Modeling, Analysis and Dynamics of The Human Jaw System

by

Francis Ng, Wai-Tsuen

B.Sc., The City University of New York, 1990

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS OF THE DEGREE OF
MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
DEPARTMENT OF ELECTRICAL ENGINEERING

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
February 1994

© Francis Ng, Wai-Tsuen, 1994

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature)

Department of Electrical Engineering

The University of British Columbia
Vancouver, Canada

Date Mar. 25, 94

DE-6 (2/88)

Abstract

This thesis deals with the modeling of the human jaw system. The model is a computer model in which the nine pairs of facial muscles and the jaw itself are represented. The study leading up to the model includes expressive Object-Oriented Programming (OOP) to encode the computer model. Different components in the jaw system are defined as objects and used as building blocks of the system. Although the studies in the thesis are confined to the human jaw system, various components of the model are designed to permit continuous modification. Direct measurements of muscles' activities are always invasive, or even impossible to measure. A dynamic simulation model offers research workers a frame to improve the concept of the matters involved before measurements are made. Lastly, a jaw model can/may gives insight into how patients will recover from facial and muscle injuries.

Studies of the behavior of other biological systems have been made to discover methods in which an artificial neural network (ANN) may contribute solutions to the dynamic control problem. Some useful results have been obtained which may indicate how ANN could be incorporated in the dynamic jaw model of the future.

The work is interdisciplinary involving the following fields: dynamic behavior of muscle; dynamic behavior of biological system; mechanical system simulation; and ANN.

Table of Contents

Abstract ii

Table of Contents iii

List of Tables vi

List of Figures vii

Acknowledgment x

Chapter 1 **Introduction** 1

1.1 System and Model 2

1.2 Existing Simulation Model 3

1.3 Applications 4

1.4 Thesis Outline 5

1.5 Considerations of Hardware and Software Platform .. 7

Chapter 2 **Muscle Mechanics and the Human Jaw System** 8

2.1 Methods of Studying the Actions of Muscles 8

2.2 The Hill-Type Muscle Model 10

2.3 Type of Contraction 11

2.4 The Mathematical Model 13

2.4.1 The Simulation Model 14

2.4.2 The Musculoskeletal Model 15

2.5 The Human Jaw System 16

2.6 Static Equilibrium Jaw Models 21

2.7 Structure of The Dynamic Jaw Model 22

Chapter 3 **Object-Oriented Programming (OOP)** 26

3.1 The Evolution of Programming Style 26

3.1.1 Chaos and Functional Programming 27

3.1.2 Structured Programming and Data Abstraction 27

3.1.3 Object-Oriented Programming 29

3.1.3.1 The World According to Objects 29

3.1.3.2 Class and Object 31

3.1.3.3 Encapsulation, Inheritance and Polymorphism 32

3.2 Design and Classification 33

3.3 The Jaw System As An Object 34

3.4 Digital Continuous Simulation Systems 36

3.5 Numerical Integration Technique 37

Chapter 4 **Considerations of The Muscular Tendon Parameters** 40

4.1 Elasticity 40

4.2 Constants that Define a Muscle 41

4.2.1 l, l_1 and l_2 41

4.2.2 k_1, k_2 and b 42

4.3 Activation Level 46

Chapter 5 **Simulation Procedures and The Results** 49

5.1 Procedures of Test Running The Simulation Model .. 50

5.2 Simulation Results 53

5.3 Discussion 58

Chapter 6 **Neural Network and Dynamic Control** 63

6.1 Artificial Neural Network 64

6.2 Backpropagation Neural Network 66

6.3 Method for Improving The Performance of Backpropagation Networks 67

6.4 Neural Network as A Digital Controller 71

6.5 Conclusions of Using Neural Networks 75

Chapter 7 **Conclusions** 77

7.1 Limitations of The Current Model 78

7.2 Future Directions of The Jaw Model 79

References 81

List of Tables

Table 2.1 Muscle Attachment Coordinates 18

Table 2.2 Tooth and Joint Coordinates 18

Table 4.1 Stress-tension Relationships 43

Table 6.1 Comparisons of The Root Mean Square Errors 71

List of Figures

Figure 1.1 System and Model 3

Figure 2.1 The Hill-Type Muscle Model 11

Figure 2.2 The Simulation Model of Hill-Type Muscle 13

Figure 2.3 Series Elastic Element 14

Figure 2.4 Force Generator, Dash Pot and Parallel Elastic Element 14

Figure 2.5 A Simple Musculoskeletal System 15

Figure 2.6 Lateral and Front View of The Human Jaw System ... 17

Figure 2.7 Reference System used by The Model 18

Figure 2.8 The Temporalis 20

Figure 2.9 Frontal View of The Movement of The Mandible 23

Figure 2.10 The Jaw Model 24

Figure 2.11 The Artificial Bolus 25

Figure 3.1 Functional Programming 27

Figure 3.2 Structured Programming and Data Abstraction 29

Figure 3.3 An Object 30

Figure 3.4 Relationships Between Objects 31

Figure 3.5 The Jaw as An Object 35

Figure 3.6 Relationships Between Muscle Objects and The Jaw Object 35

Figure 3.7 Structure of A Digital Continuous Simulation Systems 37

Figure 3.8 Numerical Integration Error 38

Figure 4.1 Stress-strain Curve of Skeletal Muscle 43

Figure 4.2 Stress-strain Curve of Tendinous Tissue 44

Figure 4.3 Response of Mammalian Muscle under Maximum Stimulation 45

Figure 4.4 Activation Level 46

Figure 4.5 B-spline Curves as The Activation Levels 48

Figure 5.1 Structure of The Simulation Program 49

Figure 5.2 Graphics Interface of The Simulation Program 50

Figure 5.3 Comparisons of Activation Levels 51

Figure 5.4 The Simulated Incisor Movement 54

Figure 5.5 Incisor and Condyle Movement 55

Figure 5.6 Reaction Force at Condyle 56

Figure 5.7 Activation Level Patterns and Muscles' Tensions 57

Figure 6.1 Typical Nerve Cell 63

Figure 6.2 Artificial Neural Node 64

Figure 6.3 The Sigmoid Function 64

Figure 6.4 General Structure of An Artificial Neural Network ... 65

Figure 6.5 Updating Procedures in Artificial Neural Node 66

Figure 6.6 A Combined Artificial Neural Network 68

Figure 6.7 The Crab Arm Example 69

Figure 6.8 State Mapping 72

Figure 6.9 Neural Network That Performs State Mapping 72

Figure 6.10 State Mapping in Continuous Time Slices 73

Figure 6.11 An Example of The Neural Network Controller 73

Figure 6.12 The Casterian Product of S and C 74

Figure 6.13 Simulation Results
of The Neural Network Controller 75

Acknowledgment

The preparation of this thesis has been a long but exciting project and one which would not have been possible without the help and encouragement of my family, friends, and colleagues.

Special thanks to my supervisor, Dr. M. P. Beddoes, for his patience, encouragement, and help have been beyond measure and generously given.

The project is a cooperation with Dr. A. G. Hannam and Dr. G. E. J. Langenbach of The Dentistry Department. I am indebted to them for their helpful discussions and assistances.

In addition, I would like to thank my parents for their support and understanding throughout my graduate study.

Finally, I wish to thank all those who have contributed in one way or another to the completion of this thesis.

Chapter 1

Introduction

Muscles are biological machines that convert signals from the nervous system into chemical energy, force and mechanical work; it is only by the use of muscles that we are able to act on our environment -- to exert forces and to manipulate objects. Although there are computer simulation models [A4, A6] which define the properties of a single muscle and which use data collected from devices which record kinetics of human motion, few of them [A3, B14, B16] are capable of describing the dynamics of skeletal muscles that provide the internal force responsible for the movement of the body. The thesis deals with the design of a dynamic computer simulation model of a complex musculoskeletal system, the human jaw, for the use in the oral biology field. Different units in the simulation model are modular enough to allow its modification and reuse. Thus, it can be easily converted to model other musculoskeletal systems.

Studies of biological systems have shown new possible uses of artificial neural networks. The *central nervous system* (CNS) of living animals is the most flexible controller. The thesis will demonstrate how to provide control with neural networks without using the conventional control theory. Neural Networks offer new alternatives of approaching problems in dynamic control, but do not replace the conventional methods. In addition, a new method, *Error-Adjusting Networks*, is found to improve the performance of neural networks.

Components of the project involve the following fields: *dynamic behavior of muscle* and *the biological systems*, *mechanical system simulation*, and *artificial*

neural networks. The following sections in this chapter define the terms -- system and model. Terms used in modeling are also stated. Existing simulation models and the outline of the thesis are presented next. The last section of this chapter provides information about computer hardware and software aspects of the project.

1.1) System and Model

In this section we will briefly discuss some essential concepts that are closely connected to the modeling and simulation approach. Terms discussed here will be used throughout the thesis.

A *system* is an arrangement of units that function together to achieve a certain goal. A system may be composed of one or more subsystems which consist again of some sub subsystems, and so on. Every system interacts with its environment through *inputs* and *outputs*. Inputs have their origins outside the system and are not directly dependent on what happens in the system. Outputs, on the other hand, are generated by the system and interact with the environment. Elements which are necessary to outline the *states* of the system are defined as *attributes* (parameters and variables). Any process which changes the attributes of a system represents an action. All existing systems change with time, but when the rates of change are significant, systems are called *dynamic systems*. Their main feature is that their output at any instant depends on their history and not just on the current input. An *experiment* is the process of extracting data (outputs) from a system by testing the system with various inputs.

A *model* contains only essential aspects of an existing system or a system which we want to build, and the model can substitute the system to conduct an

experiment (Figure 1.1). This definition does not imply that a model is a computer program. A *simulation* is an experiment performed on a model.

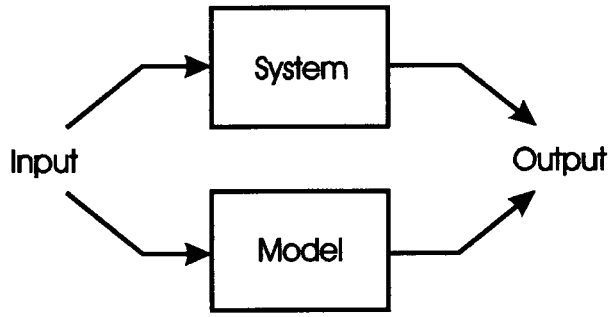

Figure 1.1
System and Model

The internal structure of the system and the initial state are usually known. With a complete system or model, we can perform one or both of the following operations:

1. When all inputs are known as functions over time, the task of the experiment is to determine the response of the system from its outputs. This problem is called the *direct* problem.

2. A second type of problem is where there is a set of desired outputs, and the goal is to solve for the unknown inputs. This is referred to as the *inverse* problem. The areas of inverse kinematics and automatic control in the robotics field are good examples.

1.2) Existing Simulation Models

Although there are many muscle contraction models [A3, B14, B16], they are usually limited by one or both of the following factors. First, they are designed to work as individual muscles instead of showing coordinate muscular actions that occur

in everyday acts such as raising the arm or chewing. Indeed, very few models have been developed to study integrated force from several different muscles. Second, they generally represent isometric contraction, in which muscle contracts without a change in length. However, despite these limitations, many existing models have produced realistic simulation results.

1.3 Applications

Computer simulation models that describe the dynamic behavior of the musculoskeletal system are very rare. For this reason, any new model offers several possibilities:

1. Transplantation of muscle attachment sites is not an unusual procedure in clinical settings. Changes in muscular function that are produced by skeletal abnormality or surgical corrections of abnormalities also occur. Hence, a dynamic simulation model which can be flexibly altered will give insight into how patients adapt to disorders, and how they might respond to surgical intervention.

2. Muscles are the interface between the CNS and the skeletal system. An understanding of the interface allows engineers to design prosthetic neuromuscular stimulation systems to restore lost or impaired motor function.

3. Direct measurements of muscle activity in living human beings are always invasive. Muscle activities that are located deep inside the human body may be impossible to measure. Measurement results are sometimes based on subjective estimates of muscle activities, and therefore, can be misleading. A dynamic simulation model can be used as a control against which to compare physiological recordings.

4. Study of artificial neural networks and methods to increase their performance may provide a possible general structure of automatic control, and can be compared with existing conceptualizations of nervous control systems in the CNS.

1.4) Thesis Outline

In Chapter 2, the study starts by defining a simulation model for the muscle, emphasizing how it contributes to the movement of the musculoskeletal system as it is attached to bone. The model is a simplified version of the real muscle but retains the essential factors to describe the biological behaviors of muscle. Hill is well known for pioneering work in muscle modeling, and the model reviewed and developed here is based on a Hill-type model. We will concentrate on the basic mechanical structure of a muscle and how its characteristics contribute to a musculoskeletal system. The way in which muscle utilizes energy sources, the way it produces heat, and the natures of the proteins that generate the force are not our concern here. The human jaw system, the existing simulation models and the basic structure of our new model will also be discussed in this chapter.

Chapter 3 introduces the concept of object-oriented programming (OOP). OOP allows one to program in the same way that one understands our world. One major focus of the project is to use of OOP's "expressive" power to model a complex mechanical system -- the human jaw. OOP is an abstract concept. In order to explain the concept clearly, a casual but adequate definition of OOP will be used instead of stating all the formal terminology of OOP. Efforts have been made to keep the discussions short and precise. Accuracy of the numerical approximation technique use in the project, and other run-time considerations are also discussed.

In Chapter 4, parameters used in the muscle model and the nerve impulses that feed into the muscle model are discussed. This chapter will utilize the conclusions of Chapter 2 and 3 to design a possible simulation structure for the jaw system.

Chapter 5 reports the simulation results, and the results are compared with known values of human mastication. As we will see, the simulation results imply some new hypotheses of movement in the musculoskeletal system.

Chapter 6 addresses neural networks, and the possible structure of neural network for dynamic control. A neural network is an engineered computational system modeled after or inspired by the learning abilities and parallelism of biological nervous systems. Neural networks are not programmed; they learn by example. Typically, a neural network is presented with a training set consisting of a group of examples (inputs and outputs) from which the network can learn. In response to this, the neural network compares its outputs to the standard outputs and adjusts the value of its internal weights. Usually the set of training examples is presented many times during training to allow the network to adjust its internal parameters gradually. The major use of a neural network is to classify different input patterns. However, by neural network for control we mean neural network that goes beyond classifying their input signals to influencing them.

Concluding remarks in Chapter 7 include current limitations of the model and suggestions for possible future directions for the project. The possible future uses of the model with abnormal muscles and jaw configurations are discussed.

1.5) Considerations of Hardware and Software Platform

The computer simulation model is designed to be a stand-alone executable program that can be run on IBM PCs or compatibles. The simulation model is also designed for use as a teaching tool in the oral biology field, therefore, a portable executable program that can run on PCs would be of great convenience.

As the simulation model is tested with object-oriented language syntax to ease complex system modeling, an object-oriented language is chosen for our purpose. C++ is found to be adequate because it is highly portable and powerful [D2, D3]. Borland C++ 3.1 is chosen from all the compilers available for its excellent integrated development environment and debugging tools.

Chapter 2

Muscle Mechanics and the Human Jaw System

Muscles and tendons are the interface between the CNS and the linked body segments. An understanding of the properties of this interface is important to scientists who interpret kinesiological events in the context of coordination of the body. Study of the musculoskeletal system with computer simulation models was not common until recently, and most studies have been based on biological measurements made in the past few decades. We start with summarizing the consequences of biological measurements as it is the basic of all the studies of biological movement.

2.1) Methods of Studying the Actions of Muscles

Human motions and muscle properties have been studied extensively in the past few decades, and methods of studying them are briefly summarized below:

1. *Anatomical* -- Dissection is used to study the location and attachments of a muscle and its relation to the joint it spans. This method provides a basis for visualizing the muscle's potential movements. Histological examination provides details of muscle fiber and tendon composition, often assisted by differential staining or labeling to reveal different fiber properties.

2. *Physiological*

   • Direct measurement of muscle length and tension changes are possible in animal preparations or in excised muscle tissue.

- Electrical activity can be recorded directly from muscles, and part of muscles, in experimental animals and living human subjects. Electromyography (EMG) is based upon the fact that a muscle generates electrical impulses when it contracts, and EMG is a technique of recording such impulses or action potentials. Surface or needle electrodes are placed close to the target muscle to do the measurement. EMG measures muscle activities during reflex and voluntary functions.

- The properties of groups of muscles can often be inferred indirectly by recording the displacement of bone, (e.g. limbs or jaw movement) or the forces generated at some target site.

3. *Physical Model* -- Elastic elements such as springs are fixed to the bones of a skeleton in such a way as to represent muscles. Tensions that develop in the springs and changes in their lengths can be demonstrated by moving the skeleton system manually, or by adjusting individual spring lengths.

Besides their contributions to the biological field, conventional measurement methods often fail to reveal correlated events in human living tissues, and limited numbers of subjects are available for experimental purposes. The requirement of expensive equipment is another factor that forbids direct measurement. In comparison, computer models can be an alternative to measurement methods using today's inexpensive computing power from PCs. Applications of the new models discussed in Chapter 1 are also new possibilities that are not possible with current measurement methods.

2.2) The Hill-Type Muscle Model

Different models of muscle have been defined mathematically, and used to estimate muscle forces during different motor tasks. Most of the latest models of muscle are based on the microscopic properties of the muscle tissue. However, the "black-box" approach (a model which only needs to be based on an input-output [I/O] description of the tissue) is more appropriate for our purpose. The selection is justified because our goal is to study the integrated force from a few different muscles instead of determining how different microscopic tissues make up the muscle force.

The Hill model [A4, A6] (Figure 2.1a) has withstood the test of time and is chosen as our base model.

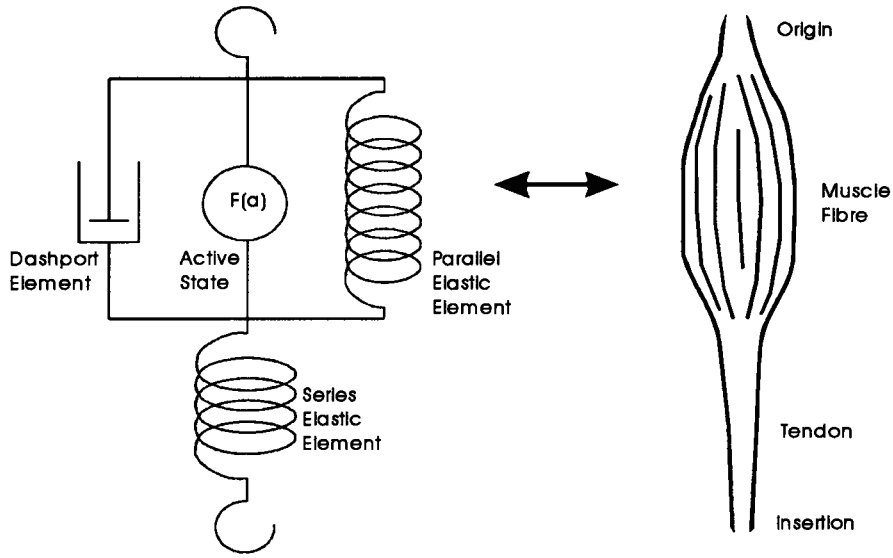

Figure 2.1a
The Hill-Type Muscle Model

Figure 2.1b
The Fusiform Muscle

The model comprises of a contractile element as a pure force generator (the *active state*) in parallel with a *dashpot element*. The contractile proteins in muscle cell that convert chemical energy into force and mechanical work make up the active state. Since muscle contains a considerable amount of water, viscosity of water

accounts for the viscous property of the dashpot element. While the muscle tendon makes up the *series elastic component*, the *parallel elastic component* resides in the muscle cell membrane, the connective tissue surrounding the muscle fibers, and the protein filaments that produce the contractile force. Together the parallel and series elastic components account for the passive tension properties of muscle [A4].

Historically, anatomy texts designated attachments of the two ends of a muscle as "*origin*" and "*insertion.*" The origin is usually characterized by stability and closeness of the muscle fibers to the bone. The insertion, on the other hand, frequently involves a relatively long tendon, and the bone into which the muscle's tendon inserts is usually the one that moves. A long tendon help prevents injury to the muscle during movement.

Figure 2.1b illustrates a fusiform-shaped muscle. This shape is characterized by its rounded muscle and gradual lessening width at either end. This is what people commonly perceive as the general shape of muscle. However, different structural forms of muscle exist, and we will study them when we reveal the musculoskeletal structure of the human jaw.

Before we look into the mathematical model of the muscle, the next section explains different types of contraction. This helps to explain the functional properties of muscle.

2.3) Type of Contraction

Although to contract literally means to "draw together" or to shorten, muscle contraction may exist when the muscle is shortening, remaining the same length, or

lengthening. A muscle contraction occurs whenever the muscle fibers generate tension in themselves. The followings reveal the three most common types of contraction.

1. *Concentric Contraction* - Concentric (toward the middle) contraction occurs when the tension generated by the muscle is sufficient to overcome a resistance and to move the body segment of one attachment toward the segment of its counterpart. The muscle shortens and, when one end is stabilized, the other pulls the bone to which it is attached and turns it about the joint axis. Usually, the muscle that undergoes a concentric contraction is directly responsible for effecting a movement and is classified as the *agonist* muscle, and muscles that cause the opposite movement from that of agonist are defined as the *antagonist*.

2. *Isometric Contraction* - Isometric means "equal length." In isometric contraction, external resistance is equal to the internal force developed by the muscle, and there is no external movement. There are two different conditions under which isometric contraction is likely to occur. First, muscles that are antagonistic to each other contract with equal strength, thus balancing or counteracting each other. The part affected is held tensely in place without moving. Tensing the biceps to show off its bulge is an example of this. Furthermore, a muscle is held in either partial or maximal contraction against another force, such as the pull of gravity or an external mechanical or muscular force. Holding a book with outstretched arm and attempting to move an object that is too heavy to move are good examples. Muscle undergoes isometric contraction can be a *fixator*, *stabilizer* or *supporting* muscle.

3. *Eccentric Contraction* - When a muscle slowly lengthens as it gives in to an external force that is greater than the contractile force it is exerting, it is in

eccentric (away from the middle) contraction. In most instances in which muscles contract eccentricity, the muscles are acting as "brake" or resistive force against the moving force of gravity or other external forces.

2.4 The Mathematical Model

In order to work interactively with other components of a musculoskeletal system, several things have been disregarded in our base muscle model. The muscle model needs an interface to the outer environment. In a real muscle, the active state is activated by nerve impulses from the CNS. While specialized sensor receptors known as *muscle spindles* within many muscles detect change of length and rate of change of length of the muscle, other receptors known as *Golgi Tendon Organs* in tendon detect the tension in the muscle [A2]. Figure 2.2 shows the model with the interfaces together with the variables needed to define the mathematical model.

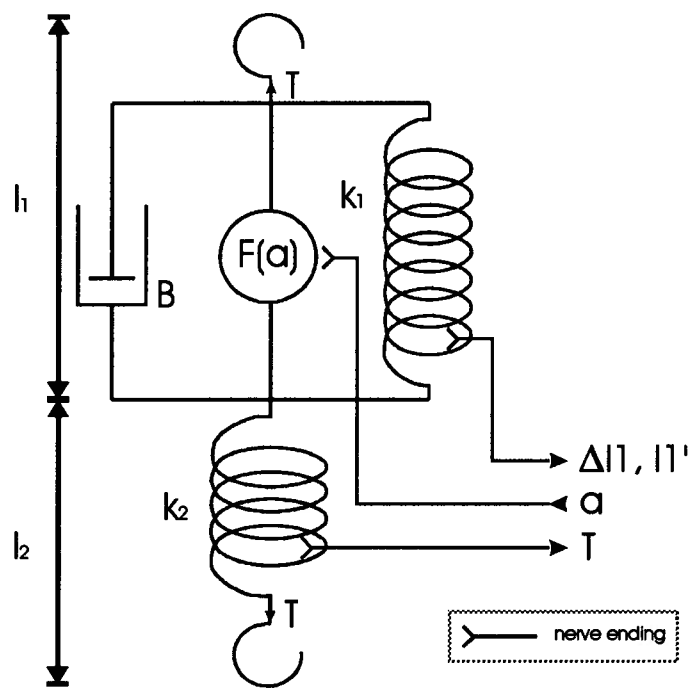

Figure 2.2
The Simulation Model of Hill-Type Muscle

2.4.1) The Simulation Model

In order to model the muscle system for simulation, we start by identifying subsystems that can be moved independently. We now cut the system open at the interfaces between the subsystems. The openings at both ends are replaced by two forces equivalent to the force act between the subsystems. These two "internal forces" are always of the same size but of opposite in directions [C1]. In our case, the muscle system can be cut between the tendon (series elastic element) and muscle fibers (active state, dash pot and parallel elastic element). The muscle system can be described with the following equations:

The reaction force on tendon:

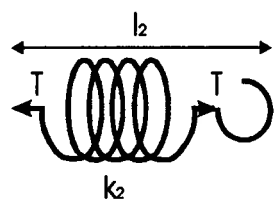

Figure 2.3
Series Elastic Element

$$k_2 \Delta l_2 = T \quad \text{.....} \quad (2.1)$$

The reaction force on active state, dash pot and parallel elastic element:

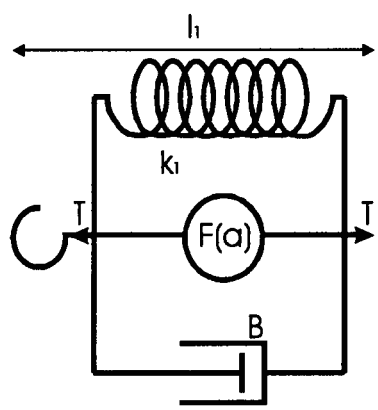

Figure 2.4
Force Generator, Dash Pot and Parallel Elastic Element

$$Bl_1' + k_1 \Delta l_1 + F(a) = T \quad \text{.....} \quad (2.2)$$

$$l_1' = \frac{T - F(a) - k_1 \Delta l_1}{B} \quad \text{.....} \quad (2.3)$$

(Note: if $\Delta l < 0$, then $k\Delta l = 0$.)

Equations (2.1) and (2.3) define the internal behaviors of a muscle and will be used as the basis to define our *simulation model* of muscle in Chapter 3.

2.4.2) The Musculoskeletal System

Up to this point, we have discussed the internal behavior of the muscle. In this section, we will go through the mechanical behavior of a single muscle in a musculoskeletal system. The basic idea is illustrated by Figure 2.3.

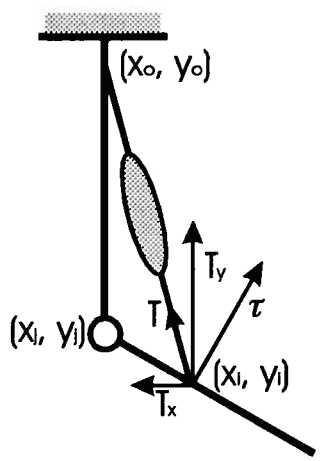

Figure 2.5
A Simple Musculoskeletal System

Resultant muscle force in the x and y-direction:

$$T_x = T \cdot \frac{(x_o - x_i)}{\sqrt{(x_o - x_i)^2 + (y_o - y_i)^2}} \quad \text{.....} (2.4)$$

$$T_y = T \cdot \frac{(y_o - y_i)}{\sqrt{(x_o - x_i)^2 + (y_o - y_i)^2}} \quad \text{.....} (2.5)$$

Torque (τ) due to the muscle force:

$$\tau = T_y(x_i - x_j) - T_x(y_i - y_j) \quad .....(2.6)$$

T_x, T_y and τ can be solved with simple trigonometric arithmetic, however, the above equations can generate faster code. Although the above illustrates a configuration of two dimensions, it is implemented with a three-dimensional design in the simulation model.

2.5) The Human Jaw System

The jaw is one of the most complicated single moving parts in the human musculoskeletal system. The jaw consists of the jaw bone with at least nine symmetric pairs of muscles pulling at different angles, and with different strengths on each side of the jaw [B12, B13]. Figure 2.6a and 2.6b show simplified lateral and frontal views of the musculoskeletal structure of the jaw. Small circles in Figure 2.6a and Figure 2.6b indicate the mandibular insertions of the muscles.

The system was modeled within a triaxial coordinate system centered on the right mandibular condyle. Figure 2.7 shows the reference system used by the model. The z-axis lays on the intercondylar axis, the x-axis runs parallel to the dental occlusal plane, and the y-axis is orthogonal to both. Coordinates describing the relative positions of vectors representing 18 principal jaw muscles, mandibular condyles and bite points are obtained from previously published data [B7, B14]. The nine muscles (or parts of muscles) on each side include the *anterior*, *middle* and *posterior temporalis*, the *superficial* and *deep masseter*, the *medial pterygoid*, the *superior* and *inferior lateral pterygoid*, and the *digastric muscle*. The attachment of the digastric muscle to the hyoid bone is fixed in space in our model.

Lateral View of The Human Jaw

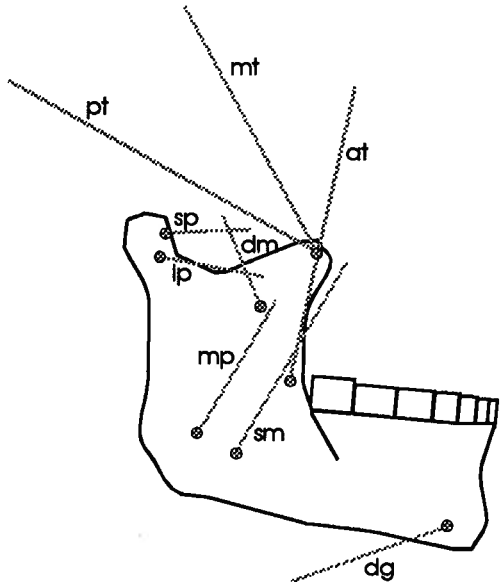

Figure 2.6a
sm = superficial masseter, dm = deep masseter, mp = medial pterygoid,
at = ant. temporalis, mt = medial temporalis, pt = posterior temporalis,
ip = inf. head lateral pterygoid, sp = sup. head lateral pterygoid, dg = digastric.

Front View of The Human Jaw

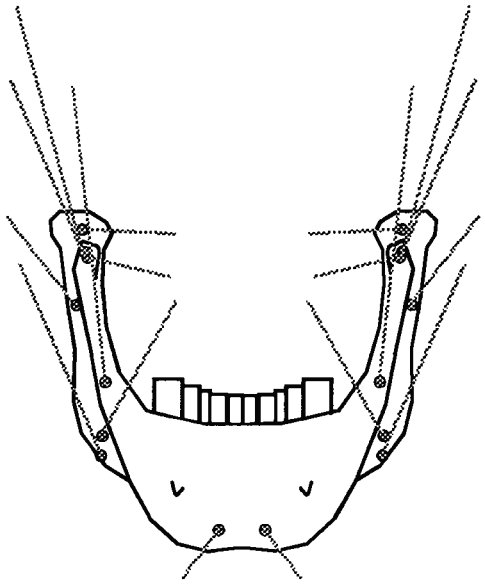

Figure 2.6b

Small circles show the insertion ends of the muscles in the jaw system.

Reference System used by The Model

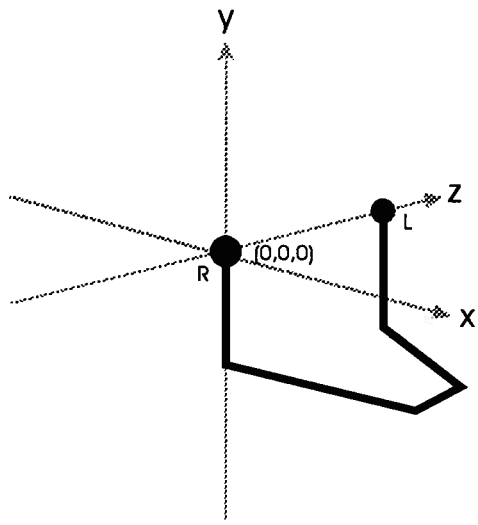

Figure 2.7
The figure illustrates the reference system
and arrows indicate the positive quadrant.

Table 2.1a and 2.1b show the physiological and anatomical parameters for each muscle group. (x_o, y_o, z_o) is the coordinate of the maxillary origin and (x_i, y_i, z_i) is the coordinate of the mandibular insertion. The first column contains abbreviations of muscles' names. Each muscle is assigned a specific cross-sectional area, and a constant of 40 N/cm^2 is then used to determine its maximum possible tension [B7, B14]. MAXF is the maximum possible contractile force of the muscle and is summarized in Table 2.1a and 2.1b.

Table 2.2 shows the coordinates of a complete set of lower teeth. The mandible's mass was assumed to be 100g, and the center of gravity is located at (0.04981, -0.048673, 0.045425). The moment of interia of the jaw is $6.084e^{-4} kg \cdot m^2$ when it rotates along the intercondylar axis. These parameters were estimated from a dry human specimen. All the coordinates refer to a normal closed jaw and the measurements are in standard SI units (meters and Newtons).

Muscle Attachment Coordinates

	MAXF	x_o	y_o	z_o	x_i	y_i	z_i
rsm	190.4	0.041501	-0.005996	-0.00885	0.015616	-0.048498	0.001675
rdm	81.6	0.017225	0.003167	-0.01245	0.024356	-0.01724	0.001625
rmp	174.8	0.025997	-0.015502	0.025275	0.007963	-0.045255	0.00595
rat	158.0	0.043005	0.041557	-0.0031	0.029027	-0.031155	0.00805
rmt	95.6	0.006519	0.057006	-0.01435	0.033807	-0.001944	0.000425
rpt	75.6	-0.02942	0.042005	-0.0161	0.03354	-0.002092	0.000275
rip	66.9	0.026238	-0.011053	0.0230	0.002947	-0.00305	0.003025
rsp	28.7	0.022312	0.001099	0.022775	0.003818	0.000931	0.0011
rdg	40.0	0.037259	-0.076011	0.0337	0.069928	-0.071325	0.0420

Table 2.1a

	MAXF	x_o	y_o	z_o	x_i	y_i	z_i
lsm	190.4	0.041501	-0.005996	0.0997	0.015616	-0.048498	0.089175
ldm	81.6	0.017225	0.003167	0.1033	0.024356	-0.01724	0.089225
lmp	174.8	0.025997	-0.015502	0.065575	0.007963	-0.045255	0.0849
lat	158.0	0.043005	0.041557	0.09395	0.029027	-0.031155	0.0828
lmt	95.6	0.006519	0.057006	0.1052	0.033807	-0.001944	0.090425
lpt	75.6	-0.02942	0.042005	0.10695	0.03354	-0.002092	0.090575
lip	66.9	0.026238	-0.011053	0.06785	0.002947	-0.00305	0.087825
lsp	28.7	0.022312	0.001099	0.068075	0.003818	0.000931	0.08975
ldg	40.0	0.037259	-0.076011	0.05715	0.069928	-0.071325	0.049

Table 2.1b

Tooth and Joint Coordinates

	Left			Right		
	x_m	y_m	z_m	x_m	y_m	z_m
condyle	0	0	0	0	0	0.09085
incisor	0.084058	-0.042117	0.045425	0.084058	-0.042117	0.045425
incisor	0.08302	-0.041706	0.0417	0.08302	-0.041706	0.04915
canine	0.079744	-0.040834	0.036425	0.079744	-0.040834	0.054425
premolar	0.073044	-0.040482	0.030975	0.073044	-0.040482	0.059875
premolar	0.06763	-0.040089	0.028025	0.06763	-0.040089	0.062825
molar	0.061696	-0.039867	0.02585	0.061696	-0.039867	0.065
molar	0.050855	-0.038728	0.023325	0.050855	-0.038728	0.067525
molar	0.041741	-0.037669	0.0199	0.041741	-0.037669	0.07095

Table 2.2

Jaw muscles are usually flat and located close to each other. Some are located deep to the mandible. The temporalis, masseter and medial pterygoid muscles are multipennate. That is they contain multiple, interleaved flat intramuscular tendon sheets to which muscle fibers insert obliquely. This close fiber packing is believed to impart greater fiber density in a minimum space. Figure 2.8 shows that pt, mt, and at actually belong to one single muscle whose fibers radiate from a narrow attachment at one end to a broad attachment at the other. Studies have shown that the line of actions of this muscle can be separated into three different parts as defined. (Muscles of this form are generally described as fan-shaped.) Partitioning also occurs in the masseter and lateral pterygoid muscles. The lines in the Figure 2.6a and 2.6b indicate the lines of action rather than the anatomical shapes of various muscles.

The Temporalis

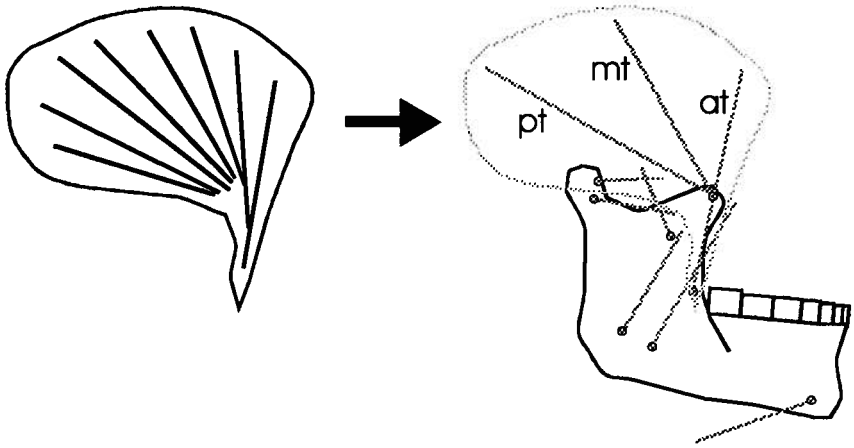

Figure 2.8
Although the temporalis is a single muscle,
the lines of actions show that the muscle can be
considered as three separated parts as defined.

The anatomical and functional complexities of the human masticatory system make it difficult to explain how muscles move the lower jaw and develop forces between the teeth, how the jaw's articulation works, and how growth, deviations in

form, and surgical or prosthetic treatment alter this process. While it is possible to measure many aspects of structure and function in living subjects, for example by imaging, electromyographic samplings, bite force recording and jaw tracking, and to shape behavior by defining voluntary tasks [B13], many important aspects of musculoskeletal function cannot be assessed because the methods used to study them are either impractical or invasive. Extrapolation of information drawn from non-human sources, an alternative approach, is unfortunately of limited value due to major inter-species differences in the face and jaws. The problem is compounded by variation in most human populations, which often makes it difficult to develop simple, working hypotheses to explain experimental observations. Increasingly, emphasis has been placed on computer models for this purpose.

2.6) Static Equilibrium Jaw Models

Static jaw models assume that the jaw is closed. Here the goal is to develop a bite force at a given bite point, and It follows from the linear algebra that six equations for static equilibrium of a rigid body have to be solved. However, there is no unique solution because the six equations contain more than six unknowns. The jaw system is a fail-safe system and this is responsible for the extra unknowns.

The idea of 'cost' is then introduced. Different costs are assigned to different muscles and joint forces, and the total cost of using the muscles and joint forces is defined as follow:

$$\text{Total cost} = \Sigma(\text{muscle force} * \text{cost of muscle force}) + \Sigma(\text{joint force} * \text{cost of joint force})$$

The objective is to find the pattern of muscle tensions that minimizes the total cost. The 'cost' technique is widely used in other static models of different musculoskeletal systems. Details of a typical static jaw model can be found in [B14].

These simulations of jaw mechanics with static equilibrium theory are useful. The jaw muscles are often active in the isometric state during symmetric and asymmetric biting and clenching, and models can provide insights that are otherwise unavailable. The expression of interactions between structure and function, including variables such as muscle activation, muscle tension, tooth and joint forces, can be developed with formal physical principles, and the models then become working hypotheses. They can predict results which are often testable. However, as mentioned earlier, a muscle driven dynamic model can offer a whole new dimension as compared to the 'cost' model.

2.7) Structure of The Dynamic Jaw Model

Few, if any, models have been developed to simulate the biology of jaw dynamics. The most sophisticated was the jaw model of a rat [B17], but the model was not based on formal physical principles.

As jaw is very complex, a few assumptions are necessary to make the jaw system feasible enough to model dynamically. The jaw model is a reduced version of the real thing, but still adequate to provide useful information.

The temporalis muscles are the largest jaw muscles in carnivores, and provide a large cutting force at the molars. The cutting action is mainly an up-down movement of the jaw. On the other hand, temporalis muscles are of relatively little use to herbivores. Temporalis in herbivores is small. They use their premolar and molar

teeth for grinding their food. The strong pterygoideus muscles in herbivores provide this side-to-side movement. Humans are omnivorous; we need both the cutting forces from the temporalis and grinding force from the pterygoideus. However, as stated in Table 2.1, temporalis are stronger than the pterygoideus in human. Figure 2.9 is an example of the frontal view of the jaw movement in a normal chewing cycle measured from the first incisor [B12]. The figure shows that magnitude of the up-down movement in the human jaw is about 3.25 times greater than that of the lateral movements. Therefore, the model will be concentrated on the up-down motion of the jaw as it is the major movement of a chewing cycle. A model that can describe both the vertical and lateral movement at the same time would be excellent, but the motion according to the two joints of the jaw makes it extremely difficult to model. In fact, the vertical motion can reveal some of the most important information in the human jaw system.

Frontal View of The Movement of The Mandible

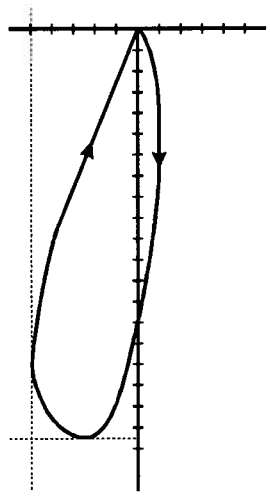

Figure 2.9
Measured at the lower central incisor teeth.
Unit is in (mm).

The up-down motion is not a pure rotation motion; the condylar heads of the lower jaw slide outward as it opens. Condylar guidance was simulated by providing a

frictionless constraint along a line angled at 30 degrees to the horizontal reference plane. Linear motion of the condylar center point was confined to this line. An unlimited sliding surface to the joint was provided in order to determine the extent to which condylar positioning could be controlled by muscle action alone. This is indicated in Figure 2.10. All muscular arrangements are expressed in three-dimensions.

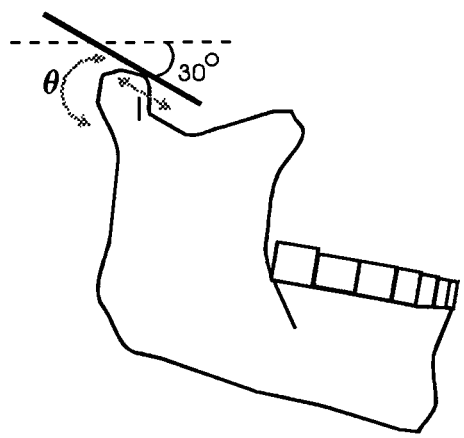

Figure 2.10
The Jaw Model

Assume that the mass of the jaw is m, and the moment of interia is I. The translational and angular acceleration (a and θ'') of the jaw are simply defined as follows:

$$a = \frac{dl'}{dt} = \frac{\Sigma T_x \cos 30^o - (\Sigma T_y + mg)\cos 60^o}{m} \quad \text{.....(2.7)}$$

$$\theta'' = \frac{\Sigma\tau + Torque_due_to_CG}{I} \quad \text{.....(2.8)}$$

The position of the first incisor is regarded as the first output of the computer model; it has often been used as the reference point to measure the motion of the jaw. In addition, changes of muscle length and tension are also valuable outputs. Most useful of all is the joint force as output. This is not measurable in living humans.

Midline chewing on a fixed food bolus in the first molar region is simulated. During simulated chewing, the "food bolus" is introduced by placing a constant force of 75N on the first molar bite point. The reaction force is effected on the first molar as it is the most common bite point, and 75N is well within the range expected during mastication. When the jaw makes contact with the bolus, the direction of the reaction force from the bolus is defined perpendicular to the line formed by the first and second molars. This reaction force remained at the same angle to the bite point irrespective of jaw position, and had to be overcome by muscle action for the jaw to return to its initial starting position, defined as *dental intercuspation*. The "bolus" is injected when the first molar bite point is 3mm from its initial, starting position during closing, and is removed when the opening cycle started. Figure 2.11 illustrates the arrangement of the bolus.

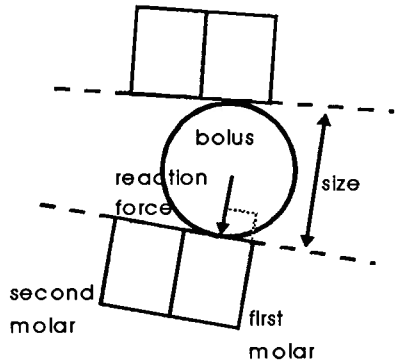

Figure 2.9
The Artificial Bolus

Chapter 3

Object Oriented Programming (OOP)

There is an extensive use of the object-oriented technology in the computer industry recently. Although the new technology can be implemented in many different areas (operating system and environment, computer hardware, etc.), the discussion here will be concentrated on *object-oriented programming* (*OOP*). Because object-oriented design is a relatively young practice, it may mean different things to different people. Therefore, the materials that follow will be based on a few different references [D1, D2, D3] and my personal experience of using object-oriented programming.

Although researchers claim that one of the main advantages of using OOP is to allow people to program the same way we understand our world, newcomers to OOP usually find the new concept difficult to learn (especially those who are already familiar with structured programming). Discussions are kept to be short and precise; main ideas will be illustrated with diagrams.

3.1) The Evolution of Programming Style

The programming community has seen different programming techniques come and go in its 40-year life span. The discussion will start with the review of the earlier programming styles then to our main subject -- OOP. A review of earlier styles is necessary to show how different styles decompose problems.

3.1.1) Chaos and Functional Programming

The earliest of programming styles is best described as *chaos programming* that has little organization either physically or logically, with jump and go-to commands sprinkled liberally throughout.

Functional Programming is the first major improvement over the chaos style, originally introduced as a way to reuse repetitive code. The most popular functional programming languages are FORTRAN and COBOL. In Figure 3.1, we see the topology of functional programming languages.

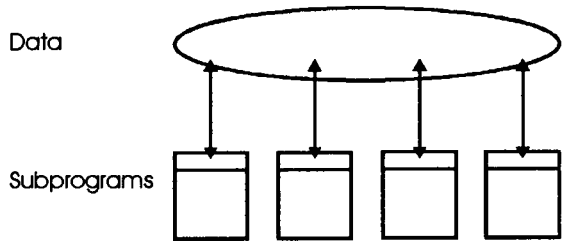

Figure 3.1
Functional Programming

Applications written in these languages exhibit a relatively flat physical structure, consisting only of global data and subprograms. The arrows in this figure indicate dependencies of the subprograms on various data. During design, one can logically separate different kinds of data from one another, but there is little in these languages that can enforce these design decisions. An error in one part of a program can have a destructive effect across the rest of the system, because the global data structures are exposed for all subprograms to see.

3.1.2) Structured Programming and Data Abstraction

In *structured programming*, the program is broken up into individual procedures that perform discrete tasks in a larger, more complex process. These

procedures are kept independent of each other, and each with its own logic. Information is passed between procedures using parameters, and procedures can have local data that cannot be accessed outside the procedure's scope. Procedures can be thought of as miniature programs that are put together to build an application.

A powerful concept was introduced with structured programming: *abstraction*. Abstraction could be defined as the ability to look at something without being concerned with its internal details. In a structured program, it is sufficient to know that a given procedure performs a specific task. As long as the procedure is reliable, it can be used without having to know how it completes its function. This is known as *functional abstraction*.

Although structured programmers were supposed to pass all data into and out through arguments, without powerful data structures this was often not possible. With *data abstraction*, data elements could be bundled together into more easily identified structures (Pascal calls these `RECORD`s; C calls them `struct`s). Data abstraction does for data what functional abstraction does for operations.

For larger programs, logically related subprograms are grouped together to form *modules*. The overall goal of the decomposition into modules is the reduction of software cost by allowing modules to be designed and revised independently. It should be possible to change the implementation of one module without knowledge of the implementation of other modules and without affecting the behavior of other modules.

Although modules are used to group logically related operations, the same data structures may be used in a few different modules as the arrows indicate. Figure 3.2 illustrates the topology of this style. As program grows in size, data types are

processed in many procedures within different modules. When changes occur in those data types, modifications must be made to every location that acts on those data types within the program. This can be a frustrating and time-consuming task in programs that contain thousands of lines of code and hundreds of functions.

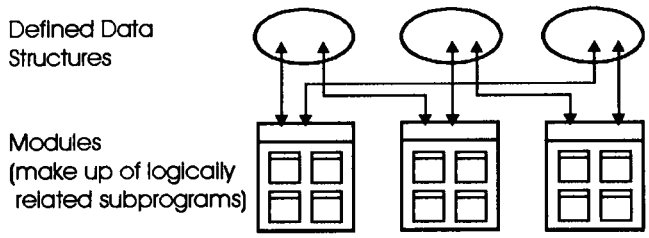

Figure 3.2
Structured Programming and Data Abstraction

What will happen if only one module will act on a single data structure? The following sections will answer this question.

3.1.3) Object Oriented Programming

While structured programming decomposes the problem into a set of operations, and modules are used to group logically related operations, OOP requires a different way of thinking about decomposition. The fundamental change is that an object-oriented program is designed around the data being operated upon, rather than upon the operations themselves. The next few sections will explain the above statement in more details. A definition of 'object' will also be given, followed by the terminology.

3.1.3.1) The World According to Objects

We experience our world largely as a vast collection of discrete objects, acting and reacting in a shared environment. An object in the real world can be simply

defined as something that can be identified and felt. *Identity* is the property of an object which distinguishes it from all other objects. The object should have a way of interacting with others in order to be felt, and *behaviors* are how an object acts and reacts to the outer environment. In addition, the *states* of an object record all the static and dynamic properties of the object, and states of an object can only be altered by its behaviors [D1]. *Message passing* to other objects is common behaviors of an object. Figure 3.3 illustrates the above ideas of an object:

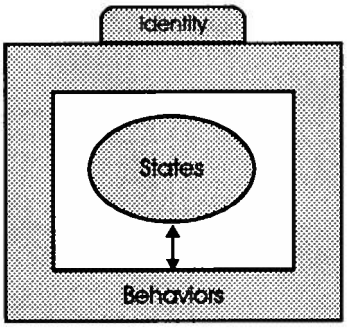

Figure 3.3
An Object

Consider a pop machine that dispenses soft drinks (i.e., the object's identity is a pop machine). The interface of this object consist of a slot for feed in of coins, a few buttons for user to make selection, and an opening for the emerges of drinks. The pop machine is usually in a state of "not ready for selection." However, the state changes to "ready for selection" after the right amount of coins are fed in. The total quantity of coins and number of pops the machine holds also make up the states of the machine. An object may contain other objects as the pop machine contains pops; it can also interact with other objects as the pop machine can interact with users.

Figure 3.4 shows a graph (not tree) structure with a few different objects interact with each other in the world they exist:

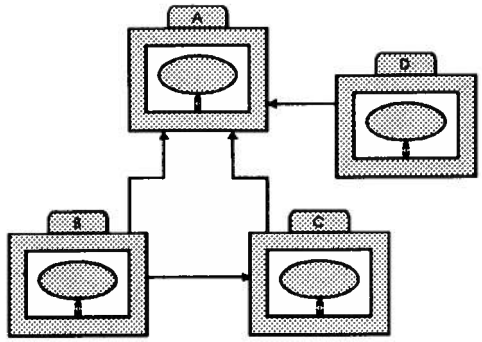

Figure 3.4
Relationships Between Objects

There are three different using relationships possible between objects. 'A' is a server object as it is only operated upon by others. 'B' and 'D' only operate upon other objects and are defined as the actor objects. Agent objects can both operate upon other objects and be operated upon by other objects, and 'C' is an agent object.

An object should represent an individual, identifiable item, unit, or entity, either real or abstract, with well-defined role in the problem domain. As you can see, almost everything in the world can be described as an object.

3.1.3.2) Class and Object

In OOP, a *class* is a template that describes both the data structures (states) and the valid actions (behaviors) for data items. When a data item is declared to be a member of a class, it is called an object. Assume we have the following C statement:

```
int  i, j, k;
```

`int` is the class while `i, j` and `k` are objects of the `int` class. Those functions that are defined as valid for a class are known as *methods* (such as `+, -, *` and `/` in integer), and they denote the way in which an object may act and react, and thus constitute the entire static and dynamics outside view (behaviors) of the object.

3.1.3.3) Encapsulation, Inheritance and Polymorphism

There are three main properties that characterize an OOP language: *encapsulation*, *inheritance* and *polymorphism*.

Encapsulation is the process of hiding all the details of an object that do not contribute to its essential characteristics. It focuses on the outside view of an object, and separates an object's essential behavior from its implementation. Actually, encapsulation and abstraction are complementary concepts.

With `struct` in C, we can define the structure and build specific operations and these specific operations only to manipulate the defined structure as in what we do with C++. In another words, it is possible to create objects with C. However, there is no compulsion in C to enforce this design. In addition, two more properties, inheritance and polymorphism are required to construct a complete OOP language.

Inheritance is the property that allows you to build new class from one or more previously defined base classes while possibly redefining or adding new data and actions. This creates a hierarchy of classes instead of building separated classes with similar properties. Besides encouraging reuse of existing codes and data structures, the main idea of inheritance is to capture the way that people classify things. We constantly relate new concepts to existing ones. We like to conceptualize the world as a tree-like structure, with successive levels of detail building on earlier generalizations. This is an efficient method of organizing the world around us.

Traditionally, operations are forced to use different names even they logically perform similar operations. In order to construct programs in a more natural way, objects are allowed to respond to the same operation (the "same" here means different

operations with the same name) with their own unique behavior. This characteristic is known as polymorphism in OOP.

Although OOP offers many new useful properties, you can use the structured programming style or even the chaos programming style to program whatever you can achieve from OOP. In fact, more computer programs have been developed with structured programming as compared to OOP. However, the advantages of using OOP are usually smaller codes. Well-defined classes are also much easier to be understood, maintained, and reused.

The followings show how a complex mechanical system can be decomposed into simpler objects. Behaviors will be assigned to the objects which make it possible for different objects function together as a complete system.

3.2) Design and Classification

A structured approach to programming is essential in OOP. In a structured program, up-front analysis is important to organize the application's functions effectively. You can use the same technique to do an object-oriented analysis of a project. You can't design classes unless you know the details about how the program's data is organized and processed.

In addition, it can require a substantial amount of preliminary work to create effective class libraries for a particular application. Defining the right set of classes for an application is critical to the effectiveness of the program. Unfortunately, there is no exact path to classification, nor the perfect class structure. Booch is well known in the OOP field and has written an excellent book on object-oriented design and analysis [D1].

To identify classes in the jaw model, however, is quite straight forward. The system basically consists of the mandible and muscles. The repetitiveness of muscles in a musculoskeletal system makes them difficult to model with conventional programming techniques as there are too many variables to be considered at the same time. This is responsible for the absence of a musculoskeletal system model that is directly driven by muscles. On the other hand, the 'repetitiveness' and 'similarity' properties of muscle make them easy to define as a class. As long as the muscle class is reliable, muscle objects define with this class can be used as building blocks of the jaw model.

There is only one moving part in the system -- the mandible. It is not a common practice to construct a class with only one object defining with it. In our case, however, by defining moving parts as a class makes the programming style more consistent and easier to cope in the complete model.

One more class is needed to handle continuous system simulation. Instead of physical existence, a process can also be an object. The following sections will explain the above designs in more details.

3.3 The Jaw System As An Object

The interface of the jaw system to the outer environment is simple. As an object, the input to the jaw consists of a bundle of nerves that activate different muscles in the system, and the useful outputs are the current status of the mandible and the muscles of the jaw system. This includes the angular and displacement information of the mandible, and muscles' tensions. Mathematical details of these displacements and tensions have already been defined in Section 2.5 and Section 2.8.

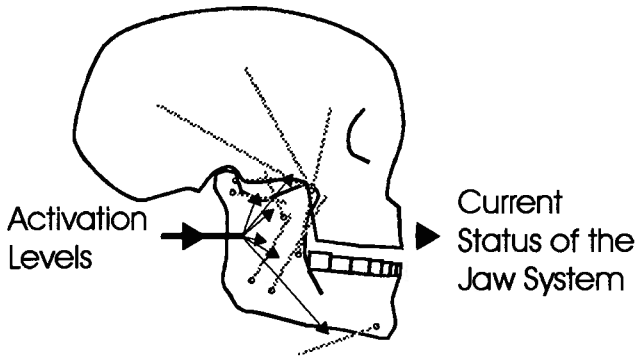

Figure 3.5
The Jaw System as An Object

The jaw in turn contains eighteen muscle objects and a mandible object. Each muscle is defined with the muscle origin, muscle insertion and maximum possible contractile force in the beginning. The first dynamic input to a muscle should be the activation level, and outputs are tension and torque on the mandible. In return, the mandible moves together with the insertion ends of the muscles. Therefore, the messages pass from the mandible to the muscles are change of insertion end locations. Figure 3.6 illustrates these relationships. The dynamic behaviors of the muscles and mandible have been defined in (2.1), (2.3), (2.4), (2.5), (2.6), (2.7) and (2.8). (2.1) and (2.3) are embedded as the internal behaviors of the muscle object. The results of using objects are that we can manipulate with the clearly defined interfaces of objects instead of a whole collection of mathematical equations. Objects can simply be used as building blocks to construct a complex system.

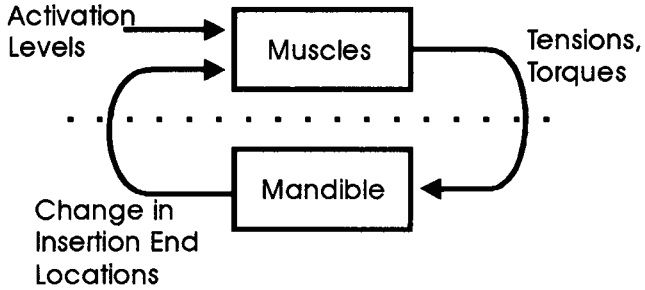

Figure 3.6
Relationships Between Muscle Objects and The Jaw Object

C++ is not a simulation language. Therefore, we have to include our own dynamic system module. The next section will discuss the concept of digital continuous simulation systems.

3.4 Digital Continuous Simulation Systems

We usually model dynamic system with a set of ordinary differential equations (ODEs), in general:

$$x'(t) = f(t, x(t))$$

In turn:

$$x(t) = \int x'(t)dt$$

Whenever we use a digital computer to simulate a continuous-time model (a set of ODEs), we must discretize the time axis in some way [C1, C2]. For instance, if simulation operates with constant independent-variable increments Δt (calculation interval), we can discretize the time axis so that differential equations become difference equations:

$$\frac{x(t+\Delta t) - x(t)}{\Delta t} = f(t, x(t))$$

or

$$x(t+\Delta t) = x(t) + f(t, x(t))\Delta t$$

The discrete event simulation is given by a two-step interation: the first step consists of the evaluation of all derivatives and the second includes the integration procedure, which evaluates the state variables for the next calculation interval. This two-step iteration is usually implemented in simulation systems with two subprograms (or behaviors in object) -- DERIV and INTEG [C2]. The basic concept of digital simulation systems is shown in Figure 3.7.

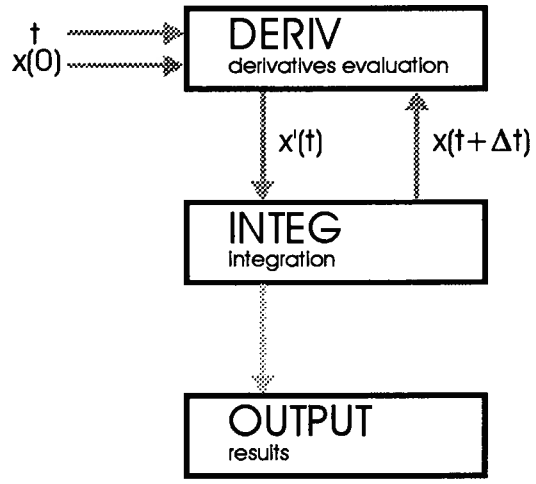

Figure 3.7
Structure of A Digital Continuous Simulation Systems

During simulation the integration procedure requires many evaluations of state derivatives (depending on the integration algorithm). In the prescribed time instants the control is given to the OUTPUT subprogram which supplies the user with simulation results.

Besides the basic mechanisms of digital continuous simulation system, our jaw model should not be contained much more than specifying the relationships between different muscle objects and the mandible object by calling the according DERIV, INTEG and OUTPUT behaviors of the objects.

3.5) Numerical Integration Technique

The integration algorithm chosen here is the simple Euler method [C2]:

$$y_{n+1} = y_n + hy'_n$$

where n = 0, 1, 2, ...

There are more efficient integration algorithms available. However, the works here are concentrated on how different features of a complex system can be

distributed into different objects, and the Euler method is easier to implement for our purpose.

No matter how good an integration algorithm is, it only gives approximation values to the true solution. Numerical approximation errors are limitations from the integration algorithm [C2]. However, a smaller calculation interval can reduce this error. On the other hand, smaller calculation interval will introduce another error -- the roundoff error. In practice, integration algorithms are implemented by computer arithmetic with finite precision (number of bits). This leads to the roundoff errors. Roundoff errors accumulate and become increasingly serious with decreasing calculation interval, since a smaller interval means more calculation intervals for given $t_{max} - t_0$. Figure 3.8 shows the relations between the numerical approximation error, the roundoff error and the total error.

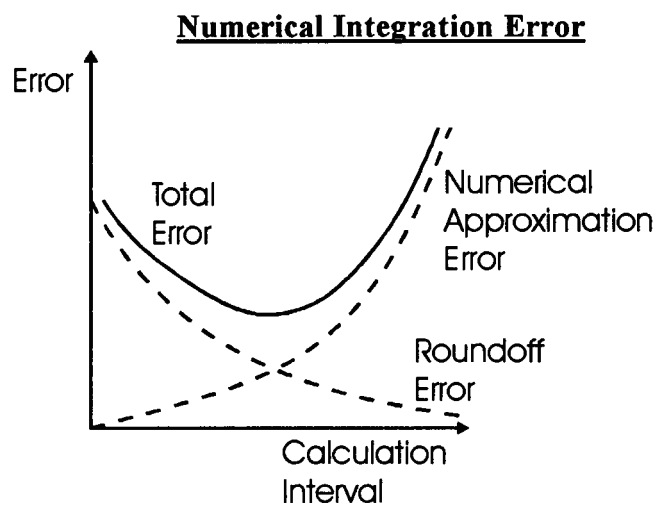

Figure 3.8
Total Error = Numerical Approximation Error + Roundoff Error

In Borland C++, there are three type of floating point numbers with different accuracy. They are `float` (32 bits, 7-digit precision), `double` (64 bits, 15-digit precision) and `long double` (80 bits, 19-digit precision). The type `long`

`double` is chosen in our simulation to minimize the roundoff error. Difference calculation intervals have been tested, and a 0.01ms calculation interval produces stable solutions for the simulation model. A smaller interval may produce a better approximation, but 0.01ms is a good compromise for efficiency and accuracy. With the calculation interval unchanged, the system has been tested by replacing `long double` with `double` and `float`. There is a big difference between the outcomes of using `long double` and `float`. However, the difference between the results of using `long double` and `double` are insignificant. We can assume that the roundoff error has been taken care of by the extra precision with `double`. All floating point numbers in the model are implemented with `long double` for better consistency of solutions.

The model has been tested with different input values, and is believed to generate reasonable outputs with the above setup. Details of the results can be found in Chapter 5. Long running time are penalties of using the Euler method; a small calculation interval and a long floating number are needed for better accuracy. The program is complied C++ code, and a total simulation time interval of 0.5s takes a 486 33MHz PC approximately two minutes to yield the simulation results.

Chapter 4

Considerations of the Muscular Tendon Parameters

In the last few chapters, we have discussed the mathematical and programming concerns of the jaw model. The mathematical model is defined and different parts of the model are clearly organized with objects. However, before we can perform simulation, the simulation model needs a few more pieces of information. First, as defined in Section 2.5, constants k_1, k_2, l, l_1, l_2 and B are needed to define a muscle. Second, we need an input signal, *activation level*, to activate the muscle. Before we look into the constants use in the muscle model, a few terms that are used to describe elasticity are defined in the next section. Different constants that define a muscle and the activation level will be presented next.

4.1) Elasticity

When a force acts on a body or material, resisting forces within the body react. These resisting forces are called *stresses*. Stress is measured by the force applied per unit area which produces deformation in a body. The unit of stress is expressed as N/m^2. Thus:

$$Stress = \frac{F}{A}$$

The ratio of length after stress is applied, to original length, is defined as a *strain*. Because it is a ratio of length, strain has no dimensions or units.

$$Strain = \frac{\Delta l}{l}$$

The numerical relationship between stress and strain was first discovered by Robert Hooke. Hooke's Law states that there is a constant or proportional arithmetical relationship between force and elongation. The *modulus of elasticity* is defined as the stress required to produce one unit strain.

$$\text{Modulus of elasticity} = \frac{F/A}{\Delta l / l} = \frac{Fl}{A\Delta l}$$

The above relationship holds until the elongation reaches a point known as the *elastic limit*. The elastic limit is the smallest value of stress required to produce permanent strain in the body. Below the elastic limit, materials return to their original length when the deforming force is removed. However, the result of applying a force beyond the elastic limit is that the stressed material will not return to its original length when the force is removed. In addition, materials elongate much further for each unit of force above the elastic limit. Elastic materials in biological systems such as muscles and tendons are arranged to work in conditions that the tissues always operate below their elastic limits. Beyond the elastic limit will cause injuries to the tissues [A1].

4.2) Constants that Define a Muscle

Constants k_1, k_1, l, l_1, l_2 and B are needed to define a muscle. The next section will define l, l_1 and l_2 which describe the dimension of the muscle, k_1, k_1 and B that describe the dynamic behaviors of the muscle will be followed next.

4.2.1) l, l_1, and l_2

Muscles are believed to be under small amount of passive tensions when they are attached to the skeleton system. If they are removed from the skeleton system, the muscle length will usually be shortened by almost ten percent as compared to the

original length. The original length of an uncontracted muscle is defined as the *resting length*, and the length of an isolated muscle is defined as the *equilibrium length*.

To simplify implementation, the muscles in the jaw system are assumed to be under no tension in a completely closed jaw. The muscle insertion and origin of a closed jaw define the total length, l, of the muscle. In fact, the absolute resting lengths of different muscle are difficult to predict mathematically.

A complete muscle is made up of muscle fibers and muscle tendons. As defined in Section 2.4, l_1 is the length of the muscle fibers and l_2 is length of the muscle tendons. Informal measurement indicates that the ratio of the fibers to tendons is about 5:1, and this ratio is used in every muscle of our model. There is no exact value for this ratio as there is a fuzzy edge between the muscle fibers and the tendons, and an exact value is not necessary either. Our concern here is to simulate the general moving pattern of a jaw pull by a set of muscles, but not the detailed internal behaviors of specific muscles.

4.2.2) k_1, k_2 and B

k_1, k_2 and B are directly proportional to the thickness or size of the muscle. In turn, the size determines the maximum contraction force possible of the muscle. Therefore, the following assumptions can be made:

$$k_1 = \textit{maximum contraction force} * \textit{constant}_1,$$
$$k_2 = \textit{maximum contraction force} * \textit{constant}_2,$$
$$B = \textit{maximum contraction force} * \textit{constant}_3.$$

The approximation values of these constants can be obtained from the behaviors of the biological tissues.

Muscles for mastication always operate within the elongation range of 45%, and muscles are stretched by about 20% in a normal chewing cycle. Figure 4.1 shows an estimated stress-strain curve of the masseter muscle [A5].

Stress-strain Curve of Skeletal Muscle

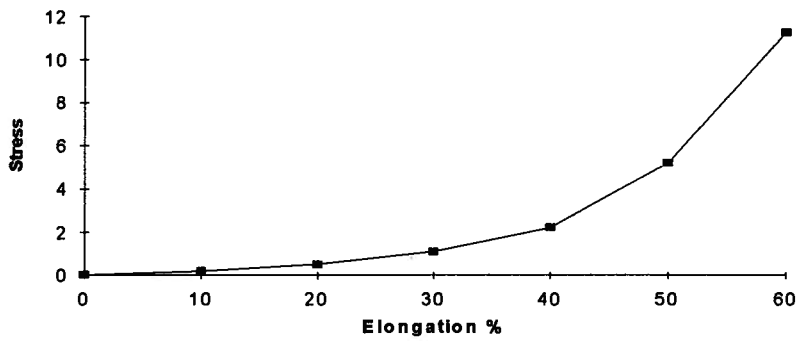

Figure 4.1
(y-axis in g/mm^2)

Base on the stress-strain curve, we have Table 4.1.

Stress-tension Relationships

Elongation	Stress	Length	Area	Tension
0%	0	1	1	0
10%	0.2	1.1	0.909	0.1818
20%	0.4	1.2	0.863	0.4165
30%	1.1	1.3	0.769	0.8459
40%	2.2	1.4	0.714	1.8708
50%	5.2	1.5	0.67	3.484
60%	11.25	1.6	0.625	7.03125

Table 4.1
Column 2 and 5 indicate the relationships between stress and tension.

Columns 3, 4 and 5 are the length, cross sectional area and the passive tension of a muscle. As these three columns represent ratios and they do not refer to any particular muscle, units are not necessary. Assuming that the muscle will reach the elastic limit when it is stretched by 50% of the original length and it can withstand its own

maximum contractile force at that point, the magnitude of the passive tension on the same muscle that is stretched by 20% is:

$$= 0.4165/3.484 * \textit{maximum contractile force},$$
$$= 0.1195 * \textit{maximum contractile force}.$$

The force required to stretch a normal spring obeys the following equation:

$$F = k\Delta x$$

In this expression Δx denotes the amount by which the spring is stretched from its unstrained length. The term k is a proportionality constant called the *spring constant* and has dimensions of force per unit length (N/m). If the muscle operates within the elongation range of 20% and assumes that the passive tension and strain have a linear relationship in this range, this equation can be implemented as following:

$$F = 0.1195 * \textit{maximum contractile force} * (\Delta l_1 / l_1) / 0.2$$

or

$$k_1 = 0.5977 * \textit{maximum contractile force} / l_1$$

Tendinous tissues are much stiffer than muscle fibers, and Figure 4.2 shows the stress-strain curve of the tendinous tissues [A5].

Stress-strain Curve of Tendinous Tissue

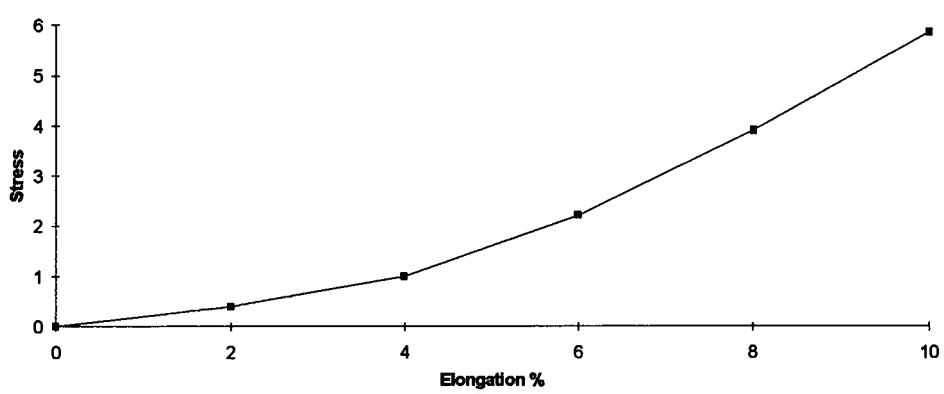

Figure 4.2
(y-axis in kg/mm^2)

Assume that a tendon is under the maximum contractile force when it is stretched by 4%, the passive force from the tendon can be described as follow:

$$F = \textit{maximum contractile force} * (\Delta l_2 / l_2) / 0.04$$

or

$$k_2 = 25 * \textit{maximum contractile force} / l_2$$

However, the ratio of the cross section area of tendinous tissue/muscle fiber in the mastication muscle is roughly four times higher than most other skeletal muscles, k_2 implemented in the model is:

$$k_2 = 100 * \textit{maximum contractile force} / l_2$$

The last constant, B, defines the response time of the muscle, and Figure 4.3 shows the response of a mammal muscle when it is fully activated.

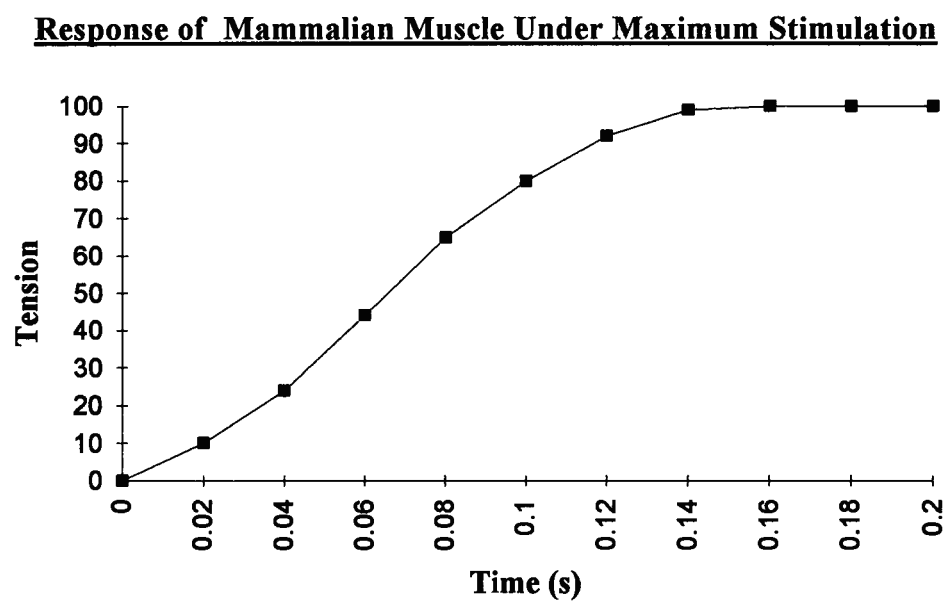

Figure 4.3
(y-axis is tension in %)

The graph shows that the muscle will reach its maximum strength at around 0.14s after it is fully activated. Simulation of the isometric contraction of a single muscle shows the magnitude of B is approximately (5 * maximum contractile force) in order to satisfy the above criteria.

This section reveals a possible way of defining the approximate values for k_1, k_2 and B, but this is by no means that there is only one way to define these values. Data gathered for biological tissues vary a lot, and sometimes contradict each other. The method discussed above is believed to be direct and easy to implement. There are limitations with the constants defined here; they are all non-linear functions in real biological system instead. However, k_1, k_2 and B define here are more adequate for the first test run of the model.

4.3) Activation Level

The jaw movements of the model are considered the result of voluntary drive by the CNS, and studies have shown that control signals from the CNS for voluntary movement can be change as fast as fifty to sixty times per second, therefore, the frequency of the activation levels is chosen to be 50Hz in our model.

In addition, the activation levels from the CNS are usually quite continuous. Figure 4.1 shows a possible measured raw EMG graph and the shape of the activation levels after it is rectified and filtered. The rectified and filtered signal is believed to be the original activation levels from the CNS.

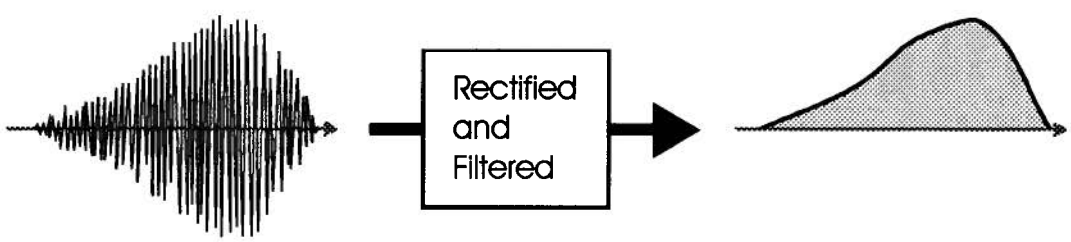

Figure 4.1
Activation Level

In order to define the activation levels freely for the model, the activation levels are designed to be defined with a B-spline curve. In drafting terminology, a

spline is a flexible strip used to produce a smooth curve through a set of plotted control points. The term spline curves, or spline functions, refer to the resulting curves drawn in this manner.

Given an input set of $n+1$ control points p_k, with k varying form 0 to n, we define points on the approximating B-spline curve as:

$$P(u) = \sum_{k=0}^{n} p_k N_{k,t}(u)$$

where the B-spline blending functions $N_{k,t}$ can be defined as polynomials of degree $t-1$. The blending function is recursively defined as:

$$N_{k,1}(u) = \begin{cases} 1 & \text{if } (u_k \le u < u_{k+1}) \\ 0 & \text{otherwise} \end{cases}$$

$$N_{k,t}(u) = \frac{u - u_k}{u_{k+t-1} - u_k} N_{k,t-1}(u) + \frac{u_{k+t} - u}{u_{k+t} - u_{k+1}} N_{k+1,t-1}(u)$$

Any terms with value of 0 as the denominators are assigned the value 0 during the recursive calculations.

The defining positions u_j for the subintervals of u are referred to as breakpoints. Breakpoints can be defined in various ways. A uniform spacing of the breakpoints is implemented here and is defined as:

$$u_j \begin{array}{ll} = 0 & \text{if } j < t, \\ = j-t+1 & \text{if } t \le j \le n, \\ = n-t+2 & \text{if } j > n. \end{array}$$

for values of j ranging from 0 to $n+t$.

The B-spline curve implements in the model has five control points. Figure 4.5 shows an example of activation level curve generated with five control points. More complex form of activation levels can be generated by joining a few different B-spline curves.

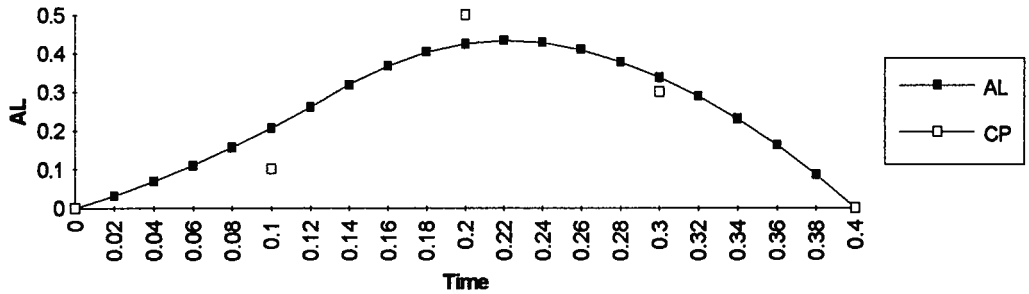

Figure 4.5
B-spline Curve as The Activation Level

With the final information described here, we are ready to test run our simulation model of the jaw. The next chapter consists of procedures of test running the model and details of the simulation result. Although the model is designed to handle abnormal settings of the jaw and activation levels, the first run is concentrated on working with a normal setting of the jaw. The model is convincing if it responds favorably, under normal circumstances, in a way compatible with the literature [B1, B12, B13].

Chapter 5

Simulation Procedures and The Results

Because of the complexity of the model, it is separated into three computer programs. ACTGEN takes the control points as inputs and generates the activation levels for different muscles. The activation levels are then fed into SIM, and SIM simulates the jaw system according to the input activation levels. The final result can be visualized with the graphics interface -- SHOW, or imported to a spreadsheet program. The ACTGEN and SIM can be put into a batch program to simulate a few different patterns. The structure of the programs is shown in Figure 5.1.

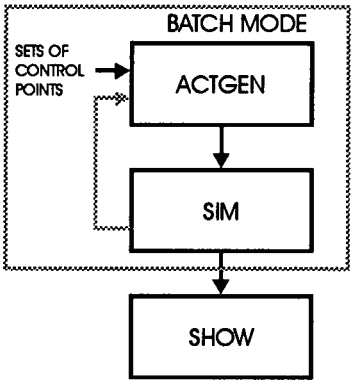

Figure 5.1
Structure of The Simulation Program

SHOW is a graphics interface that shows informations about the jaw and different muscles, and most of the effort has been spent on SIM that performs the simulation. However, the graphics interface helps people to understand the simulation results much better. Figure 5.2 shows the output of the graphics interface.

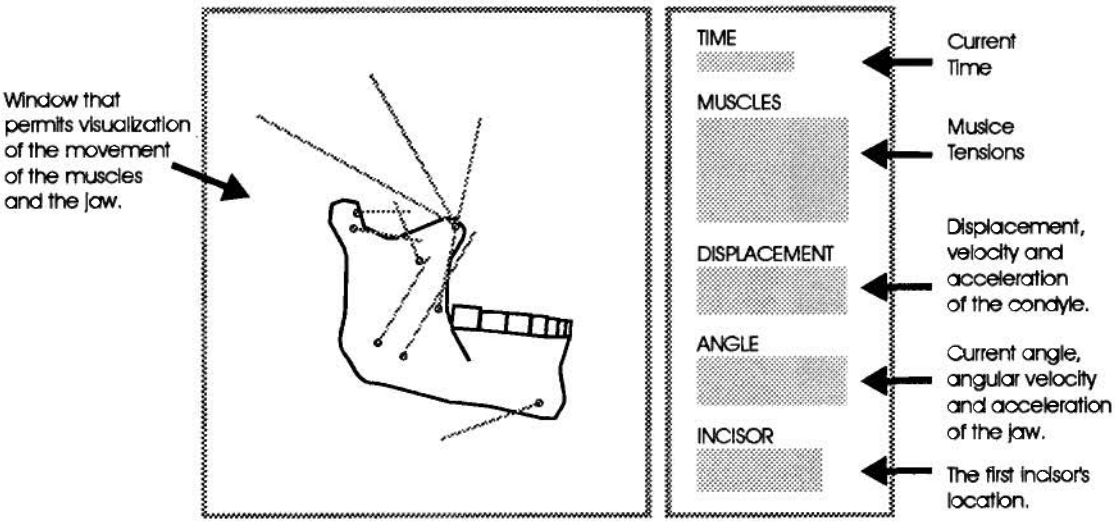

Figure 5.2
Graphics Interface of The Simulation Program

5.1) Procedures of Test Running The Simulation Model

The computer simulation model of the jaw was fully tested in the Faculty of Dentistry's Craniofacial Laboratory at UBC, and provided satisfactory results. The patterns of muscle activations for simulated chewing by the model are shown as heavy lines in Figure 5.3. Where possible, comparisons have been made with muscle activation patterns derived from Moller [B13] as indicated with the light lines in the figures. Moller's measurements of the muscle activations in the chewing cycle are believed to be the most accurate and complete in the oral biology field.

The duration of the chewing cycle was fixed at 700ms in our simulation model, representing a value within ranges reported in the literature [B1 (600-1000ms), B13 (435-865ms)]. Chewing cycle durations depend upon the nature of the food, tougher and stickier foods requiring longer times [B1]. The duration selected for this study is typical for apple and gum chewing, and very close to the duration cycle of the mean electrical activity for several jaw muscles published by Moller, (1966).

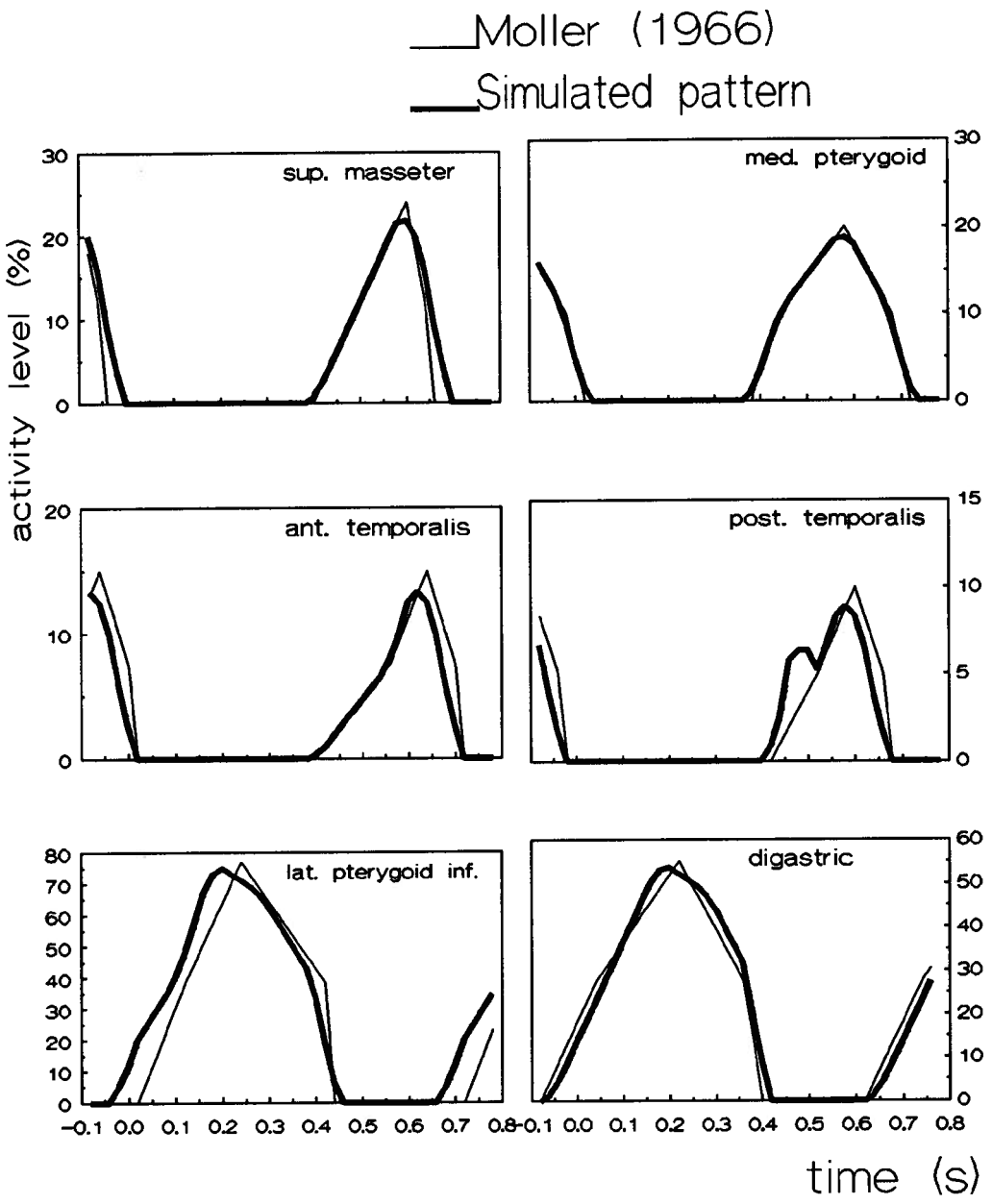

Figure 5.3
Comparisons of Activation Levels

Initially, each activation level was generated by assigning the onset, 50% peak, peak amplitude, 50% peak, and cessation of activity from published, mean electromyographic data [B13, p.103, Fig. 143]. Straight-line curve fits between these points were smoothed with a simple three-point averaging filter. Moller's data did not include values for the middle temporal muscle or deep masseter, nor did they distinguish between activity in the upper and lower heads of the lateral pterygoid muscle. Therefore values were assigned to the middle temporal muscle which were between those for the anterior and posterior parts. Since activity in the deep masseter is similar to that in the anterior temporal muscle during gum chewing [B2] values for the deep masseter were matched accordingly. Although Moller's data describe biphasic activity in a single lateral pterygoid muscle, newer data suggest that the inferior part of the muscle is inactive in jaw closing during mastication [B5, B12, B22]. The upper and lower parts thus activate reciprocally, i.e., the superior lateral pterygoid is activated synchronously with the anterior temporal muscle during the closing phase of the chewing cycle [B5]. In the present study, Moller's data for the opening phase was assigned to the inferior lateral pterygoid, and his data for the closing phase to the superior part of the muscle. Since mid-line chewing was simulated, all muscle groups were considered to act symmetrically, i.e., they were activated in matching pairs on the right and left sides.

The model was driven from a position of assumed rest where there was no activity in any muscle. Motion of the jaw was analyzed from the lateral aspect, with emphasis on the incisor point and the center of the mandibular condyle. Minor corrective changes were then made to overall muscle amplitudes (but not timing) to create an incisor point trajectory which fell within the mean range of data published for human chewing [B1]. Minor scaling was considered acceptable, because electromyographic data itself is a relative measure of muscle activation. Since

condylar motion was unconstrained except for angle, muscle "balance" became critical during the last part of the closing sequence. Many muscles were active, each with a different line of action, and the goal was to bring the mandible to its start point without sliding past it. Accordingly, small adjustments, especially to the temporalis muscle group, were important in this phase. Figures 5.3 illustrates the similarity between the simulated patterns of contraction, and those believed to occur in the biological jaw. Notable features include slight differences in symmetry between the digastric and inferior head of the lateral pterygoid muscles during jaw opening, and early activation of the medial pterygoid, coupled with relatively late peaking of activity in the temporalis muscles during jaw closing. The small, early burst of activity in the middle and posterior temporal muscles was needed in the model to maintain a good trajectory of jaw movement when the bolus was hit, and may or may not be present in human electromyographic responses when a similar bolus is used. Once the first open and close sequence of chewing was achieved, the model was allowed to cycle by driving it repetitively with the same muscle activation patterns. This was done to observe any effect of phasic changes in muscle properties on subsequent cycles.

5.2) Simulation Results

The shape and temporal characteristics of the chewing stroke produced by this muscle drive are shown in Figure 5.4. The opening and closing strokes nearly superimposed, and were angled towards posteriorly at approximately 70 degrees to the dental occlusal plane. The gape at maximum jaw opening was 20.9mm. The movement trajectory, which completed one cycle every 700ms, reached maximum gape at 360ms. The opening phase was slower than the closing phase, which showed characteristics of natural chewing such as a pause in closing when the bolus was hit,

slowed movement through the bolus, and the dwell phase in the "*intercuspal*" or start position before opening began for the next cycle.

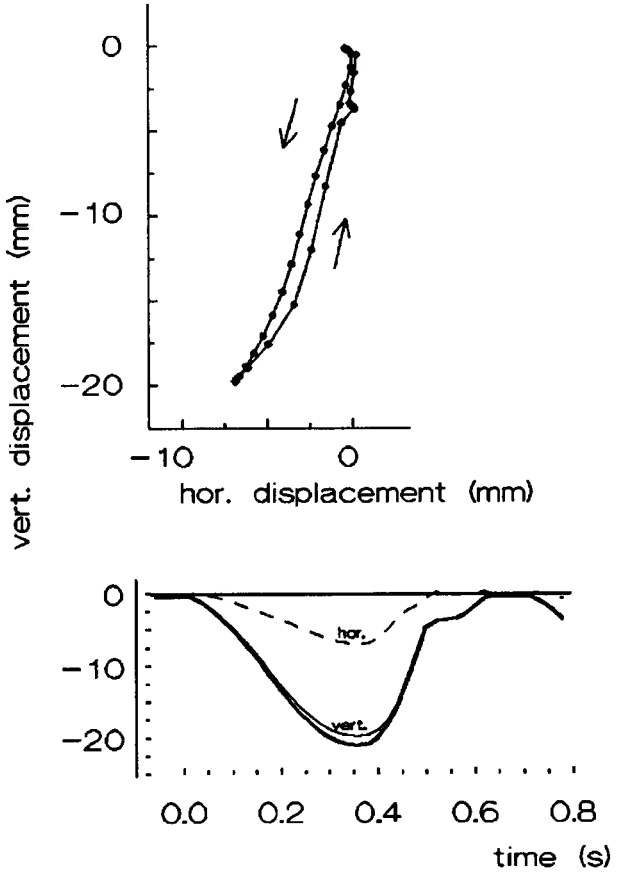

Figure 5.4
The Simulated Incisor Movement

Figure 5.5 illustrates changes in position and linear velocity of both the incisor and condylar points during the same cycle. At the incisor point, the linear velocity was 91mm/sec when the jaw was approximately halfway open, and reached 192mm/sec when it was between a third and halfway closed. At the condyle, the peak opening linear velocity was 26mm/sec about halfway through forward condylar translation, which reached 5.2mm at maximum gape. During closing, condylar velocity reached 56mm/sec at the same time as the incisor point reached its peak velocity. Both the

incisor and condylar closing velocity curves showed second, smaller peaks coincident with the molar's passage through the "bolus."

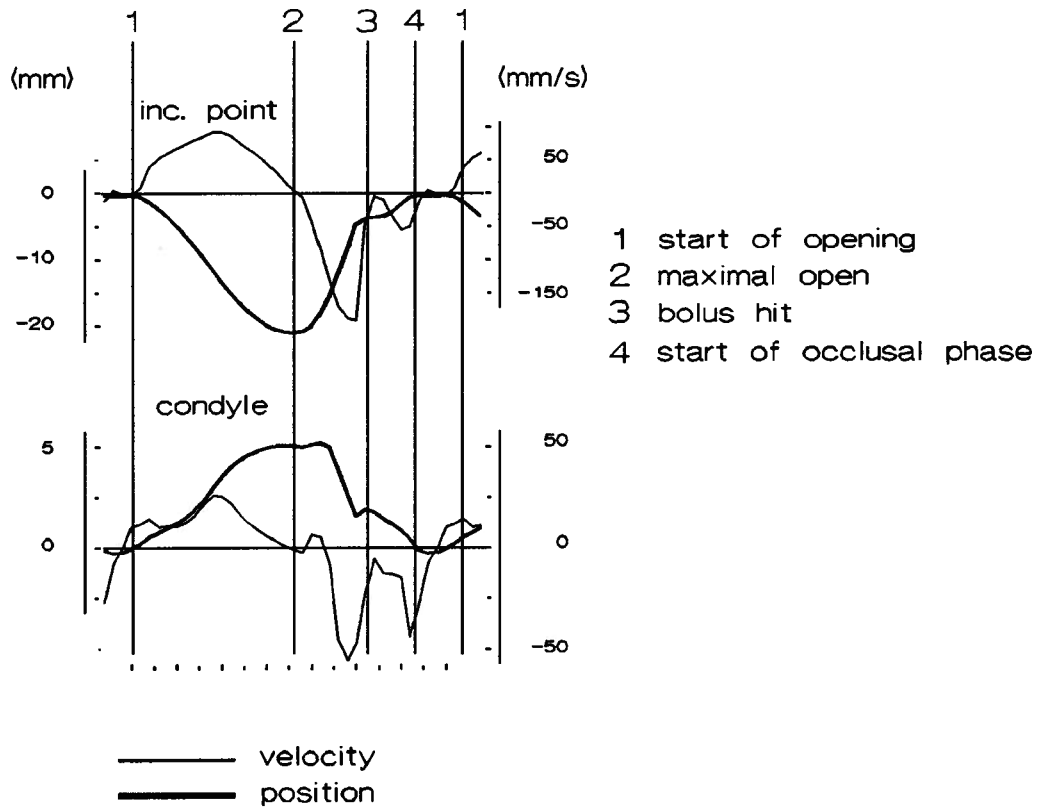

Figure 5.5
Incisor and Condyle Movement

Changes in force on the condyle are shown in Figure 5.6. Condylar force was essentially biphasic, reaching a peak of 43N towards the end of condylar translation, and a slightly greater peak of 55N when the jaw reached its initial starting position, i.e., at the end of "bolus" compression. A third, transient peak of the same magnitude occurred when the "bolus" was struck.

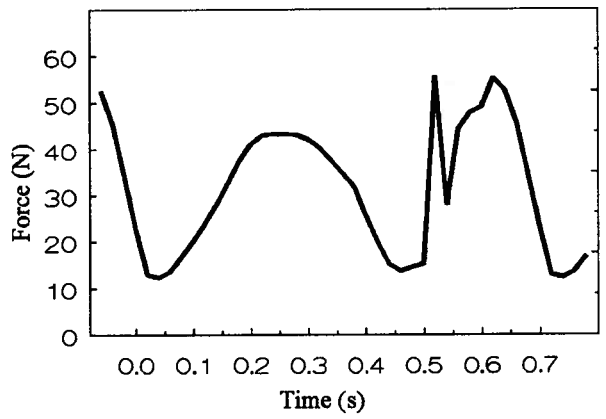

Figure 5.6
Reaction Force at Condyle

Recently, researchers have had success in gaining access to animal musculotendious units to measure their tensions, but these techniques are available in humans in isolated instances only. The simulation model, however, provides estimated values for the jaw muscle. In Figure 5.7, changes in muscle tension (continuous lines) have been superimposed on the muscles' corresponding activation curves (bar lines). In addition, displacement curves for the incisor point are provided for comparison. The elevator muscles all showed marked increases in tension during the opening phase, and again during the closing and compressive phase of the cycle. The amplitudes and timing of these changes were unique to each muscle. The closing/opening tension ratios were noticeably greater in the superficial masseter, medial pterygoid and anterior temporal muscles, but approached unity in the remainder, i.e., the opening and closing tensions were roughly equal. With the exception of the medial pterygoid, all the elevators showed least tension (approaching or reaching zero) before, and at the time the "bolus" was struck, i.e., just after jaw closing velocity reached its maximum. This effect was particularly evident in the three temporal muscles, especially in the anterior temporalis. Another decrease in tension was observed at the end of the "dwell" phase, just before jaw opening.

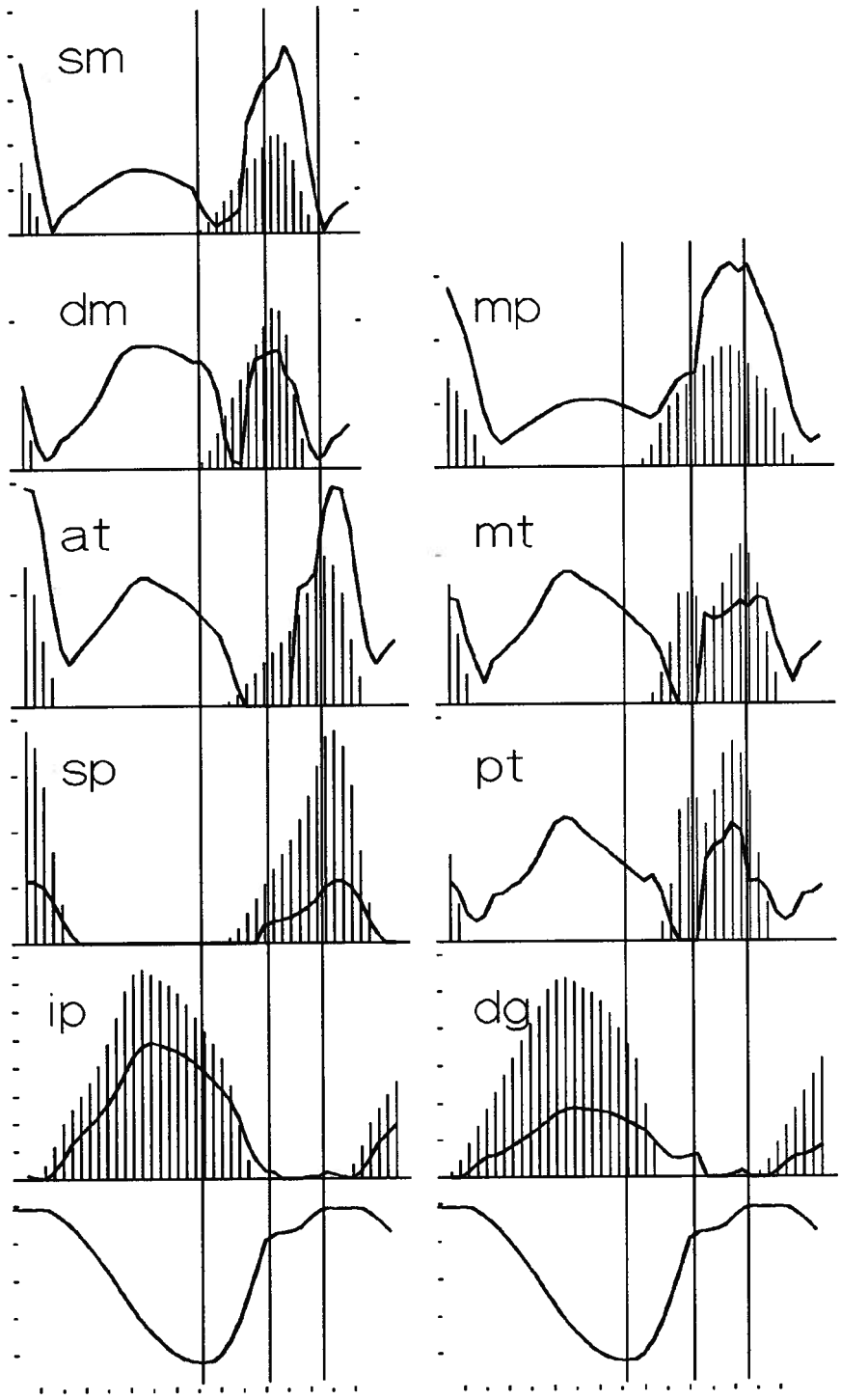

Figure 5.7
Activation Levels and Muscles' Tensions

5.3) Discussion

Despite simplification of the biomechanics, the model produced a very realistic chewing cycle when known patterns of muscle activation were chosen to drive the mandible through a simple resistance. Many features of the simulated cycle compared favorably with known values for human mastication.

In the model, the movement trajectory, jaw gape, and condylar motion were all determined by the timing and amount of activity in the inferior lateral pterygoid and digastric muscles, both working with the assistance of gravity against the combined passive tensions of the jaw-closing muscles. These passive tensions differed according to each muscle's cross-sectional size, location and length, although they shared common length-tension curves and visco-elastic characteristics. The pattern of activation in the inferior lateral pterygoid and digastric muscles during jaw opening creates unique movement trajectories at each closing muscle's insertion, and the various differences between the muscles' sizes and insertion sites then cause their lengths and shortening speeds to vary. The passive tensions are therefore specific and task-dependent.

The model's prediction of low to zero tensions in several closing muscles (particularly the temporalis group) at maximum closing velocity, just before the "bolus" is hit, may indicate a tendency rather than an actual event in the biological system. The behavior of the model is explained as follows. The induction of fast closing by the medial pterygoid (which did not show any decreased tension) created phase lag in muscles not yet activated to the same extent. Their insertions were rapidly displaced so as to slacken muscles which at best were marginally active. This effect was most obvious in the anterior temporalis, which was the most susceptible

due to its location and length. Although possible, the phenomenon of "slackness" is unlikely to occur in the biological jaw. Differences in muscle properties compared with the model, or low levels of activity in the muscles concerned (possibly reflex-driven), could maintain low muscle tensions irrespective of any actions of the jaw system. The maintenance of residual tensions would have considerable advantage, for example more efficient force coupling and faster response times when the bolus is struck. In the model, these zero tensions could be avoided by earlier activation of the muscles concerned, but the trajectory of the closing cycle changed when this was done.

The model shows how interrelated muscle activation patterns and musculoskeletal mechanics must be. Any patterned drive to the inferior lateral pterygoid and digastric muscles that is intended to produce a particular jaw movement has to do so under the influence of passive tensions in the closing muscles. This tensioning system responds differently according to the direction and the speed the jaw is driven. Thus, both rate and position-dependent factors of the mandible must be taken into account when the goal is to move the jaw into a particular position within a set time. Although at best an approximation, our model provides at least some idea of the way these tensions alter during function, and it invites speculation regarding the way the central nervous system learns to select the appropriate pattern of activation in advance.

The jaw-closing speed in our simulation was faster than that reported by Ahlgren [B1] for unrestrained human mastication (about 75 mm/sec for carrot-chewing) but is consistent with that reported previously for forced rapid chewing, which can be as high as 274 mm/sec [B6]. Jaw-closing speeds vary considerably according to food type. The kind of "chopping strokes" simulated in this study are

more like shorter duration chewing cycles than the wider, more ruminant strokes used in the mastication of hard foods [B12].

When the "bolus" was struck, 50-60ms passed before combined muscle tension reached a sufficient magnitude to overcome the resistance and move the jaw upwards. In practice, most foods do not present such an abrupt transition, and a softened "leading edge" to the onset of force would alter this movement-time relationship. The thickness and resistance profile of the bolus has a critical relationship to the timing of muscle activation and most important, the generation of muscle tension. The activation pattern has to be generated with an expectation of probable jaw velocity, muscle tension on impact, and tension required to compress a given bolus in a pre-selected direction of jaw movement. Failure to tune the model in this way produced an unwanted trajectory of jaw movement. Nevertheless, surprisingly small modifications to the average muscle pattern provided by Moller, (1966) were needed to "chew" the bolus in a typical manner.

When the initial starting position of the jaw was reached at the end of closing, the drive to most elevator muscles had begun to decrease, although their tensions remained. This was required to complete "bolus" penetration, and the tensions had to dissipate before the next opening stroke. Dissipation occurred when the jaw became stationary in a spontaneous "dwell" phase, which is also a characteristic of mammalian and human mastication [B1, B6, B9]. It is significant that force dissipation in this "dwell" phase was so balanced between the different muscles that the jaw remained stationary despite the location of its "condyle" on an inclined plane with no posterior limit. Mutual balance between elevator muscle groups obviates the need for a posterior limit. The condylar head can move and resist forces during opening, closing and bolus compression into an assumed "intercuspal" position without any passive

articular restraints, other than rear slope of the articular eminence itself. Here, the superior lateral pterygoid muscle had a critical role. Unless this muscle contracted synchronously with the closing muscles, the condyle continued to slide posteriorly and upwards beyond its starting position as closing muscle tensions decreased. The superior lateral pterygoid is generally considered to tension the articular disk, stabilizing it during bolus and tooth compression [B5, B10, B15]. Although many fibers of the superior lateral pterygoid attach to the condyle itself [B12] there are few opinions about the role of this attachment. The model suggests that the muscle provides an anterior tension vector to the condyle at a critical time in the late closing phase. Without this, a posterior limit to condyle movement e.g. by a ligamentous restraint would be essential. The alternative possibility would be for the inferior lateral pterygoid to contract biphasically (i.e., be active in both the opening and closing phases) as originally proposed by Moller, (1966), but this notion is not supported by the current literature.

A tedious aspect of working with models is developing muscle contraction strategies. However, the process is educational in that it provides one with insight into the problems facing a central nervous system, and a major advantage of the model is that the generation of muscle activation patterns is a contained operation. The availability of many descriptors such as changes in muscle tension and length, joint translation and rotation, movement velocities and articular forces, makes it readily possible to derive variables used as feedback by the central nervous system. It would be comparatively simple to use these data to simulate neural sensory information, and build this into a separate, linked model of the nervous control mechanisms responsible for jaw movement. Thus any future models of the nervous system, perhaps including aspects such as artificial intelligence, can readily be added to the system.

The next chapter will discuss an existing artificial neural network architecture, a method of improving it, and a feasible dynamic controller with neural network that could be used to control the jaw model in the future.

Chapter 6

Neural Network and Dynamic Control

Neural networks provide a unique computing architecture that can be used to address problems that are unmanageable with traditional methods. These new computing architectures, inspired by the structure of the brain, are radically different from the computers that are widely used today. Neural network architectures are motivated by models of our own brains and nerve cells. Although our current knowledge of the brain is limited, the basic anatomy of an individual nerve cell or neuron is known. A typical nerve cell in the human brain is shown in Figure 6.1.

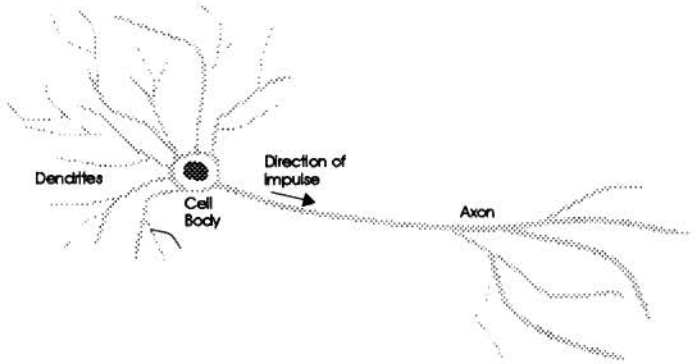

Figure 6.1
Typical Nerve Cell

The output area of the neuron is a long, branching fiber called the axon. An impulse can be triggered by the cell, and sent along the axon branches to the ends of the fibers. The input area of the nerve cell is a set of branching fibers called dendrites. The connecting point between an axon and a dendrite is the synapse. When a series of impulses is received at the dendritic areas of a neuron, the result is usually an increased probability that the target neuron will fire an impulse down its axon.

6.1) Artificial Neural Network

A great deal of biological detail is eliminated in the computing models. However, the artificial neural networks retain enough of the structure observed in the brain to provide insight into how biological neural processing may work. Figure 6.2 illustrates an example of a typical processing unit for an artificial neural network.

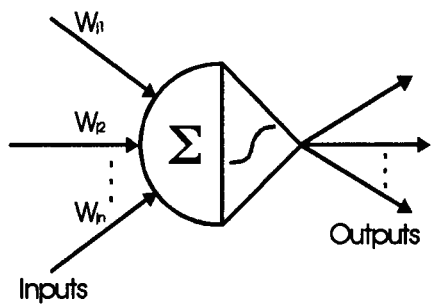

Figure 6.2
Artificial Neural Node

On the left are the multiple inputs which are connected to the processing unit; each arriving from another unit. Each interconnection has an associated connection strength. The processing unit summing up all the inputs and uses a nonlinear threshold function to compute its output. The calculated result is sent along the output connections to the target cells. The nonlinear threshold is usually implemented with the sigmoid function. The equation for the sigmoid function is:

$$f(x) = \frac{1}{1+e^{-x}}$$

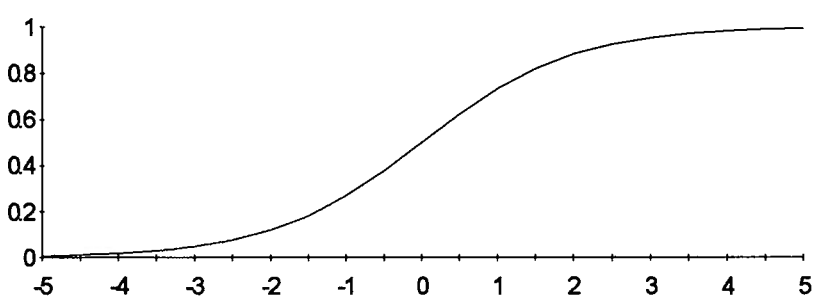

Figure 6.3
The Sigmoid Function

Figure 6.4 shows an example of neural network with two layers of processing units, a typical organization of the neural network known as feedforward network.

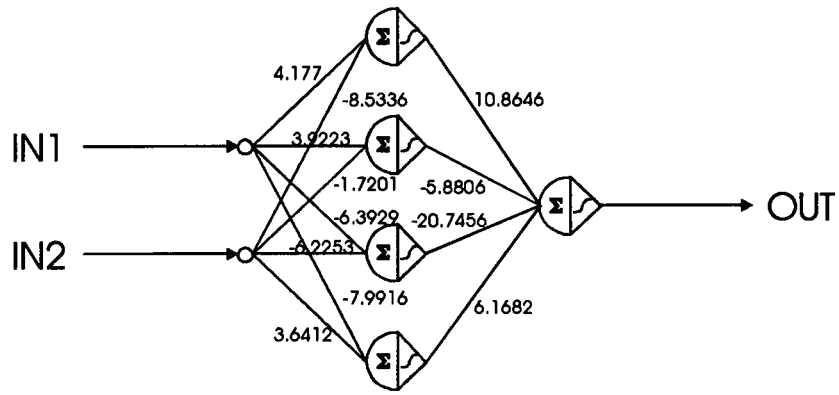

Figure 6.4
General Structure of An Artificial Neural Network

First is a layer of input units. The input patterns are represented as vectors to the network. The middle, "hidden," layer of this network consists of "feature detectors" -- units that respond to particular features that may appear in the input pattern. Sometimes there is more than one hidden layer. The activities of the last layer are read as the output of the network. In the example, there are two inputs, four hidden nodes and one output, however, configurations may be different for different applications.

Despite the complex form of the network, it can be easily implemented as follows:

$$\mathrm{OUT} = \mathrm{sigmoid}\left(\mathrm{sigmoid}\left(\begin{bmatrix}\mathrm{IN1}\\ \mathrm{IN2}\end{bmatrix}\begin{bmatrix}4.177 & -8.5336\\ 3.9223 & -1.7201\\ -6.3929 & -6.2253\\ -7.9916 & 3.6412\end{bmatrix}\right)\begin{bmatrix}10.8646 & -5.8806 & -20.7456 & 6.1682\end{bmatrix}\right)$$

Assume that IN1 and IN2 are {[0, 0], [0, 1], [1, 0], [1, 1]}, OUT will be {0.017719, 0.980323, 0.980321, 0.025326} accordingly, indicating that the neural network is performing the XOR function. The problems remaining are to find the right

connection weights for the desired function in the neural network. The next section will briefly discuss a training algorithm which is commonly used in artificial neural network.

6.2) Backpropagation Neural Network

Backpropagation neural networks are the most widely used of the neural network models and have been applied successfully in a broad range of areas. Backpropagation neural networks can handle any problem that requires pattern mapping. Given an input pattern, the network produces an associated output pattern. A backpropagation neural network is also one of the easiest networks to understand because its learning and update procedure is a relatively simple concept. If the network gives the wrong answer, then the errors back-propagate along the connections. The weights of the connections are corrected so that the error is lessened. Figure 6.5 illustrates these updating procedures.

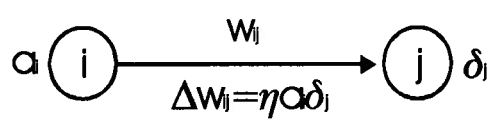

Figure 6.5
Updating Procedures in Artificial Neural Node

where

a_i = activation level
δ_j = error value
w_{ji} = connection weight
η = learning rate

In the context of such training, the feedforward network is often referred to a "backpropagation neural network."

Although biological systems have neurons that perform a type of summation of inputs, and have varying interconnection strengths, direct back-error propagation along the same nerve has not yet been identified (actually not possible) in biological systems. However, the uses of a trained net are completely forward in order, as in biological systems.

Backpropagation neural networks have a few disadvantages. First, the largest drawback with backpropagation appears to be its long training time. Second, because of the long training time, on-line retraining of the net is not easy. Finally, backpropagation is susceptible to training failures in which the network never converges to a point where it has learned the training set.

6.3) Method for Improving The Performance of Backpropagation Networks

Backpropagation networks are layered, and usually with each layer fully connected to the layers below and above. Backpropagation networks do not have to be fully interconnected, but, most applications that work have used fully interconnected layers. The more complex the training patterns, the bigger the net we have to use. This is true to a certain extent, but simulations show that there will be no improvement after the network reaches a certain size. Instead of holding everything in one fully interconnected network, the biological neural networks tend to store information in a more distributed manner. Therefore, I suggest a way of combining artificial neural networks. Figure 6.6 illustrates the structure.

Two neural networks in parallel make up this new configuration. One of the two networks is to train on the original training pattern, while the second network is to train on the error of the first network. The second network is named the Error

Adjusting Network (EAN) because its function is to minimize the error of the first network

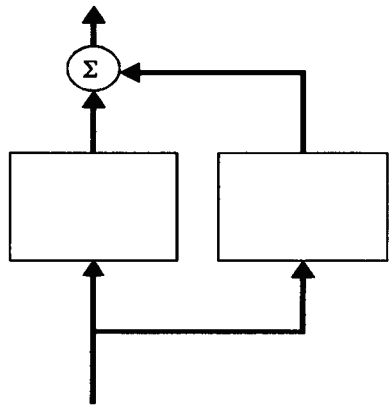

Figure 6.6
A Combined Artificial Neural Network

Neural networks tend to learn patterns that are more continuous in the initial training state that is fast, and slow down as only less continuous patterns are left behind. The less continuous patterns are responsible for the inconsistent performance of a neural network. EAN can deal with these situations.

The results of using the parallel structure are shorter training time and smaller network. The new structure is even able to handle more complex patterns that don't converge with a single network. The major drawback of a neural network is that it only gives approximate solutions, and is not good for precise controlling. The new configuration also shows a possible way to adjust the precision of a neural network to an acceptable level.

The uses of the above structures are illustrated with solving the inverse kinematics of a two-jointed crab arm. Figure 6.7 shows the arrangements of the crab arm. Although the structure of the crab arm here is not common in artificial robot arm, the purpose of the crab arm model is to illustrate the performance of the new configuration of the neural network.

The trajectory of the open end of the arm is specified with the angle θ and the distance d. By specifying θ and d, we want to know the values for θ_1 and θ_2 respectively. If the inverse kinematics of the crab arm is solved with conventional techniques, there is no reason to replace it with a neural network. Therefore, the training data for the neural network is generated with the direct kinematics that is straight forward as compared to inverse kinematics. As long as we have the input and output patterns, neural networks don't care how the patterns are generated. The training patterns should characterize the complete range of the input and output patterns, and a powerful feature of neural networks -- generalization -- will fill in appropriate values in the empty gaps that are not included in the training patterns.

The Crab Arm Example

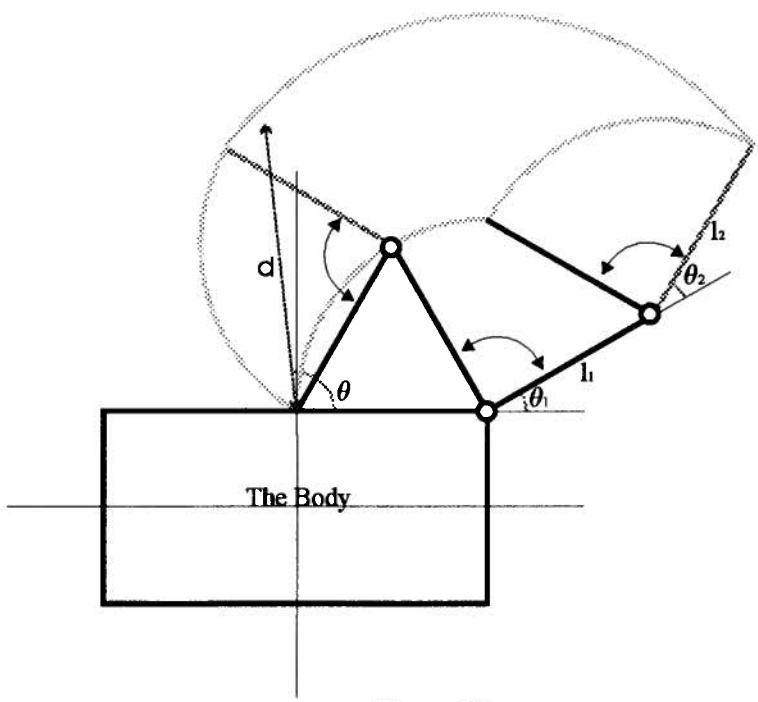

Figure 6.7
l_1 = length of the proximal arm
l_2 = length of the distal arm
$\theta_1, \theta_2 : 30^o \rightarrow 120^o$
(shaded lines indicate the possible area of movement as constraint by θ_1 and θ_2)

The training patterns are generated as follows:

```
for θ1 = 30° to 120°, step 5°
{
      for θ2 = 30° to 120°, step 5°
      {
            // direct kinematics equation of
            // the crab arm

            x = l1cos(θ1) + l2cos(θ1+θ2) + 2
            y = l1sin(θ2) + l2sin(θ1+θ2) + 1

            θ = atan(y/x)
            d = (x^2+y^2)^0.5
      }
}
```

Angles θ_1 and θ_2 are separated into two neural networks. Neural networks which are fully connected have the configurations of two inputs (θ and d), two hidden layers and each with eight nodes, and one output. The neural networks that are built from two networks in parallel have a different configurations. Each of them have two hidden layers with four nodes. Although there are the same number of nodes in both configurations, there are 88 connections in the first configuration as compare to 56 in the second.

In the new configuration, the first network is trained until the root mean square error reaches 0.08 and the second network will pick up the rest. With half the training time, two networks in parallel still work considerably better than the standard configuration. The root mean square errors of these two different networks are compared in Table 6.1.

Comparisons of The Root Mean Square Errors

	RMS in θ_1	RMS in θ_2
Standard Configuration	0.002928	0.001809
Two Networks in Parallel	0.000242	0.00074

Table 6.1

As indicated in Table 6.1, error in the new configuration could be up to ten times smaller than the standard configuration. The running time of the new configuration is also faster as there are 32 fewer connections as compared to the first. On-line retraining is also possible because we can always add a second network in parallel with the first. In addition, there is a better chance for the neural networks to converge because informations can be distributively stored in different networks. These are all significant improvements to the standard backpropagation networks.

6.4) Neural Network as A Digital Controller

There is much research in the area of neural networks for control. However, most of them depend on feedback controller, and the neural network learns from the controller and finally replaces it. Again, there is no reason to replace a conventional controller with a neural network controller. The best a neural network can do is to copy the function of a conventional controller and the performance will never be better but only worse than the original controller.

In feedback control, the parameter that is being controlled is continually measured (feedback), and compared to a reference (error calculation), and the action modified according to a control law to overcome the error. However, one has to think in terms of pattern mapping when working with neural network. The objective is to

map the current state and the desired "next" state to the according control signal. In other words, we want to determine the correct control signal that will transform the system from the current state to a desired new state. Figure 6.8 illustrates this relationship.

State Mapping

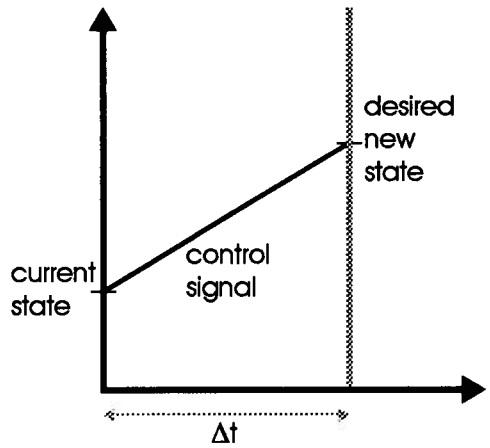

Figure 6.8
The objective is to determine
the correct control signal to transform the system
from the current state to the desired new state.

The input and output connections of the neural network that performs this function is simple. Figure 6.9 shows this configuration.

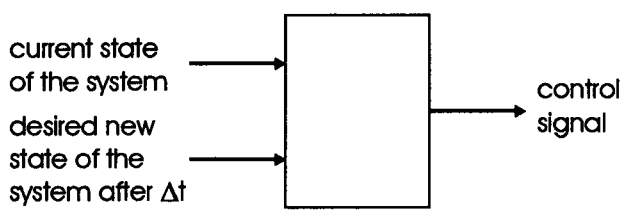

Figure 6.9
Neural Network That Performs State Mapping

The training set is prepared with forward dynamics. With different possible states of the system, different possible control signals are fed in to the system and the outcome is a set of possible "next" state of the system. The neural network is trained

with the current state and the "next" state as inputs, and the according control signal as output as indicated in Figure 6.9.

Continuous controlling can be performed by dividing the time axis into different time slices, and each time slice can be considered as separate pattern to the neural network. The idea is illustrated in Figure 6.10.

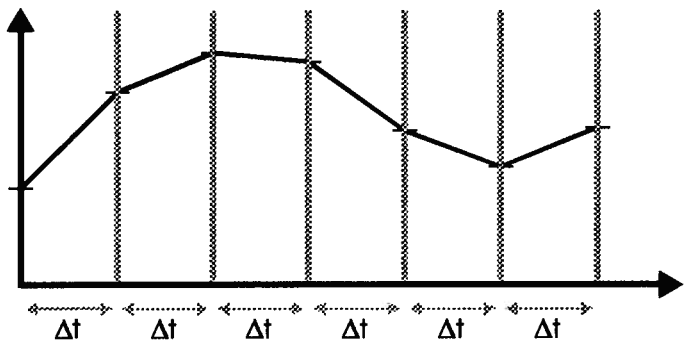

Figure 6.10
State Mapping in Continuous Time Slices

As the neural network controller always tries to give a control signal that matches the desired path in different time slices, the error is not cumulative.

The above idea is illustrated with the isometric contraction of a single muscle. Figure 6.11 shows the complete arrangement of this example.

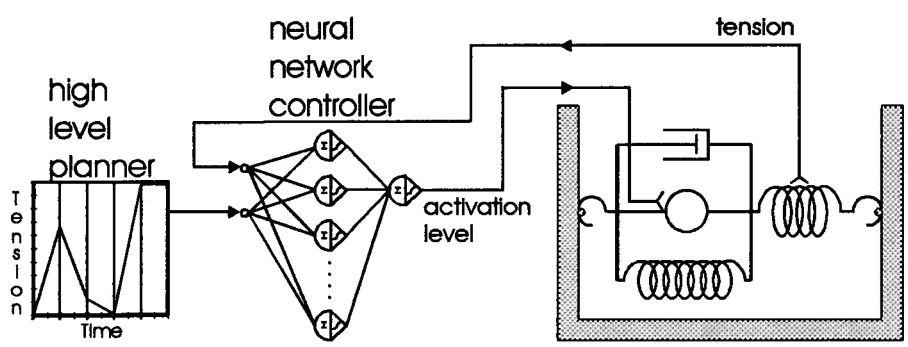

Figure 6.11
An Example of The Neural Network Controller

In the above example, the only variable that defines the state of a muscle is the tension and the possible control signal is the magnitude of the neural drive. With the current tension known, we want to find out the magnitude of the activation level to generate a desired tension after Δt.

Recently, there are studies that the neural network controller learns from its environment. These methods are based on a trial-and-error scheme and are passive and slow. Instead of the passive trial-and-error scheme, the data is generated once and for all. A complete set of training patterns can be generated by the cartesian product of S and C in which S is made up of elements within the possible initial states of the system and C is the set of possible control signals that can be fed into the system. Figure 6.12 illustrates the cartesian product of S and C in the previous example.

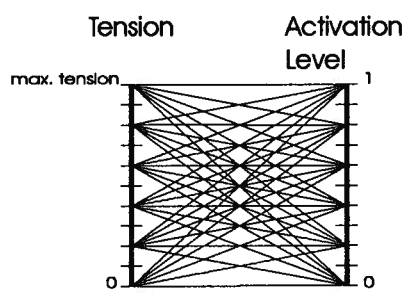

Figure 6.12
The Cartesian Product of S and C

Please note that the above figure shows the cartesian product of S and C instead of a time slice, and the neural network is trained with patterns that are generated as follows:

```
for current_tension = 0 to max_tension, step max_tension/5
{
      for activation_level = 0 to 1, step 1/5
      {
            new_tension = tension_after_Δt(current_tension,
                                           activation_level);
      }
}
```

Change in tension as according with time is planned (Figure 6.11), and Figure 6.13 illustrates the simulation result.

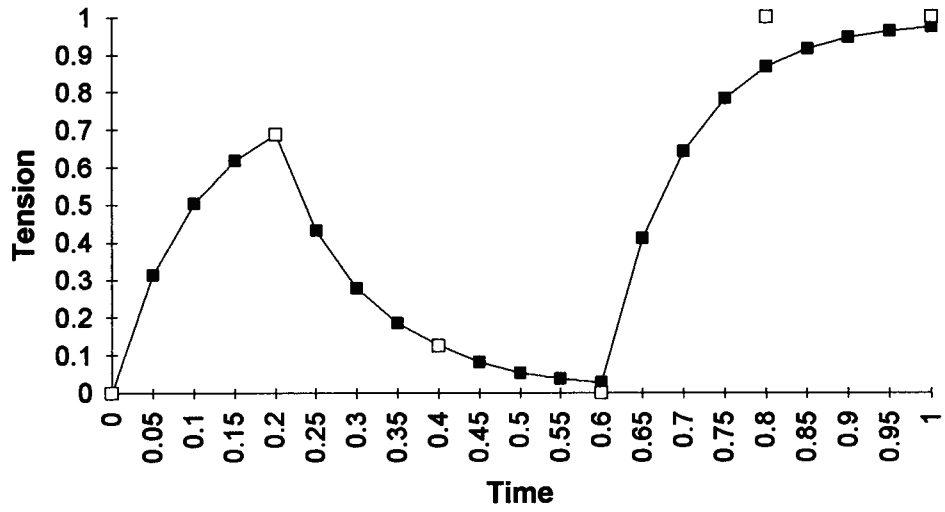

Figure 6.13
Simulation Results of The Neural Network Controller

The white dots in Figure 6.13 indicate the desired tensions at the specified time, and the curve indicates the tension developed within the muscle with the control of neural network. The outcome at 0.08s does not match the requirement as it is not possible to develop maximum tension of the muscle within this short period of time (0.02s), however, the neural network controller will always try to provide the best match to the requirement. As indicated in Figure 6.13, the simulation results are quite good with this simple test.

6.5) Conclusions Regarding Neural Networks

This chapter has demonstrated how to perform inverse kinematics and dynamic control with neural networks in a direct manner. Neural networks that handle inverse kinematics and dynamic control are not designed to replace the conventional algorithms. Indeed, no matter how accurate a neural network is, it only gives an approximate solution to the problem. Therefore, an error will always be presented. On

the other hand, conventional techniques are more accurate and predictable. A neural network should be used when the solutions are impractical to solve with the conventional algorithms, and when approximate solutions are acceptable.

In our example, there is only one variable that defines the state of the muscle. However, no matter how many variables that define the state of a system, it makes no difference to the way in which the neural network performs. One of the strongest points of a neural network is that it can deal with multi-dimensional inputs as well as two or three-dimensional inputs.

In addition, EAN is believed to improve the performance of neural networks to a more acceptable level. Although the demonstrations here only show the combination of two neural networks, there is no reason to limit the number of neural networks to two. Instead, different neural networks should be thought of as the building blocks of a more complex system.

Chapter 7

Conclusions

The simulation results from the computer model in Chapter 5 are realistic in the following way. The model produces trajectories of opening and closing movements in the incisor region which resemble those reported in the literature for human mastication, both in time and space. This kind of dynamic simulation, can help explain the interplay between active and passive muscle tensions during jaw motion, the physical consequences of muscle coactivation, and loads upon the mandibular condylar during translational and rotational motion. The basic design can be modified to explore the question of articular stability, the action of specific muscle groups on articular function, and wider associations between patterns of muscle contraction and craniofacial shape. Long-term goals might include the simulation of an active neural control system, mandibular mechanics during whiplash injury, and prosthetic designs for joint replacement.

Although studies in the project are confined to the human jaw, the completed muscle model is flexible enough to model other mammalian musculoskeletal systems as well. The jaw model that utilized the muscle model provided valuable information which was not obtainable in any other way, and it could do the same for any other musculoskeletal system developed with it.

Studies of the biological system seemed to indicate a way of increasing the performance of an artificial neural network. Neural networks may provide a feasible structure for automatic control, and could be incorporated into future musculoskeletal

system model of the kind developed in this study. The examples in Chapter 6 are simple, but the structures of the neural networks suggested are sufficient for general use.

7.1) Limitations of the Current Jaw Model

There are several limitations in the present jaw model. They include simplifications of the form and properties of the jaw muscles, the adoption of an artificial food bolus, reduction of the condylar guidance to an unlimited and frictionless sliding surface, assumptions regarding the center of gravity and moment of interia, and the limitation of jaw movement within a two dimensional space.

Human jaw-closing muscles are all multipennate (containing radiating patterns of individual muscle fibers), and they are notable for their relatively wide areas of attachment. It is also possible that all the closing muscles are capable of at least some degree of regional activation, depending upon the task being performed [B3, B11, B21, B22]. Given their complexity, various of attachment sites, and the possibly local, graded patterns of intramuscular activation, it is presently difficult to predict the true nature of active and passive length-tension curves for individual human jaw muscles. The relatively simple assumptions for muscle-tendon actuator behavior in this study were based on data available for whole skeletal muscles generally, and did not take specific pennation patterns into account. Although the muscle model may not be the ideal, approximating to the actual system, it provides reasonable estimate of the general qualitative changes which occur in the biological system. However, the fact that the jaw model with realistic patterns of muscle activation produced jaw movement patterns which closely resembled those in the literature suggests that our assumptions regarding muscle properties were quite reasonable first approximations.

A similar argument can be proposed for regarding the mass, center of gravity and moment of interia of the human lower jaw. The estimation of jaw weight is likely to have been low, since it was based on a dry specimen, and the measurement of the mandible's true center of gravity and moment of interia is not simple. However, both vary in life, and it would be simple to modify these constants when better data become available. Again, the behavior of the model system indicated that the values used were not unreasonable. Even in the absence of other variables, such as the passive visco-elastic properties of other soft tissues in the region and the weight of the tongue, the model was still able to behave well. With the great strength of the jaw muscles, these factors seem to have only minor effects on the jaw.

7.2) Future Directions of The Jaw Model

The model was designed to permit continuous modification and improvement. The muscle attachments which were modeled three-dimensionally can be altered to produce different musculoskeletal configurations. Similarly, muscle constants can be altered as data change. The specification of "bolus" properties can be changed to include different thicknesses, different bite point locations, and different compressive properties. Elements of friction can be introduced at the dental occlusal level, and within the temporomandibular articulation. The shape of the articular eminence can be made curvilinear if desired, and various forms of posterior buttressing and elasticity can be added. Conversion of the model to accommodate three-dimensional jaw motion is not as simple a proposition as the above changes even though the construction of a twin-joint system is a very desirable goal.

It would be a comparatively simple matter to use many of the behavioral descriptors such as changes in muscle tension and length, joint translation and

rotation, movement velocities and articular forces as variables describing feedback by the central nervous system, and to build these into a separate, linked model of the nervous control mechanisms responsible for jaw movement. Future models of the nervous system, including control of muscle drive by artificial intelligence, can readily be added to the system which invites further development.

References

The references are classified into five sections, labeled from A to E.

A) Muscle Mechanics

1. Alexander R. M. (1988). Elastic Materials. In: Elastic Mechanisms in Animal Movement. Cambridge University Press, pp 1-21.

2. Alter M. J. (1988). The Neurophysiology of Flexibility: Neural Anatomy and Neural Transmission. In: Science of Stretching. Human Kinetics Books, pp 43-50.

3. Jongen H. A. H., Denier van der Gon J. J. and Gielen C. C. A. M. (1989). Activation of Human Arm Muscles During Flexion/Extension and Supination/Pronation Tasks: A Theory on Muscle Coordination. Biological Cybernetics 61: 1-9.

4. McMahon T. A. (1984). Fundamental Muscle Mechanics. In: Muscles, Reflexes, and Locomotion. Princeton University Press, pp 1-26.

5. Yamada H. (1970). Locomotor System. In: Strength of Biological Materials. pp 82-105.

6. Zajac F. E. (1989). Muscle and Tendon: Properties, Models, Scaling, and Applications to Biomechanics and Motor Control. Crit Rev Biomed Eng 17: 359-404

B) Human Jaw System

1. Ahlgren J (1976). Masticatory Movements in Man. In: Mastication (eds. Anderson DJ and Matthews B). John Wright and Sons Ltd., Bristol, Great Britain, pp 119-130.

2. Belser UC and Hannam AG (1986). The Contribution of The Deep Fibers of The Masseter Muscle to Selected Tooth-clenching and Chewing Tasks. Journal of Prosthetic Dentistry 56: 629-636.

3. Blanksma NG and van Eijden TMGJ (1990). Electromyographic Heterogeneity in The Human Temporalis Muscle. Journal of Dental Research 69: 1686-1690.

4. Blanksma NG, van Eijden TMGJ and Weijs WA (1992). Electromyographic Heterogeneity in The Human Masseter Muscle. Journal of Dental Research 71: 47-52.

5. Gibbs CH, Mahan PE, Wilkinson TM and Mauderli A (1984). EMG Activity of The Superior Belly of The Lateral Pterygoid Muscle in Relation to Other Jaw Muscles. Journal of Prosthetic Dentistry 51: 691-702.

6. Hannam AG, De Cou RE, Scott JD and Wood WW (1977). The Relationship Between Dental Occlusion, Muscle Activity and Associated Jaw Movement in Man. Archives of Oral Biology 22: 25-32.

7. Korioth TWP, Romilly DP and Hannam AG (1992). Three-dimensional Finite Element Stress Analysis of The Dentate Human Mandible. American Journal of Physical Anthropology 88: 69-96.

8. Korioth TWP and Hannam AG (1990). Effect of Bilateral Asymmetric Tooth Clenching on Load Distribution at The Mandibular Condyles. Journal of Prosthetic Dentistry 64: 62-73.

9. Lund JP (1991). Mastication and Its Control by The Brain Stem. Critical Reviews in Oral Biology and Medicine 2: 33-64.

10. Mahan PE, Wilkinson TM, Gibbs CH, Mauderli E and Brannon LS (1983). Superior and Inferior Bellies of The Lateral Pterygoid Muscle EMG Activity at Basic Jaw Positions. Journal of Prosthetic Dentistry 50: 710-718.

11. McMillan AS and Hannam AG (1992). Task-related Behavior of Motor Units in Different Regions of The Human Masseter Muscle. Archives of Oral Biology 37: 849-857.

12. Miller AJ (1992). Craniomandibular Muscles: Their Role in Function and Form. CRC Press Inc., Boca Raton, Florida, USA.

13. Moller E (1966). The Chewing Apparatus: An Electromyographic Study of The Action of The Muscles of Mastication and Its Correlation to Facial Morphology. Acta Physiologica Scandinavica 69, suppl. 280: 1-229.

14. Nelson GJ (1986). Three Dimensional Computer Modelling of Human Mandibular Biomechanics. MSc. Thesis, University of British Columbia.

15. Okeson JP (1993). Functional Anatomy and Biomechanics of The Masticatory System. In: Management of Temporomandibular Disorders and Occlusion, Mosby-Year Book Inc., St. Louis MO, pp. 21-27.

16. Osborn JW and Baragar FA (1985). Predicted Pattern of Human Muscle Activity During Clenching Derived From A Computer Associated Model: Symmetric Vertical Bite Forces. Journal of Biomechanics 18: 599-612.

17. Otten E (1987). A Myocybernetic Model of The Jaw System of The Rat. Journal of Neuroscience Methods 21: 287-302.

18. Smith DM, McLachlan KR and McCall WD (1986). A Numerical Model of Temporomandibular Joint Loading. Journal Dental Research 65: 1046-1052.

19. Throckmorton GS (1985). Quantitative Calculations of Temporomandibular Joint Reaction Forces. II. The Importance of The Direction of The Jaw Muscle Forces. Journal of Biomechanics 18: 453-461.

20. Throckmorton GS and Throckmorton LS (1985). Quantitative Calculations of Temporomandibular Joint Reaction Forces. I. The Importance of The Magnitude of The Jaw Muscle Forces. Journal of Biomechanics 18: 445-452.

21. van Eijden TMGJ, NG Blanksma and P Brugman (1992). Amplitude and Timing of EMG Activity in The Human Masseter Muscle During Selected Motor Tasks. Journal of Dental Research 72: 599-606.

22. Wood WW, Takada K and Hannam AG (1985). The Electromyographic Activity of The Inferior Head of The Human Lateral Pterygoid Muscle During Clenching and Chewing. Archives of Oral Biology 31: 245-253.

C) Continuous System Simulation

1. Francois E. Cellier (1991). Principles of Planar Mechanical System Modeling. In: Continuous System Modeling. Springer - Veriag, pp 79-109.

2. Matko D., Karba R and Zupancic B (1992). Simulation and Modeling of Continuous Systems. Prentice Hall.

D) Object-Oriented Analysis and Programming

1. Booch G. (1991). Object Oriented Design with Applications. The Benjamin / Cummings Publishing Company, Inc..

2. Davis S. R. (1992). C++ Programmer's Companion. Addison Wesley.

3. Ladd S. R. (1990). C++ Techniques & Applications. M & T Books.

E) Artificial Neural Network

1. Levine D. S. (1991). Introduction to Neural and Cognitive Modeling. Lawrence Erlbaum Associates, Inc..