraint layer in the network.

7.2.1 The Variational Autoencoder

In order to best explain the structure of the VAE, we return to thinking in terms of nonlinear operators rather than neural network architecture. Suppose that each x_i is a Gaussian random variable with a distribution given by $\mathcal{N}(\mu_x, \Sigma_x)$. Then the encoder is chosen such that the $h_i = \mathcal{E}(x_i)$ is a Gaussian random variable with standard normal distribution $\mathcal{N}(0, I)$. In addition, the decoder must be chosen such that for any h_i sampled from a standard normal, we have that $\hat{x}_i = \mathcal{D}(h_i)$ is a random variable with a distribution $\mathcal{N}(\mu_x, \Sigma_x)$. In general, it is possible to impose other distributions on these random variables but the Gaussian is generally the easiest to work with. Under these constraints it is possible to use $\mathcal{D}$ as a generative model that maps from points not seen in the training data onto the distribution $\mathcal{N}(\mu_x, \Sigma_x)$ [71].

In order to construct such a network, we first consider how the Gaussianity constraint is imposed on the latent space and define the objective function for which the network will be optimized. Figure 7.3 depicts the general architecture of the VAE when considered at the operator level where the Gaussianity constraint is imposed using what is known as the 'reparameterization trick'. The 'reparameterization trick' can be thought of in two parts. First, the output of the encoder is defined to be the parameters of a Gaussian distribution, namely the mean and variance. A random sample is then drawn from that Gaussian distribution and used as input to the decoder. The random sampling is accomplished by converting a random sample ϵ from a standard normal via multiplication and addition with the standard deviation and mean respectively.

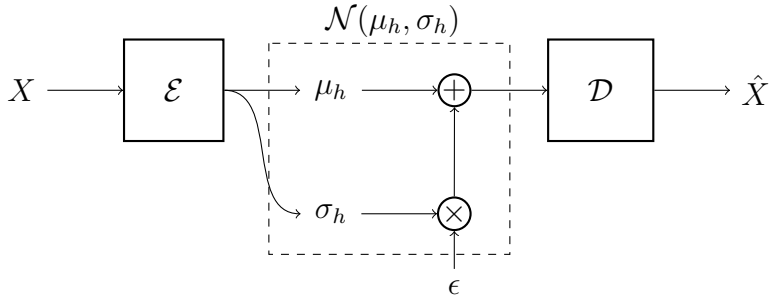


Figure 7.3: An implementation of the VAE using the 'reparameterization trick'.

The objective function that this network is trained to maximize is given by L in Equation 7.2, where the first term can be thought of as the reconstruction cost and the second imposes the Gaussianity constraint on h. Specifically, if X is Guassian then $P(X|h)$ is maximized when $\|X - \hat{X}\|^2$ is minimized. Additionally, since we assume that h must be a standard Gaussian then the second term is minimized when $\mathcal{E}(x)$ is also a standard Gaussian.

$$L = \mathbb{E}_h \left[\log(P(X|h))\right] - \mathrm{KL}\left[P(h|X, \mathcal{E}) \| P(h)\right] \tag{7.2}$$

Because of the randomness induced in the reparameterization trick, the encoder and decoder operators are robust to samples that may not have been seen in the original training set. It is important to note, however, that enforcing this topology on the latent space comes with a direct trade-off in the model's accuracy in its approximation of the identity operator. This commonly manifests in the form of 'fuzzy' outputs from the decoder. This trade-off can be more explicitly seen in the more general implementation of VAE known as the β-VAE, in which a weighting variable β is inserted into the objections function shown in Equation 7.3. The extra weighting variable allows for control of the balance between Gaussianity and reconstruction error; however, the selection of a value of β cannot be determined analytically before training and instead must be found through trial and error [72].

$$L = \mathbb{E}_h \left[\log(P(X|h))\right] - \beta \cdot \mathrm{KL}\left[P(h|X, \mathcal{E}) \| P(h)\right] \tag{7.3}$$

7.2.2 The Channel Autoencoder

We now extend the concept of the VAE by observing that the 'reparameterization trick' used in the constraint layer can be further generalized. Specifically, the reparameterization trick is the implementation of a differentiable operator[1] that is parameterized by the output of the encoder operator. This concept can then be extended by noting that the constraint layer can be replaced with any other operator for which the gradient can be analytically

[1]The differentiability of the operator is of particular importance because the gradient is necessary for the backpropagation algorithm that is used to train the weights of the neural net through gradient descent. It is possible to train the network using gradient-free optimization methods; however, such algorithms are computationally much more expensive and typically grow exponentially with network depth [73].

solved. The general architecture of this *Channel Autoencoder* (ChAE) is depicted as the graphical model shown in Figure 7.4.

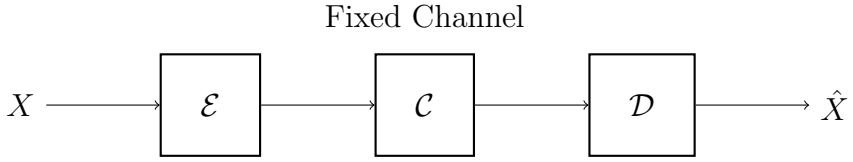


Figure 7.4: The notional form of the Channel Autoencoder.

This notion of the Channel Autoencoder has been applied to a simple RF channel model where the encoder and decoder are trained to map a binary vector through the channel [74]. Using a convolutional network architecture, this application of the Channel Autoencoder formed a nonlinear mapping that blurred the lines between modulation and coding in order to operate near the channel capacity of an arbitrary differentiable channel. Additionally, recurrent neural nets were employed in a similar architecture to learn codes for feedback channels [75][76][77]. Although these results are compelling, the requirement that the gradient of the channel be known *a priori* limits the application of this direct approach rather severely. Given that our model of speech production is not easily differentiable, we seek a method of extending this notion of the Channel Autoencoder to channel models for which the gradient is *unknown*.

7.3 Generalized Channel Autoencoder

In the case of utilizing the vocal tract as the channel operator, a closed form solution to the gradient cannot be directly computed. As such, we seek an alternative formulation of the problem that will still allow the neural network to be efficiently trained via back-propagation. In order to do this, we once again return to thinking about the problem in terms of operator decomposition rather than thinking in terms of neural network architecture. In the channel autoencoder, the neural net learns a composition of operators (namely the encoder, channel, and decoder) that approximate the identity operator denoted by $\mathcal{I}$. We can then graphically represent the mappings between these operators as loop shown in Figure 7.5.

The important observation in this diagram is that the inverse of each

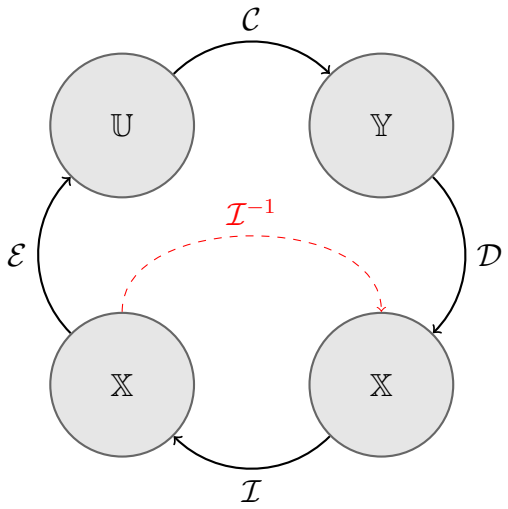


Figure 7.5: A visualization of the operator mappings employed in the channel autoencoder with the operators indicated by arrows mapping between mathematical spaces.

operator can be approximated as a composition of other three. In the case of the channel autoencoder, the identity operator is its own inverse and so this relationship is not as readily apparent. Importantly, so long as the network architecture is appropriately constructed, it is possible to learn the encoder and decoder operators by approximating the inverse of either the identity operator or the *channel*. The latter approach is the primary contribution of this chapter and outlined in further detail in the following sections.

7.3.1 The Inverse Channel Encoder

From this operator perspective, the ChAE is trained using the inputs and outputs of the identity operator in order to learn its inverse. The network can then be thought of as breaking the loop shown in Figure 7.5 at the identity operator to form the feed-forward network represented by Figure 7.4. Given that we are interested in a channel for which the gradient is unknown, back propagation cannot be used to estimate the encoder and decoder operators directly. Instead, we can break the loop at the channel itself and learn an approximation of the channel inverse as a composition of the opposing operators $\mathcal{C}^{-1} \approx \mathcal{D} \circ \mathcal{I} \circ \mathcal{E}$. With this form of feed-forward network, we can then impose the Gaussian constraint on the latent space that exists between the decoder and encoder operators in a manner equivalent to the traditional

VAE.

By learning the inverse channel model using this VAE-like architecture, we have that the encoder maps the output of the channel onto a Gaussian distribution and the decoder maps that Gaussian distribution back onto the inputs of the channel. As a result, the communication channel that is formed by the composition $\hat{h} = \mathcal{E} \circ \mathcal{C} \circ \mathcal{D}(h)$ is effectively a Gaussian communication channel over the latent space $\mathbb{H}$. The resulting network illustrated in Figure 7.6 is computationally very similar to the traditional VAE with the exception that the overall network is trained to learn the inverse channel model rather than the identity operator.

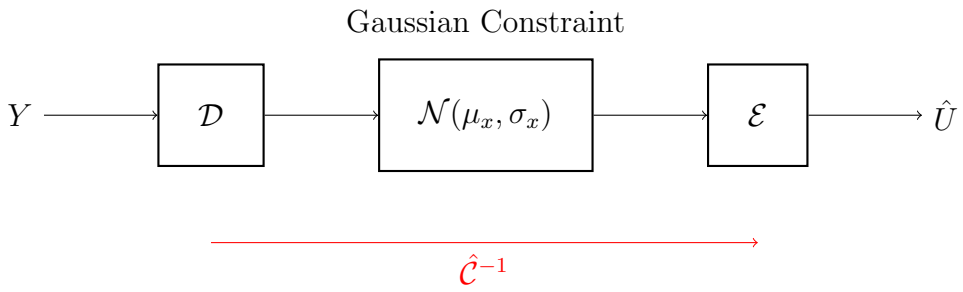


Figure 7.6: The notional form of the Inverse Channel Autoencoder.

Notably, there is no need to know anything analytic about the channel's internal structure, although this information is relevant to choosing a neural network architecture. All that is necessary for training this model is access to sufficient training data consisting of samples taken from the input and output of the desired channel. Lastly, this *inverse channel encoder* (ICE) effectively learns the same set of operators shown in Figure 7.5, which can then be rearranged to form an approximation of the channel autoencoder. Since the encoder and decoder map through a Gaussian constraint, the input and output of this channel autoencoder are similarly constrained. In essence, the newly learned operators effectively convert the arbitrary channel $\mathcal{C}$ into a Gaussian channel. As such, this transformed channel can be analyzed using well established methods of analyzing Gaussian channels in order to impose a coding scheme on the channel that maps bits of information onto points in $\mathbb{H}$.

7.4 An Example Application with Written Digits

In order to illustrate the ICE, we will consider the application to the MNIST database [78] using two 'toy' channel models. The first will consider written digits without any distortion, i.e. the channel is the identity operator, which is equivalent to the traditional VAE. The second channel will correspond to a horizontal blurring operator and the inverse channel encoder is then trained to 'unblur' the digits.

Furthermore, we will consider the problem of selecting the dimensionality for the latent space $\mathbb{H}$. We begin by first constraining the latent space to three dimensions, which is generally insufficient for representing the information contained in the images but allows for a straightforward visualization of the space. We then examine the behavior in the case that the latent space is allowed to utilize up to 10 dimensions, which is more difficult to visualize but demonstrates the behavior when more dimensions than necessary are employed. As we will see, the Gaussian constraint in the utility function imposes an elegant method of constraining the dimensionality of the latent representation.

In each case, the encoder and decoder networks are both composed of three fully connected layers with each layer consisting of 50 rectified linear units (ReLU). Encoder and decoder are then connected by a three-dimensional Gaussian constraint layer using the reparameterization trick. The network was then trained using the Adam optimizer provided within the Tensorflow framework [79]. The choice of network architecture, activation function, and backpropagation optimizer may be adjusted to improve performance; however, in this application we simply require a function approximator with sufficient representational power that can be trained in a relative short period of time. Each network discussed in this chapter was trained over 20 epochs, after which the loss function begins to plateau.

7.4.1 Over-Constrained Latent Space

In this first example, the latent dimension of the VAE is constrained to three dimensions. In order to get a sense of the representational power of the network, we first visualize input and output images of the VAE. Figure 7.7 illustrates a sampling of such input-output pairs as sorted by digit. As we

can see, the limited dimensionality of the latent space results in somewhat 'fuzzy' output images and a nontrivial amount of confusion between digits that are visually similar.

In addition, we can visualize the latent space as a 3D scatterplot where the colors of the points indicate the digit labels provided in the MNIST dataset, as shown in Figure 7.8. Notably, the training labels are not used in training the VAE; they are simply used for the purpose of illustrating that each of the digits are effectively clustered together. As can be seen, the digits appear to separate into clusters within the latent space but there is a moderate amount of overlap between them. This is largely the cause that drives mixed digits seen in Figure 7.7.

Lastly, we use the encoder and decoder networks to form a 3D Gaussian channel that transmits information through the space of handwritten digits. We can then measure the average error in the channel by transmitting three-dimensional vector valued numbers that have been sampled from a standard normal distribution $\mathcal{N}(0, 1)$. The mean square error is then estimated using the average squared error between the input and output for each of these random samples. The MSE estimated from this 3D channel is 0.26 as measured from 5000 random samples. We can visualize the channel error in Figure 7.9, where the inputs are uniformly sampled along each axis from -1 to 1 and plotted alongside the outputs.

Next we look at the performance of the inverse channel encoder when applied to the horizontally blurred channel. As with the null channel example, we begin by examining the input and output digits of the channel in order to get a sense of the neural net's representational power. Figure 7.10 illustrates a sampling of such input-output pairs as sorted by digit.

As would be expected, the performance of the communication channel is degraded compared with the undistorted channel but the results are still quite promising. In particular, digits that were already somewhat confused are much worse under this distortion. This is largely due to the fact that the horizontal blur obscures some of the features that might be used to distinguish certain digits.

To visualize the channel, we randomly sampled points from the input as was done with the null channel example. The MSE estimated from the data is approximately 0.47, which is much higher than that measured in the null channel. As such, the visualization in Figure 7.9 is ill-suited for a channel

Figure 7.7: Example inputs and outputs of the MNIST VAE, with input images shown above the reconstructed output.

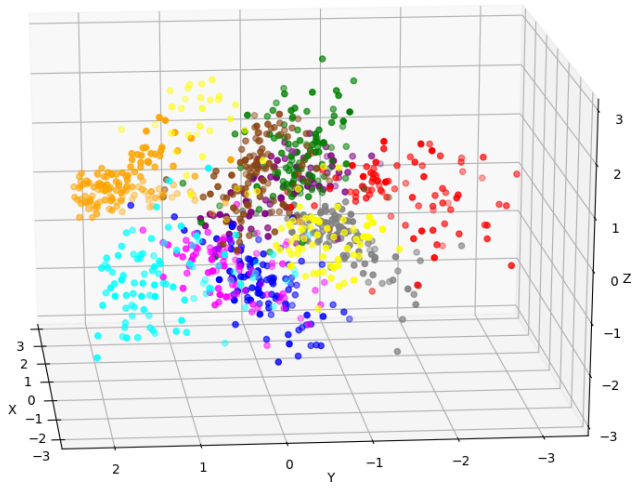


Figure 7.8: Scatterplot of the latent space illustrating the clustering performed by the VAE over written digits. Each color in the scatterplot corresponds to a digit label provided within the MNIST dataset.

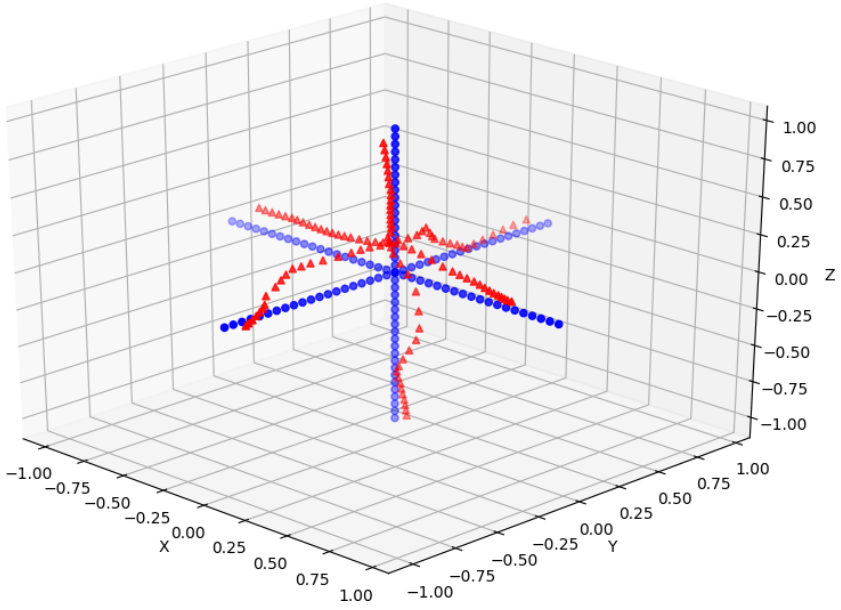


Figure 7.9: View of the channel distortion by transmitting points along the axes of the latent space. The points in blue represent the inputs to the channel and the points in red represent the outputs.

Figure 7.10: Example inputs and outputs of the MNIST VAE, with input images shown above the reconstructed output.

with such high error as the structure of the output would be difficult to recognize. Instead, we can visualize the distortion of the channel as shown in Figure 7.11. Here the points are plotted based on the coordinates of the inputs to the channel and they are colored based on the measured MSE. As we can see, the over-constrained inverse channel encoder appears to organize the latent space such that points with high error are separated from those with low error.

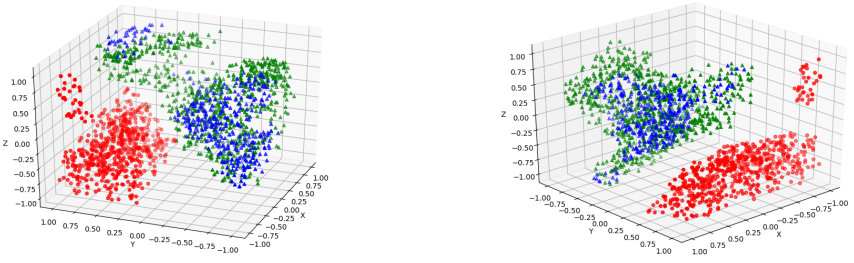


Figure 7.11: View of the channel distortion by transmitting points along the axes of the latent space.

Since the encoder operator produces the mean and standard deviation of a probability distribution as its output, we can also look at the variance and its relation to the error magnitude. The relationship for the null channel is shown in Figure 7.12, and the blurred channel is shown in Figure 7.13. Interestingly, the null channel does not appear to exhibit any clear relationship between standard deviation and measured MSE; however, the blurred channel shows a notable linear relationship between the two. It should be noted that this relationship is not perfectly predictive, but it suggests that the variance could be used as an indication of uncertainty in the output of the channel.

7.4.2 Under-Constrained Latent Space

Next we consider the case that the latent space consists of ten dimensions. Recall that the two terms in the objective function in Equation 7.3 drive the latent space to be as close to a standard normal as possible while approximating the output as accurately as possible. As a result, the encoder and decoder operators only use as many dimensions as are needed for reconstruction and

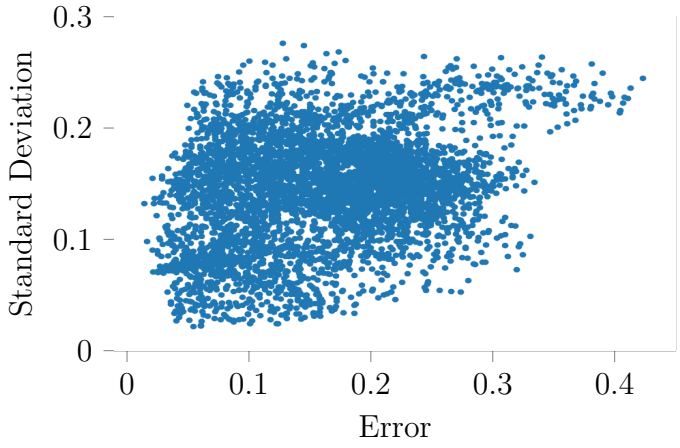


Figure 7.12: Scatterplot of the measured MSE of the null channel model versus standard deviation output by the encoder operator as the horizontal and vertical axes respectively.

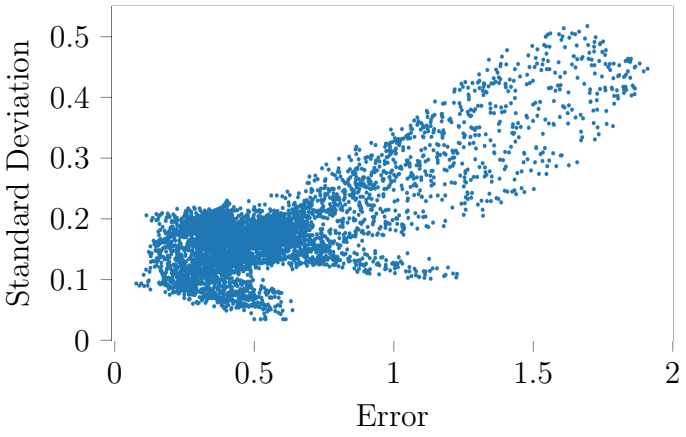


Figure 7.13: Scatterplot of the measured MSE of the blurred channel model versus standard deviation output by the encoder operator as the horizontal and vertical axes respectively.

fix the remaining dimensions of the space as standard normal distributions. We begin by considering the case of the null distortion, where we note that only five of the dimensions are actually used. Once again, we use example digits sampled as input and output of the channel in order to visualize the network's representational accuracy. As is seen in Figure 7.14, the output images appear to more accurately represent the input images than when the latent dimension was over-constrained.

Figure 7.14: Example inputs and outputs of the ICE trained on the MNIST dataset with no distortion operator applied. Input images are sampled from MNIST and shown above the reconstructed output.

Next, the encoder and decoder operators are used to form a channel and the transmission error is estimated by randomly sampling points from the

five-dimensional latent space. Using 5000 data points normally distributed within the 5D space, we find that the MSE is approximately 0.11, which is much reduced from the 3D example.

Although it is hard to visualize the 5D latent space, we can visualize the digits that correspond to given coordinates. Figure 7.15 shows the digits corresponding to the input and output coordinates of the inverse channel encoder when used as a communication channel. Each pair of rows shows a sweep across one of the latent dimensions and thus gives a form of visual representation analogous to that shown in Figure 7.9.

Figure 7.15: A visualization of the 'digit' represented by points swept along each of the 5 dimensions of the latent space. The upper row in each pair represents the input to the channel and the lower represents the output.

The same procedure is then repeated for the case of a horizontal blur applied to the digits. Figures 7.16 and 7.17 show example inputs and outputs as well as a sweep of each dimension of the latent space, respectively. We find that the higher dimensional latent space does much better at approximating the inverse of the blurring channel than when the latent space was constrained to 3 dimensions. In addition, we note that the latent space employs a total of 6 dimensions in contrast with 5 used in the case of the null distortion. Lastly, we find that the MSE of the channel formed using the encoder, decoder and blurring operator has an estimated MSE of 0.11, which is effectively unchanged from the null example. This last result is particu-

larly significant in that it demonstrates that the ICE naturally balances the cost of using additional latent dimensions with representing a more complex channel operator.

7.5 Conclusion

In this chapter, we have developed a notion of the *inverse channel encoder* as a method of learning a pair of operators that map into and out of any arbitrary channel operator. Importantly, this model can be applied to any arbitrary channel model to learn a pair of encoder and decoder operators that map Gaussian random variables through the channel. The issue of transmitting discrete symbols over a Gaussian channel is a well established application of information theory [80]. In addition, by representing the space of primitives in terms of continuous variables, this approach lends itself to a straightforward interpretation under the framework of exemplar theory [21].

Relating back to speech, we will next consider a communication channel that extends from speech production through perception. Using the Praat speech production model, we can readily generate the necessary training for approximating the inverse model of speech production using the inverse channel encoder. With such an inverse model, we then have a pair of operators that map from the latent space onto the space of articulatory gestures and map from the acoustic feature space back onto the latent space. The following chapter will detail the application of the ICE for learning sensorimotor primitives of the acoustic speech communication channel.

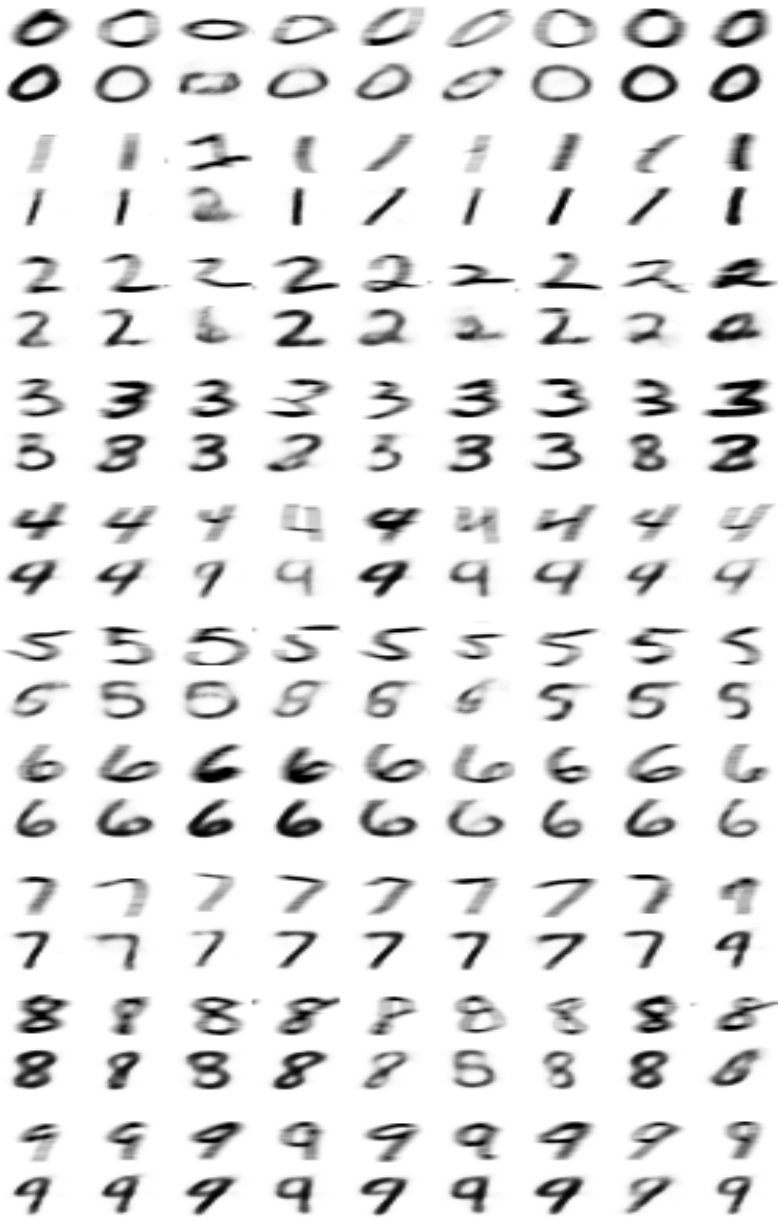


Figure 7.16: Example inputs and outputs of the ICE trained on the MNIST dataset with a horizontal blur operator applied. Input images are sampled from MNIST and shown above the reconstructed output.

Figure 7.17: A visualization of the 'digit' represented by points swept along each of the 6 dimensions of the latent space. The upper row in each pair represents the input to the channel and the lower represents the output.

CHAPTER 8

THE SPEECH COMMUNICATION CHANNEL

As shown in Chapter 6, a linear control system implementation of sensorimotor primitives is ill-equipped for modeling communication using the acoustic speech signal. In order to extend the sensorimotor primitive model of speech to the domain of the acoustic signal, we utilize a nonlinear model of both the high level controller and acoustic observer, or in the taxonomy of communication, the encoder and decoder respectively. In this chapter we demonstrate the inverse channel encoder (ICE) neural network architecture as a unsupervised method of learning this encoder/decoder pair.

The following sections detail the application of the ICE to speech communication. We begin by specifying the speech communication channel, giving special attention to the channel inputs and outputs as they are defined in this model. Next, we provide specifications of the neural network architecture used to implement the ICE, as well as the procedure used to generate data and train the network. We then examine some experimental results demonstrating basic communication over the speech communication channel using the encoder and decoder derived from the trained ICE neural network. We conclude by discussing the implications and limitations of these results.

8.1 Channel Specifications

We define the speech communication channel as the mapping from the linear primitive control inputs through the acoustic features produced by the cochlea. The basic layout of this channel is graphically represented in Figure 8.1.

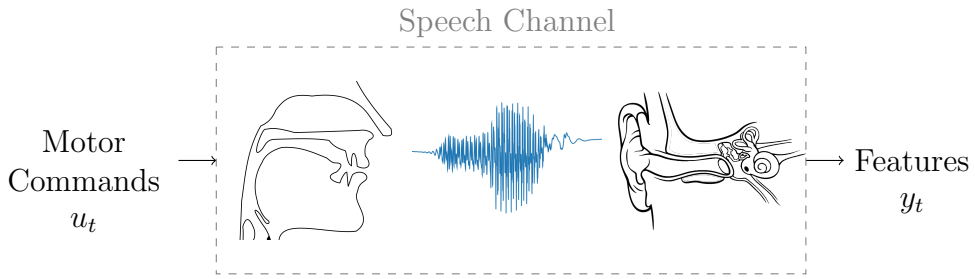


Figure 8.1: Notional depiction of the speech communication channel that maps from motor control commands to acoustic features.

8.1.1 Channel Inputs: Primitive Control

The channel inputs are defined using the linear sensorimotor primitives that are detailed in Chapter 5. Importantly, the linear primitives are only implemented using proprioceptive feedback as a means of controlling the area function of the vocal tract. The details of the training procedure and specifications for the linear articulatory primitives are provided in Chapter 6. In this case, we utilize a 10 dimensional primitive control input as a means of driving the vocal tract.

A critical advantage of this approach, as opposed to applying the ICE directly to the muscle inputs of the Praat model, is that the linear primitives provide a form of coordinated control over the muscles. Although it could be expected that the encoder learned by the ICE architecture could presumably learn this same coordinated control, the articulatory primitives serve as a form of shortcut in training the neural network, since the linear primitives have already been trained on a very large dataset. Additionally, the reduced dimension allows for training data to be created much more efficiently. Rather than generate training data by randomly exciting all muscle inputs, we can randomly excite the primitive inputs and cover the same subset of vocal tract configurations.

8.1.2 Channel Outputs: Acoustic Features

We define the output of the speech communication channel as the features extracted by the cochlea. In Section 1.1.4, we noted that a model of the cochlea is generally much less important than a model of the vocal tract dynamics. Instead, we select a simplified model of acoustic features that is

inspired by the cochlea, namely mel frequency cepstral coefficients (MFCC). For the experiments described in this chapter, we represent the acoustic features produced at the output of the vocal tract using an MFCC feature vector containing 13 coefficients. The stream of feature vectors are computed over a 30ms hamming window, that is stepped along the speech signal in 10ms increments. This analysis window length and step size are selected to coincide with typical values used in analyzing speech as a short-time stationary signal [22].

8.1.3 Training Data Production

Training data is generated using the linear sensorimotor primitives detailed in Chapter 5 to drive the Praat model. Because of the coordinated control inputs produced by the linear primitives, the discovery of sound-producing gestures through random inputs is much more frequent than when compared with direct muscle control. As a result, the linear primitive inputs were used to generate training data that involved driving all muscle inputs through coordinated random inputs from the linear primitives. The data was collected and saved in the same manner as the direct muscle control, using an energy threshold of 10^{-3} to collect 1000 one-second-long utterances that each contain an audible segment.

8.2 ICE Specifications

The structure of the neural network used to implement the ICE is detailed in this section. Although the layout of the network does play a significant role in the nature of the resultant primitive operators, it is important to note that our selection of this architecture is largely arbitrary. In this case, we simply choose a neural network architecture that strikes a balance between representational power and training efficiency.

8.2.1 Neural Network Architecture

The ICE neural network used in this experiment can be considered in two parts: the encoder network and the decoder network. The decoder and en-

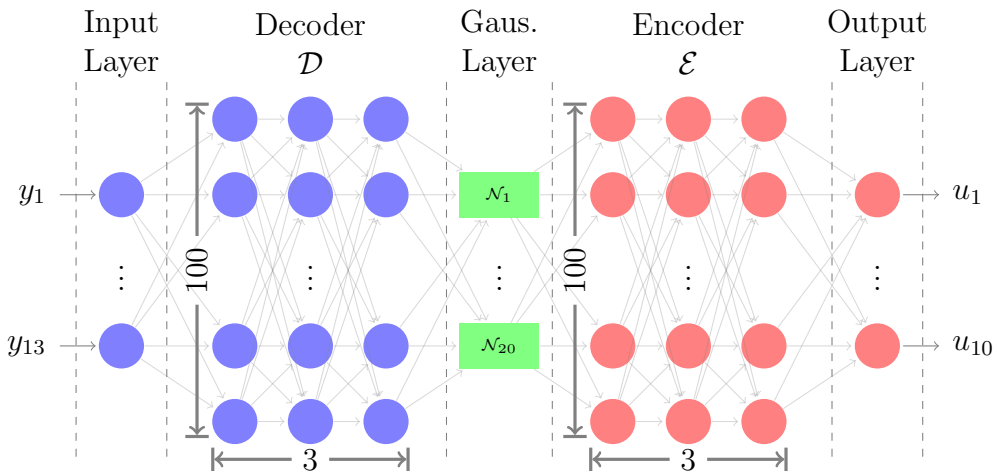


Figure 8.2: Topology of the ICE neural network with annotations indicating the dimension of each layer that was used in the experiments discussed in this chapter.

coder are then connected by a Gaussian constraint layer. The annotated topology of the ICE neural network is shown in Figure 8.2. The decoder network consists of an input layer for the 13 acoustic features followed by 3 fully connected layers. The output of the decoder produces the mean and variance parameters for a 20 dimensional Gaussian constraint layer. Similarly, the encoder consists of three fully connected layers that map from the 20 dimensional output of the Gaussian constraint layer to a 10 dimensional output layer for the control inputs. In both the encoder and decoder, the fully connected layers each consist of 100 relative linear unit (ReLU) nodes.

As stated, the choice of fully connected layers is primarily driven by ensuring that the networks have sufficiently large representational power. In addition, ReLU is selected as the activation function due to the fact that its gradient is computationally simple, and, as such, backpropagation can be computed rapidly. Lastly, we take a moment to emphasize that this selection of architecture is largely arbitrary. What is important is that the neural network be able to adequately represent the inverse channel operator in a manner than can be decomposed into an encoder/decoder pair. As such, this experiment simply illustrates the efficacy of the ICE architecture at a high level.

8.2.2 Neural Network Training

The ICE neural network is trained as a feed-forward network using the MFCC features as input and primitive control actions as output. The training data consists of 1000 seconds of randomly generated utterances generated based on the procedure outlined in Section 8.1.3. Prior to training, the data are normalized to the interval of $[0, 1]$, and the input-output pairs are shuffled into random batches. The network weights were learned using the Adam optimizer as implemented in the Tensorflow framework. The training procedure is then iterated over 100 epochs, which is sufficiently long for the loss term to plateau.

8.3 Experimental Results

Using the trained ICE network, a communication channel is formed using the ICE-encoder to map the primitive feature space onto articulatory primitive actions and the ICE-decoder to map the resulting MFCCs back onto the primitive feature space. Given that the network is trained on only 1000 seconds of random utterances, it is important to note that this model of speech primitives is actually quite limited. Since speech production is learned over several months in humans, it is reasonable to expect that the training dataset would need to be orders of magnitude larger in order to fully represent all possible speech sounds.

In looking at these results, it is interesting to note that the dimensions of the primitive feature space are not used equally. This can be effectively measured by examining the variance parameter at the output of the decoder network. This is represented by the graph in Figure 8.3 which shows the variance output on each latent dimension, as averaged across the training data inputs.

As we examine the performance of this model, we will look at two metrics of performance. First, we will consider the quality with which the model can reproduce a given utterance. Second, we will consider the performance of this model as a communication channel that maps the primitive feature space through the speech channel.

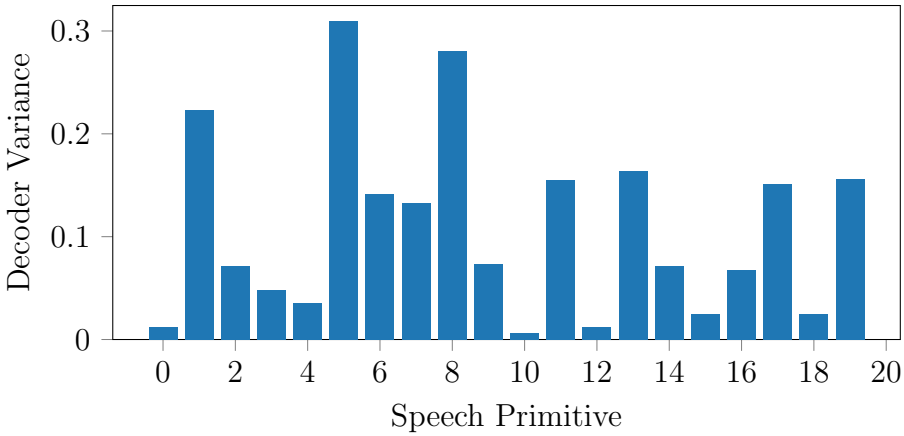


Figure 8.3: A graph showing the average variance of each primitive output by the decoder network.

8.3.1 Utterance Reproduction

We first consider a qualitative demonstration of utterance reproduction using this primitive model. We do this by examining a randomly generated utterance that is reproduced using the ICE primitive model. Utterance reproduction is achieved by feeding the MFCC features into the ICE network in its original feed-forward layout. The articulatory primitive actions that are generated at the output of the ICE network are then used to drive the Praat vocal tract model to produce an imitation of the original utterance.

The spectrograms of the original and reproduced utterances are shown in Figure 8.4, where we see a strong resemblance between the two. If we first consider the error in reproduction of the motor inputs shown in Figure 8.5, we see that the quality of reproduction appears to be somewhat poor. Even more interestingly, we see that the motor inputs are all defaulted to a value near zero for the first part of the utterance, where no sound is being produced. Finally, we consider the primitive features produced by both the original and reproduced utterance as well as the "error bounds" indicated by the variance in the primitive features. What we see in Figure 8.6, is that the primitive features appear to be reproduced quite well. Furthermore, the primitive features show a large variance during the quiet segment of the utterance, effectively indicating that little information is being transmitted during that segment.

More quantitatively, the reproduction error can be measured by considering the error in MFCC value and the error in control input values. Since

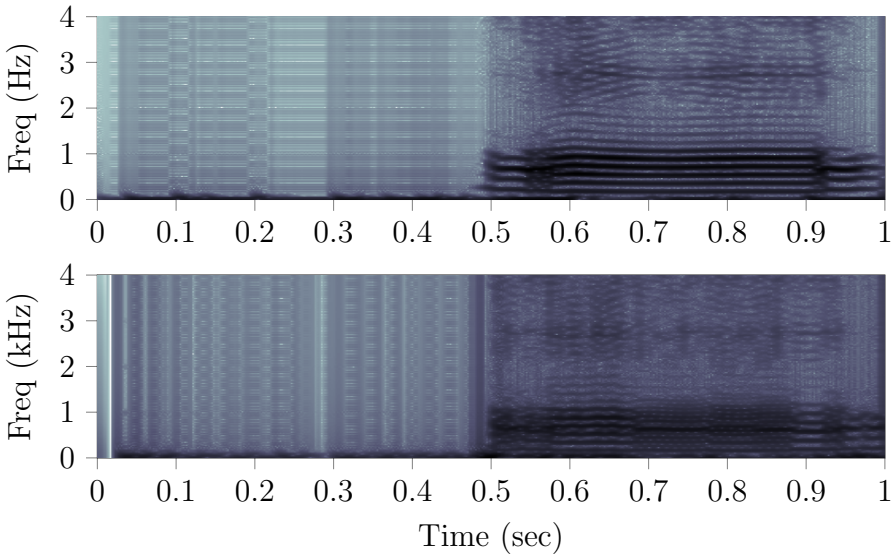


Figure 8.4: An example of an imitated utterance produced using the ICE primitive model. The spectrogram of the original is shown above the spectrogram of imitation.

the utterances are imitated directly in 5ms time steps, the error can be computed directly without the need for advanced methods of time alignment like dynamic time warping [81]. The errors values shown in Figure 8.7 were computed for 100 randomly generated utterances that were reserved as the validation set during training.

8.3.2 Communication Performance

Next we consider the communication channel formed using the full communication channel as illustrated in Figure 8.8, which is formed using the encoder, articulatory primitive controller, Praat simulator, and decoder. We evaluate the performance of the encoder and decoder operators by estimating the channel capacity under two simple conditions. The first is a measure of the capacity under random input sequences, constructed in the same manner as those used for training the ICE neural network. We then consider the channel performance under static inputs, which roughly correspond to vowel sounds.

We then estimate the channel capacity based on the 'waterfilling' method for channels with colored Gaussian noise [80]. Effectively, we find an upper bound for the mutual information of the channel inputs and outputs defined

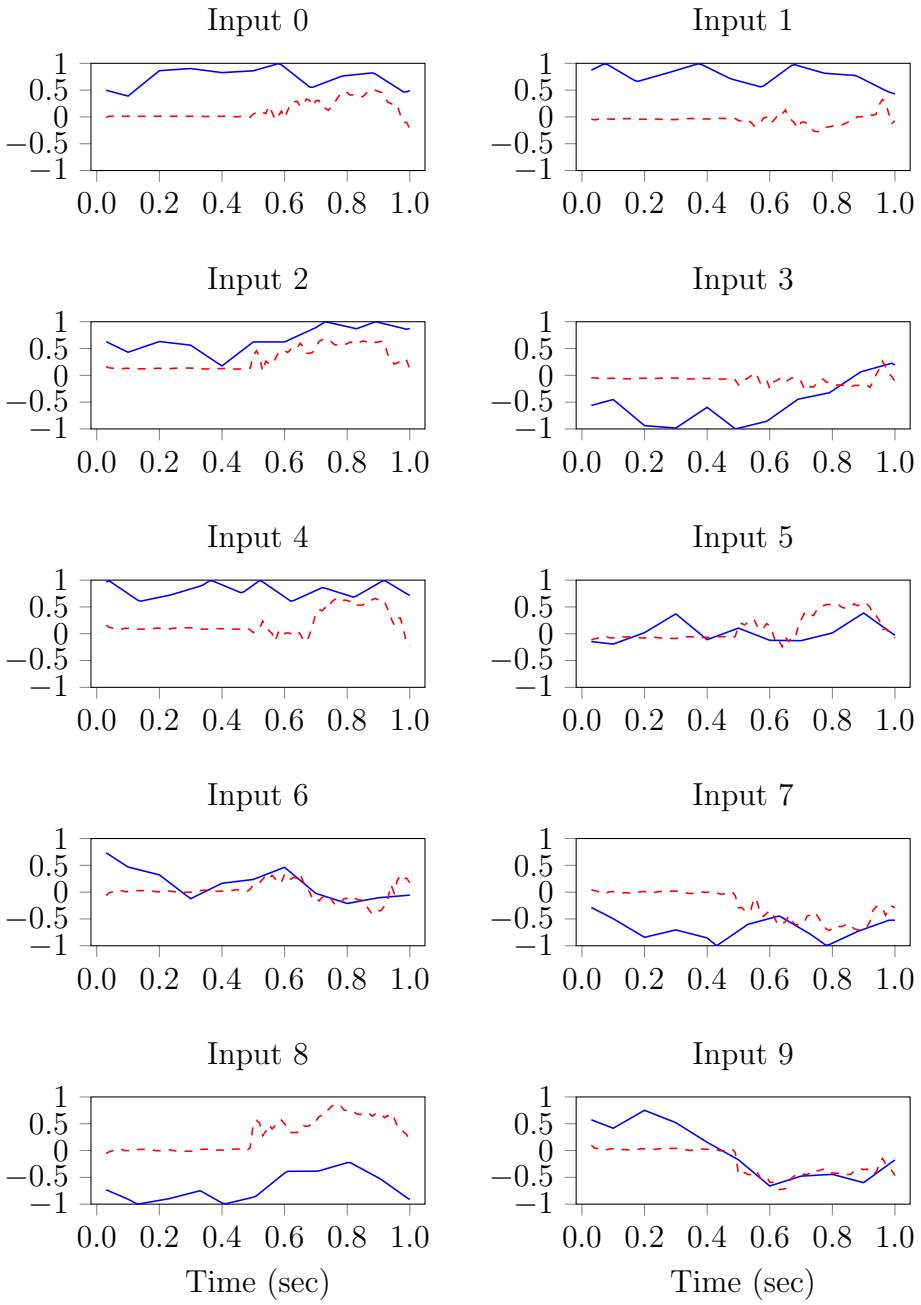


Figure 8.5: The 10 control inputs used to produce the original and imitation utterances. The original control inputs are shown in blue and the imitation in dashed red.

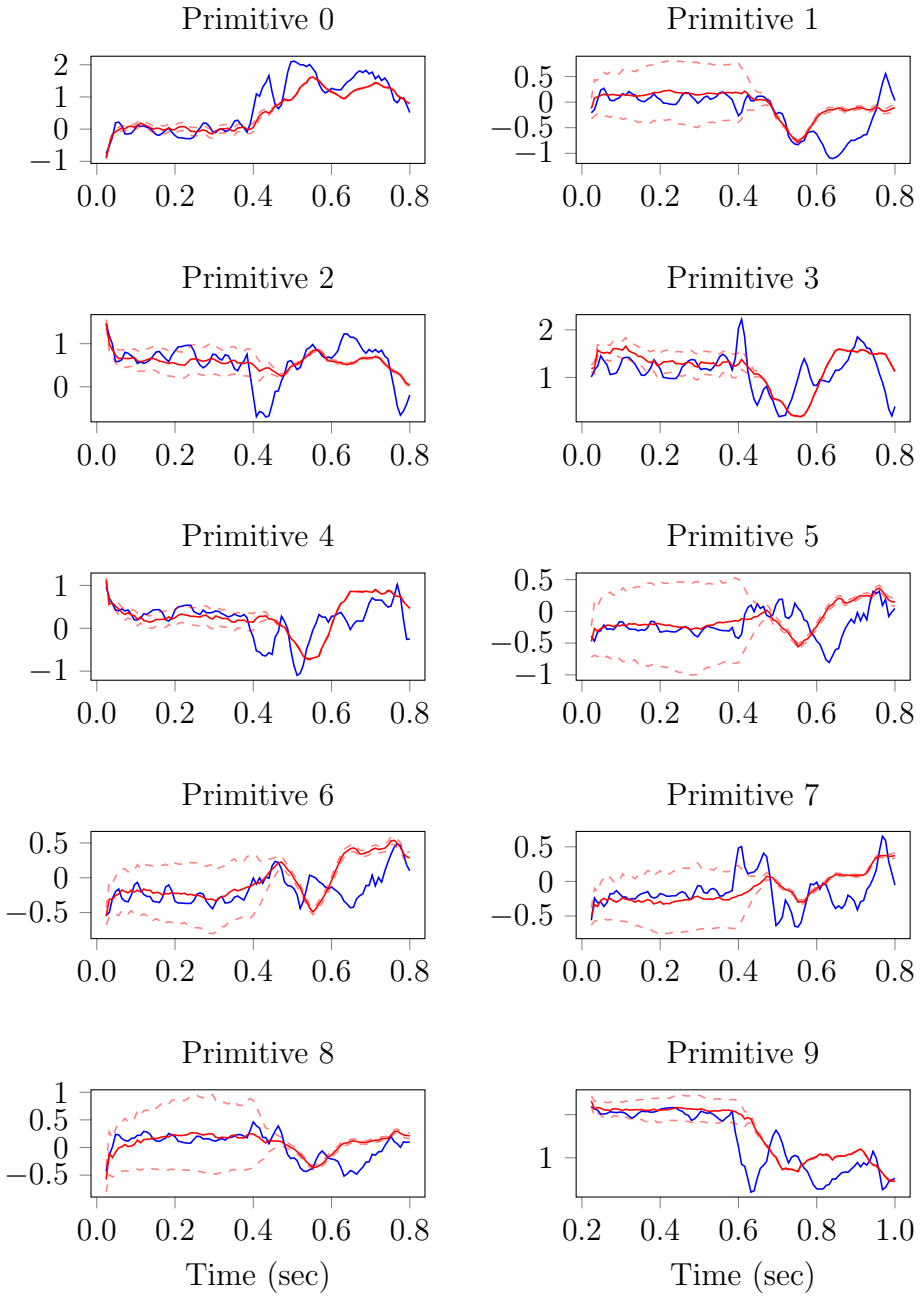


Figure 8.6: The first 10 dimensions of the primitive values corresponding to the original and imitation utterances. The original is shown in blue and the imitation in red.

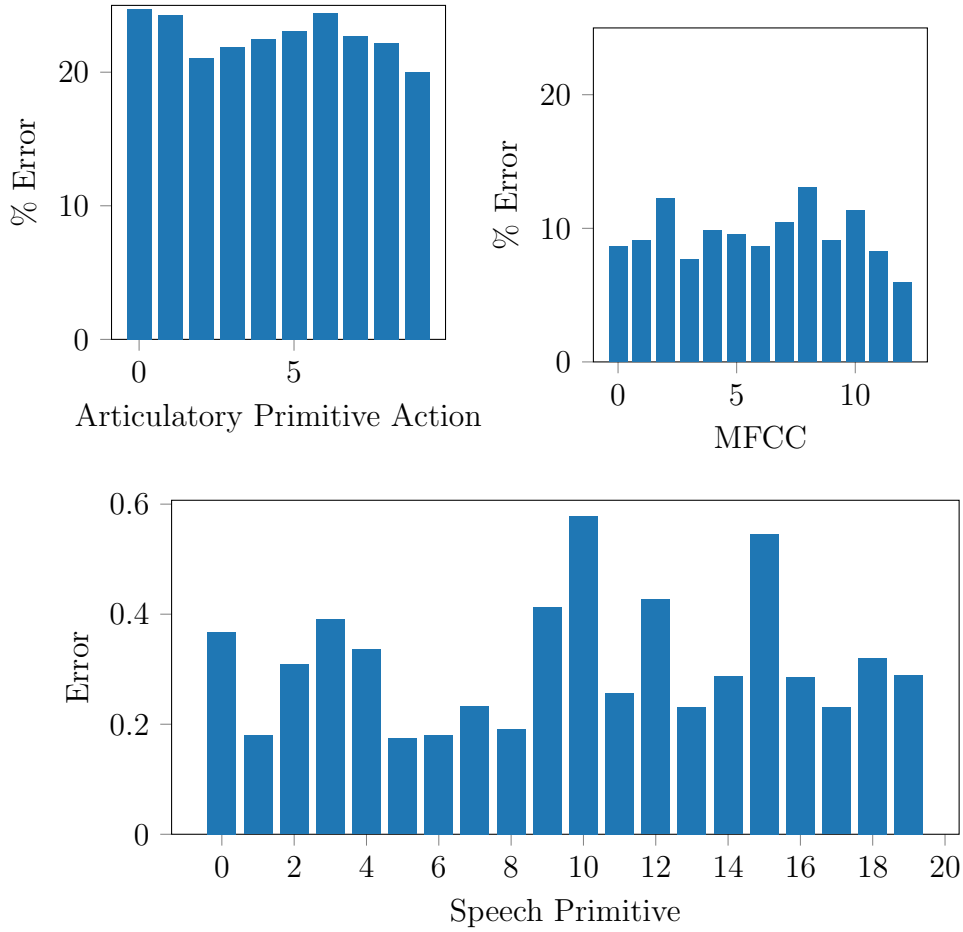


Figure 8.7: Average error between original and reproduced utterances computed over each dimension of the primitive control inputs, MFCC values, and primitive feature values.

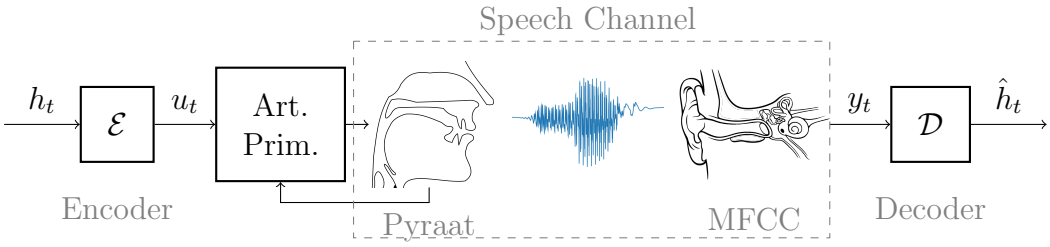


Figure 8.8: Speech communication channel implementation.

in Equation 8.1, where we use Σ to represent the covariance matrix of a random variable and $E = X - Y$ is the error measured at the output of the channel.

$$\begin{aligned} I(\mathbf{X};\mathbf{Y}) &= h(Y) - h(Y|X) \\ &= h(Y) - h(E) \\ &= \log\left(\frac{|\Sigma_Y|}{|\Sigma_E|}\right) \end{aligned} \tag{8.1}$$

We find an upper bound for the mutual information in Equation 8.1, using the properties of the determinant operator and eigen-decomposition. Specifically, we define $\Sigma_E = Q\Lambda Q^\top$, where $QQ^\top = I$. Then the determinant may be computed as the product of eigenvalues, $|\Sigma_E| = \prod_i \lambda_i$. In addition, we have Hadamard's inequality which states that, for a positive definite matrix A, the determinant is bounded by the product of the diagonal elements, $|A| \leq \prod_i A_{i,i}$. We then define $A = Q\Sigma_Y Q^\top$ and compute an upper bound on the mutual information using Equation 8.2. In effect, this is a reverse engineering of the waterfilling procedure where the waterline is fixed and defined by $A_{i,i} > \lambda_i$.

$$I(\mathbf{X};\mathbf{Y}) \leq \sum_i \left[\log(A_{i,i}) - \log(\lambda_i)\right] \quad \text{for } \forall i : A_{i,i} > \lambda_i \tag{8.2}$$

We apply random gesture sequences h_t to the input of the encoder and compute the estimation error at the output of the channel $E = h_t - \hat{h}_t$. The covariance matrices of the channel output and the error are then visualized in Figure 8.9. In addition, we can visualize the 'waterline' of the channel as shown in Figure 8.10. Using Equation 8.2, we compute the upper bound on the mutual information to be 15.48 bits per channel usage.

Next, we consider the behavior of the channel when constant inputs are provided and consider the error at the output over time. This provides us with a means of parameterizing the error versus time spent transmitting a given symbol - i.e. a way of selecting an appropriate symbol rate for static speech sounds. We then plot the average error as a function of integration time as shown in Figure 8.11.

We can compute the average error per utterance for each channel dimension and compute the corresponding covariance matrix for this Gaussian channel. Applying the same procedure for computing an upper bound on the mutual information using Equation 8.2, we estimate a bit rate of approximately

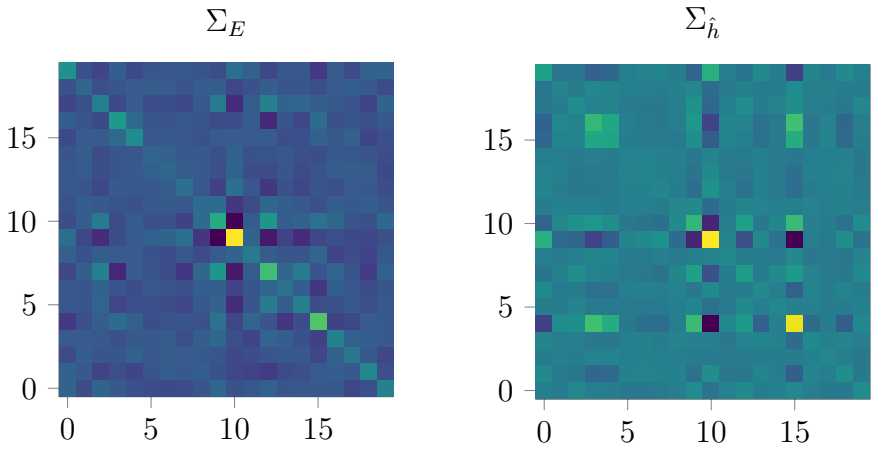


Figure 8.9: A visualization of the error covariance and channel output covariance when random gesture sequences are applied at the speech channel input.

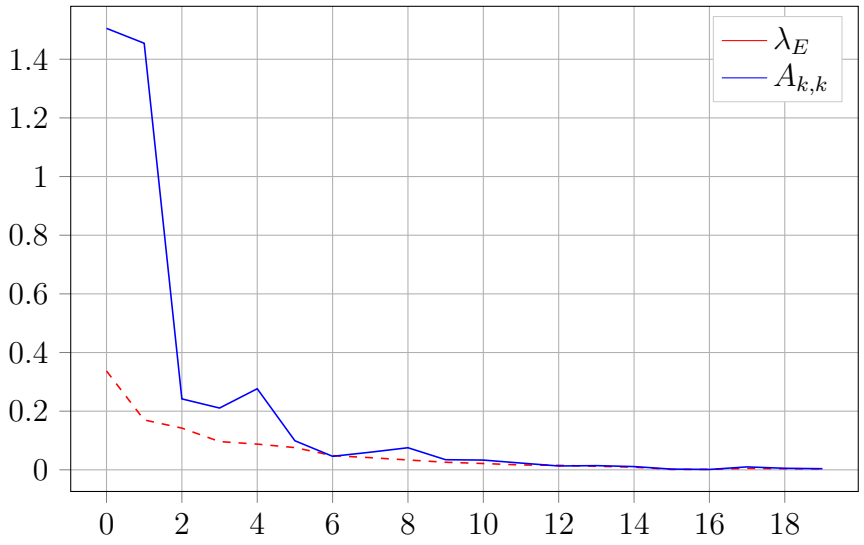


Figure 8.10: A plot of the eigenvalues of the error covariance λ_i versus the corresponding channel covariance $A_{i,i}$.

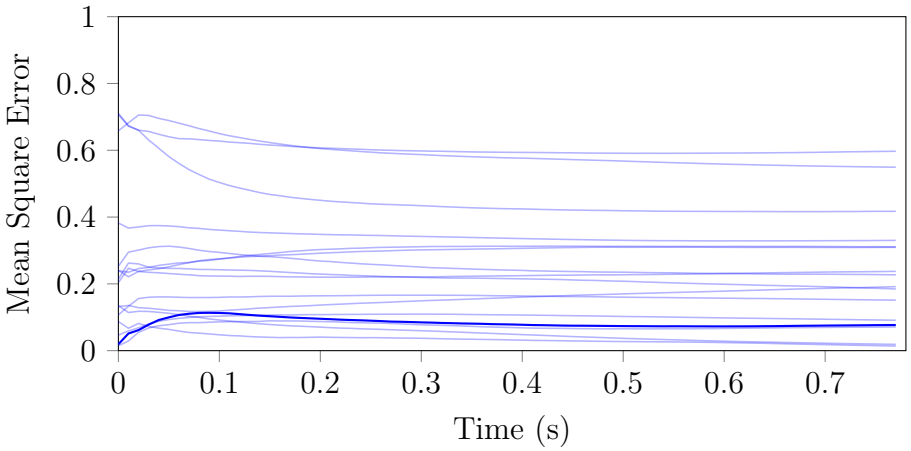


Figure 8.11: The average error as a function of integration time for constant valued inputs to the speech communication channel.

3.08 bits per channel usage. Effectively, this implies that this model of the speech channel can produce at most 8 vowel-type sounds that can be reliably distinguished from one another.

In both of these examples, we note that the computed bit rate is much smaller than what might be found by breaking speech down into phones or phonemes and computing a corresponding bit rate. The reason for this is largely due to the narrow scope of the data used to train this model. Specifically, this model is trained using a total of 1000 seconds of data produced by random excitation of the articulatory primitives. This represents only a minuscule subset of the total space of articulatory gestures. If the model were to be trained on a dataset that more fully sampled the space of possible gestures, we would expect the corresponding channel model to accurately represent the actual channel capacity of speech.

8.4 Discussion

This chapter demonstrated the application of the ICE neural network architecture as a means of learning a nonlinear speech primitive model. While these results show the efficacy of this approach to characterizing the speech signal within the framework of a communication channel, it is important to note the limitations of these results. The most prominent limitation is the limited amount of training data used to train the neural net. Although this

limitation is severe, it is notable that the resultant model was still able to form a functional speech communication channel. Given sufficient time to generate a large enough training set, this approach would be usable as a general framework for speech analysis in natural speech. An additional limitation of this work is in the simplistic neural network design. Namely, the network architecture is only selected to satisfy the basic ICE structure and does not employ any advanced neural network design in selecting the internal structure of the network. If a more advanced network design were employed, it is possible that a more robust set of speech primitives could be learned.

Given these strong limitations on this model design, it is significant that the model developed here was capable of forming a simple speech communication channel. Although this model does not yet represent a universal framework for analyzing speech, reaching that goal is largely a matter of increasing the size of the training data.

CHAPTER 9

CONCLUSION

In this thesis, we have developed the general framework of sensorimotor primitives as a means of interfacing with a complex environment. The primitive operators serve as a map between the environment and an abstract primitive feature space. By training the primitive operators using random excitation, the dynamics of the environment are effectively abstracted away. As a result, an autonomous agent is able to use the primitive features as a simplified representation of its interaction with the environment.

This work is motivated by speech communication through the use of a high fidelity articulatory synthesizer, namely the Praat model. Specifically, the Praat model of articulatory speech production is both high dimensional and nonlinear. To simplify the interface to this speech production model, we have developed two specific realizations of sensorimotor primitives as a means of articulatory control and acoustic communication. Articulatory primitives were developed in terms of a linear feedback control system in order to implement a reduced dimension articulatory controller. Acoustic primitives were developed in terms of a nonlinear communication channel in order to implement a channel encoder and decoder for the synthesized acoustic signal. The resultant primitive operators were then demonstrated to interface with the Praat model in order to form a simple speech communication channel.

9.1 Future Directions

The work presented in this thesis leaves two important avenues for the future development of sensorimotor speech primitives. Although our development of sensorimotor primitives for linear feedback control is largely complete, there remains much room for improvement in the design of a specially tailored implementation of the inverse channel encoder (ICE) for speech analysis.

9.1.1 Larger Datasets and Simulation Time

A strong constraint in our implementation of the ICE is the amount of available training data. In the case of the sensorimotor primitives for linear control systems, the model could be trained using an arbitrarily large set of data without the need to store that data in memory all at once. In the case of the nonlinear primitive operators, they are trained using back-propagation and thus the training data must be stored in memory in order to iterate over the dataset for multiple epochs. As such, we expect that this development of speech primitives would be able to serve as a general model of speech so long as enough training data can be stored in memory. Unfortunately, the size of such a training set is likely to require significant memory storage resources.

An additional constraint in this approach is the available computational power. As noted in previous chapters, the Praat model is computationally expensive and requires a long run time in order to generate large amounts of speech production data. In the case of the 500 minutes of training data detailed in Chapter 6, it took approximately 5 days of run time on a general purpose computer. This problem is further compounded by the fact that only a small percent of the simulated data produces audible speech sounds. Given that humans learn speech production over the course of months, it would be reasonable to assume that a full exploration of the system would require a comparably long set of training data. As such, it would be advantageous to apply more computational power in order to shorten simulation run-time or run multiple parallel simulations.

9.1.2 Improved Neural Net Architecture

In the development of primitives as a nonlinear channel decomposition, a very simplistic form of neural network was used to represent the nonlinear operators. It is expected that more advanced neural network architectures may improve upon the results presented in this thesis. In particular, a convolutional architecture or a recurrent neural network (or perhaps a combination of the two) may lead to a more robust representation of primitive operators. Given that convolutional models have deep roots in speech processing that date back to the source filter model discussed in Chapter 1, it seems likely that our representation of speech production would be improved by incor-

porating this into the network architecture. Similarly, feedback control has played a large role in our formulation of speech primitives and thus it is reasonable to expect that a network architecture that incorporates feedback would be an improvement over a simple feed-forward architecture. Alternatively, it may be that some other parameterization of nonlinear operators could be well suited to this problem; the only true restriction is that the representation must allow for operator decomposition.

9.2 Final Notes: The Primitive Framework

An important aspect of the development of speech primitives presented in this thesis is that they are an inherently general representation of speech that is not language specific. This generality is drawn from the fact that the primitives were trained using random excitation of the vocal tract. It is notable that this approach to training is generally slower than if the model were trained to directly imitate human speech sounds; however, this approach to learning primitives causes them to learn a representation of the system dynamics rather than the existing structure of some existing language. By training in a manner that is not language specific, the resulting primitive feature space can be used as a universal representation through which disparate languages could be compared. In addition, the primitive feature space provides a representation that can be used to quantify and analyze the information encoded in the speech signal. As the concept of speech primitives is further developed, we expect that it will provide a quantitative framework through which speech can be analyzed in terms of an information theoretic framework.

9.2.1 Broader Applications

Although this thesis is motivated by the study of speech and language, is important to consider the broader implications of our mathematical development of primitives. Fundamentally, we consider primitives in terms of operators that are used to transform the problem space into a simpler representation. It is important to note that the implementations of sensorimotor primitives developed in this thesis could easily be applied to other forms

of communication and control. This could take the form of human communication such as sign language, animal behaviors such as bird songs and bee dancing, or technological implementations of communication over some nonlinear medium.

9.2.2 The Primitive Mindset

The most important contribution of this work does not come from the specific mathematical developments that were presented; rather, it lies in the way of thinking that led to these developments. The study of primitives in both control theory and biology has yet to reach a unified notion of what primitives are. In this thesis, we posit that primitives serve the function of simplifying a *problem space* rather than breaking down some specific problem that is at hand. By thinking in terms of simplifying a space of problems, we can more readily come to a mathematical development of primitives that enable a broad range of learning and adaptation to be implemented in the abstract primitive space.

REFERENCES

[1] C. H. Shadle and R. I. Damper, "Prospects for Articulatory Synthesis: A Position Paper," *Fourth ISCA Tutorial and Research Workshop on Speech Synthesis*, pp. 121–126, 2001.

[2] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," 2016. [Online]. Available: http://arxiv.org/abs/1609.03499

[3] S. E. Levinson, *Mathematical Models for Speech Technology*, ser. Prentice-Hall signal processing series. The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England: John Wiley and Sons Ltd, 2005.

[4] A. M. Turing, "On Computable Numbers, with an Application to the Entsheidungsproblem," *Proc. of London Math. Soc.*, vol. 2, no. 42, pp. 230–265, 1936.

[5] A. M. Turing, "Computing machinery and intelligence," *Mind*, no. 49, pp. 433–460, 1950.

[6] O. Sacks, *Seeing Voices: A Journey into the World of the Deaf*. London: HarperCollins, 1990.

[7] O. Schaller, *A Man Without Words*. Berkley: University of California Press, 1995.

[8] S. R. Waxman, "Links between object categorization an naming: Origins and emergence in human infants," in *Early Category and Concept Development*, D. H. Rakison and L. M. Oakes, Eds. 198 Madison Avenue, New York, New York 10016: Oxford University Press, 2003, ch. 9, pp. 213–241.

[9] G. Thierry, P. Athanasopoulos, A. Wiggett, B. Dering, and J.-R. Kuipers, "Unconscious effects of language-specific terminology on preattentive color perception." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 11, pp. 4567–70, 2009.

[10] A. Franklin, G. V. Drivonikou, L. Bevis, I. R. L. Davies, P. Kay, and T. Regier, "Categorical perception of color is lateralized to the right hemisphere in infants, but to the left hemisphere in adults," *PNAS*, vol. 105, no. 9, pp. 3221–3225, 2008.

[11] C. Kobayashi, G. H. Glover, and E. Temple, "Switching language switches mind: linguistic effects on developmental neural bases of 'theory of mind'," *Social Cognitive and Affective Neuroscience*, vol. 3, no. 1, p. 62, 2008.

[12] T. Ullman, A. Stuhlmüller, N. Goodman, and J. B. Tenenbaum, "Learning physics from dynamical scenes," 2014, pp. 1640–1645.

[13] Z. Ghahramani, "Probabilistic machine learning and artificial intelligence," *Nature*, vol. 521, no. 7553, pp. 452–459, may 2015.

[14] N. D. Goodman, J. B. Tenenbaum, and T. Gerstenberg, "Concepts in a Probabilistic Language of Thought," *To appear in Concepts: New Directions*, pp. 1–25, 2014.

[15] B. M. Lake, C.-y. Lee, J. R. Glass, and J. B. Tenenbaum, "One-shot learning of generative speech concepts," in *Proceedings of the 36th Annual Meeting of the Cognitive Science Society*, 2014.

[16] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[17] I. Meir, W. Sandler, C. Padden, and M. Aronoff, "Emerging sign langauges," in *Oxford Handbook of Deaf Studies*, M. Marschark and P. E. Spencer, Eds. New York: Oxford University Press, 2010, ch. 18.

[18] L. Deng, G. Ramsay, and D. Sun, "Production models as a structural basis for automatic speech recognition," *Speech Communication*, vol. 22, pp. 93–111, 1997.

[19] G. Fant, *Acoustic Theory of Speech Production*, ser. D A C S R Series. Mouton De Gruyter, 1970.

[20] A. M. Libermant, "Perception of the speech code," *Psychological Review*, vol. 74, no. 6, pp. 431–461, 1967.

[21] R. Port, "How are words stored in memory? Beyond phones and phonemes," *New Ideas in Psychology*, vol. 25, pp. 143–170, 2007.

[22] T. F. Quatieri, *Discrete-Time Speech Signal Processing*, ser. Prentice-Hall signal processing series. Upper Saddle River, NJ: Prentice Hall PTR, 2002.

[23] Li Deng and Xiao Li, "Machine Learning Paradigms for Speech Recognition: An Overview," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 1060–1089, may 2013.

[24] A. M. Libermant and I. G. Mattinglyit, "The motor theory of speech perception revised," *Cognition*, vol. 21, pp. 1–36, 1985.

[25] S. Levinson, D. Davis, S. Slimon, and J. Huang, "Articulatory Speech Synthesis from the Fluid Dynamics of the Vocal Apparatus," *Synthesis Lectures on Speech and Audio Processing*, vol. 8, no. 1, pp. 1–116, jul 2012.

[26] K. Livescu and J. Glass, "Feature-based pronunciation modeling for speech recognition." Association for Computational Linguistics, 2004. [Online]. Available: http://dl.acm.org/citation.cfm?id=1614005 pp. 81–84.

[27] S. King, J. Frankel, K. Livescu, E. McDermott, K. Richmond, and M. Wester, "Speech production knowledge in automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 121, no. 2, pp. 723–742, 2007.

[28] M. Hasegawa-Johnson, K. Livescu, P. Lal, and K. Saenko, "Audiovisual speech recognition with articulator positions as hidden variables," 2007. [Online]. Available: http://202.114.89.42/resource/pdf/656.pdf

[29] A. Nieto-Castanon, F. H. Guenther, J. S. Perkell, and H. D. Curtin, "A modeling investigation of articulatory variability and acoustic stability during American English /r/ production," *Journal of the Acoustical Society of America*, vol. 117, no. 5, pp. 3196–3212, 2005.

[30] C. A. Fowler, D. Shankweiler, and M. Studdert-Kennedy, "'Perception of the speech code' revisited: Speech is alphabetic after all." *Psychological Review*, vol. 123, no. 2, pp. 15–150, 2016.

[31] A. G. Bruderer, D. K. Danielson, P. Kandhadai, and J. F. Werker, "Sensorimotor influences on speech perception in infancy," *Proceedings of the National Academy of Sciences*, p. 201508631, oct 2015.

[32] M. R. Schomers and F. Pulvermüller, "Is the Sensorimotor Cortex Relevant for Speech Perception and Understanding? An Integrative Review," *Frontiers in Human Neuroscience*, vol. 10, p. 435, sep 2016.

[33] C. Read, Y. Zhang, H. Nie, and B. Ding, "The contributions of phonology, orthography, and morphology in Chinese-English biliteracy acquisition," *Cognition*, vol. 24, pp. 31–44, 1986.

[34] T. J. Palmeri, S. D. Goldinger, and D. B. Pisoni, "Episodic encoding of voice attributes and recognition memory for spoken words." *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 19, no. 2, p. 309, 1993.

[35] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 1, pp. 379–423, 623–656, 1948.

[36] R. Jakobson, *On the Identification of Phonemic Entities*. Nordisk Sprog-og Kulturforlag, 1949.

[37] R. Jakobson, C. G. Fant, and M. Halle, "Preliminaries to speech analysis: The distinctive features and their correlates," MIT, Tech. Rep., 1951.

[38] R. Jakobson and M. Halle, *Fundamentals of Language*. Walter de Gruyter, 2010.

[39] N. Chomsky and M. Halle, "The sound pattern of English," 1968.

[40] N. Wiener, *Cybernetics: Control and Communication in the Animal and the Machine*. Wiley New York, 1948.

[41] J. F. S. Lin, M. Karg, and D. Kulic, "Movement Primitive Segmentation for Human Motion Modeling: A Framework for Analysis," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 3, pp. 325–339, 2016.

[42] E. L. Saltzman and K. G. Munhall, "A Dynamical Approach to Gestural Patterning in Speech Production," *Ecological Psychology*, vol. 1, no. 4, pp. 333–382, 1989.

[43] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," Tech. Rep., 2002. [Online]. Available: http://infoscience.epfl.ch/record/58507/files/ijspeert02.pdf

[44] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors." *Neural Computation*, vol. 25, no. 2, pp. 328–73, 2013.

[45] E. Todorov and M. Jordan, "Optimal feedback control as a theory of motor coordination." *Nature Neuroscience*, vol. 5, no. 11, pp. 1226–1235, 2002.

[46] E. Todorov and Z. Ghahramani, "Unsupervised learning of sensory-motor primitives," in *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE*, vol. 2. IEEE, 2003, pp. 1750–1753.

[47] E. Todorov and W. Li, "Hierarchical optimal feedback control of redundant systems," *Advances in Computational Motor Control IV, Extended Abstract*, pp. 1–2, 2004.

[48] V. Ramanarayanan, A. Katsamanis, and S. S. Narayanan, "Automatic Data-Driven Learning of Articulatory Primitives from Real-Time MRI Data Using Convolutive NMF with Sparseness Constraints." in *INTERSPEECH*, 2011, pp. 61–64.

[49] V. Ramanarayanan, L. Goldstein, and S. S. Narayanan, "Spatio-temporal articulatory movement primitives during speech production: Extraction, interpretation, and validation," *The Journal of the Acoustical Society of America*, vol. 134, no. 2, pp. 1378–1394, 2013.

[50] J. A. Tourville and F. H. Guenther, "The DIVA model: A neural theory of speech acquisition and production." *Language and Cognitive Processes*, vol. 26, no. 7, pp. 952–981, Jan 2011.

[51] C. H. Coker and O. Fujimura, "Model for Specification of the Vocal-Tract Area Function," *The Journal of the Acoustical Society of America*, vol. 40, no. 5, pp. 1271–1271, Nov 1966.

[52] P. Mermelstein, "Articulatory model for the study of speech production," *The Journal of the Acoustical Society of America*, vol. 53, no. 4, pp. 1070–1082, Apr 1973.

[53] S. Maeda, "A Digital Simulation Method of the Vocal-Tract System," *Speech Commun.*, vol. 1, pp. 199–229, 1982.

[54] K. Iskarous, L. M. Goldstein, D. H. Whalen, M. K. Tiede, and P. E. Rubin, "CASY: The Haskins Configurable Articulatory Synthesizer," in *Proceedings of the 15th annual International Congress of Phonetic Sciences*, Barcelona, Spain, 2003. [Online]. Available: http://www.haskins.yale.edu/Reprints/HL1299.pdf pp. 185–188.

[55] F. H. Guenther, "A Neural Network Model Of Speech Acquisition And Motor Equivalent Speech Production Running title: Speech acquisition and motor equivalence," *Biological Cybernetics*, vol. 72, pp. 43–53, 1994.

[56] S. Maeda, *Compensatory Articulation During Speech: Evidence from the Analysis and Synthesis of Vocal-Tract Shapes Using an Articulatory Model*. Dordrecht: Springer Netherlands, 1990, pp. 131–149.

[57] P. Perrier, D. J. Ostry, and R. Laboissière, "The equilibrium point hypothesis and its application to speech motor control," *Journal of Speech, Language, and Hearing Research*, vol. 39, no. 2, pp. 365–378, 1996.

[58] R. Shadmehr, "The Equilibrium Point Hypothesis for Control of Movements," *Motor Control*, pp. 370–372, 1998.

[59] P. Boersma, "Functional Phonology," Ph.D. dissertation, University of Amsterdam, 1998.

[60] W. J. Wagner, "Unsupervised learning of vocal tract sensory-motor synergies," M.S. thesis, Unversity of Illinois at Urbana-Champaign, Urbana, IL, 7 2017.

[61] G. Kapetanios and M. Marcellino, "A parametric estimation method for dynamic factor models of large dimensions," *Journal of Time Series Analysis*, no. 5620, 2009. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1111/j.1467-9892.2009.00607.x/full

[62] D. Bauer, "Subspace algorithms," *IFAC Proceedings Volumes*, vol. 36, no. 16, pp. 993 – 1004, 2003, 13th IFAC Symposium on System Identification (SYSID 2003), Rotterdam, The Netherlands, 27-29 August, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667017348899

[63] E. Hannan and M. Deistler, *The Statistical Theory of Linear Systems*. Society for Industrial and Applied Mathematics, 2012.

[64] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, no. 2, pp. 387–392, Apr 1985.

[65] B. Hajek, *Random Processes for Engineers*. Cambridge University Press, 2015.

[66] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980.

[67] X. Huang, A. Acero, H.-W. Hon, and R. Reddy, *Spoken language processing: A Guide to Theory, Algorithm, and System Development*. Prentice hall PTR Upper Saddle River, 2001, vol. 1.

[68] D. H. Ballard, "Modular learning in neural networks." in *AAAI*, 1987, pp. 279–284.

[69] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[70] D. Holden, J. Saito, T. Komura, and T. Joyce, "Learning motion manifolds with convolutional autoencoders," *SIGGRAPH ASIA 2015 Technical Briefs on - SA '15*, pp. 1–4, 2015.

[71] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[72] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "β-VAE: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations*, 2017.

[73] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. John Wiley & Sons, 2012.

[74] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention," in *Signal Processing and Information Technology (ISSPIT), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 223–228.

[75] H. Kim, Y. Jiang, R. Rana, S. Kannan, S. Oh, and P. Viswanath, "Communication algorithms via deep learning," *arXiv preprint arXiv:1805.09317*, 2018.

[76] H. Kim, Y. Jiang, S. Kannan, S. Oh, and P. Viswanath, "Deepcode: Feedback codes via deep learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 9458–9468.

[77] Y. Jiang, H. Kim, H. Asnani, S. Kannan, S. Oh, and P. Viswanath, "Learn codes: Inventing low-latency codes via recurrent neural networks," *arXiv preprint arXiv:1811.12707*, 2018.

[78] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[79] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[80] T. M. Cover and J. A. Thomas, *Gaussian Channel*. Hoboken, New Jersey: John Wiley and Sons, Inc., 2006.

[81] L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson, "Considerations in Dynamic Time Warping Algorithms for Discrete Word Recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-26, no. 6, pp. 575–582, dec 1978.