An investigation into motor pools and their applicability to a biologically inspired model of ballistic voluntary motor action

by

Mark Paul Norman

A thesis submitted to the University of Plymouth
in partial fulfilment for the degree of

DOCTOR OF PHILOSOPHY

Neurodynamics Research Group
School of Computing
Faculty of Technology

April 1996

An investigation into motor pools and their applicability to a biologically inspired model of ballistic voluntary motor action

Mark Paul Norman

Abstract

This study investigates the properties of motor pools in the human motor control system. The simulations carried out as part of this study used two biologically inspired neuronal models to simulate networks with properties similar to those observed in the human motor system (Burke, 1991). The Synchronous neuronal model developed as part of this study explicitly models the input/output spike train and frequency relationship of each neuron. The motor pool simulations were carried out using the INSIGHT TOO simulation software developed as part of this study. INSIGHT TOO is a flexible neural design tool that allows the visual interactive design of network connectivity and has the power of a node specification language similar to that of BASIC that allows multi-layer, multi-model networks to be simulated. The simulations have shown that the motor pools are capable of reproducing commonly observed physiological properties during normal voluntary reaching movements. As a result of these findings a theoretical model of ballistic voluntary motor action was proposed called the Recruitment Model.

The Recruitment model utilises the "recruitment" principle known to exist in motor pools and applies this distributed processing methodology to the higher levels of motor action to explain how complex structures similar to the human skeletal system might be controlled. A simple version of the Recruitment Model is simulated showing an animation of a running "stick man". This simulation demonstrates some of the principles necessary to solve problems relating to synergy formation.

Index

2.0 Overview of the control of movements

3.0 Neurophysiological and biological review

4.0 Neuronal models

5.0 Motor pool simulations

6.0 The Recruitment model of ballistic voluntary motor action

6.4 Higher motor centres (HMC)

I. List Of Figures

II. List of tables

III. Acknowledgements

I would like to thank the Neurodynamics Research Group for their input and comments during the course of this research and in particular the support and assistance of my supervisor Prof. M. Denham. I would also like to thank my wife, family and friends for the support and understanding they afforded me during the production of this thesis.

IV. Authors Declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award.

This study was entirely financed by the University of Plymouth.

Relevant scientific seminars and conferences were regularly attended at which work was presented. A number of papers were prepared for publication.

Publications:

Denham M.J., Patel S., Troup L.J, Norman M.P. (1991) Oscillatory neural networks and their application to sensory motor control in adaptive robots. Proc IEE Conf on Artificial Neural Networks, Bournemouth, November 1991. IEE Press 1991, pp328-332.

Denham M.J., Patel S., Troup L.J, Norman M.P. (1991) Sensory-motor control of intelligent robots using oscillatory neural networks. Proc 30th IEEE Conf on Decision and Control, Brighton, December 1991. IEEE Press 1991, pp168-169.

Norman M.P. Denham M.J. Voluntary motor control using biologically plausible neural networks with application to advanced biologically inspired robots. Proc. Int. Conf on Neural Networks ICANN'92, Brighton, September 1992. Artificial Neural Networks, 2, I Alexsander & J Taylor (eds), Elsevier, 1992, 491-494.

Presentations and Conferences Attended:

International conference on Neural Networks, Brighton, 1992

Signed ..M. P. Norman..........

Date ..20/11/96....................

1.0 Introduction

The standard 3-joint robotic arm currently seen in many factories throughout the world has brought significant improvements in the efficiency and quality of manufacture for a vast range of products. At present the usefulness of these robotic machines is limited by the control strategies that govern them and therefore the tasks to which they can be applied. Existing control strategies limit the application of robotics to only relatively simple production processes. A control strategy that could match that of the human motor control system would find a multitude of new applications.

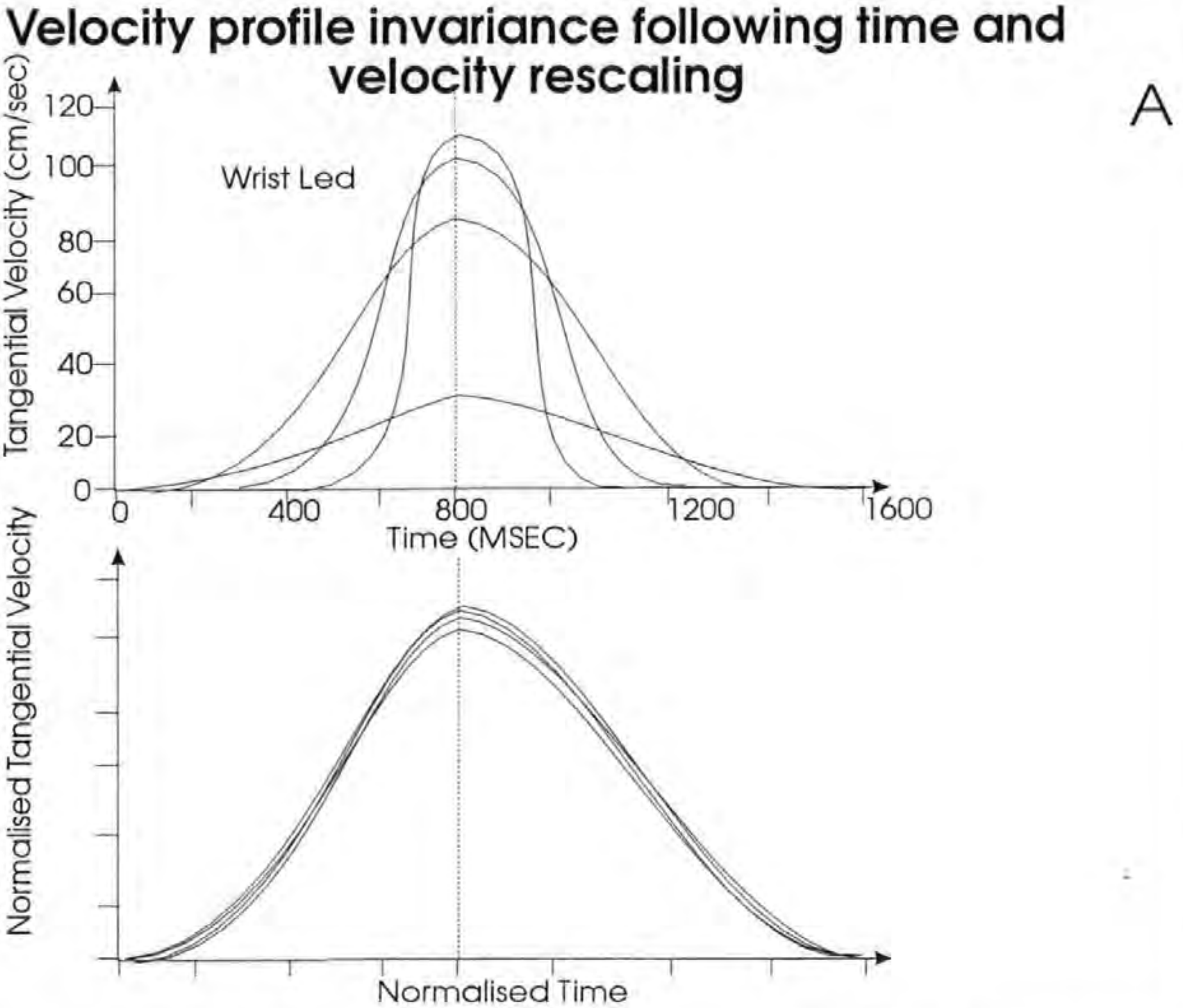
Control techniques have often revolved around mathematical analysis of a problem space and an analytical solution to the problem. Although the implementation of these models has changed, in general the philosophies and approaches have changed by only small amounts.

Classical robotic manipulators have typically been controlled using standard control theory using mathematical analysis of the problem space to send rigid control signals to rigid robotic manipulators. These control models are based around inverse dynamic and inverse kinematic analysis (see for example Wada and Kawato 1993). For simple problems this approach has proved to be acceptable. For many tasks that humans take for granted this approach stalls almost immediately. As the problem space complexity increases the analysis complexity explodes combinatorially. The reason for this failure is that the complex interacting three dimensional geometries require more time to compute than is available in a real-time time environment with the present common computing facilities. The next stage adopted by many researchers was to build models implementing classical control theory using neural networks (see chapter 2). This approach led to control strategies that can adapt to small perturbations in simple problem spaces but still could not solve more complex problems.

There is a control system that is already capable of carrying out tasks of significantly greater complexity than those currently available for artificial systems. This control system is implemented in the human brain and is capable of highly complex control behaviour. Although we cannot build the control system at present, it is possible to learn as much as we can about the methods employed in it and to use this information to improve our control models.

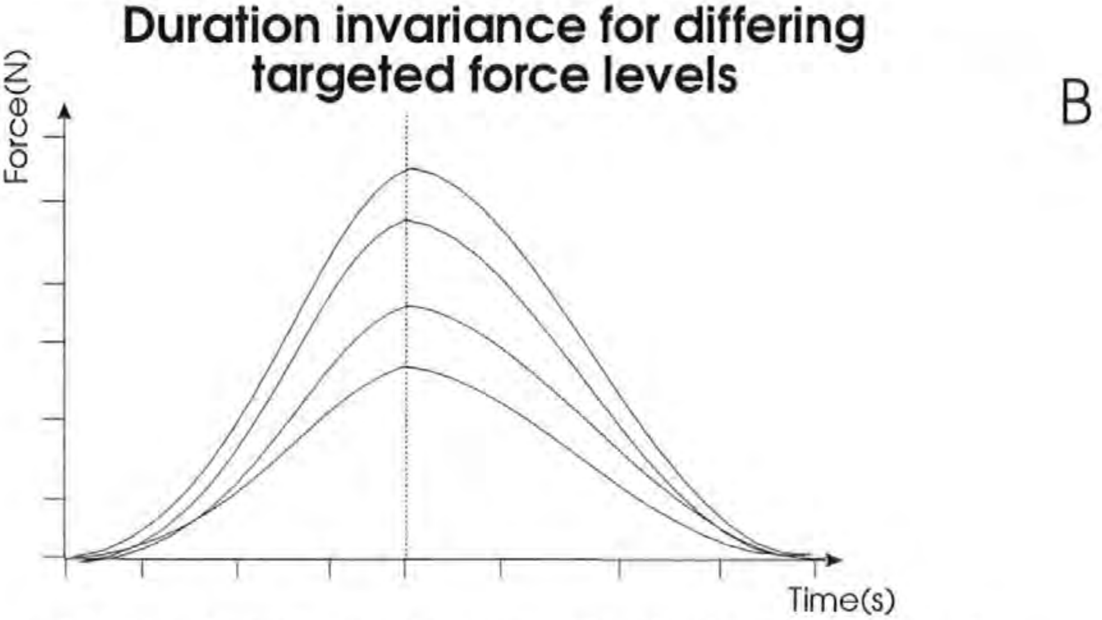
The design of the human brain is less rigid in philosophy than the techniques we can apply with mathematics and computation. In a computer controlled process, if an event occurs that was not originally planned for by the systems designers then often the system will produce an unpredictable result, fail ungracefully or just fail completely. In many control processes it is not possible or desirable to allow for every possible contingency. The techniques employed in the human motor control system enables it to adapt to new situations in a way that does not cause the system to fail. The chosen response is often predictable as the best choice given the information available.

The rigidity in the design of standard control systems is also reflected in the design of the physical device being controlled (and vice-versa). An example of this is a standard robotic arm; the motors and joints that move the limbs have precise characteristics that require precise control parameters and as such every parameter must be explicitly controlled.



Velocity profile invariance following time and velocity rescaling

A. Velocity profiles from movements of a similar duration are approximately superimposable following velocity and time axis rescaling. (Atkeson and Hollerbach, 1985).



Duration invariance for differing targeted force levels

B. Curves for subjects' approach to various targeted force levels are reached at nearly The same time, indicating duration invariance across different force "distances". Freund and Budingen (1978).

Figure 1.1

The human motor system is entirely different. Human muscles are "sloppy" in comparison and the feedback from them is slow and imprecise (Rack, 1981). The control signals that govern them are also imprecise. The net result of the human design is obviously better. A hypothetical account of this is that some of the complex parameters in the control sequence are designed-out, that is they are accounted for at a local level and do not need to be directly controlled. From this hypothesis we can infer that the human motor control system has learnt some useful techniques that could be used to solve real world control problems.

When a reaching movement is required (a voluntary movement), the human brain activates the appropriate units to make a movement in that general direction and at that distance. The initial stimulus that is sent to the muscles is usually larger than that required. Subsequent feedback of the position of the arm as the movement progresses is used to modify this signal. The movement of the arm must be smooth, it cannot be jerked into action as this might cause the muscle to tear or become strained. Equally it should not be brought to an abrupt halt either. The ideal overall velocity profile of each muscle for a given movement would be approximately "bell-shaped" when re-scaled for velocity/duration (see figure 1.1(a)) and for a wide range of movements in the human motor system this is indeed observed (Abend, Bizzi & Morasso, 1982; Atkeson & Hollerbach, 1985). Many control models omit to produce bell-shaped velocity profiles and other physiological observations made in human arm movements, as they are invariably implemented and designed on rigid robotic manipulators. In the human motor control system the motor pools are located in the spinal cord and brain stem, and are known to play a major role in transforming descending motor commands into physical tensions in the target muscles. The motor pools are therefore also referred to as the output motor drive and are in an excellent position to produce physiologically observed data.

1.1 Objectives of research

The objective of this research was to:- (i) understand the neurophysiological and biological processes associated with the output drive of voluntary motor control during the initial ballistic phase of movement (before feedback is available), and to understand the position of the output motor drive in the context of the overall ballistic voluntary motor control system; (ii) to assess the applicability of the major motor control systems in the human brain with regard to building an artificial biologically inspired control system; to achieve this a review of the neurophysiological motor areas is undertaken in chapter 3; (iii) to test this understanding by building biologically inspired models which will be designed, simulated and assessed in this context.

The ballistic phase of voluntary motor action occurs before feedback is available to alter the trajectory (Rack, 1981); the control system is therefore open loop control (see chapter 2.1). By distinguishing between the initial ballistic phase and subsequent corrected phases of movements, important simplifications can be made to help focus on the motor production problem, i.e. the generation and form of control signals. A number of complex brain structures such as the cerebellum, basal ganglia (which consists of many sub-areas) and the thalamic nuclei can all be disregarded as their input is known not to be available until much later corrected phases of movement occur (Rack, 1981).

Voluntary motor control is distinguished from other types of motor control since it is a conscious action. Other types of motor action, such as reflexive motor actions occur without cognitive effort and are controlled by localised control mechanisms and are essentially low-level processes. There are however, many different types of voluntary motor action, some involve planning from current information, some from recalled information. Other types rely on complex recall of motor sequences, requiring the interaction of many sensory modalities.

In chapter 6 where the Recruitment model of motor control is described in detail an important distinction is made between higher motor levels (or motor planning levels) and motor production levels. The motor planning areas issue sequences of synchronised "simple instructions" onto a common area known as the motor production level, which integrates and processes these inputs. The definition of "simple instructions" in this case does not refer to a single muscle or joint movement, but instead refers to a movement that can be realised in a single action. For example, an arm reaching movement can require the synchronisation and movement of several joints and many muscles in a complex way. For the majority of arm movements in free-space this would be defined as a "simple movement". A movement requiring that an obstacle be avoided en-route would be defined as a "complex movement", and would be completed by means of several "simple movements" released from higher motor areas or motor planning levels. For this reason it is important that transitions between successive simple movements occur smoothly.

Figure 1.1(b) shows another important physiological property where a movement of differing fixed amplitudes is made in similar times, demonstrating duration invariance (Freund and Budingen, 1978). The second half or downside of the profile shows natural decay back to a resting position. The simulations in chapter 5 and chapter 6 focus on this property as being significant to building better motor control models. The Recruitment Model suggests a key role for the motor pools in the production of this property.

Chapter 2 gives an overview of contrasting approaches to motor control problems and a review of the major issues relating to the control of movements. A mix of biologically plausible and artificial models are described. In chapter 3 a review of the motor pools in the human brain is presented. These reviews form the basis for a number of experimental simulations, which are presented in chapter 5, the results of which are contrasted with the control review presented in chapter 2.

The structure and operation of the motor pool simulations are then considered in the context of an overall motor control model. Chapter 6 therefore discusses how the information presented thus far could be combined into a uniform Control Model (the Recruitment model).

The Recruitment Model of ballistic voluntary motor action breaks down into two main areas:- those concerned with motor planning and those concerned with motor production. The planning modules are concerned with issues such as high level trajectory planning, object avoidance and recall from memory of complex motor sequences. The motor production modules are concerned with carrying out these sequences of positions correctly. The higher motor planning areas are numerous, complex, distributed and diverse in design and in the human motor system are located both cortically and sub-cortically (see chapter 3). The motor production area will therefore take many input signals, integrate them and then ensure correct execution. This involves monitoring the progress of any single movement and providing corrections where necessary, as well as adapting to any physical changes in the device being controlled (for example allowing for a strained muscle). The motor production areas must provide smooth control of a multitude of muscles and muscle fibers, each of which needs a unique control signal which must be synchronised with the operation of all the other muscles and muscle fibers. The motor pools (a constituent part of the motor production areas) are responsible for the final smooth motor output. For each muscle there is a motor pool, and each motor pool is made up of many other independent elements.

Each motor pool is controlled by higher motor production levels. These include the Joint Recruitment Level (JRL) and Muscle Recruitment Level (MRL). The arrangement of these levels can be seen in figure 1.2.

The JRL is responsible for the control of joints, which in turn operate on areas of the MRL. The MRL which has control of each individual muscle in turn instructs the operation of each motor pool.

A simplified form of the Recruitment control model is then demonstrated in the form of computer simulation (Chapter 6). A description of the neuronal model and simulation parameters used during these experiments is given in Chapter 4.

The Recruitment Model Of Ballistic Voluntary Motor Action.

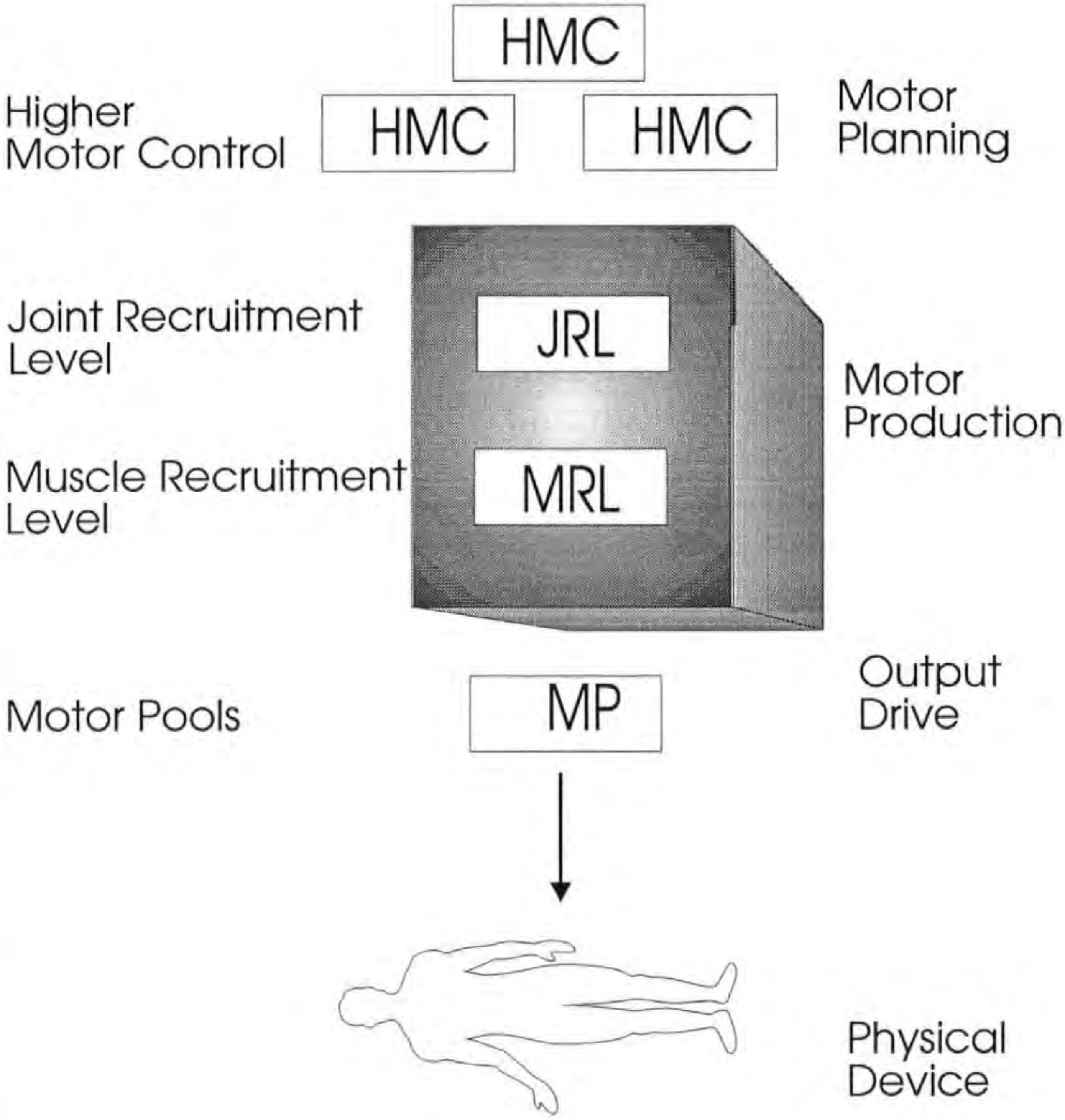


Figure 1.2

1.2 Summary

In chapter 5 simulations designed to examine motor pool behaviour are studied. To provide a background to these simulations, chapter 2 reviews the primary motor control issues and concepts and provides a contextual background for the detailed neurophysiological review undertaken in chapter 3. These chapters in combination provide the theoretical basis for the Recruitment Model, which is outlined in detail in chapter 6.

The simulations in chapter 5 and chapter 6 are based on two neuronal models, the first being the Leaky Integrator (Bressloff and Taylor, 1988), and the second is the Synchronous neuronal model, which was developed as part of this study. The details of these models are presented in chapter 4. The operation of the INSIGHT TOO simulation software developed as part of this study, is outlined in chapter 5, where it is applied to both Leaky Integrator and Synchronous network simulations.

In chapter 7 the research aims and objectives are measured against the output of this research, and the results are put into a practical context. The concluding chapter also discusses how this research might be progressed in the future.

2. Overview of the control of reaching movements.

2.1 Introduction

In any control problem there are a number of important issues that need to be resolved, these include choosing a control strategy, a co-ordinate system and how the various control tasks are to be segmented. In section 2.1 a broad description of the available control strategies is given and the human motor control system is described in this context. This is followed by a discussion on co-ordinate systems biased towards research results reported about human neurophysiology.

Section 2.3 describes an arbitrary breakdown of tasks that are commonly seen in control models implemented to solve voluntary reaching problems. The processing necessary at each step is then described in more detail, and in the context of the human motor control system.

The final sections focus down on models and issues relating solely to the human motor system; specifically on reported physiological observations and how this has affected the design of existing motor control strategies.

The aim of this chapter is to give a background of control issues and solutions, which can be used to assess the more detailed neurophysiological results reported in chapter 3 and the experimental data reported in following chapters.

There are a number of differing approaches to any control problem; the approach chosen will depend on the task, and is affected by factors such as the required response speed, control complexity and available feedback information. Closed loop and open loop are the two main control strategies. These strategies can be combined in many ways to form hybrid strategies. The variations on these basic forms are numerous, a brief overview is given below.

2.1.1 Closed loop control

A closed loop control strategy does not plan the entire trajectory in advance. The trajectory is allowed to unfold in real-time using feedback from the periphery to alter the path.

The advantage of closed loop control is that the motor plan may be relatively simple and may overlook non-linearities in the device being controlled. Spinal reflex circuits have been viewed as feedback regulated servo systems and is an example where closed loop theory has enjoyed much popularity (Houk and Rymer 1981).

A requirement for closed loop control is that the feedback of information be fast and accurate relative to the process being controlled. The "sloppiness" of biological proprioceptors and the delays inherent in the reflex feedback loops (Rack 1981) make it impossible that the brain uses this feedback for the control of fast movements.

2.1.2 Open loop control

In open loop control no feedback is used. The entire trajectory including both spatial and temporal aspects is determined in detail in advance. Without the advantage of feedback the planning of the motor program is far more expensive than in the closed loop case (Hollerbach and Flash 1982).

The human motor control system does not use solely open-loop control either, since once a plan has been set into action this strategy allows no mechanism for its alteration. The human motor system is clearly able to alter targets and react to changes without difficulty.

2.1.3 Hybrid control strategy

In practice the majority of control strategies employ a combination of both open and closed loop controls. The human motor control system certainly has many examples of both strategies combined at various levels.

The ballistic phase of voluntary motor action is the phase that occurs before feedback is available to alter the trajectory (Rack, 1981); this phase then, is by definition open-loop. The ballistic phase is however part of a larger control strategy that is primarily closed loop. The voluntary motor control system is therefore a hybrid control strategy.

In this research, only the ballistic phase of motor action is considered. This is an important point as it significantly reduces the complexity required for the description of the Recruitment Model, and allows this research to focus on a considerably narrower area of brain function. The consequences of this choice are a major point of discussion in the following chapters of this thesis.

2.2 Co-ordinate systems

One of the current issues in the study of reaching control concerns the coordinate system used by the nervous system in the planning and generation of movements.

The choice of coordinate systems is limited by the type of information that is fedback from the control system periphery. In the human motor control system this still leaves a choice of:- visual co-ordinates, egocentric head or trunk-based coordinates, joint angles, muscle lengths and many others.

There is still considerable controversy concerning the co-ordinate frame in which trajectories are planned. Morasso (1981) observed that the hands of subjects, when making horizontal planar arm movements between two points, follow approximately straight line trajectories. This led to the hypothesis that arm movements are planned in terms of hand position in space. These hand movements were made using highly non-linear joint excursions and in the presence of complicated limb dynamics (Hollerbach and Flash, 1982) making this interpretation a persuasive one.

Other studies (Soechting and Lacquaniti 1981; Laquaniti and Soechting 1982) have shown that, for other types of arm movements, the trajectory of the hand is not always straight, and that there appears to be instead invariances between joint angles. While these results have been disputed (Hollerbach and Atkeson 1987), more evidence has accumulated that there are joint angles (specifically the elevation and yaw angles of the upper and lower arm) which are used for the estimation of static arm orientation (Soechting and Ross 1984) and for the generation of arm movements in the absence of visual feedback (Soechting and Flanders 1989b). The use of these joint variables in the planning of movements is also consistent with neurophysiological data suggesting that movement direction is encoded in a coordinate system which rotates with the shoulder (Caminiti et al. 1992). Also, Helms Tillery et. al. (1991) have shown the position of the hand in space can be derived purely from kinaesthetic information, and more importantly they showed that this kinesthetic information can be used to reproduce arm configurations in the absence of visual information more accurately than the hand position itself can be deduced. This supports the hypothesis that kinesthetic information is not used to derive the hand position in space, and that trajectory planning occurs in body centred joint coordinates (Helms Tillery et. al. 1991).

2.3 Control tasks

Any controlled movement requires the following tasks to be carried out:-

1. calculate the start position, or the localisation of the hand or limb to be moved;
2. calculate the end position, or the localisation of the target;
3. calculate the trajectory profile, or the joint angles and torques;
4. calculate the actuator activations, or the muscle activity;

The nature of the calculation for each of these steps depends on: (i) the coordinate system being used (see section 2.2); (ii) the physical device being controlled and (iii) implementation. As a result of these three factors some of the above tasks may not be explicitly calculated and are often merged with other tasks, but each must in some way be accounted for.

The above steps are useful for providing a methodical breakdown of tasks when solving real-world engineering problems. It is unwise to go looking for an implementation of these explicit tasks in the human brain. The task list above implies an inherently sequential order, this is diametrically opposed to the intrinsically parallel nature of the brain.

The human control system uses fundamentally different processing techniques which are parallel by design; the Recruitment mechanism is an example of this (see section 3.4.3) and is exploited throughout the Recruitment control model. In contrast a "standard implementation" of the above tasks is known as a hierarchical controller, which proceeds through tasks similar to those above in a sequential order. The majority of implementors of this hierarchical model have to perform complex computations when calculating tasks 3 and 4, these calculations are known as inverse kinematic and inverse dynamic analysis. These calculations are discussed below.

2.3.1 Inverse kinematic analysis

Tasks 1 and 2 above (section 2.3) define the start and end points of the path. The joint angles and velocities then need to be calculated; this highly non-linear calculation is known as "inverse kinematics" (Saltzman,1979). The human forelimb has seven degrees of freedom, one more than the six necessary to position and orient to any object in space. The additional degree of freedom implies that the inverse kinematics problem is a one-to-many mapping (An, 1988); that is, for any given hand position there are a number of possible joint configurations. Thus, for the nervous system the inverse kinematics problem not only represents a difficult calculation, but one of selecting criterion for the reduction of the number of degrees of freedom. Many criterion have been tried in different models including minimisation of time, acceleration, energy and jerk (Nelson, 1983).

2.3.2 Inverse dynamic analysis

At this stage in the control process the necessary joint angles and velocities have been computed. The nervous system has control over joint muscles, just as other control systems have control of robot actuators. The angles and velocities computed in the previous stages are converted into joint torques at this stage. This process is termed "inverse dynamics" and is complicated by inertial, centripetal, and coriolis torques generated in a moving multijoint system (Hollerbach, 1982; Hollerbach and

Flash, 1982). In addition, the forces which are to be applied to the environment through the hand must be taken into account (An, 1988). The inverse dynamics step is possibly the most computationally intensive of the entire hierarchical process, and this computation is further complicated by the fact that there is no analytic solution to the inverse dynamics problem.

In robotics, the inverse dynamics problem is solved by lengthy numerical approximation methods (Saltzman 1979, Orin et. al. 1979).

2.3.3 Muscle activation

The final task is to take the joint torque information and to convert this into actuator values or muscle activation patterns. In the actuator case this is now a straightforward process, the actuator characteristics are precisely defined and easy to control. Human muscles are not so simple, they consist of many smaller components known as muscle fibres.

Each muscle fibre has a point of origin and insertion. The point of origin is defined as the fixture point on the bone that does not move under the muscle contraction, and the point of insertion is the fixture point on the bone that moves. The origin and insertion points may vary within a muscle (section 3.4.3). The characteristics of the fibres (and therefore the muscle) vary as the joint rotates and the muscle length changes. Each muscle has a characteristic length-tension relationship. This scenario is further complicated by bi-articular muscles, i.e. muscles with multiple insertion or origin points (section 3.4.3). Muscle fibres also will not operate continuously like a robotic arm actuator; the fibres need to be used for short periods and then allowed to rest to prevent muscle fatigue (see section 3.4.2).

The majority of control models ignore the properties of the human muscle since it has been shown that they can be approximated to behave in a linear fashion by assuming muscle stiffness regulation that compensates for muscle irregularities (Nichols & Houk, 1976). This is the case in virtually all models that are eventually implemented on a robotic arm. The Recruitment model of motor control proposes a model of muscle activation (see chapter 5) which implies not only a useful role for this complicated physiology, but also a necessary one.

2.4 Physiological observations

This section provides details on measurements and observations which have been made for differing categories of arm movements. The calculations are typically used as reference parameters by which models that claim to be biologically plausible are judged.

2.4.1 Bell-shaped profiles and profile invariances

Many researchers have observed that the velocity profiles of simple arm movements are approximately bell-shaped (Abend, Bizzi and Morasso, 1982, Atkeson and Hollerbach, 1985). Furthermore, the shape of the bell if re-scaled appropriately, is approximately preserved for movements that vary in duration, distance or peak velocity. The findings of this research suggest that not only does the shape of the profile reduce wear and tear on the muscles themselves, but that the invariant aspects of these profiles is an essential component of synergy formation. Figure 1.1(a) shows how velocity profiles when rescaled appropriately have a similar bell-shaped profile. These velocity profiles were generated over a fixed distance at several different velocities (Atkeson and Hollerbach, 1985). Figure 1.1 (b) shows how differing targeted force levels are reached in equal amounts of time, demonstrating duration invariance.

2.4.2 Fitts law

In 1954 Fitt conducted a systematic analysis of the relationship between speed and accuracy that has remained nearly unmodified for over forty years. In Fitts'paradigm (or Fitts' task), a person is to tap a hand-held stylus alternately between two target plates as rapidly as possible for a 20-sec trial. The two targets are usually rectangular and oriented as shown in Figure 2.1, with the long dimension perpendicular to the line between the two targets. Both the width of the targets (W) and the amplitude of the movement between them (A) can be altered from condition to condition producing a large number of possible combinations of A and W. The task is scored as the number of taps (regardless whether they are correct) in 20 sec, but subjects are cautioned to make no more than about 5% errors in their movements. Fitts found that the relationship between the amplitude (A) of the movement, the target width (W), and the resulting average movement time (MT) was given by the following equation:

$$MT = a + b[\mathrm{Log}_2\,(2A/W)]$$

where MT is the average movement time for a series of taps, computed as the trial duration (20 sec) divided by the number of taps completed in that time. The values a and b are empirical constants and are required in order that the mathematical equation of the line actually fits the observed data from the experimental setting.

The form of the Fitts equation is linear and shows how $\mathrm{Log}_2(2A/W)$ is related to the movement time i.e. it relates the "difficulty" of the movement to the time taken. For this reason Fitts called this value the index of difficulty.

Fitt's law is unique in that it relates to virtually all motor acts under many conditions, and has caused consternation as to how a simple mathematical relationship can be used to describe such a complex movement process.

2.4.3 Synergy formation

It has been observed that during human arm movements groups of muscles become active and inactive during the course of a movement. The group of muscles acting in cooperation at any instant in time are known as a synergy.

It is not sufficient to calculate a movement in terms of x, y, and z displacements, and then to simply activate the muscles or joints to move in direction x, followed by y and z (see Figure 2.2). The displacement in each direction must start and finish at the same time. The muscles used during a typical synergy will span multiple joints and will need to contract unequal amounts in equal time.

The problem is further complicated when the more detailed physical arrangement in individual muscles is considered. Due to leverage effects, rotations about joints are not simply achieved by one muscle, but many. Each individual muscle is itself broken down into compartments and groups of muscle fibres. These fibres and components join in the movement of the joint at differing positions and under differing conditions (section 3.4.3).

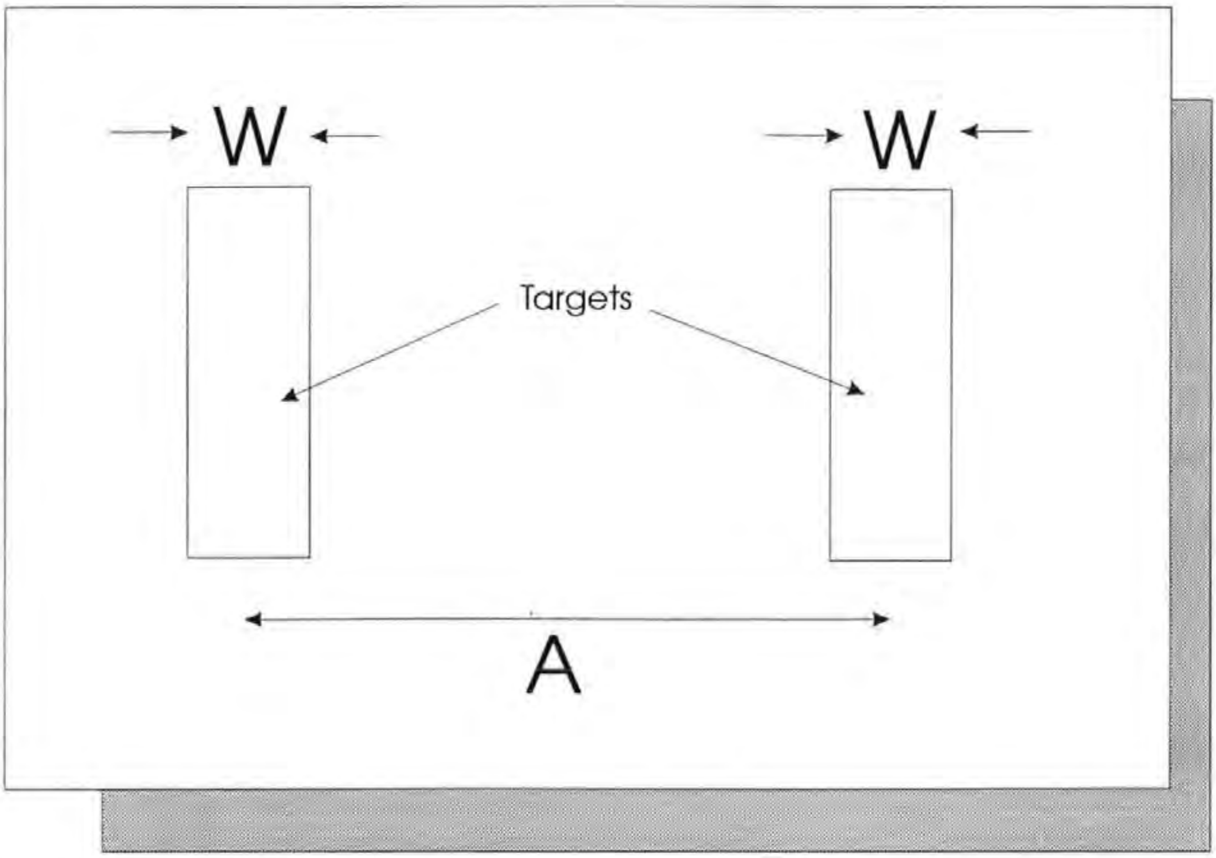
To determine the muscles cooperating in a synergy and their activation levels, a number of parameters have to be considered. Firstly it is necessary to decide which muscles are going to be used; there may be a large number of choices since there is a great deal of redundancy. Having selected the criterion for the reduction of this selection, each synergist must be controlled temoporally and spatially. Certain muscles might not be active at the start of a movement but might instead start at a half way point for reasons of biomechanical efficiency. Synergy formation is therefore a complex selection making process with strong temporal aspects.

Caminiti et. al. (1990) analysed the patterns of muscle activation for a given three - dimensional movement and noted that the patterns of activation in the human motor cortex changed as a result of the changes in the activity of many component muscles. The degree of change in muscle pattern and the muscles responsible for these changes varied as a function of the direction of movement.

Due to the complexity of the synergy formation task, a key question is:-

what is the mechanism used by the human motor system that enables these interacting components to be combined so effectively? This investigation suggests that the synergy formation problem is fundamental to building good control systems. Synergy formation is therefore a central issue which is discussed in detail in the following chapters.

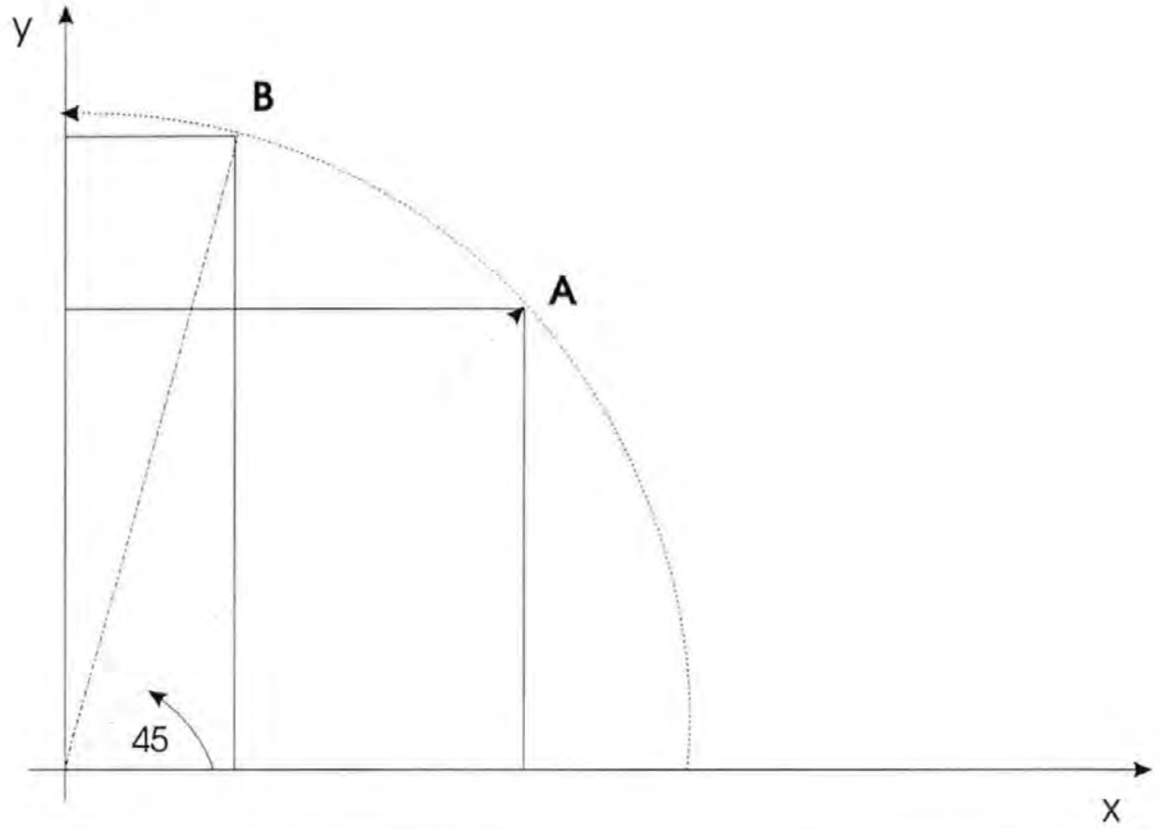
Fitts Paradigm



The performer taps a stylus alternately between two targets of width W separated by a distance A.

Figure 2.1

Synergy Formation



Point A indicates a path that requires the acting muscles to contract by equal amounts in equal time.
Path B shows a path that requires the acting muscles to contract by different amounts in equal time.

Figure 2.2

2.5 Control strategies

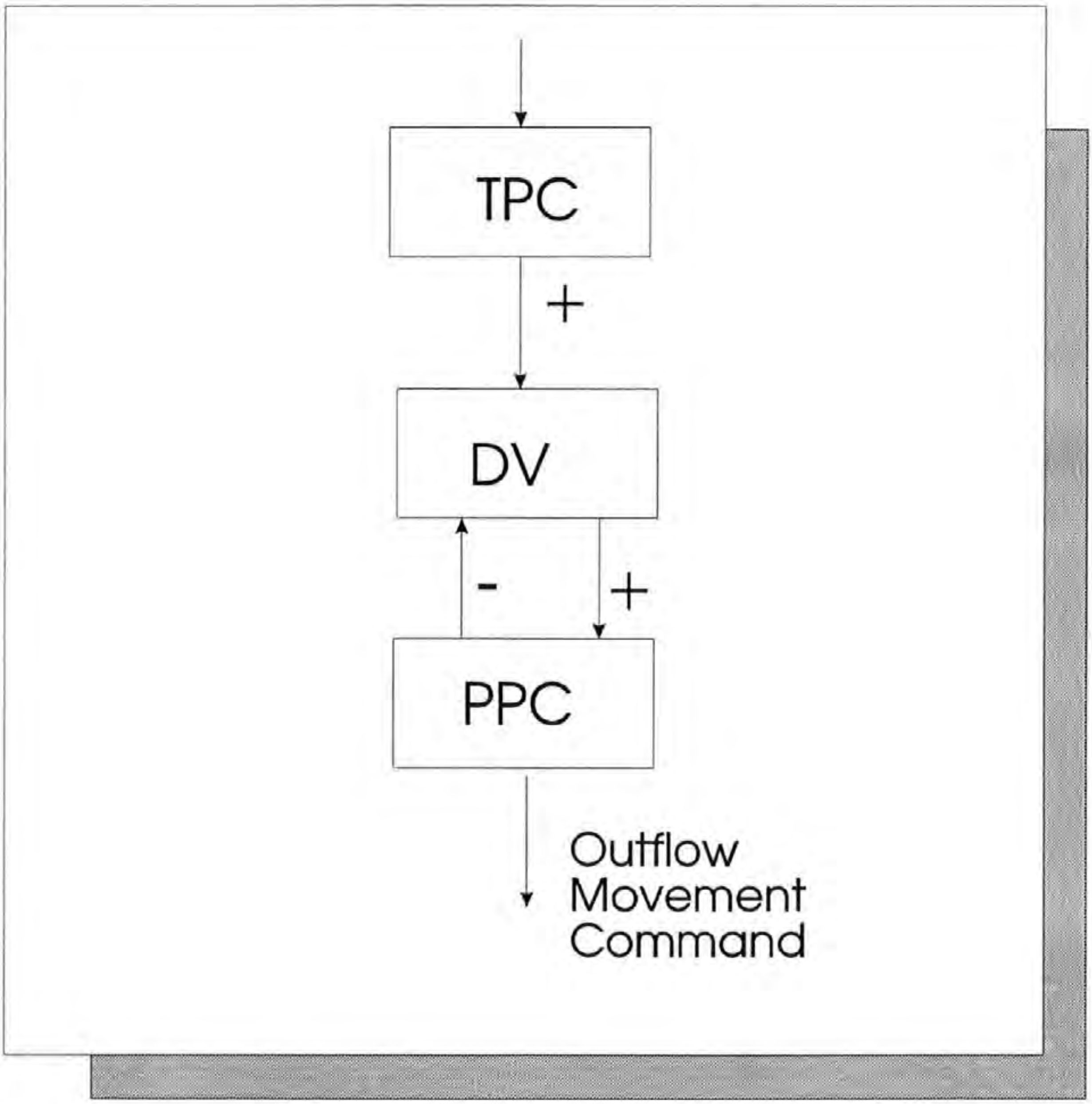
The focus of this study is on the details of trajectory formation both at a theoretical and practical level. Theories as to how trajectories are formed can be divided into two broad areas; those that suggest that the whole trajectory is pre-planned such as the Minimized Cartesian Jerk theory (Hogan, 1984; Flash & Hogan, 1985), and others that suggest that the trajectory is formed as the movement progresses such as those described in sections 2.5.1 (VITE Model) and 2.5.2 (Mass Spring models) below.

The details of these trajectory formation models are discussed in more detail in chapter 4 and chapter 5 where they relate to the description of the Recruitment model of motor control.

2.5.1 VITE/AVITE model

The VITE (Vector Integration To Endpoint) model (Bullock & Grossberg, 1988) describes a simple linear pair model of a muscle (agonist and an antagonist). VITE is based upon knowing the present muscle position (PPC - Present Position Command) and the required target position (TPC - Target Position Command). The difference between the PPC and the TPC is then calculated, and this is known as the Difference Vector (DV). A speed controlling GO signal multiplies the DV before it is integrated with the PPC. The PPC is gradually updated by integrating the DV through time. This arrangement is shown in Figure 2.3. The VITE model therefore allows the trajectory to unfold on the basis of these parameters at each instance in time.

VITE / AVITE Model



TPC - Target Postion Command

DV - Difference Vector

PPC - Present Position Command

Figure 2.3

The VITE model uses a series of equations that together accurately predict a variety of physiological data as described in section 2.4. It has been successful in explaining the form feedback takes in the various stages of the motor control process. Bullock & Grossberg(1988a) recognise duration invariance as an essential component in synergy formation, but this property is implemented directly through the system equations. The Recruitment Model postulates that the human motor pools are ideally suited to this role, and as a result have importance with regard to other roles as well.

The VITE model does not explain all the neurophysiological data reported in chapter 3, nor does it encompass information regarding muscle behaviour. The VITE model although very simple is undeniably an important contribution to the understanding of motor control.

2.5.2 Mass Spring modelling

Another approach to simplifying inverse dynamic and inverse kinematic equations has been proposed. In this scheme, the equations are not explicitly solved, and the biomechanical properties of the arm (elasticity, viscosity) are used to generate the trajectory (Feldman 1966; Polit and Bizzi 1979). The agonist and antagonist muscles activity would be set to the value appropriate for the final position of the arm, and the forces generated by the spring-like muscles would tend to carry the arm to an "equilibrium" position. It has been shown that the original formulations of this so called "equilibrium point" model, in which movements result from step changes in the equilibrium point, is not adequate to explain the trajectories of perturbed movements (Bizzi et. al. 1982, 1984, 1990). This data is instead consistent with the idea of a gradually shifting equilibrium point. The path of the equilibrium point through space has been termed the "virtual trajectory" of the arm (Bizzi et al. 1984; Hogan 1988). Because of the uncompensated dynamic torques generated by the moving arm, the hand does not precisely follow the virtual trajectory. The nervous system can however, control the degree of precision with which the arm follows the virtual trajectory by regulating the stiffness of the arm (amount of contraction of agonist antagonist muscle pairs; Hogan 1988). While this model permits complicated dynamics problems to be treated as a simpler statics problem (Hestenes, 1994), exactly how the appropriate muscle activities are generated from the virtual trajectory has not been addressed.

2.5.3. Trajectory formation utilising parameter optimisation

There are many examples of control models based upon minimisation or optimisation of movement parameters. A good example of this is the "minimized Cartesian jerk theory" (Hogan, 1984; Flash & Hogan, 1985) which posits the existence of a high level stage involving the explicit computation and internal representation of the velocity profile as a whole. This representation is then used for performing the desired action. The concept of computing the whole trajectory in advance has many draw backs and rightly no longer appears to have much backing. Problems include the inability of such models to react to trajectory changes or perturbations in the environment during movement. Other such optimisation models have subsequently been proposed such as Minimum Torque Change model(Wada,Y & Kawato,M 1993) which predict data better than the Minimum-Jerk model (Uno et. al., 1989) but still requires several passes to produce these movements.

2.6 Summary

In this chapter an overview has been given of important control issues that will relate to the later discussions on the Recruitment model of motor control.

The Recruitment model is at present a theoretical model of ballistic motor production i.e. it is the phase of a voluntary reaching movement that occurs before feedback is available to modify the movement. This chapter outlined some issues relating to co-ordinate schemes and synergy formation, both are important issues which are discussed in detail in later chapters.

A key question examined in this research is whether the additional complexity of modelling structures similar to human arm muscles will lead to any emergent benefits in a trajectory formation model as a whole. This question is pursued experimentally and theoretically in the following chapters. The trajectory formation models outlined in section 2.5 are known to be able to reproduce features observed in human arm movements, such as bell-shaped profiles and duration invariances (section 2.4.1) under certain conditions. These will be used to assess the performance of neuronal modelling carried out as part of the Recruitment model of motor control in chapters 4 and 5.

3. Neurophysiological & Biological Review

3.1 Introduction

The human brain is an immensely complex structure, the sheer number of individual cells and the myriad of connections between them are beyond visualisation. The actual form of the processing undertaken is fundamentally different from that of symbolic computing. It is inescapable however, that the human brain carries out many functions superbly well, visual and auditory acuity are just two of the many notable examples.

The human motor system is also an example where artificial systems fall well short of the performance attainable in the human motor system. There is however no single area that can be identified in the human brain as carrying out motor function, instead it is spread throughout many areas both cortical and subcortical. All the areas associated with motor function interact to affect the outcome. Although some of the main motor areas now have fairly well defined contributions there still exist numerous areas, of which our understanding is very limited.

3.2 Nomenclature

Historically most of the designations for cortical brain regions have arisen as a result of lesion studies, followed by coarse electrode mapping studies and more recently by detailed electrode mapping studies. Due to the inaccuracies and coarseness of early studies many investigators adopted their own naming conventions. This of course has led to great confusion and made the accurate interpretation of data (especially early studies) difficult. In order to partially alleviate this problem Figure 3.1 lists a number of prominent researchers and their naming designations for different regions. The definitions are as reported from a group meeting of Wise et. al. 1991. The cortical motor areas all form parts of the cerebral cortex, of which the main areas are the primary motor area (M1) and the nonprimary motor areas, which may include parietal as well as frontal fields. The frontal nonprimary areas can be further subdivided into premotor cortex (PM), supplementary motor cortex (M2), and medial motor areas. This thesis has been produced adopting the naming convention of Wise et. al. (1991). The finer subdivisions shown in Figure 3.1 are not discussed in this thesis. The positions of the primary motor areas on a monkey's cerebral cortex can be seen in Figure 3.2, defined according to Wise et. al. (1991) nomenclature. Investigations into motor behaviour are often undertaken on monkey's as they have motor abilities most similar to our own. The relative proportions of corresponding areas do however vary: in a human brain the cerebral cortex is more developed and larger, whereas in a monkey the phylogenetically older basal ganglia structures are more developed.

Nomenclature in motor cortical areas

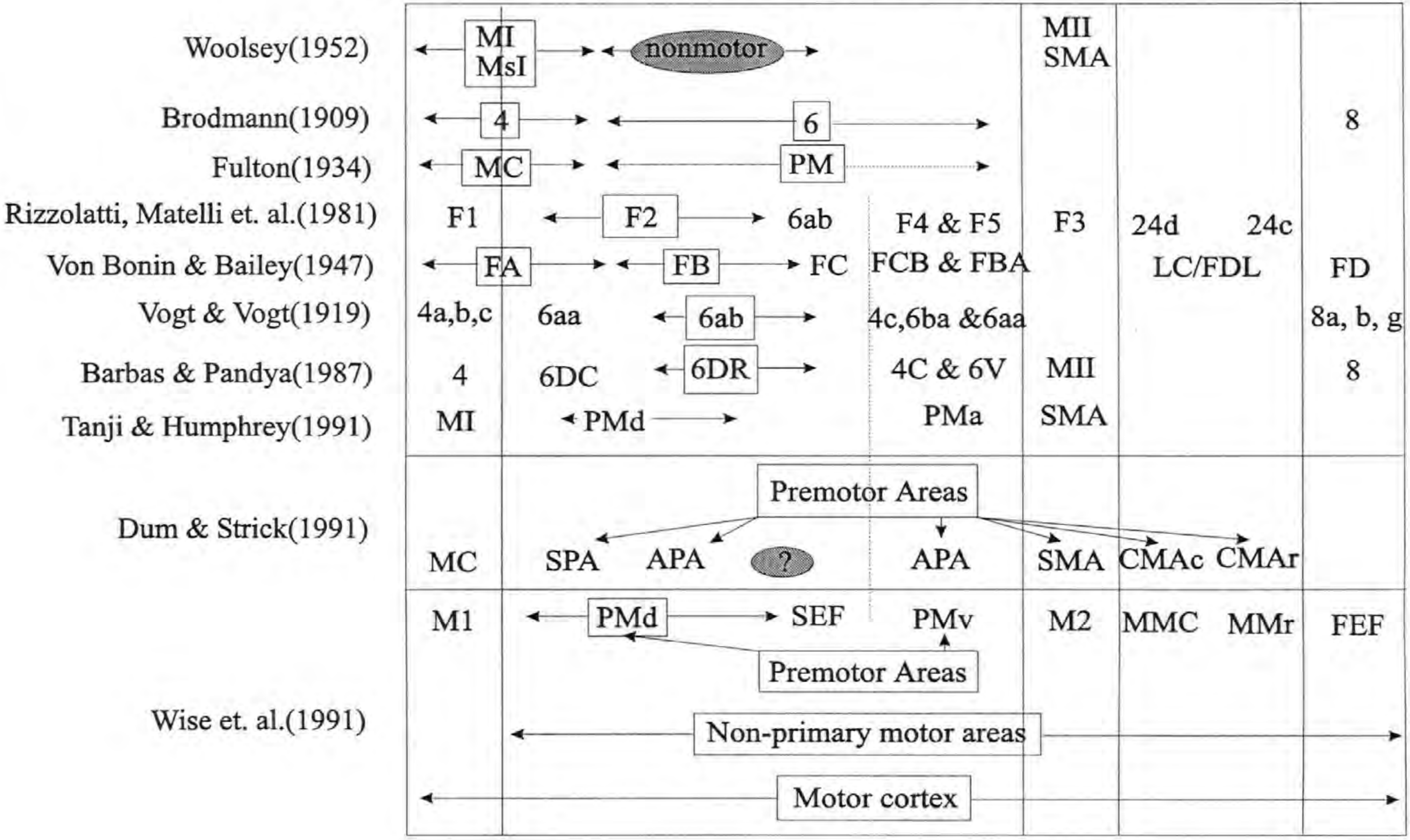
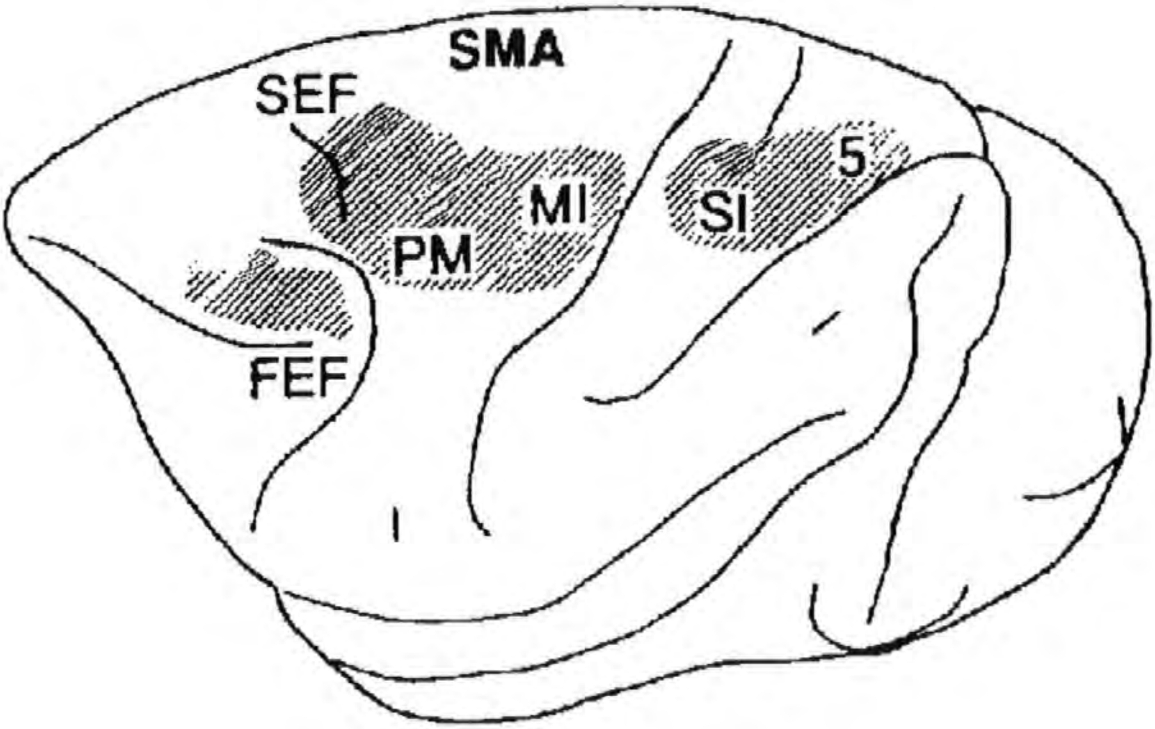


Figure 3.1

Primary Motor Areas Of A Monkey Cortex



(Diagram of the left lateral view)

During this study the main motor areas of interest are M1, which is the primary motor cortex, PM which is the premotor cortex and SMA which is the Supplementary Motor Area.

Figure 3.2

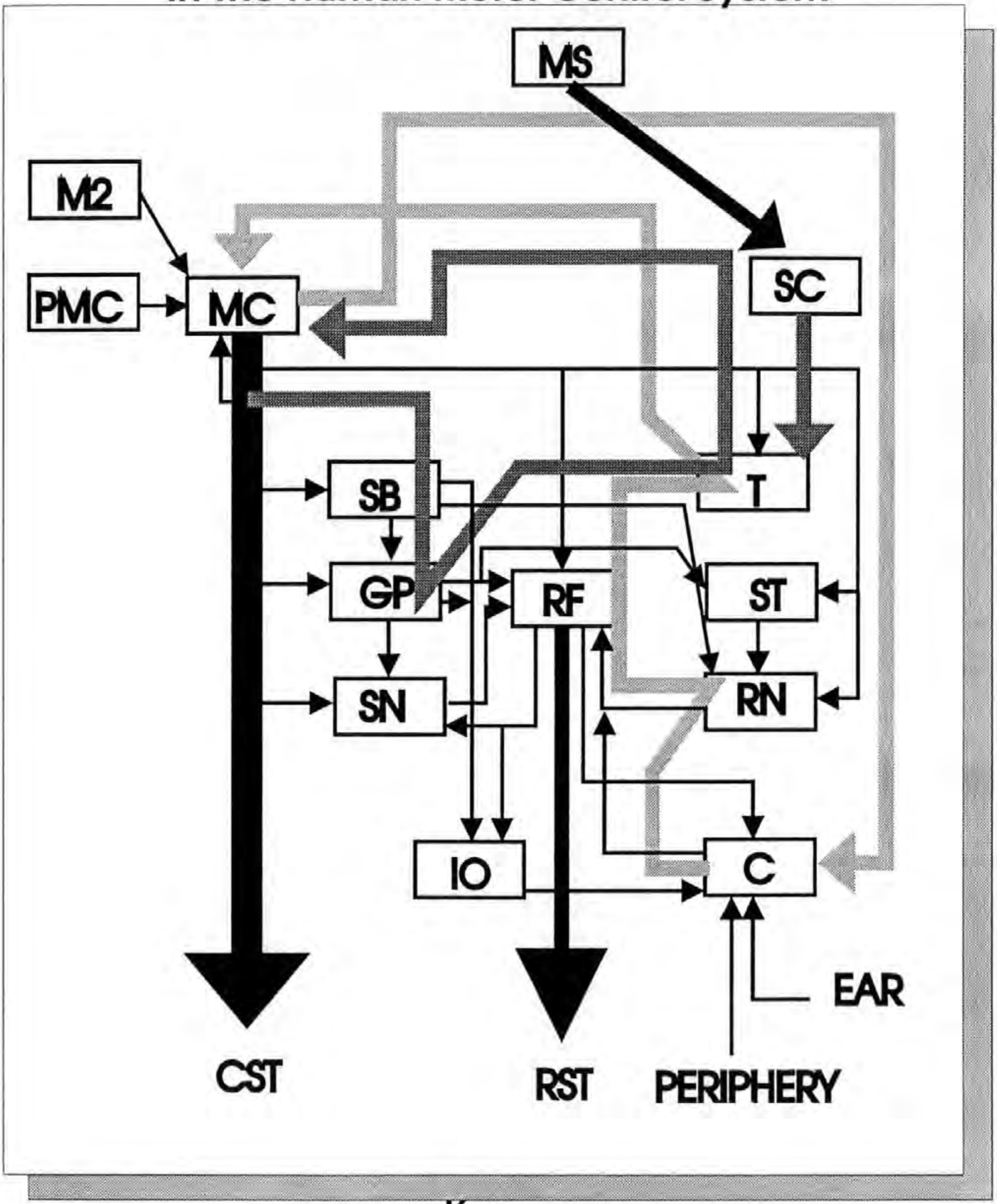
3.2 Overview of the human motor control system

The motor system in its entirety breaks down into four broadly different areas:- cortical regions, sub-cortical regions, spinal/brain stem areas and muscles. The cortical areas subdivide as shown in Figure 3.1 (see section 3.1) and are discussed in detail in section 3.3. The subcortical motor areas consists of two primary pathways.

The first is the pyramidal pathway (or corticospinal tract (CST)) which is by far the most significant, and secondly, the extra-pyramidal pathways (all the other tracts in the basal ganglia, including the rubrospinal tract) which consists of many smaller specialised tracts. The extra-pyramidal pathways pass through the area known as the basal ganglia and thalamic regions. The basal ganglia are made up of many highly interconnected sub-areas; in Figure 3.3 areas designated as being part of the basal ganglia are the striate body(SB), globus pallidus(GP), substantia nigra(SN), subthalamus(ST), red nucleus(RN) and inferior ollivellecur(IO). Figure 3.3 shows the main feedback pathways within these areas. Due to the highly interconnected nature of the basal ganglia it is extremely difficult to determine the exact function of each of the separate basal ganglia areas. The pyramidal tract (CST) is relatively simpler, targeting directly onto brain stem and spinal cord regions. The neurons in these regions that are targeted by descending cortical commands are known as motor neurons, and are organised into structures known as motor pools. The motor pools are discussed in detail in section 3.4.1 and are a major topic in this study. The motor neurons within these motor pools then directly innervate skeletal muscles.

This study is concerned with only the ballistic phase of motor production, that occurs before feedback is available by way of extrapyramidal pathways. The basal ganglia and thalamic regions are therefore precluded from further discussion in this thesis.

Key Motor Areas And Pathways In The Human Motor Control System



Key

ST - Subthalamus
RN - Red Nucleus
C - Cerebellum
SN - Substantia Nigra
CST - Corticospinal Tract
RST - Rubrospinal Tract

MC - Motor Cortex
PMC - Premotor Cortex
SC - Sensory Cortex
SB - Striate Body
GP - Globus Pallidus
RF - Reticular Formation

T - Thalamus
IO - Inferior Ollivellecur
MS - Higher Motor System
M2 - Supplementary motor areas
▨ Pathway A
▩ Pathway B

Figure 3.3

3.3 Cortical Areas

Pylogenetically the cortical areas represent the most recent structures of the brain. These cortical regions envelop and surround the subcortical regions. The cortical areas are relatively thin but highly interconnected with many downward connections. The most important motor cortical area is the primary motor cortex (M1 - Figure 3.2), which is in an excellent position to mediate information flow between the many other cortical areas as shown in Figure 3.3.

The additional non-primary motor areas all communicate either directly or indirectly with M1. Although some of these non-primary areas do target sub-cortical areas, the main connections are aimed at M1.

Various functions are postulated for the role of M1 from complete planning and control of all movements down to merely passing on instructions from functionally higher areas.

In the Recruitment Model of motor control the Joint Recruitment Level (JRL) and the Muscle Recruitment Level (MRL) correspond to M1 in the human brain. Both the MRL and the JRL are concerned with motor production only and are not involved in any high level planning. The neurophysiological basis for this and other aspects of the Recruitment control model (see chapter 6) are discussed in section 3.3.1. A discussion of the role of the non-primary motor areas is given in chapter 3.4. to give some background information to aid understanding of the role of M1, but they are not the primary focus of this study.

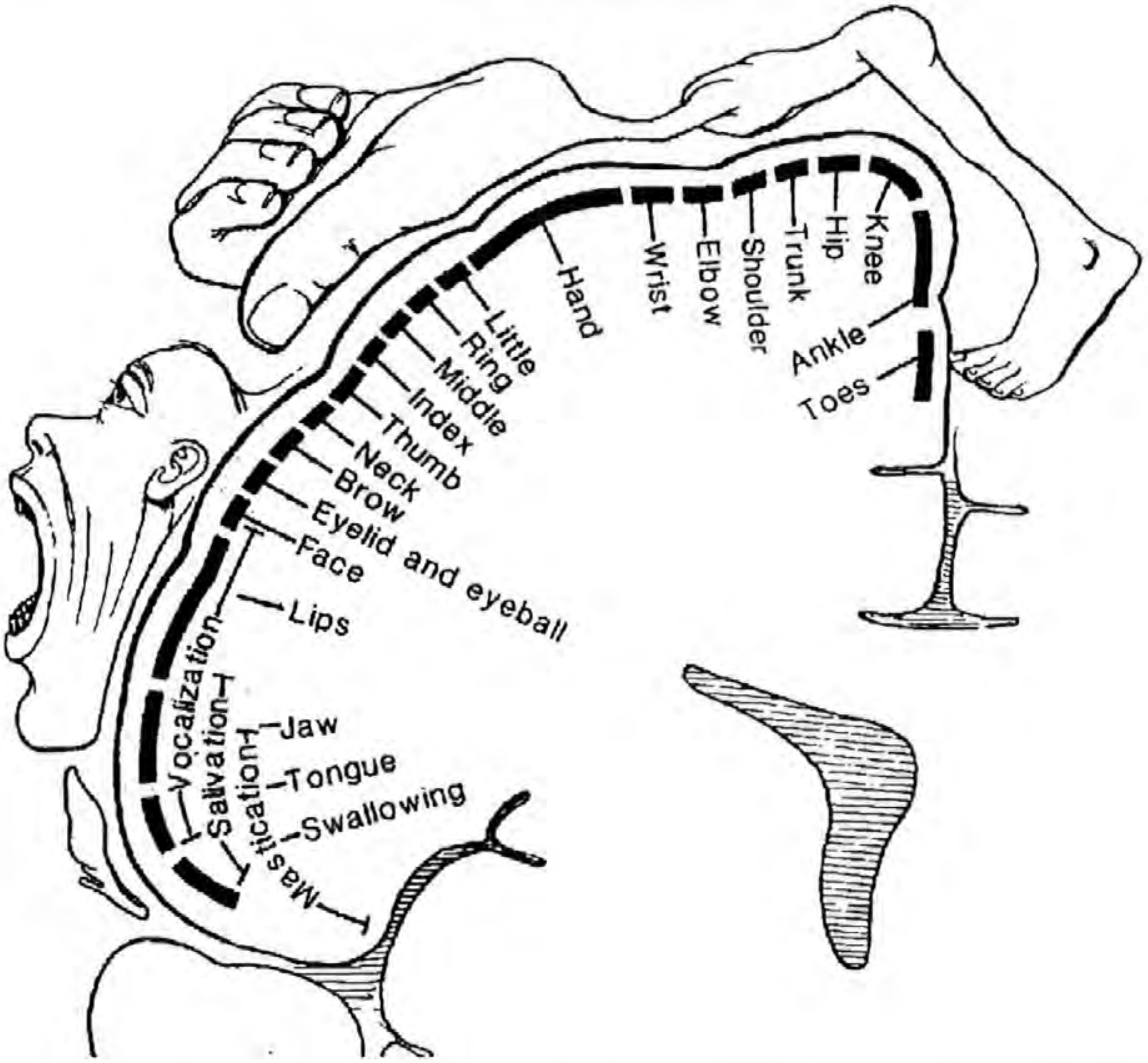
3.3.1 Motor Cortex

The primary motor cortex was the first cortical area to be discovered as having motor function. A multitude of primitive experiments established the link between the primary motor cortex (M1) and motor function by applying electrical stimuli to various points on the cortex and noting any corresponding elicited movement. These studies then proceeded to map out various topographical relationships between adjacent cortical areas and the resulting bodily movements.

One of the most striking observations was that the motor cortex (M1) is organised somatotopically (Woolsey et. al. 1952), i.e. M1 is organised as a map of all the muscles that are controlled, the relationship between the representation for neighbouring muscles in M1 is the same as the physical relationship between muscles (Figure 3.4). The size of the relationship in M1 reflects the function of the muscle being controlled, for example a muscle requiring gross degrees of control like a thigh muscle would have a proportionally small representation compared to that of the hand which has a much larger area so that fine, controlled movements can be made (Figure 3.4). This somatotopy in the motor cortex is sometimes referred to as the motor homonoculas or homonocular, and the other commonly referred to map is in the sensory cortex and is referred to as the seminicular map.

The reason for this organisational arrangement is not clear, but some degree of somatotopy is preserved throughout the entire motor system including sensory areas. The Recruitment Model exploits this aspect of cortical design and suggests that it is necessary to facilitate the efficient use of available muscles (see chapter 6).

The phenomenon of somatotopy in the motor cortex



The figure above schematically depicts the phenomenon of somatotopy, observed in areas such as the motor cortex and the sensory cortex. The figure shows that muscle representations in the motor cortex have similar relationships to their physical arrangement e.g. hand muscles adjoin arm muscles. The size of representation is dependant on the function of the target muscle. For example the representation of the hand and face areas is correspondingly larger than an arm muscle.

Figure 3.4

The neurophysiological data has been broken down into sections relevant to the functional subcategories for building a computational model. Firstly, section 3.3.1.1 describes the major input/output pathways and the type of information conveyed in them. Section 3.3.1.2 reviews the single electrode mapping studies which are less sophisticated than the population mapping studies in section 3.3.1.3. The population mapping studies have enabled significant advances in understanding the nature of all motor cortical areas and in particular M1. The early studies focused on 2-dimensional movements, but have subsequently been expanded to 3-dimensional studies.

The final section reviews how the neurophysiological data may be applied to the synergy problem (see section 2.4.3 and 3.3.1.4) and by what techniques these motor processes might occur (section 3.3.1.5).

3.3.1.1 Efferents and afferent

The motor cortex has been established as the central area for motor control and as such has the most complex connectivity with adjoining structures. This section will briefly outline the major incoming (efferent) and outgoing (afferent) connections.

The pyramidal (or corticospinal) tract is the largest afferent pathway from M1 and this projects directly onto the spinal cord and brain stem grey matter (Figure 3.3). This is the major pathway for the issue of voluntary motor commands and targets onto the motor pools which then control the individual muscles.

There are two main extrapyramidal pathways, the first being the cerebellum thalamacortical pathway (Schell & Strict, 1984 - Figure 3.3 pathway A) , and the second being the somatosensory thalamacortical pathway (Schell & Strict, 1984,

Figure 3.3 pathway B).

The first of these pathways is involved in providing feedback as a voluntary movement proceeds (particularly from the cerebellum), and the second is involved in issuing commands on the basis of proprioceptive sensory input. The other extrapyramidal pathways within the basal ganglia seem concerned primarily with information sharing between the many disparate areas, and probably are specialised systems concerned with things such as posture control and locomotion (Johnson, 1992).

The inputs to the motor system are equally diverse coming from adjoining nonprimary cortical areas such as the supplementary motor areas (M2), premotor (PM) and parietal areas (e.g. SEF and FEF) (Figure 3.2). All these cortical areas exchange information directly. The adjoining motor areas however do not communicate through M1 exclusively and have been shown to target sub-cortical areas directly. Other cortical areas such as auditory, visual and sensory communicate via the thalamacortical pathways. From this information it is known that the motor cortex is linked either directly or indirectly to every sensory modality available. The outputs of M1 are tightly linked to the output motor response and not the incoming stimuli. The motor cortex primarily responds to proprioceptive information and shows only loose association with visual and auditory stimuli (Tanji and Kurata, 1985). The motor cortex does however respond to incoming proprioceptive stimuli even if a movement is not evoked (Fetz et. al, 1980).

Using a technique of spike triggered averaging, Fetz and Cheney (1980) and Fetz et. al. (1990) discovered that each M1 cell that had connections along the pyramidal tract, targeted on average between 2 or 3 motor neuron pools, although the possible range was between 1 and 6. Of the 65 M1 cells studied, 60% exhibited pure facilitation of one or more agonist, 30% exerted reciprocal facilitation on agonist or antagonist, 8% exhibited mixed effects, and 2% co-facilitated motor neuron pools (Fetz et. al. 1990). These experimental results give an interesting upper limit on the level of divergent output available from M1. This data is further discussed with reference to the Recruitment model of motor control in chapter 6.

3.3.1.2 Single unit recording studies

M1 has been the subject of more single unit recording studies than any other cortical area. Initially the data gathered was used to provide a picture of the topographical structure of the motor cortex and to give us the representation known as the motor homonoculus (Woolsey et. al. 1952). Since then more sophisticated studies have attempted to show links between neuronal cell activation and kinematic or dynamic properties.

Various flexion/extension experiments were performed by Evarts (1968) in which a sample of pyramidal tract neurons were observed under three differing movements: unloaded, with a flexion load, or with an extension load. Evarts found that 90% of the cells in his sample varied firing rate with muscle activation patterns. A further series of experiments were undertaken (Schmidt et. al. 1975, Smith et. al. 1975, Hepp-Raymond and Diener 1983) as a consequence of these results, all of which showed a correlation between cell activity and force and thus implied that the cortex represents muscles and not movements.

Additional studies into the somatotopical organisation of M1 have shown that individual muscles could be activated by microstimulation over a relatively large area of the cortex, and that within this area multiple low threshold foci can exist (Humprey & Mitz 1989; Sato & Tanji 1989; Sanes & Donoghue,1992). These foci from different muscles tend to overlap forming a "mosaic" of multiple muscle representations. The multiple representation of joints within MI has also been reported (Humphrey and Mitz 1989), and the output from areas identified as controlling both muscles and joints can be weighted (Donoghue et. al. 1989) to change the level of facilitation of target motor neuronal pools.

3.3.1.3 Population recording studies

The population studies started with Georgopoulos et. al. in 1982 when they discovered that cell population activity in the motor cortex varied as a function of the required direction of an upcoming movement. This approach has achieved much success in the following years as it has proven to be a very flexible idea and has subsequently been expanded to encompass three-dimensional data and loading effects.

The technique of recording the activity of a population of cells, when compared to that of recording single cells is of considerably more use when the structural nature of M1 is considered. M1 as with all brain regions is a massively parallel structure. Examining individual elements of this structure is prone to error and misconception in the same way that looking at a single neuron in a perceptron type network will yield no useful information. With this premise in mind the relevance of the population studies below should be correspondingly weighted against single unit studies.

Using a "center-out" task in which the monkey was required to make two-dimensional movements using a manipulandum from a central starting position to targets located on a surrounding circle, Georgopoulos et. al. (1982) observed that the activity of each individual neuron was broadly tuned to a preferred movement direction. This activity can be described by a mathematical model in which the cell activity is proportional to the cosine of the angle between the preferred direction and the actual direction of movement. They found that in spite of complicated muscle and joint geometries involved in reaching movements, a simple relationship exists between the proximal arm region of the motor cortex and the direction of the hand movement in space.

An extension of the two-dimensional model was made by allowing free-arm reaching movements in three-dimensions (Scwartz et. al. 1988; Caminiti et. al. 1990). In this model activities of all the neurons in the population are summed to produce a population vector. It has been shown that this population vector accurately predicts movement direction in both two and three dimensional movements (Georgopoulos et. al. 1983, 1988,1993; Kalsaka et. al. 1989; Caminiti et. al. 1990, 1991).

In further analyses of the movement direction in the motor cortex, Caminiti et al. (1992) extended the three-dimensional reaching task to cover most of the extrapersonal workspace of a monkey. They found that the preferred direction vectors of individual cells shifted as the monkey performed in different parts of space. These shifts varied from cell to cell as might be expected if the cells were encoding functional groups of muscles (Burnod et al. 1992). Global measures of the population's preferred directions, however, tended to rotate with the initial position of the arm in space, suggesting that the arm movements are being represented in an arm centred reference frame. In all these studies of the directional properties of M1 neurons, the monkey has performed essentially straight line movements and the net direction between the movement origin and movement endpoint has been used as the direction of movement.

3.3.1.4 Coding mechanisms

Although the direction of movement seems to be represented in the motor cortex it is still necessary to determine in what form it is stored, i.e. are movements represented in the form of Cartesian co-ordinate system or more probably as Humphrey et. al. (1970), Thach (1978) and Murphey et. al. (1982) have suggested that a relationship exists between cortical cell activity and the angular displacements of the joints involved in the arm movement. Other recent studies have indicated that a representation of the trajectory, not just the net direction of the movement may be presented in motor cortical activity (Hocherman and Wise 1991).

Another unknown is how movement amplitudes are encoded; it is possible they are encoded in the motor cortex, but single cells showed little modulation with movements of varying lengths (Hamada and Kubota 1979; Schwartz et.al. 1987; Riehle and Requin 1989). However, this does not rule the possible representation of movement amplitude by population code. Another possibility is that movement direction and amplitude are encoded by separate systems (Riehle and Requin 1989; See also Favilla et al. 1989)

The fact that neuronal activity varied with movement direction in the above experiments could be explained by the encoding of movement endpoint (Giszter et al. 1989; Hestenes, 1994) as well as by the encoding of direction since these two variables are confounded in the standard "center-out" tasks.

Another task, the "out-center" task was employed to test the hypothesis that movement endpoint was being encoded (Georgopoulos & Massey 1985). In this task the monkey made movements of varying direction to a common central endpoint.

Cell activity recorded during the task still exhibited broad tuning to movement direction. These data show that for many cells in M1, movement direction is more important than endpoint, and the movement endpoints are probably not represented by single units at the motor cortical level but more probably represented at more of a population level.

3.3.1.5 Summary

From the single unit studies M1 can be summarised as representing:-

- force related information;
- joint related information;
- muscle related information.

From the population recording studies M1 can be summarised as representing:-

- direction;
- movement endpoint;
- trajectory;
- movement amplitude.

This does not appear to be much of a summary; one view is that for all these parameters to be represented, functionally separate system must coexist in M1 (Riehle and Requin, 1989). The Recruitment Model demonstrates a strategy that encompasses all these parameters in some form, and endorses the view that M1 encodes a transition from kinematic (joint related) related properties to dynamic related properties (muscle related properties) (Kalasaka 1992), and that the cells involved in representing these properties are organised somatotopically and therefore encode direction also. The motor cortex must be able to continuously integrate incoming signals. If M1 represents the direction of movement then over a period of time it will encode trajectory. From the single unit studies more force or amplitude related information would seem to be represented; this taken in combination with the population studies which indicate more spatially related information, gives an approximate indication of movement endpoint. The endpoint represented at any instant in time is unlikely to be the actual endpoint, due to the fact that as the temporal sequences of endpoints are integrated, the inability to actually implement each position will result in a virtual trajectory and virtual end point being encoded. This idea concurs with the Mass Spring modelling tradition (Bizzi et. al. 1984, Bizzi and Mussa-Ivaldi, 1990) and the difficulty of confirming if endpoints actually are encoded.

The population studies on the motor cortex has proved to be the most successful approach for understanding the behaviour of M1. Populations of cells have a preferred direction of action at both the joint and muscle levels. The distributed coding approach to understanding M1 has proved to be successful in other motor areas as well, such as the Premotor cortex (Caminiti et. al., 1991) and the cerebellar nuclei (Fortier et. al. 1989) as well as the coding of arm movement direction in parietal area 5 (Kalasaka et. al. 1983; Kalasaka, 1988). From this information it can be deduced that M1 has a common communications protocol with the many other motor areas (Godschalk et.al. 1984). This protocol is preserved through connectivity, activation levels and probably temporally. This concept predicts a role for a time signal with which to synchronise all contributing motor areas.

From the information contained within M1 and its position with respect to surrounding motor areas it appears to be a central area through which all primary motor function is executed. Certainly M1 is well placed for selection and control of both joints and muscles. It does not however appear to carry out any form of planning or complex sequences of control.

Although it is known that a transition between joint information and muscle information exists, the method of transition is still unknown. The process by which the populations of cells are selected has not been fully described, although "recruitment" is the proposed mechanism in the Recruitment Model.

The Recruitment Model also reconciles data concerning the level of divergent output from M1 that engages with motor pools. Consider a cell that has a preferred direction of action that is the same as the direction of action of a muscle with respect to the joint it acts around. In this (the ideal) case it would have only one divergent axon, which would engage the motor neurons of the appropriate motor pool (section 3.4.1) related to the target muscle. In cells that encode directions that fall between the actual muscles directions of actions, the divergent axons would engage multiple motor pools with appropriately weighted connections. In the worst case the maximum would be twice the maximum number (allowing for agonist and antagonist) of degrees of freedom available at the joint being represented. For the shoulder joint which has three degrees of freedom this would be six, which is the reported maximum number (Cheney and Fetz, 1985, Lemon et. al. 1986) of divergent axons from M1 that target motor pools. In the Recruitment Model the process of encoding a trajectory is one of recruiting the appropriate populations of functional cells at the appropriate time. This process is explained in more detail in chapter 6.

3.3.2 Other Cortical Areas

Of the remaining non-primary motor areas, the following structures are the most significant:- premotor cortex (PM), supplementary motor area (M2). Their designation as "non-primary" motor areas is however misleading since their reported role in the motor system is more "supra-motor" rather than "sub-motor" as both have been found to be an important part of the motor system.

The roles that these areas play are more subtle and less easy to define than M1. This fact is compounded further by historical nomenclature difficulties and by the lack of more reliable population studies. Historically the bulk of information available has been made via lesioning experiments and only latterly by more sophisticated single electrode and population studies.

The premotor cortex (PM) occupies a large area of the frontal lobe. In the human brain PM is six times larger than M1. The medial surface of the frontal lobes is designated as the supplementary motor area (M2). Both these areas are complex and are typically subdivided into 2 or 3 other functional areas. For the purposes of this review however they will be summarised more generally with respect to their relationship with M1.

3.3.2.1 Efferents and afferents

Projections to PM and M2 arise from a wide area of the thalamus (Darian-Smith et. al. 1990), and thus from all areas inputting into the thalamus(Figure 3.3); primarily, including the cerebellum, basal ganglia (De Vito and Anderson, 1982) and all other sensory areas. Reciprocal connections occur between M1, M2 and other parietal areas (concerned with association).

The properties of both M2 and PM were once thought to be mediated purely via pyramidal tract connections. Now it is known that they both possess direct links to the subcortical motor centres (Keizer and Kuypers, 1989).

3.3.2.2 Summary of supplementary motor area (M2)

Data on M2 is not easily synthesised into a single theory. This can be illustrated by a brief listing of the current hypothetical roles of M2: the proprioceptive guidance of movement (Passingham, 1987); the preparation for movement (Wise, 1984); the planning of motor routines (Roland, 1987); the coordination of bilateral hand movements (Brinkman, 1984); the initiation of voluntary movement (Goldberg, 1985). Most of these hypothesis stress a "supra" as opposed to a supplementary role for M2 with respect to M1. The anatomical connections with M2 make it a possible gateway by which large parts of the nervous system not involved in motor control can influence the motor apparatus. This idea is compatible with metabolic ERP (event related potential) data implicating this brain region in the control or initiation of movement based on internal information (Orgogozo and Larsen, 1979). Lesion and single-unit data are equivocal on such an interpretation, but the results to date are too variable to rule it out. The recent findings showing strong projections to M2 from subcortical and spinal regions, raises doubt as to whether M2 can be considered as purely "supramotor" either (Dum and Strick, 1991).

In the Recruitment Model M2 would be categorised as a Higher Motor Center (HMC), primarily as a result of its connectivity with M1. At present the Recruitment Model makes no further analysis of M2's role.

3.3.2.3 Summary of premotor cortex (PM)

Anatomically, PM is in a position to mediate information flow to M1 from prefrontal and parietal cortices. PM receives a lot of information which is thought to be visually related. Lesion studies indicate that PM is normally involved in the processing required to generate or select sequences of complex movements on the basis of visual information (Luria, 1980). Many studies have found PM cells to be particularly active before movements in response to sensory (usually visual) stimuli (Godschalk et. al. 1985; Okano and Tanji, 1987).

The "instructed delay" task is where a monkey is informed what the upcoming movement will be by an instruction stimulus (IS). Following a delay period (known as "set" related activity) during which the monkey is required to remain motionless, a trigger stimulus is given to inform the monkey that the movement is to be performed. In such a task cell activity has been correlated with the IS (Godschalk et. al. 1981, 1985), "set" related activity (Evarts et. al. 1984) and movement direction (Kurata and Wise, 1988a).

Such data strongly suggests that PM is involved in the preparation for, or planning of, upcoming reaching movements, but is strongly influenced by sensory signals.

In the Recruitment Model of motor control PM would be categorised as a Higher Motor Center (HMC), but at present the model makes no strong predictions about the exact function of PM.

3.4 Subcortical Areas

3.4.1 Motor Pools

The motor pools are located in the brain stem and in the spinal cord grey matter. The smallest element of a motor pool is a motor neuron. A motorneuron has a cell body and dendrites which are in the central nervous system (CNS), a peripheral axon, and a set of muscle fibers, which are innervated by it. These fibers are collectively referred to as a motor unit (Liddell and Sherrington, 1925).

There is general agreement that, in fully mature muscles, an individual muscle fibre receives innervation from one, and only one, motorneuron (Burke, 1991). A given motorneuron, on the other hand can innervate tens or hundreds of individual muscle fibers. This innervation ratio is one of the several factors that determine the force output from an individual muscle unit (Burke 1981).

The mammalian neuromuscular system is generally considered to be organised in a hierarchical manner. Individual muscle fibres are organised in motor units, each of which is controlled by a single motorneuron, and the ensemble of motor neurons innervating a single muscle is organised into a contiguous nucleus of cell bodies in the spinal cord or brain stem and are known as a motor pool. The force output of the motor pool may be controlled by both the number of such motor units "recruited" from this functional pool and the firing rates of the active units (Burke, 1981).

If a motor unit is active within a pool it is deemed to have been "recruited". If a unit changes from a active to inactive state it is deemed to have been "derecruited". The mechanism by which units are activated and deactivated is known as recruitment. The following section describes the mechanism that allows this to occur.

3.4.2 Recruitment

Recruitment refers to the activation and deactivation ("de-recruitment") of motor units from a motor pool during reflex or voluntary motor acts. The traditional view is that a motor pool includes all the motorneurons that have axons within the defined muscle nerve and which innervate muscle units within the confines of an anatomically defined muscle.

Henneman's contribution to current understanding of motor pool recruitment began over 30 years ago when he proposed a "size principle" to explain the phenomenon of a fixed recruitment order(Henneman, 1957). Although the precise wording of this hypothesis has been modified over the years, it is sufficient to quote the version published two decades after the original presentation (Henneman, 1977):-

"The amount of excitatory input required to discharge a motorneuron, the energy it transmits as impulses, the number of fibres it supplies, the contractile properties of the motor unit it innervates, its main rate of firing and even its rate of protein synthesis are all closely correlated with its size. This set of experimental facts and interrelations has been called the "size principle":-

" ... a particular voluntary movement appears to begin always with the discharge of the same motor unit. More intense contraction is secured by the addition of more and more units added in a particular sequence .... This "recruitment" of motor units into willed contraction is identical with that occurring in certain reflexes .. The early motor units in normal gradual voluntary contraction are always in our experience small ones ... The larger more powerful motor units, each controlling many more muscle fibres, enter contraction late."

Motorneuron-muscle unit interrelations have been explored by comparing work on motorneuron discharge properties to work on a wide variety of muscle properties (anatomical, biochemical, physiological etc.). Based upon these relationships, investigators have drawn inferences on the functional significance of the various associations for graded development of muscle force.

Unit Types	FF	FR	S
Motorneuron Properties			
Input resistance	low	intermediate	high
Total membrane area	largest	intermediate	smallest
Axonal conduction velocity	high	high	low
Muscle unit properties			
Contraction time	fast	fast	slow
Force output	large	intermediate	small
Fatigue resistance	low	high	very high
Functional properties			
Usual recruitment threshold	high	intermediate	low
Output force grading by recruitment	course	moderate	fine
Metabolic optimum activity	shortening	shortening	isometric
Relative duty cycle	low	moderate	high
"typical" usage	gallop,jump	walk, run	posture
Metabolic maintenance cost	low	moderate	high

Table 3.1 - Motorneuon-Muscle unit interrelations.

If all muscle units had identical mechanical and metabolic properties, specification of recruitment order would have no important functional consequence (Burke 1991). However, virtually all muscles in vertebrates have widely varying mechanical properties, which furthermore exhibit systematic interrelations with motorneuron and synaptic input characteristics (Table 3.1) (Burke 1981).

Although many muscle unit properties vary continuously, others show discrete boundaries that lend themselves to identification of motor unit types, which have both heuristic and descriptive value (Burke 1981). Three categories are now recognised in most mammalian muscles, based upon the mechanical properties of the muscle unit portion (Table 3.2).

Unit Types	Contraction	Force Output	Fatigue
FF	Fast	Large	Fatigable
FR	Fast	Moderate	Resistant
S	Slow	Small	Very Resistant

Table 3.2 - Recruitment order.

The original size principle formulation (Henneman and Olson 1965) , with its association between increasing force output, contraction speed, and threshold, fits with the notion that type S units generally have the lowest functional thresholds, type FF have the highest, and type FR have intermediate thresholds under most conditions (Table 3.1) Zajac and Faden (1985). The usual recruitment order is S -> FR -> FF (Burke 1991) and this is known as invariant recruitment.

There are many unresolved questions concerning motor unit recruitment (Stuart and Enoka, 1983; Enoka and Stuart, 1984). In addition to the key issue of its precision, these questions include its presence or absence during strong isometric contractions, and muscle contributions to force in different directions (e.g. abduction versus flexion). Moreover, much must still be learnt about the modifications of orderly recruitment by descending command signals and sensory feedback, as well as understanding the effects of ongoing (Miles & Turker 1986) and prior (Dubose et al., 1987) muscle activity, including fatigue (Kossev et al., 1987).

Several recent observations on fatiguing muscle in conscious humans have shown an association between the relaxation rate of the whole muscle and the rate of motor unit discharge necessary to optimise force output (e.g. Bigland-Ritchie et al., 1983). This finding and some subsequent work led Bigland-Ritchie and collegues (1986) to hypothesise that "during fatigue, motor neurone firing rates may be regulated by a peripheral reflex originating in response to fatigue related changes induced within the muscle". However Dubose et.al. (1987) and Stuart et.al. (1988) suggest that fatigue induced changes within the muscle must occur largely in the higher-force (threshold) motor units and that slowing of motor neuron firing rates should not be anticipated during contractions. Furthermore, there is evidence that the absolute threshold for motor unit discharge during sustained low force contraction is altered as fatigue sets in, thereby implying that a change in recruitment order must have taken place (Kossev et. al. 1987)..

This data implies a complex dynamic relationship where the so called invariant recruitment order is not always invariant. The corresponding extreme to this is to claim a process of selective recruitment which implies there is no intrinsic recruitment order, and that neurons are selected on the basis of arbitrary parameters. The Recruitment Model of motor control explores a third possibility; it proposes the existence of invariant recruitment, but within the framework of a selective recruitment order by means of connections between neurons. This could perhaps account for data reported indicating stochastic variance in recruitment order (Burke 1991).

3.4.3 Muscles

The notion of a "muscle" carries functional as well as anatomical implications. The anatomical origin and insertion of a muscle define the force vector (or vectors) that active muscle fibers joining origin and insertion can produce on the skeleton, leading directly to inferences about its function(s) as "extensor", "flexor", "abductor" etc. (see Sherringtons functional classification (1910)).

There are many complexities which surpass these simple classifications. Although many muscles produce output force in only one direction there are also a significant number that have more complex specialisations (Burke, 1991). For example, muscles that have a broad origin or insertion contain regions with quite different and even sometimes antagonistic behaviour (e.g. human deltoid or trapezius). There are two different kinds of regional specialisation (Burke 1991):-

- muscles that appear to have a single force vector on the skeleton (type A);
- muscles with subregions that produce clearly different force vectors on the skeleton (type B);

Type A muscles can exhibit simple internal architectures but sometimes have multipinnate subregions without obviously different force vectors, as occur in the lateral gastrocnemius (LG) in the cat. In the case of the LG, the five identified compartments have somewhat different histochemical fiber composition, and are innervated by discrete subbranches of the LG nerve, and exhibit little or no crossover between compartments (English and Weeks 1984). The electromyographic activity in the LG compartments during postural adjustments

(Russell et. al. 1982) and locomotion (English 1984) roughly, but not exactly, match expectations based upon a fixed pattern of motor unit recruitment (Burke 1991). This type of data implies that subtle graduations of intramuscular force can be utilised by the animals even when a multipinnate muscle is not multifunctional.

Type B muscles usually exhibit functional differences between regions. Examples include the human deltoid and trapezius, the anterior and medial portions of the cat sartorius (Hoffer et. al. 1987), all of which show relative independence of subregions during normal movements. At present there is no identifiable difference between muscles that have large insertions/origins and those that have separate heads, although there is considerable overlap in the cells that project to them (Weeks and English 1985).

Muscles that span more than one joint introduce further complexities, since they can acts as extensors about one joint and flexors at another. The movement arcs of these disparate actions can also depend on the joint angle. Multi-joint muscles that must function efficiently through a wide range of physiological length changes often exhibit muscle fibers that are arranged parallel to the muscle axis because pinnate muscles sacrifice extensibility for power (Loeb et. al. 1987).

3.5 Summary

This chapter has reviewed all the key areas involved in ballistic voluntary motor action including adjoining areas which interact indirectly. From the data reviewed it is clear the motor cortex (M1) is the most significant area involved in motor production and that this targets directly onto motor pools located within the brain stem and spinal cord grey matter. The motor pools in turn mediate the amount of contraction within the muscles themselves.

This review has shown that although there is still much debate about the exact details and nature of cortical areas, the amount as well as the quality of information is increasing rapidly. Recent studies of M1 have shown it to have a particular topographical organisation and to be encoding numerous control parameters in a population code. In a similar vein the operation of the motor pools is also unique, and with respect to the Recruitment Model provides a valuable insight into how the "recruitment" paradigm might be extended to explain how population codes are implemented in other cortical regions involved in the voluntary motor control process.

Although much work has been carried out by researchers investigating the individual areas, very little research has been carried out into how this new information might be combined to give an enhanced view of the motor production process. Chapter 6 draws upon the information presented in this chapter as the basis for proposing the Recruitment Model of ballistic voluntary motor action.

4. Neuronal models and simulation software

4.1 Introduction

The Recruitment model of motor control is based upon biologically inspired techniques, therefore the neuronal models thought to be most applicable and realistic were selected, restricted only by the practicalities of simulation. The root of most neuronal models can be traced back to the original Hodgkin and Huxley(1952) experiments on squid axons, and indeed the models used in this work are strongly derived from these early experiments. The simulations carried out in chapter 5 and chapter 6 are based on two distinct neuronal models. The Leaky Integrator (Bressloff and Taylor, 1990) neuronal model is described in Section 4.2. The Leaky Integrator models the key features of neuronal behaviour as well as still being practical to build complex networks based around it. The Synchronous neuronal model is described in section 4.4. This model was used for the later simulations in chapter 5 and subsequent simulations in chapter 6. It was developed to model spiking behaviour between neurons more explicitly than the Leaky Integrator equations. Both models were simulated using the INSIGHT TOO simulation software (see chapter 5) and later for reasons of simulation efficiency in the C and C++ programming languages.

4.2 Leaky Integrator equations

The Leaky Integrator equations (Bressloff and Taylor,1990) model membrane potential and are governed by the equations given below.

$$\frac{d}{dt}V_i(t) = -\frac{1}{\tau_i}V_i(t) + \sum_{j=1}^{m} a_{ij}(t)[S_{ij} - V_i(t)] \qquad (1)$$

where V_i is the membrane potential of the ith neuron in a network of m neurons, τ_i is the time constant of the membrane leakage current, S_{ij} is the membrane reversal potential at the ijth synapse, and a_{ij} is the input signal at neuron i from neuron j, which takes the form of a sequence of impulses:

$$a_{ij}(t + t_d) = g_{ij}\sum_{h=1}^{N} \delta(t - T_n^j) \qquad (2)$$

where t_d is the synaptic delay, g_{ij} is a constant related to the synaptic efficiency and T_n^j is the time at which neuron j fires for the nth time. These firing times are determined by the iterative threshold condition:

$$T_n^j = \inf\{t | V_j(t) \geq K_j : t \geq T_{n-1}^j + t_r\} \qquad (3)$$

where t_r is the refractory period and K_j is the threshold for the jth neuron.

4.3 Leaky Integrator simulations

The leaky integrator simulations are based upon an Euler discretisation of the above equations. Each leaky integrator neuron within a network is governed by the following parameters:

- synaptic time delay t_d;
- refractory period t_r;
- leakage time constant τ_i;
- threshold K_i;
- membrane reversal potential S_{ij};
- synaptic efficiency g_{ij}.

An impulse arriving via an axon at a synaptic knob on a dendrite is subject to a time delay, t_d (see Figure 4.1(A)) before taking effect on the cell body by either increasing or decreasing the cell membrane potential (depending whether the connection is excitatory or inhibitory). If the membrane potential exceeds a predefined limit known as the threshold and if it is outside its refractory period, the neuron is said to fire. In doing this a pulse is sent along all the connections leading away from the cell body (axons). After a pulse has been fired there is a rest period known as the refractory period during which no further pulses can be generated. The refractory period therefore determines the maximum firing rate of the cell. When no external forces act upon the cell body the membrane potential decays away with a CR (electrical capacitance-resistive network) time constant τ_i.

When a pulse arrives at the cell body, the relationship between the membrane reversal constant S_{ij} and the threshold K_i indicates whether the incoming pulse has an excitatory or an inhibitory effect. The magnitude of the effect is determined by the efficiency term, known as the conductance (g_{ij} in equation 4.1).

There are many connectional arrangements but it is important to note that an outward connection (axon) can feedback into the cell via a dendrite, causing either direct excitation or direct inhibition, determined by the nature of the connection. This is an important type of connection since it can be used to sustain cell based activity.

4.4 Frequency based equations

The second type of neuronal model used during this study is based on the mean frequency of spiking of the cell but explicitly models the temporal spike trains to and from neurons.

The equations given below govern the behaviour of the model.

$$f_{in}(t) = \frac{1}{n\Delta t}\sum_{k=0}^{n-1}\theta(\sum_{j=1}^{m}W_{ij}.S_{ij}(\mathrm{t}-\mathrm{k}-\mathrm{t}_d)) \qquad (4)$$

where $f_{in}(t)$ is the mean excitation level of the neuron over the time window of length $n\Delta t$, and where:-

k = step index;

j = index of input line;

m = number of inputs;

n = number of the time steps in the window;

Δt = length of the time step;

$n\Delta t$ = size of the time window;

$S_{ij}(t)$ = 1 if a spike exists at time t on input j

= 0 otherwise;

$\theta(x)$ = 1 if $x \geq K_1$, for some synapse threshold K_1

= 0 otherwise;

W_{ij} = Weight of a connection on input j;

U_r = Update rate constant;

$$u(t) = \sum_{n=1}^{N}\delta(t - T_n) \text{ where, } T_n = T_{n-1} + U_r \qquad (5)$$

Here, u(t) is the frequency update time of the model (determined by the update rate constant, U_r), and T_n is the time at which the update rate of the model will be met for the nth time.

If u(t) = 1 then:-

$$f_{outd}(t) = f_{in}(t).A \tag{6}$$

otherwise,

$$f_{outd}(t) = f_{outd}(t-1)$$

where A is a constant, and $f_{outd}(t)$ is the desired mean spiking rate of the neuron at the output.

If the input frequency F_{in} is above the soma threshold K_2 and:-

if $f_{outd}(t) <= F_{min}$ then $f_{outd}(t) = F_{min}$

or

if $f_{outd}(t) >= F_{max}$ then $f_{outd}(t) = F_{max}$

otherwise

$$f_{outd}(t) = 0 \tag{7}$$

Let

$$f_{outa}(t) = \frac{1}{n\Delta t}\sum_{k=0}^{n-1} S_i(t-k) \tag{8}$$

be the actual mean spiking rate over the time window. where $S_i(t-k)$ = value of output spike at time (t-k). Then,

If $f_{outa}(t) < f_{outd}(t)$, $S_i(t) = 1$ else $S_i(t) = 0$. (9)

The model's behaviour can be described as follows. The model summates at each time step the input across all its incoming connections. This input is then subject to a synapse threshold function θ, and the result of this is summated over time and averaged over a time window of size nΔt, where Δt is the size of each time step and n is the number of time steps (equation 4). The result of equation 4 represents the mean excitation level $f_{in}(t)$ of the cell body or soma. $f_{outa}(t)$ is the actual mean frequency of spiking of the cell (equation 8), and is calculated by summating the number of spikes leaving the cell over time and then averaging in the same way as in equation 4.

At any instant in time there is a desired output f_{outd}. f_{outd} is related to f_{in} as described in equation 6. The desired output is only recalculated as the update condition u(t) of the model is met (equation 5). The actual input to the soma must be above the soma threshold K_2, otherwise the desired output will be 0 (equation 7). The desired output frequency must also be within a minimum(F_{min}) and a maximum(F_{max}) frequency condition, and is modified appropriately if not (equation 7). A spike is output from the cell if the actual output is less than the desired output (equation 9).

The Synchronous neuronal model has the advantage of maintaining spatial and temporal correlations between the input and output, this is in contrast to the Leaky Integrator equations which only maintain spatial relationships. This means that coherence of incoming spike trains within a time window is detected and can be processed by this model, yielding significantly enhanced processing ability.

4.5 Synchronous model simulations

The force output of a motor pool is controlled by both the number of recruited motor units and the firing rate of these units (Burke, 1981). The Synchronous model was developed to overtly model the firing rate relationship between motor neurons. This requirement was identified as a result of the Leaky Integrator simulations described in chapter 5.

The model was implemented in the INSIGHT description language (see appendix A for source code) initially and more recently in C and C++. It was necessary to re-implement the simulation code to increase the simulation speed beyond that of the interperated INSIGHT language.

The following parameters are used during the simulation to govern the behaviour of each neuron within the network (the variables down the left hand side are as described in equations (4) to (9)). The symbols in brackets relate to the simulation parameters in Appendix B (Synchronous motor pool simulations):-

- K_1 - Synapse threshold (THRESHI)
- U_r - Update rate (UPDATERATE)
- K_2 - Soma Threshold (SOMA)
- F_{min} - Minimum firing frequency (FREQMIN)
- F_{max} - Maximum firing frequency (FREQMAX)
- A - Gain term (RF)
- W_{ij} - Connection weight (CONNECTVALUE)
- T_d - Connection delay (CONNECTDELAY)
- n = Window size (TBASE)

The examples below show how the main parameters may be combined to affect the neurons behaviour. In the simulations below F_{min} is set to 0 and F_{max} is set to 100 such that they have no affect in these simulations.

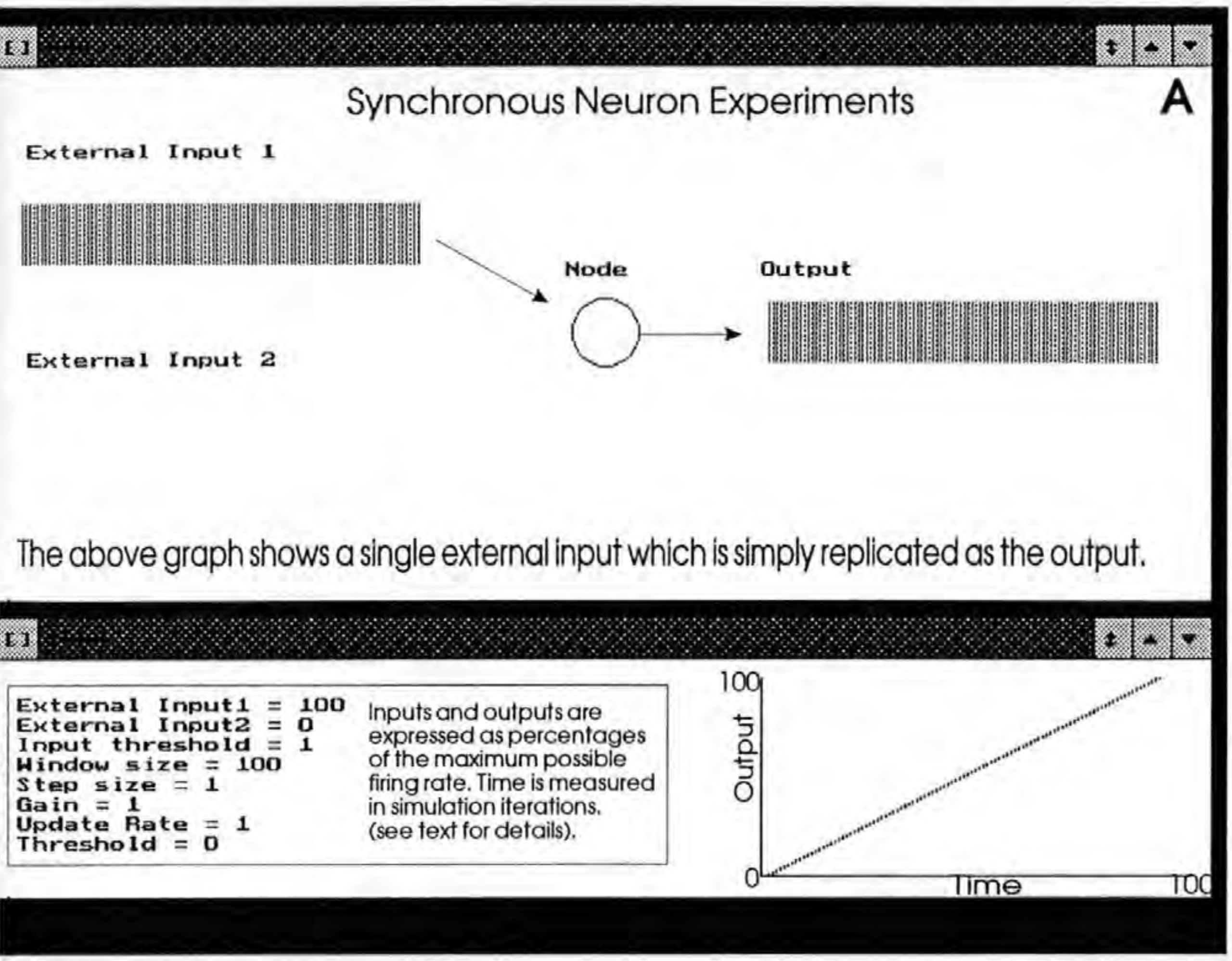
Figure 4.1a shows the behaviour of a single Synchronous neuron with a single external input. The input shown on the left shows the temporal form of the spike input train, and on the right the resultant output spike train is shown. Below in the graph the output (frequency) of the neuron is shown with respect to time. For the simulations in this section the maximum possible spiking frequency is determined by nΔt which is set to 100. Input and output frequencies are specified as percentages of the maximum possible firing rate, for example if a spike occurs at every other time step then the value would be 50%, if a spike occurs at every possible time step then the value is 100%. In Figure 4.1 a spike is present at each time step for duration of the simulation (100 time steps of size $\Delta t = 1$) repesenting an input of 100%. As the simulation proceeds the averaged input frequency gradually increases to a maximum of 100%, the output on the right is also seen to increase at the same rate. With the input threshold(k_1)=0, gain(A)=1 and update rate(U_r) set to 1, the neuron simply copies the input to the output.

If the gain term(A) is set to a constant greater then 1 then the output is scaled by A. For example in 4.1b the gain has been set to 2, the input was set at 50% and the output was 100%, exactly double the input.

The temporal form of the output from the cell body can be modified by altering the update rate (U_r) (Figure 4.2a) and soma threshold(K_2) (Figure 4.2b). In both these cases the output is given in a bursting form. An important feature of the model is how it combines inputs from multiple incoming spikes. Figure 4.3a demonstrates the neuron behaving as an AND gate; this is achieved by changing the input threshold to 2, requiring coincidence between incoming spikes on separate input lines. Figure 4.3b also shows the AND gate behaviour with lower input spike rates. By specifying that one input into the cell is inhibitory the model can behave like an XOR gate as in Figure 4.4a and Figure 4.4b.

By having multiple inputs and an input threshold of 1 the neuron will behave like an OR gate, combining both input lines into a single output line (Figure 4.5a). This output can be modulated as before by changing the update rate and gain terms as in Figure 4.5b.

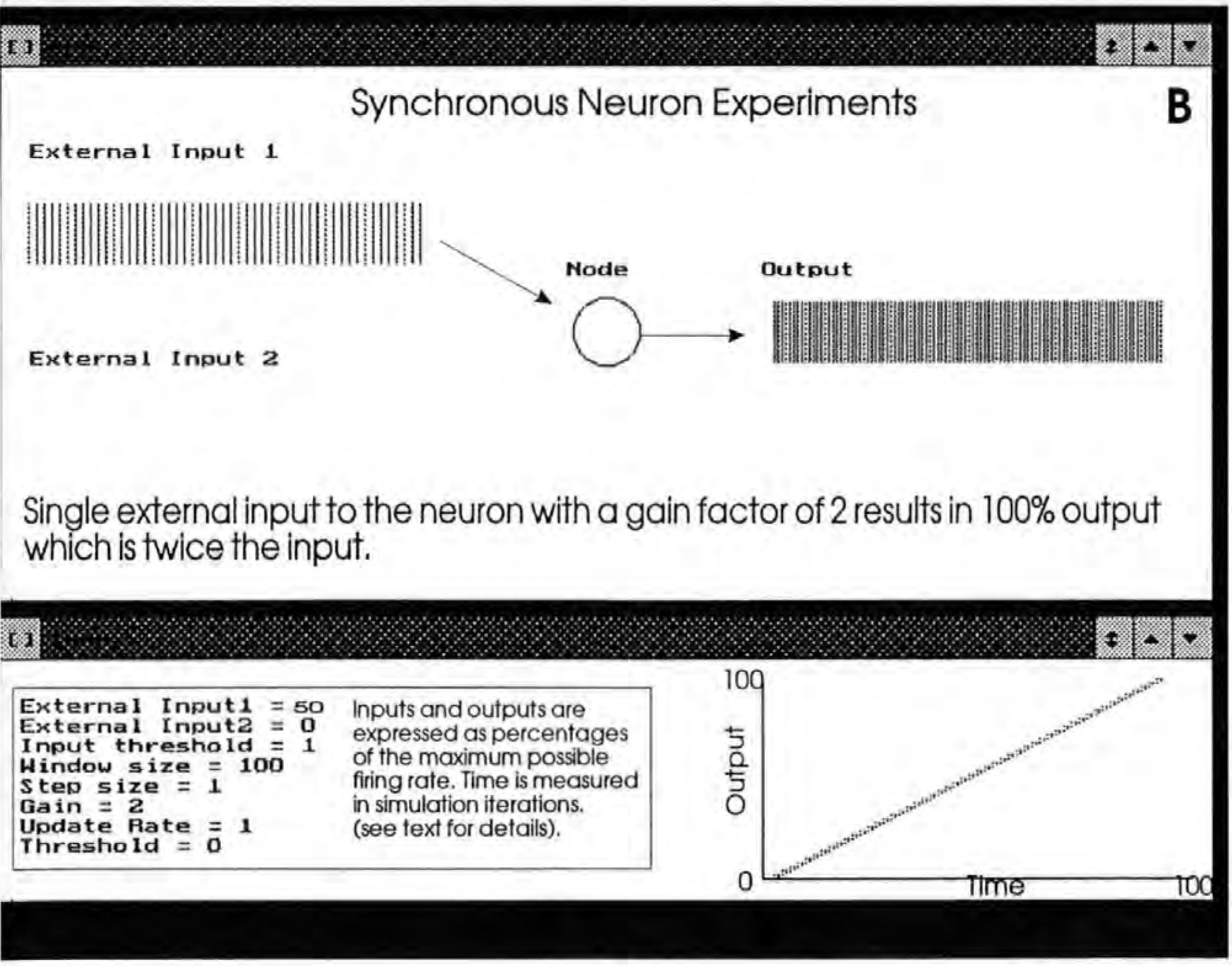
By manipulating these parameters all the basic logic functions can be produced. These cells can be combined in any way to produce complex logic operations. Further simulations with this model are described in chapter 5 and chapter 6.



The above graph shows a single external input which is simply replicated as the output.

External Input1 = 100
External Input2 = 0
Input threshold = 1
Window size = 100
Step size = 1
Gain = 1
Update Rate = 1
Threshold = 0

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

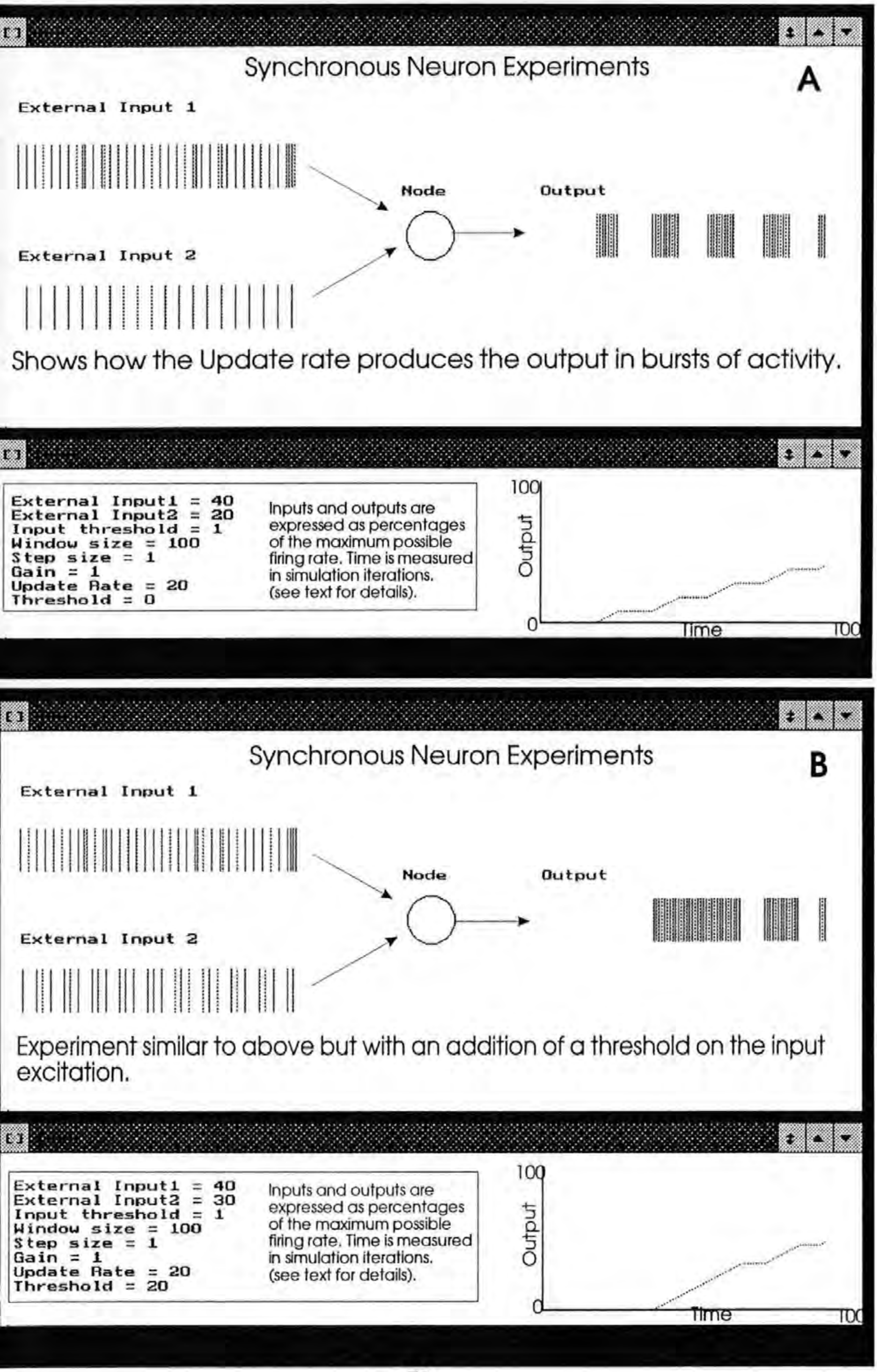


Single external input to the neuron with a gain factor of 2 results in 100% output which is twice the input.

External Input1 = 50
External Input2 = 0
Input threshold = 1
Window size = 100
Step size = 1
Gain = 2
Update Rate = 1
Threshold = 0

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

Figure 4.1



Synchronous Neuron Experiments

Shows how the Update rate produces the output in bursts of activity.

External Input1 = 40
External Input2 = 20
Input threshold = 1
Window size = 100
Step size = 1
Gain = 1
Update Rate = 20
Threshold = 0

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

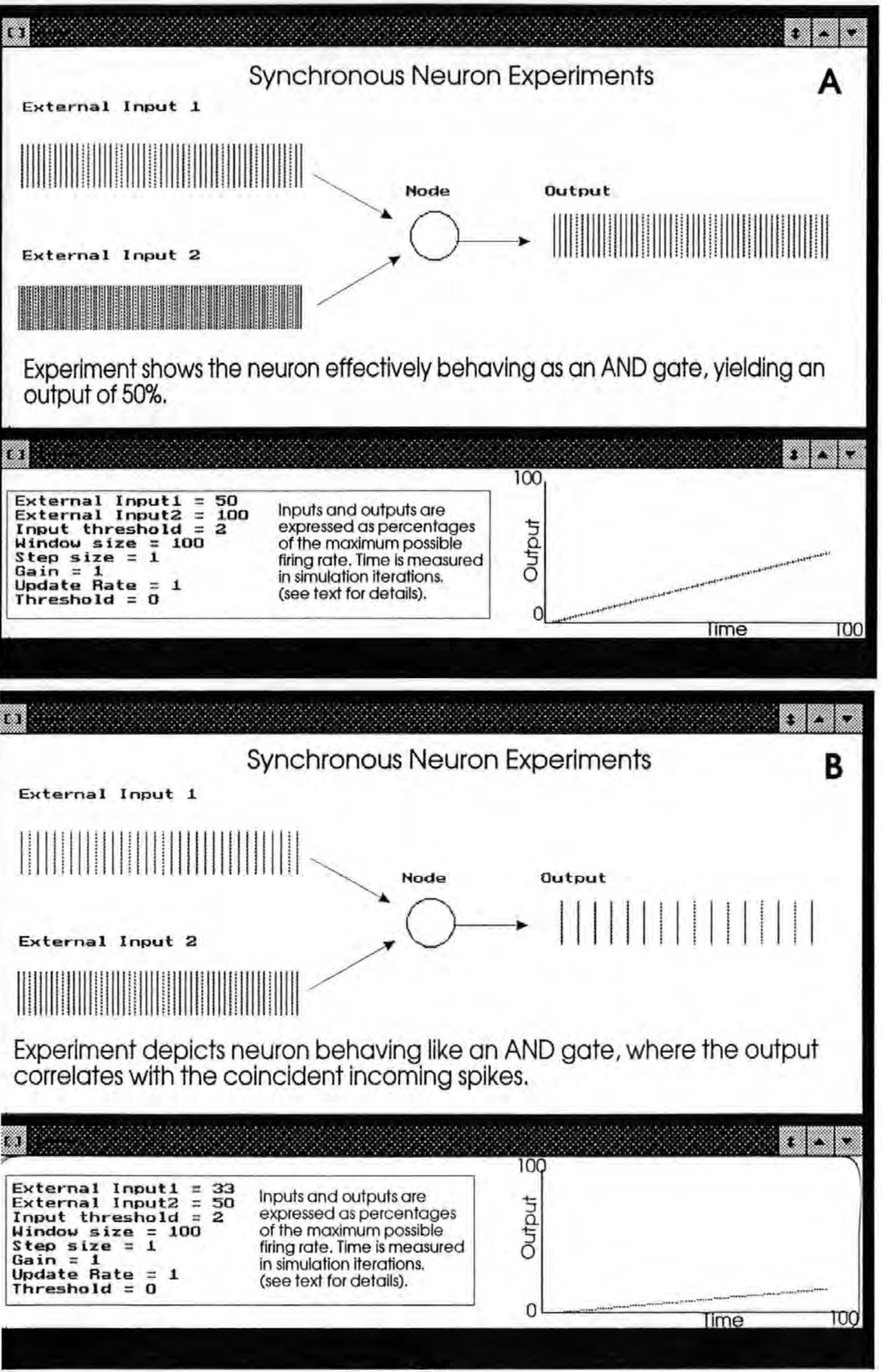
Synchronous Neuron Experiments

Experiment similar to above but with an addition of a threshold on the input excitation.

External Input1 = 40
External Input2 = 30
Input threshold = 1
Window size = 100
Step size = 1
Gain = 1
Update Rate = 20
Threshold = 20

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

Figure 4.2



Synchronous Neuron Experiments

Experiment shows the neuron effectively behaving as an AND gate, yielding an output of 50%.

```
External Input1 = 50
External Input2 = 100
Input threshold = 2
Window size = 100
Step size = 1
Gain = 1
Update Rate = 1
Threshold = 0
```

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

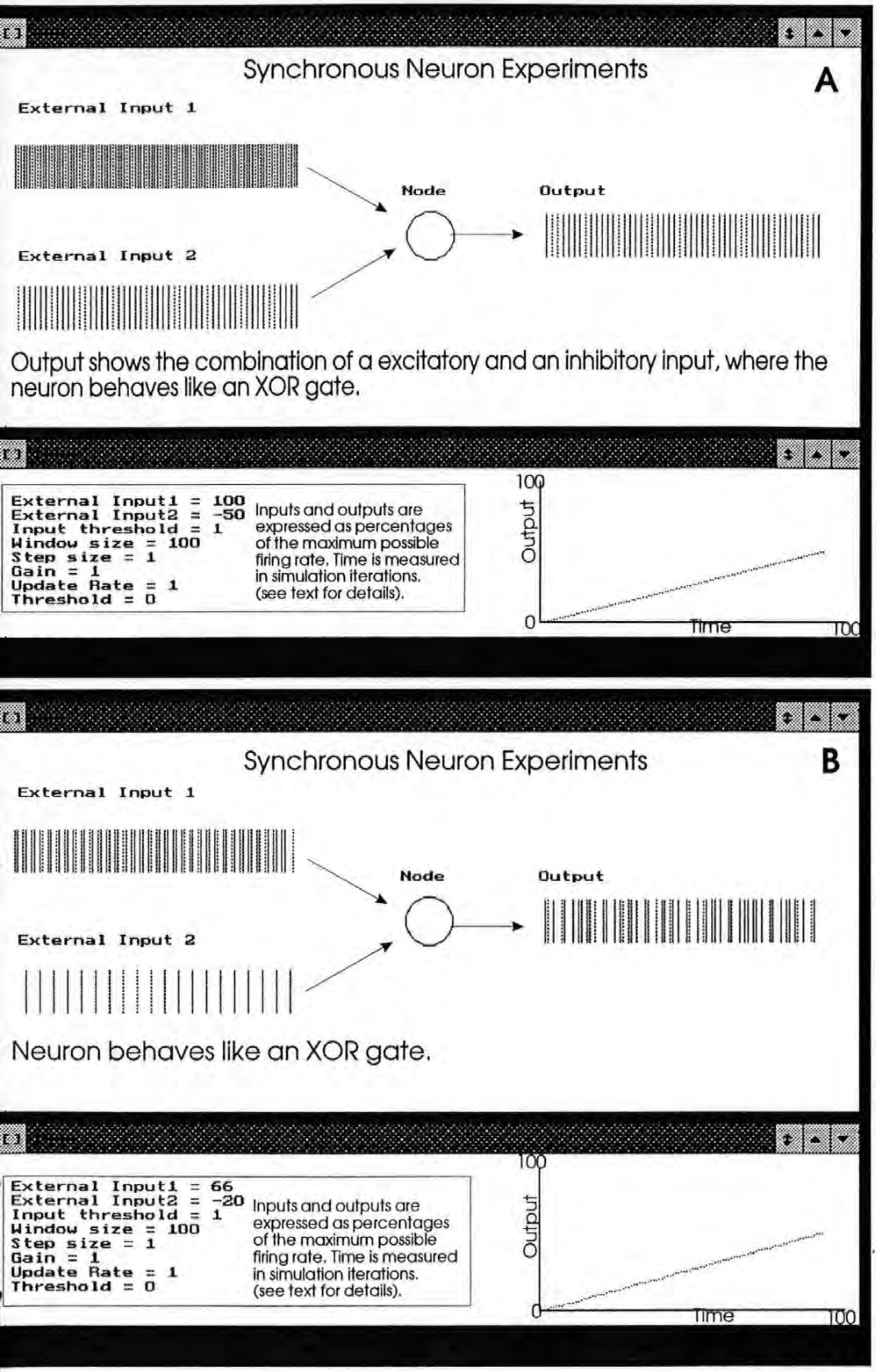
Synchronous Neuron Experiments

Experiment depicts neuron behaving like an AND gate, where the output correlates with the coincident incoming spikes.

```
External Input1 = 33
External Input2 = 50
Input threshold = 2
Window size = 100
Step size = 1
Gain = 1
Update Rate = 1
Threshold = 0
```

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

Figure 4.3



Output shows the combination of a excitatory and an inhibitory input, where the neuron behaves like an XOR gate.

```
External Input1 = 100
External Input2 = -50
Input threshold = 1
Window size = 100
Step size = 1
Gain = 1
Update Rate = 1
Threshold = 0
```

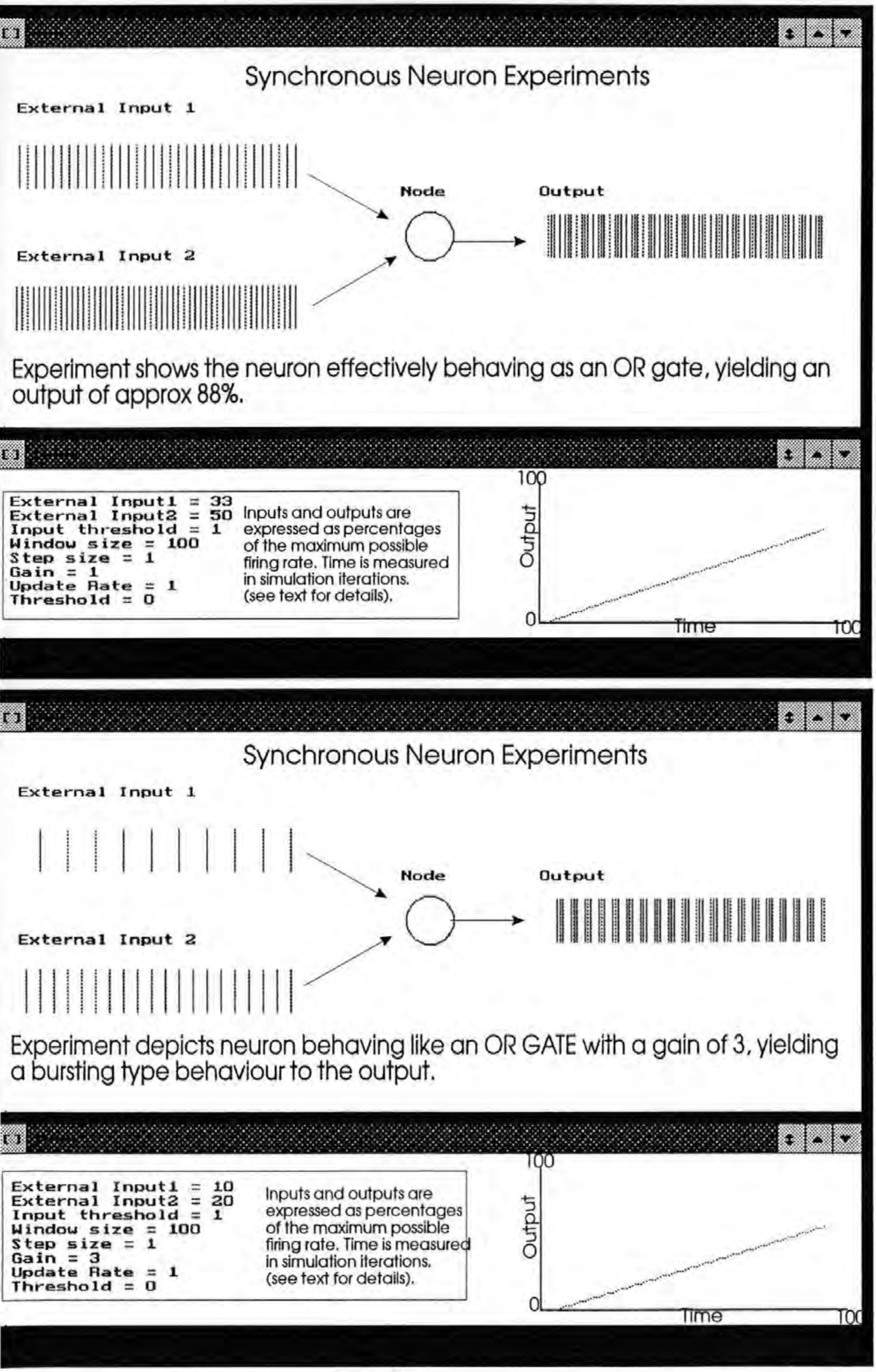
Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

Neuron behaves like an XOR gate.

```
External Input1 = 66
External Input2 = -20
Input threshold = 1
Window size = 100
Step size = 1
Gain = 1
Update Rate = 1
Threshold = 0
```

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

Figure 4.4



Experiment shows the neuron effectively behaving as an OR gate, yielding an output of approx 88%.

```
External Input1 = 33
External Input2 = 50
Input threshold = 1
Window size = 100
Step size = 1
Gain = 1
Update Rate = 1
Threshold = 0
```

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

Experiment depicts neuron behaving like an OR GATE with a gain of 3, yielding a bursting type behaviour to the output.

```
External Input1 = 10
External Input2 = 20
Input threshold = 1
Window size = 100
Step size = 1
Gain = 3
Update Rate = 1
Threshold = 0
```

Inputs and outputs are expressed as percentages of the maximum possible firing rate. Time is measured in simulation iterations. (see text for details).

Figure 4.5

4.6 Summary

Both the Leaky Integrator and Synchronous neurons fire in response to excitatory current. The current induced in a motorneuron by a connection from another neuron can be calculated as the firing rate of the neuron multiplied by the number of synapses and the charge transferred to the soma from a synapse due to the arrival of an impulse (Redman, 1976). In the Synchronous model the averaged spiking between the neurons is modelled explcitly, whereas the Leaky Integrator models current transfer between neurons.

The relationship between excitatory current and motorneuron firing is described by two linear curves (Kernell, 1983, 1984). The primary range is the range in which a motorneuron can maintain repetitive firing. The secondary range describes higher rates which can not be maintained under normal conditions. The simulations in the proceeding chapters only model the primary range.

The simulations in Chapter 5 are designed to explore the behaviour of both the neuronal models described here when arranged into varying network topologies to explore motor pool physiology. The resultant group behaviour of these networks is assessed in the context observations reported in Chapter 2 and Chapter 3.

5. Motor pool simulations.

5. 1 Introduction

In the proceeding chapters in this study a high level overview of the human motor control system has been described, and a general review of motor control issues and concepts discussed in chapter 2. In this chapter the focus changes, and a much more detailed study is undertaken on an area of human neurophysiology known as the motor pools. The theoretical background for this topic area was given in section 2.5 and in detail in section 3.4. The motor pools are situated in the brain stem and spinal cord grey matter. They are in an excellent position to mediate cortical control signals descending from the motor cortex. Motor pool neurons output directly onto muscle fibres. Due to the position and possible function of motor pools the areas are also known as the motor output stage (Hultborn and Illert 1991), and it is motor pool functionality in this role that is examined.

A large number of simulations were undertaken using both Leaky Integrator and Synchronous neuronal models (see chapter 4) using the INSIGHT TOO neuronal modelling software.

The analysis of motor pool behaviour undertaken in this chapter focuses on the computational features of motor neuron behaviour. The simulations make no attempt to be exact biological models of motor neuron behaviour, instead a focus is maintained on computational functionality, production of generally observed physiological properties (section 2.4) and the process of recruitment (Hennemann, 1957).

5.2 Simulation Software

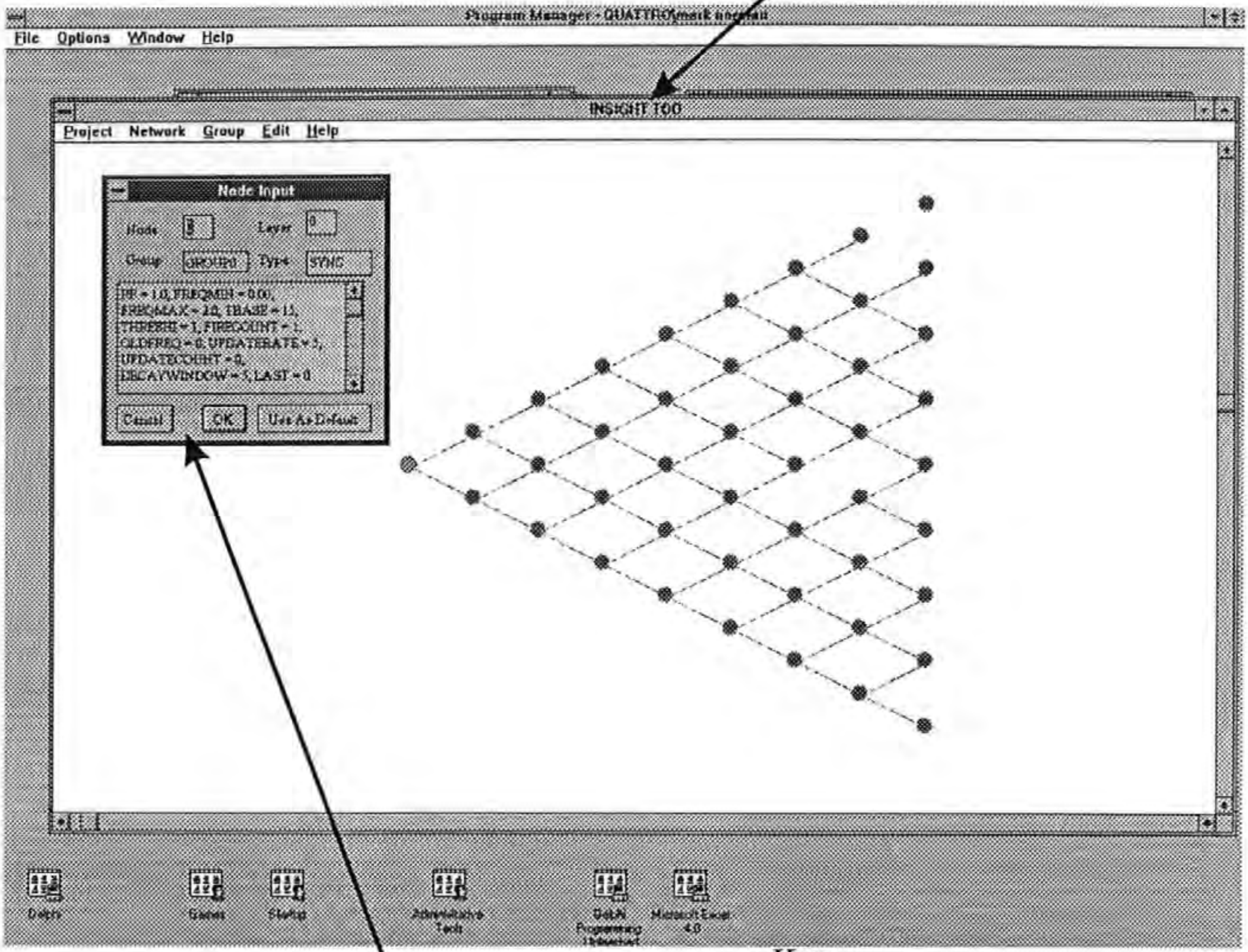
Many of the experiments undertaken in the early part of this study were simulated using the INSIGHT TOO modelling and simulation software. The INSIGHT TOO simulation software was developed as part of this study. The INSIGHT software (developed as part of an earlier study (Norman, 1991)) was suitable for the visualisation and simulation of small single layer neural networks. The original INSIGHT software was also only able to model Leaky Integrator neurons (Chapter 4). The INSIGHT TOO software removes these limitations by:-

- using the simple INSIGHT language for the description of neuronal model characteristics;
- a simple windows interface allows the visualisation of multi-layer large scale networks;
- node and connection definitions are "soft" and are determined at run-time, allowing for networks containing dynamic multi-model networks. This means that in theory a neuron could utilise different simulation code part way through a simulation;
- the INSIGHT TOO software has been developed using the Windows 3.1 operating system allowing INSIGHT simulations to be performed as background processing tasks.

Figure 5.1 shows the INSIGHT TOO simulation software running under Windows 3.1. The screen shot shown in Figure 5.1 is of a single layer neural network. The circles represent nodes and the lines that join them represent the interconnectivity between them. The colouring of the lines indicates the direction

INSIGHT TOO Simulation Software

Insight TOO - Generic neuronal simulation software tool. Allows the visual design and editing of large scale mixed neuron multi-layer networks.



Node 8 Layer 0

Group GROUP0 Type SYNC

PF = 1.0, FREQMIN = 0.00,
FREQMAX = 1.0, TBASE = 15,
THRESH = 1, FIRECOUNT = 1,
OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0,
DECAYWINDOW = 5, LAST = 0

Free format neuron properties dialog allows any neuronal model parameters to be defined.

Key

● Node

◍ Selected Node

┈ Axonic Connection

— Dendritic Connection

Insight TOO software requires an IBM compatible 486dx33 with 4mb RAM and Windows 3.1 minimum to run.

Figure 5.1

of the connection (see the key in Figure 5.1). A green(dotted)/red(solid) connection indicates an axonic connection which outputs the result of that node's innervation. A red/green connection indicates an dendritic input connection.

Each node and each connection is described by a series of parameters which must be defined. A starting value for each parameter is defined using the properties dialog box (see Figure 5.3). The parameters available for the definition of nodes and connections is determined by the node or connection type. The dialog performs no validation on these parameters; how and when these parameters are used is dependant upon the INSIGHT control program being used. Figure 5.2 shows the INSIGHT TOO software design cycle. The INSIGHT TOO windows software is used to design and manage topographical relationships(connections) between nodes. A text editor is used to enter the program control files written using the INSIGHT software language. INSIGHT is a simple instruction language similar to BASIC (Beginners All Purpose Symbolic Instruction Code). Table 5.1 contains a list of INSIGHT commands that are supported by the INSCOM interpreter.

The INSIGHT language allows network data files and INSIGHT program files to be combined in an arbitrary way at run-time. This is a flexible process that allows the simulation of multi-layer, multi-node networks in almost any combination. Examples of INSIGHT programs can be found in appendix A.

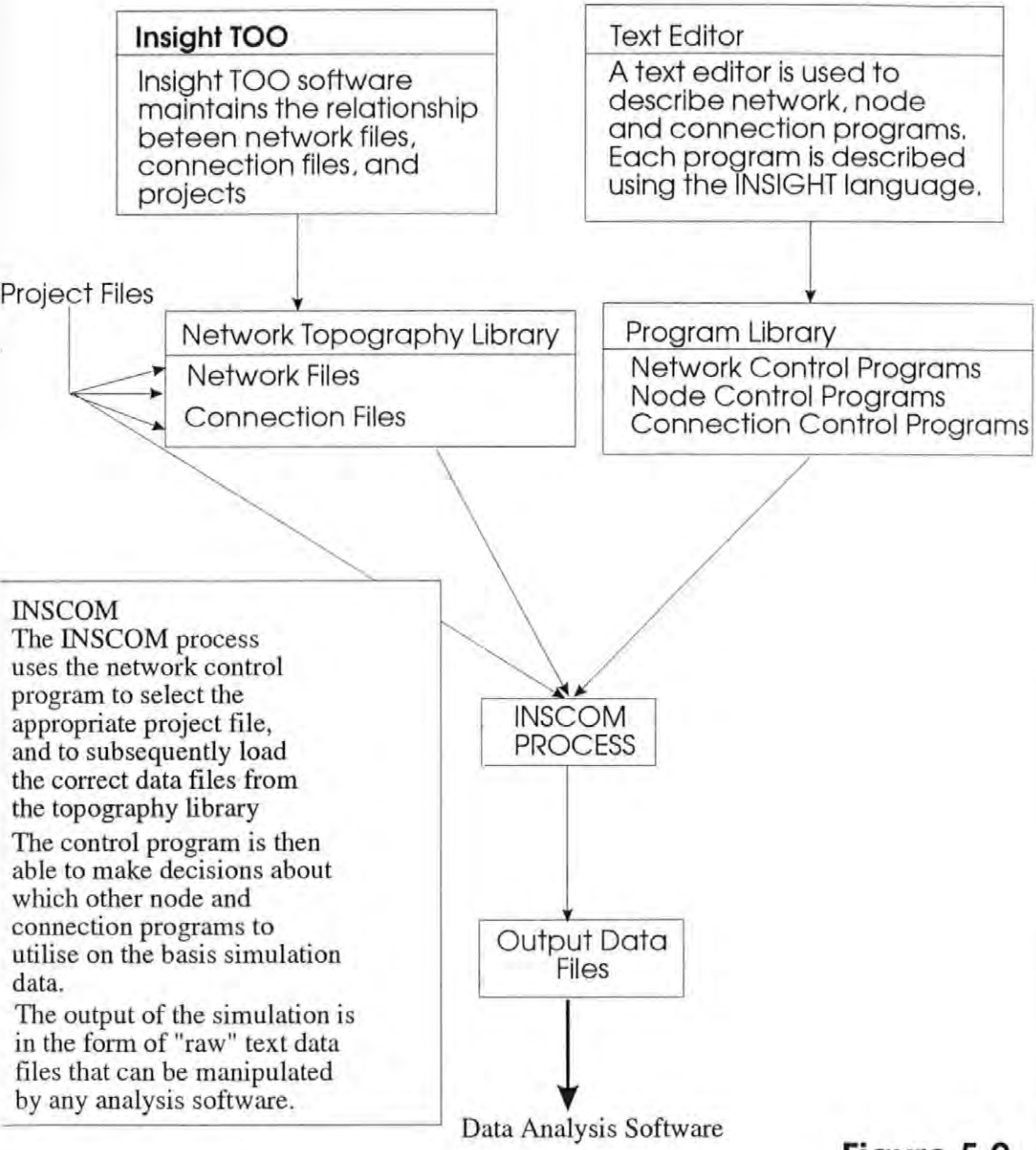
For all the simulations in this study the output is in the form of "raw" text data which can be examined using a variety of analysis tools, although the INSIGHT language is capable of providing a simple visual output also.

All the experiments described in this chapter using the INSIGHT TOO software have been analysed use the EXCEL 4.0 spreadsheet and charting software. Appendix B gives a list of EXCEL data files that apply to each of the simulations in this chapter as well as the exact experimental details.

INSIGHT COMMAND	DESCRIPTION
OPEN(<Filename>, <FileMode>,<Stream>)	Opens the specified file using the mode and stream specified. Returns a handle to the opened stream.
INITPROJECT(<Stream>)	Resets all node and connection parameters to their start values.
GOTOXY(<X>,<Y>)	Move the "cursor" to the x,y co-ordinate specified.
PRINT <String>	Outputs the specified string at the cursor position specified.
WRITE <Stream>, <Data>	Outputs the comma separated <data> to the specified stream.
FOR <var> = <startvalue> TO <endvalue>	Executes the code between the FOR and NEXT for the number of iterations indicated by the <startvalue> and <endvalue>
NEXT	Terminates the FOR..NEXT construct
REM	Source code annotation
CLOSE(<stream>)	Closes the file stream.
LOADNEURON(<Stream>,<NodeNum>)	Loads the specified <nodenum> from the open project stream <Stream>.
SAVENEURON(<Stream>,<NodeNum>)	Saves the specified <nodenum> to the open project stream <Stream> saving any associated state information.
GOTO <Label>	Move the execution pointer to the specified label.
GOSUB <Label>	Pushes the current execution point onto the stack and moves execution to the specified label. Allows execution to return using the RETURN command.
RETURN	Returns to a previously specified execution point specified by a matching GOSUB command.
IF <expr1> THEN <expr2>	If <expr1> evaluates to TRUE then expr2 is executed.
LOADCONNECT(<Stream>,<ConNum>)	Loads the specified connection from the open project stream <stream>
SAVECONNECT(<Stream>,<ConNum>)	Saves the specified connection to the open project stream <Stream>
CHAIN <FileName>	Loads and executes the specified INSIGHT program file.

TABLE 5.1 INSIGHT commands

INSIGHT TOO
Simulation Software Design Cycle



Insight TOO software maintains the relationship beteen network files, connection files, and projects

A text editor is used to describe network, node and connection programs. Each program is described using the INSIGHT language.

INSCOM
The INSCOM process uses the network control program to select the appropriate project file, and to subsequently load the correct data files from the topography library

The control program is then able to make decisions about which other node and connection programs to utilise on the basis simulation data.

The output of the simulation is in the form of "raw" text data files that can be manipulated by any analysis software.

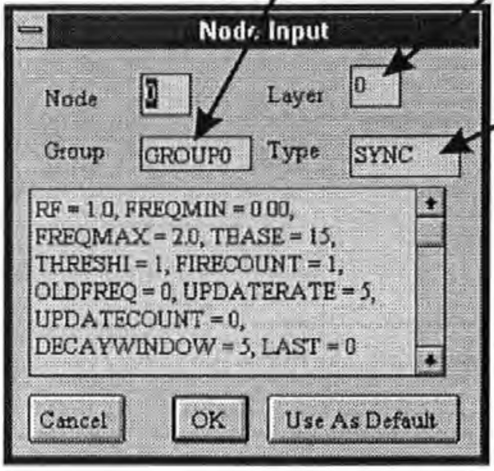
Figure 5.2

INSIGHT TOO - Simulation Parameters

Allows neurons to be functionally grouped (not used in these experiments).

Indicates the "layer" in multi-layer network projects.

Determines the control file loaded at run-time.

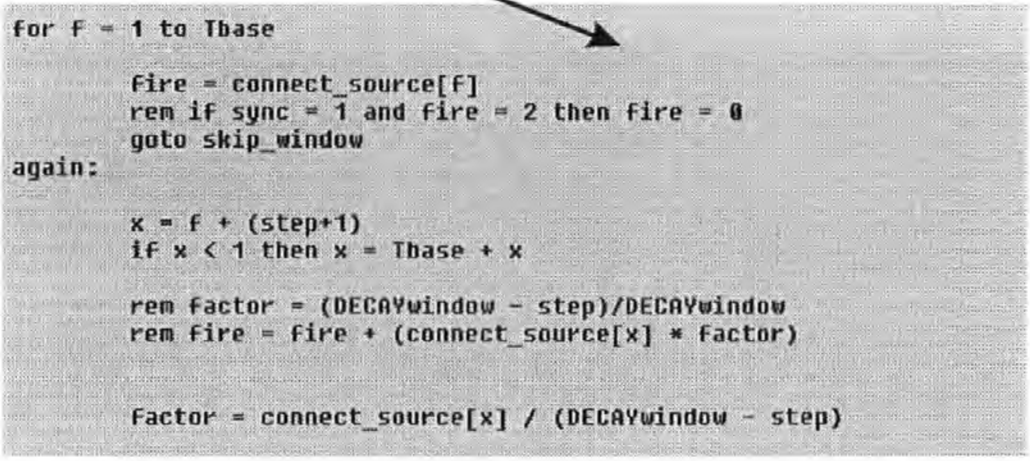


Node 0
Layer 0
Group GROUP0
Type SYNC

RF = 1.0, FREQMIN = 0.00,
FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1,
OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0,
DECAYWINDOW = 5, LAST = 0

Simulation Parameters	
RF = 1.0 FreqMin = 0.00 FreqMax = 0.00 Threshl = 1 UpdateRate = 5	Node Parameters
DecayWindow = 5 TBase = 15	Simulation Constants
FireCount = 1 OldFreq = 0 UpdateCount = 0	Simulation State Variables

INSIGHT program code excerpt from SYNC2.INS



```
for f = 1 to Tbase

        fire = connect_source[f]
        rem if sync = 1 and fire = 2 then fire = 0
        goto skip_window
again:

        x = f + (step+1)
        if x < 1 then x = Tbase + x

        rem factor = (DECAYwindow - step)/DECAYwindow
        rem fire = fire + (connect_source[x] * factor)


        Factor = connect_source[x] / (DECAYwindow - step)
```

Figure 5.3

5.3 Leaky Integrator motor pool simulations

The initial aim for the Leaky Integrator experiments was to build a network capable of taking a single input signal and then produce a spatiotemporal output sequence, that when plotted produces a profile that is approximately bell-shaped. Figure 5.4 depicts a network with similar topographic connectivity to those used in the following experiments. After examining a number of topologies it was felt the pyramidal connectivity structure shown in Figure 5.4 had a number of properties that would be a positive influence in the production of smooth output profiles of this form. In all the "pyramidal" experiments a stimulus is presented at the head of the "neuron pool", and over time the output from all the neurons is summated to form a total "pool" output. The output measured is the number of neurons that have "fired" at any instant in time. Nichols & Houk, 1976 have shown that underlying stiffness regulation compensates for muscle irregularities, therefore the actual force output at the muscle (muscle tension) in this model is assumed to have a linear relationship with the number of nodes that have fired at any instant in time.

A key question of interest is whether a single input signal into a motor pool is sufficient to control the peak amplitude, velocity, force and output profile of a muscle. This question has major implications for the design of the higher levels of motor function. If a second or third "command" channel is required for each muscle then the complexity required to integrate/distribute this information will be correspondingly more complex.

Figure 5.5B shows the structure of a "two-tier" motor pool simulation. Input from higher cortical levels is targeted on the head node of a pyramidal type motor pool network. An external input of varying intensity was targetted on the head node of the network for the first 10 time steps of the simulation. Each successively deeper layer within the pyramidal network has a progressively higher "recruitment" threshold. As the initial cortical stimulus progresses through the network it elicits a response from each node in turn proportional to the input stimulus received, the current state of the node and the settings for that node. Each node within the pool is a Leaky Integrator neuron (see chapter 4) defined as described in appendix B. Output from each node in the first motor pool is sent directly to the muscle and also to the "head node" in the second pyramidal motor pool (by direct axonic connection). At a certain activation level the summation of the inputs from the first motor pool will cause the second motor pool to become active (or "recruited"). The second motor pool was designed to behave in the same way as the first. The output from the second motor pool is summed with the first to produce the total output shown in Figure 5.5A.

This network design tests a number of principles, the first is the process of recruitment. This network imposes a recruitment order based on the "size" of each node (its threshold), which is invariant recruitment, but it also imposes a recruitment order on the basis of connectivity, which is selective recruitment. The experiment embodies the principle the "harder you push the further you get". In this way successively higher stimuli evoke activity in more neurons producing the larger peaks and greater activity shown in Figure 5.5A. The connectivity in a pyramidal motor pool implies a temporal aspect, which can also be seen in Figure 5.5A in the way in which the peak outputs and total output duration are slightly offset for successively higher input stimuli. The pyramidal topology seemed to be a positive influence in both the temporal and spatial form of the output profile.

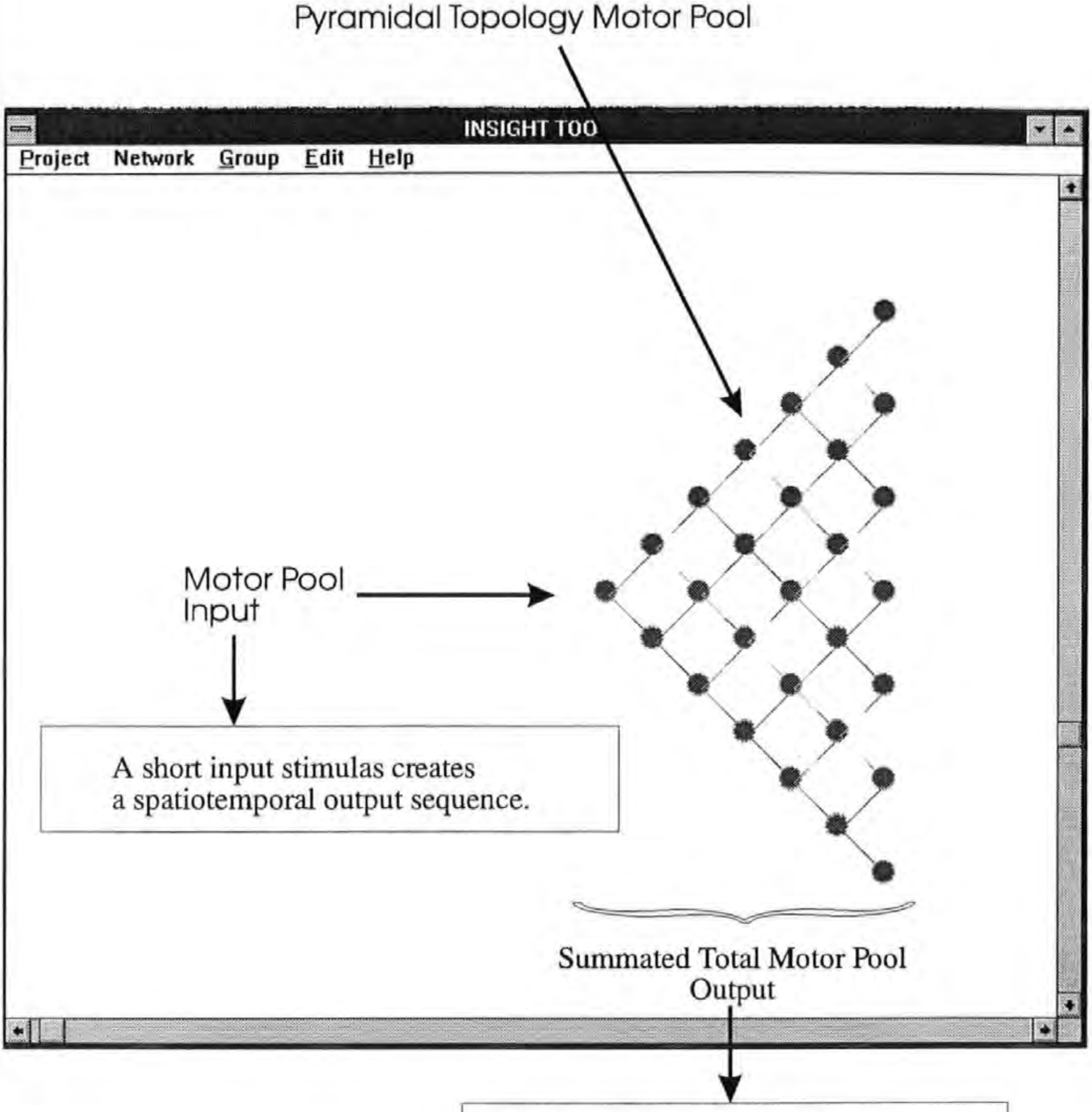
There are three categories of motor unit that are recognised in human mammalian muscle (Zajac and Faden, 1985; see section 3.4.2). The two tier motor pool experiment i.e. using two motor pools, as described above in Figure 5.5B is an unconventional attempt to examine the boundary between two of these three types. It is unconventional because recruitment is normally described on the basis of invariant recruitment only, whereas this experiment demonstrated invariant and selective recruitment.

The second motor pool output was scaled to be twice that of the first, i.e. each output pulse from the second pool counted as two output force units. The decay time constant of each of the second pool motor neurons was set such that the membrane potential decayed at a much higher rate representing the relationship between output force and fatigue shown in Table 3.2. The limitations of computer simulation speed prevented all three types being represented in this way.

Although the two types of neuron are represented in an overt manner i.e. by a second pool, the thresholds within each pool layer represents the progression between the two types in a graduated fashion.

How much better is the two-tier model over a single tier model? The two-tier motor pool was better than the single-tier motor pool (Figure 5.6A), in that the bell-profile had marginally better definition. A factor of greater significance is that the amplitude peaks for all the simulations was not correlated with the external input stimulus (Figure 5.6A - External Input, EI). A clear difference in peak output for different input stimuli is the minimum requirement for the application of these networks in a control model. One of the reasons the peak output profile is not correlated with the input signal is that the firing activity of each neuron is not tightly correlated with the "head node" input, the incoming stimulus becomes effectively translated into an activation duration rather than activity intensity i.e. the individual firing frequency of each node is not directly related to the firing frequency of the "head node". As a result of this it was concluded that the Leaky Integrator neuronal model was not well suited to the motor pool task.

Motor Pool Simulations Using INSIGHT TOO Software

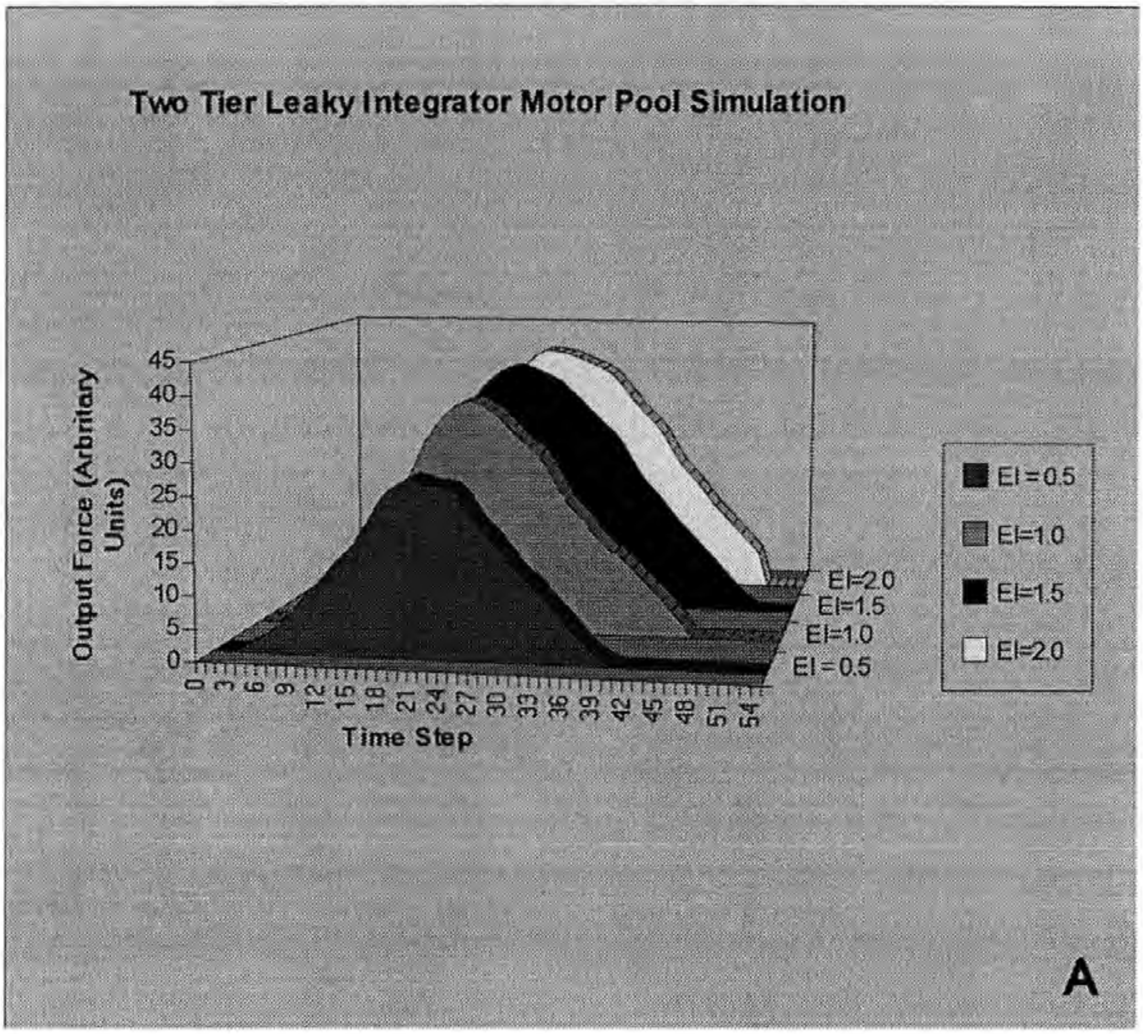


A short input stimulas creates a spatiotemporal output sequence.

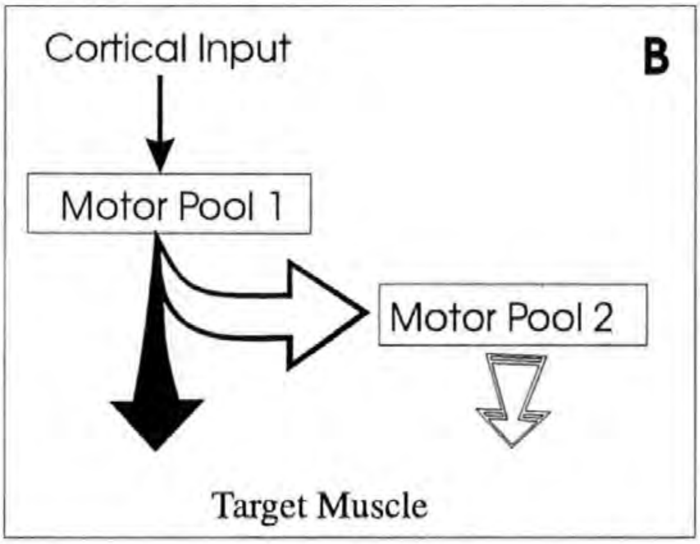
At each instant in time the total motor pool output is the number of neurons that have fired at that time step.

Figure 5.4

Leaky Integrator "two-tier" motor pool simulations.



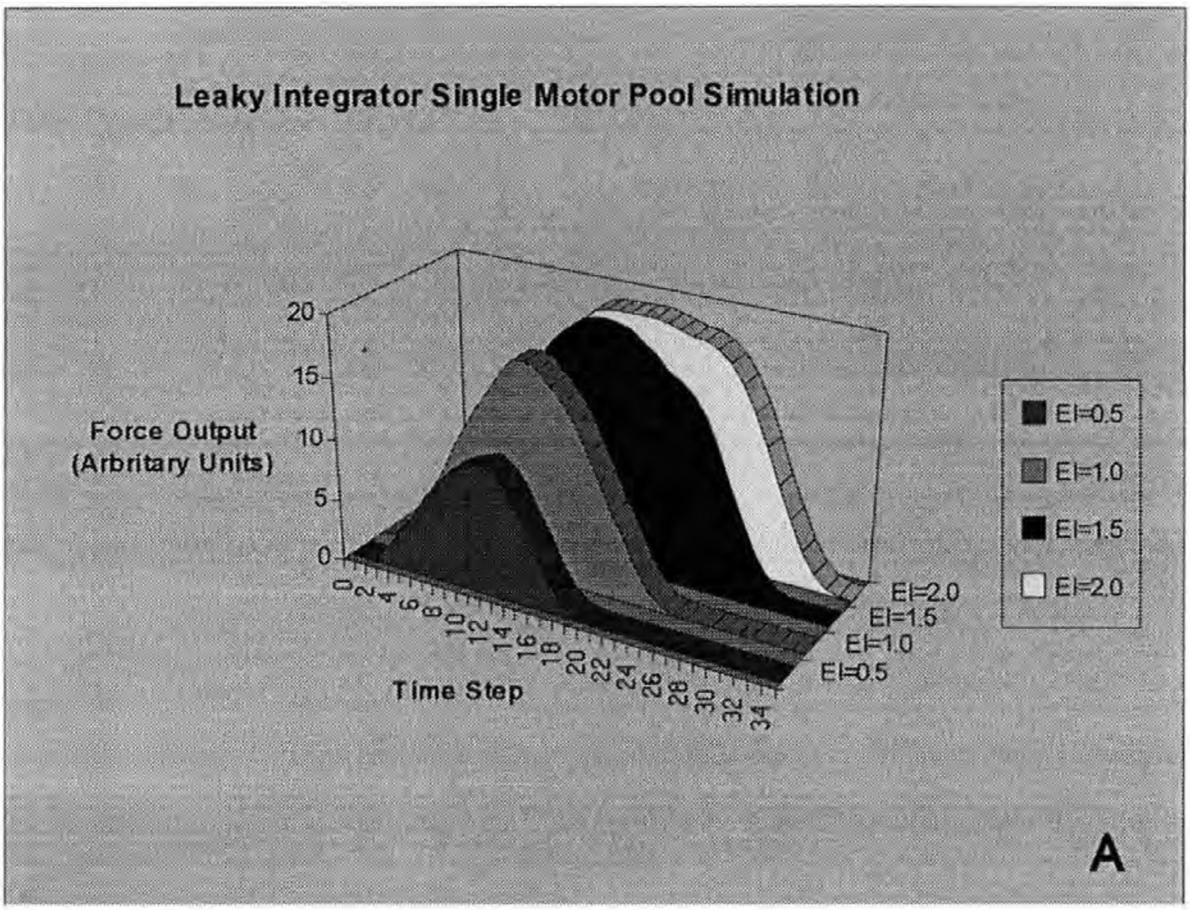
The figure above represent a series of experiments conducted on a "two-tier" motor pool (see text). EI represent the external stimulus presented during each simulation for 10 time steps. The output at each time step is plotted in arbitrary units.



Output from the first motor pool targets onto the "head" node of the second motor pool. The total muscle output force is the sum activity of both motor pools.

Figure 5.5

Leaky Integrator single tier motor pool simulations.



The figure above represent a series of experiments conducted on a "single-tier" motor pool (see text). EI represent the external stimulus presented during each simulation for 10 time steps. The output at each time step is plotted in arbritary units.

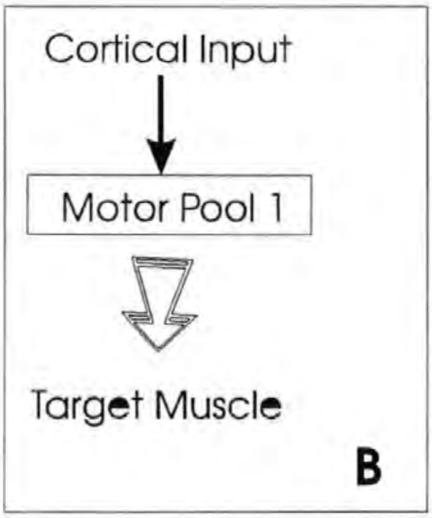


Figure 5.6

5.4 Synchronous motor pool simulations

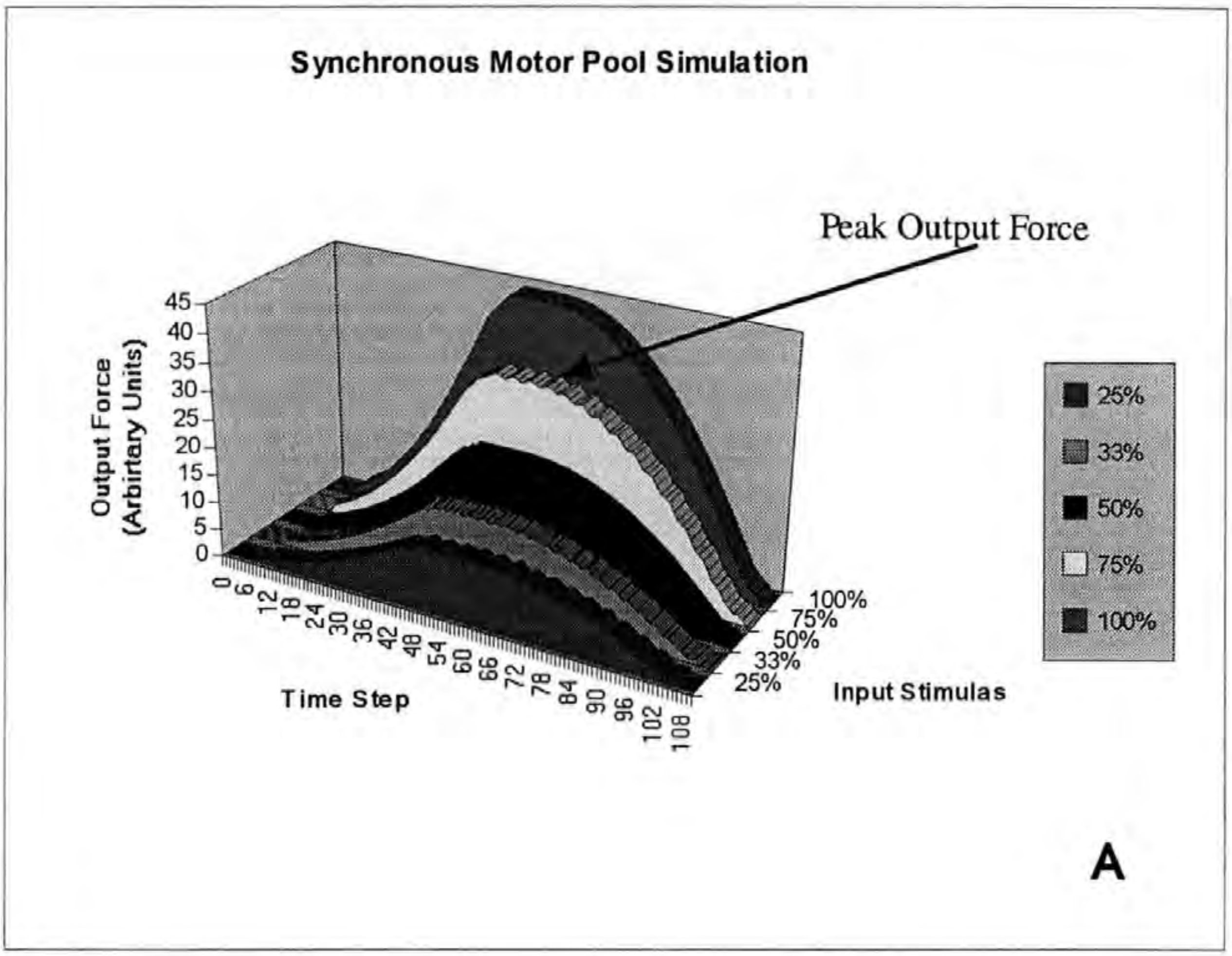
The Synchronous neuronal model directly relates output firing activity with the average input frequency, whereas the Leaky Integrator neuronal model is more focused around modelling the membrane potential of the neuron soma, from which the firing behaviour is determined in a relatively simplistic manner. The details of the Synchronous model are described in chapter 4.

Figure 5.7 shows the result from the first set of Synchronous motor pool simulations. The motor pool connectivity and topography is as described before and can be seen in Figure 5.7C (detailed information given in Appendix B). Figure 5.7B shows that the simulation was based on a single tier motor pool design. The experiments were once again designed around arbitrary simulation parameters and are not specifically related to any real muscle model.

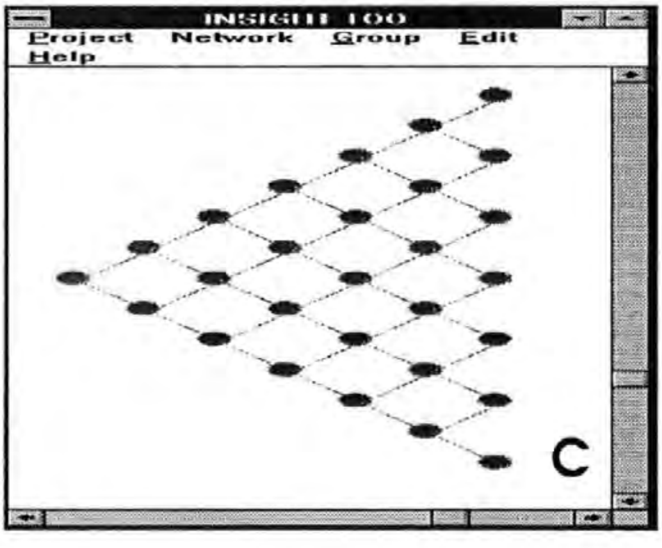
The presentation of the input stimulus was different to that of the Leaky Integrator simulations, Figure 5.8B shows the difference between the step input (Figure 5.8B Top) used during the leaky simulations and the frequency based input spike trains (Figure 5.8B Bottom) used in the Synchronous model. The charts shown in Figure 5.7A and Figure 5.8A were produced by presenting the higher input stimulus for the first 50 iterations, which is approximately half the total simulation time. The down side of the bell-shaped profile therefore represents the natural decay of the network (Figure 5.8A is therefore comparable with Figure 1.1B).

In the Leaky Integrator experiments the input stimulus was presented to the head node in the network for only 10 time steps. The Synchronous network stimulations require a continuous presentation of an input stimulus unless a full

Synchronous neuron single tier motor pool simulations.



Output force is the measured as the number of active neurons at each time step. Output force is linearly related to muscle force and is shown here in arbitrary units. The input stimulus is presented as a percentage of the maximum possible firing



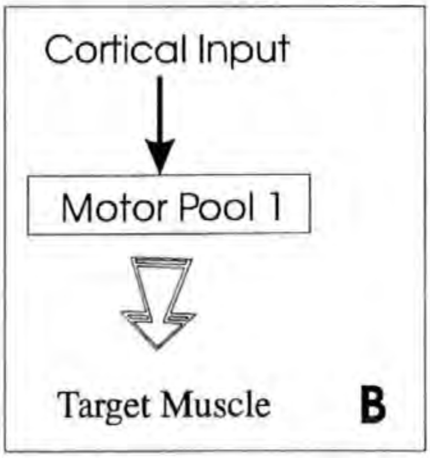
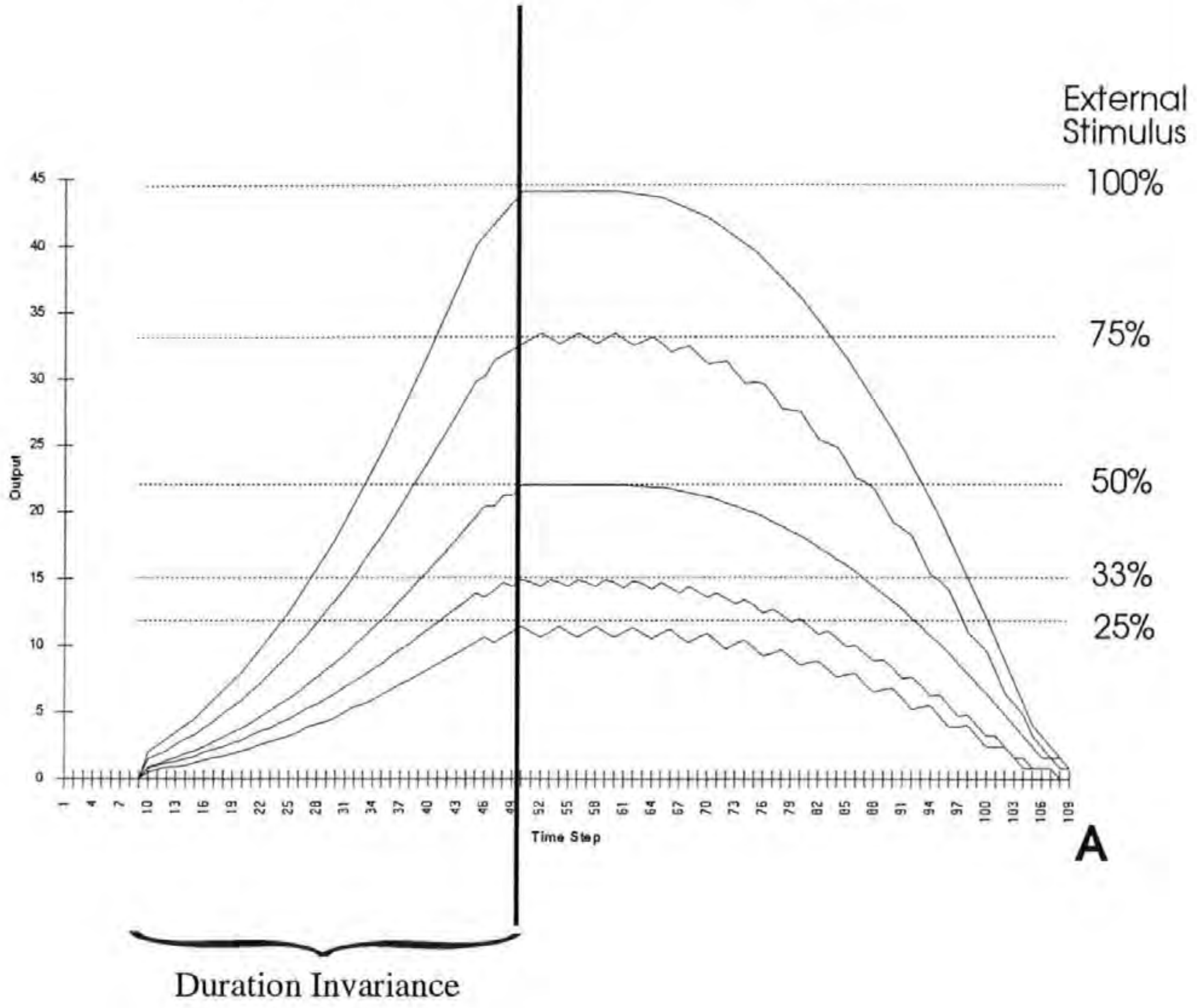


Figure 5.7



Synchronous neuron single tier motor pool simulation

The graph above charts force over time for a Synchronous motor pool for a variety of input stimuli. A stimulus was presented and maintained for 50 iterations. The activity in the motor pool was then allowed to decay naturally.

The output results show that the motor pool exhibits good duration invariance for a number of targetted force levels, and a strong relationship between peak output force and the input stimulus.

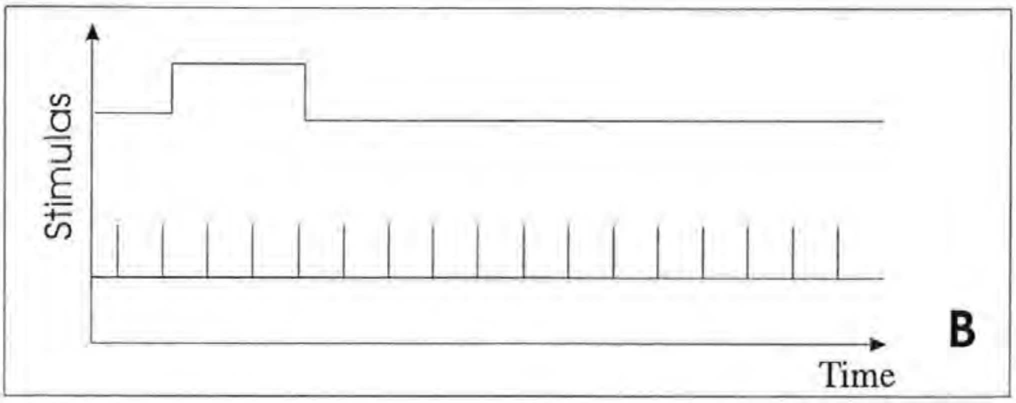


Figure 5.8

resting state is required. The Synchronous network is therefore more responsive to successive varying stimuli and requires a continuous input stimulus. This is in fact a significant benefit as the network is able to react to changes in input in a dynamic fashion (this feature is explored in more detail later in this chapter).

From Figure 5.8 it can be more clearly seen that the duration from onset to peak force is equivalent for a variety of input stimuli, this network is therefore capable of exhibiting the physiologically observed property of duration invariance observed in human movements (see section 2.4).

From Figure 5.8 it can also be seen that there is a linear relationship between the input stimulus and the resultant elicited peak force, an essential requirement for controlling the amplitude of reaching movements.

The duration invariance and the peak profile relationship indicate a tight temporal relationship in this network. By careful manipulation of the timing of input sequences it is possible to control amplitude, speed and force of reaching movements using a single input signal. The coding of the input stimulus, not just its average value but the spatial relationship between the spikes in the input train is therefore of paramount importance. In these experiments the input stimulus was proportionally spaced (i.e. the spike train form was uniform) and not in the form of a single burst of activity (which is the equivalent form of the Leaky Integrator external input). The modelling of spiking in the Synchronous neuronal model is a key feature and is referred to in subsequent experiments described in chapter 6.

These results were produced with a single layer motor pool of only 15 nodes. Each successive layer within the network had progressively higher minimum frequency thresholds, therefore utilising the "harder you push the further you get" paradigm mentioned earlier. Each node also fires at a rate proportional to the input. This relates more directly to motor pool properties observed by Burke (1991) in human motor pools, than the previous simpler Leaky Integrator simulations. The output shown was produced with a single pass in-line exponential smoothing function. This contrasts with the Leaky Integrator experiments which required several smoothing operations to produce the profiles shown (see Appendix B for details).

5.4.1 Motor Unit Simulations

The preceding experiments in this chapter were simulated using arbitrary parameter values, instead of biologically reported ones. Furthermore, the force output on the muscle from each node was constant across the pool. This simplification meant that the functional grouping between a neuron and the muscle fibres it innervates (i.e. a motor-unit) were ignored. In addition to this the rest periods for each neuron necessary to prevent muscle fatigue were not modelled either.

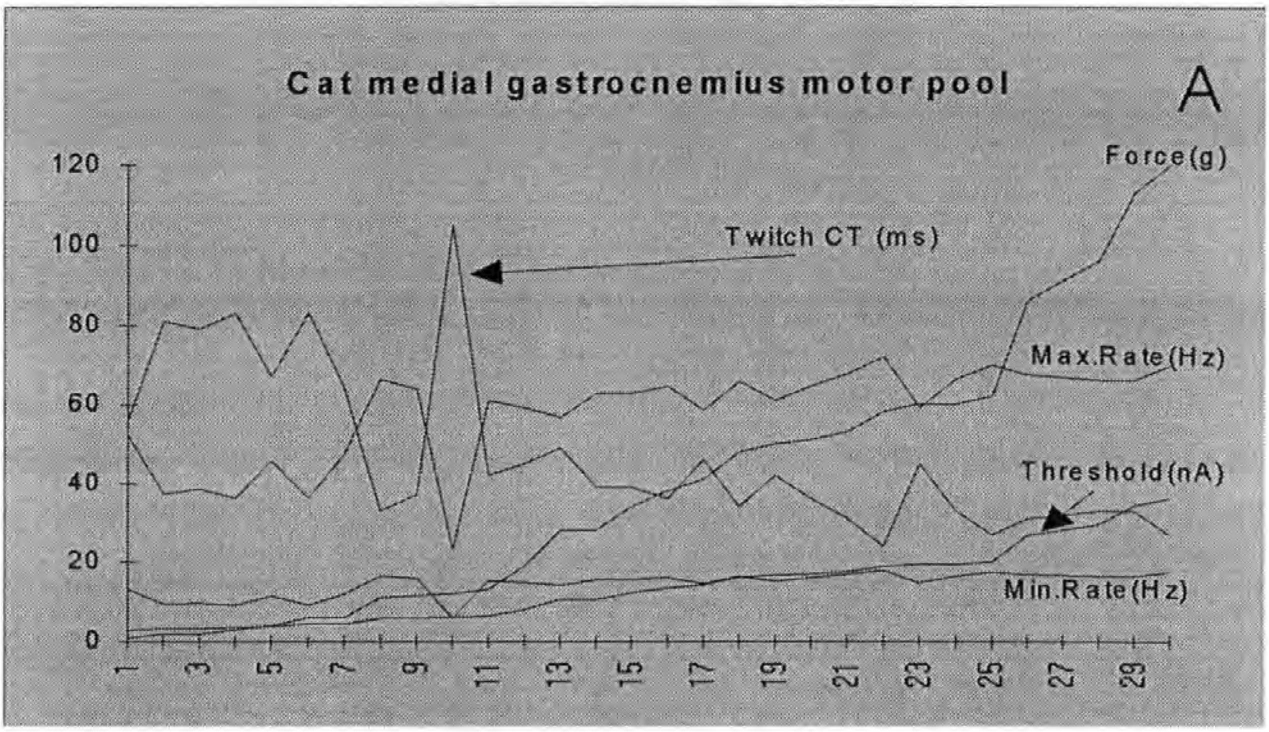
The following simulations are designed to add in these additional properties and specifically to reproduce the observed property of velocity profile invariance. These simulations will also differ from the previous ones because they will be based solely upon invariant recruitment, ignoring selective recruitment. This means that the external input stimulus to the pool is presented to all the nodes in the pool directly. There are also no connections between the neurons within the pool. Each neuron within the motor pool is simulated with the Synchronous Neuronal model as described in chapter 4.

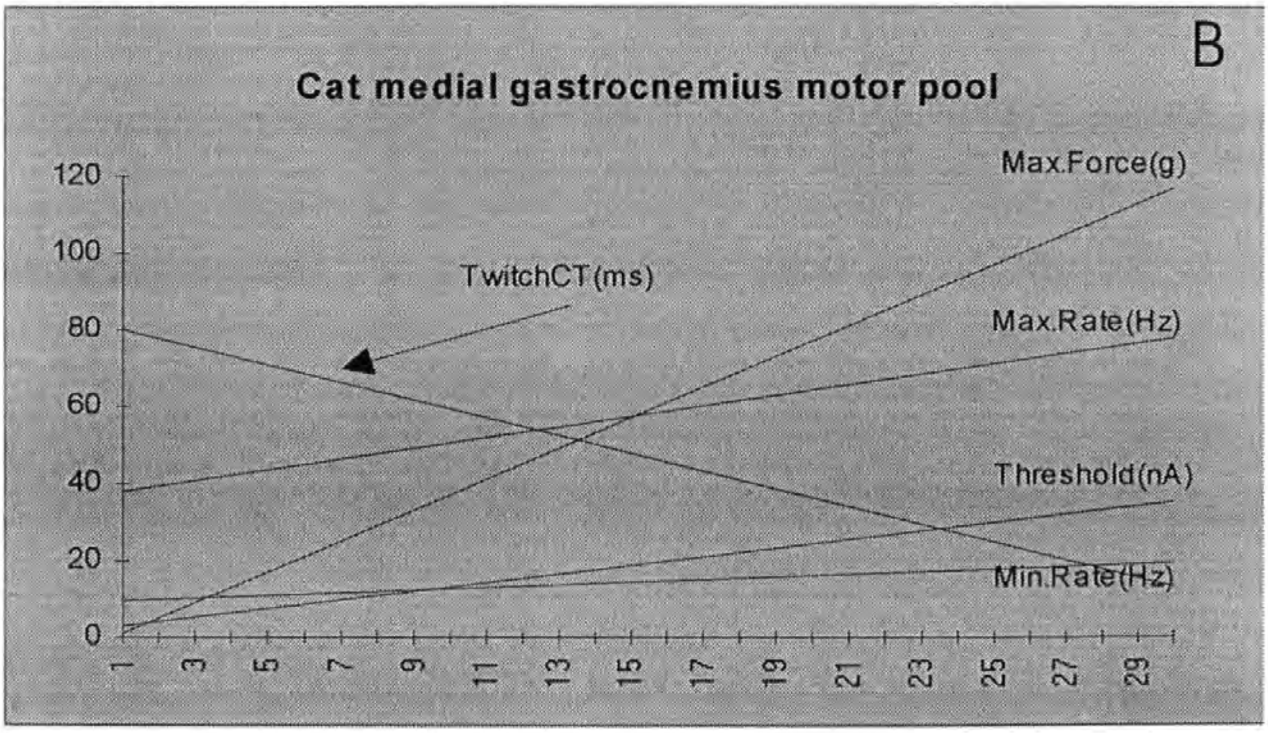
The simulations are based around the data reported by Burke (Burke, 1981) for the cat medial gastrocnemius (MG) muscle. A complete motor pool of a cat MG consists of around 300 motor neurons. Graham & Redman(1993) modelled the cat MG using 30 neurons with a distribution similar to that of an actual cat MG. The exact composition of a motor pool will be dependant upon the functional requirement of that muscle. For example a finger muscle requiring finely graded muscle function would be different to a leg muscle (such as the MG) requiring strong gross movement. Table 5.2 therefore lists a population of thirty neurons used during these simulation which have a linear property distribution. The relationship between a MG pool (that has a specific muscle function) and a similar pool but with a linear distribution of reported property ranges and relationships is shown in Figure 5.9.

Threshod (nA)	Min.Rate (Hz)	Max.Rate (Hz)	Max.Force (grammes)	TwitchCT (ms)
3.0	9.5	38.0	1	80.0
4.12	9.845	39.38	5	77.7
5.24	10.19	40.76	9	75.4
6.36	10.535	42.14	13	73.1
7.48	10.88	43.52	17	70.8
8.6	11.225	44.9	21	68.5
9.72	11.57	46.28	25	66.2
10.84	11.915	47.66	29	63.9
11.96	12.26	49.04	33	61.6
13.08	12.605	50.42	37	59.3
14.2	12.95	51.8	41	57.0
15.32	13.295	53.18	45	54.7
16.44	13.64	54.56	49	52.4
17.56	13.985	55.94	53	50.1
18.68	14.33	57.32	57	47.8
19.8	14.675	58.7	61	45.5
20.92	15.02	60.08	65	43.2
22.04	15.365	61.46	69	40.9
23.16	15.71	62.84	73	38.6
24.28	16.055	64.22	77	36.3
25.4	16.4	65.6	81	34.0
26.52	16.745	66.98	85	31.7
27.64	17.09	68.36	89	29.4
28.76	17.435	69.74	93	27.1
29.88	17.78	71.2	97	24.8
31	18.125	72.5	101	22.5
32.12	18.47	73.88	105	20.2
33.24	18.815	75.26	109	17.9
34.36	19.16	76.64	113	15.6
35.48	19.505	78.02	117	13.3

Table 5.2 Cat Medial Gastrocnemius (MG) motor pool

Cat medial gastrocnemius(MG) motor pool simulation parameters





The cat medial castrocnemius (MG) motor pool has approximately 300 neurons. Figure A shows a motor pool of 30 motor neurons with a similar distributive mix to that reported in the original (Burke, 1991).

Figure B shows a similar motor pool of 30 neurons but with a linear distribution of the parameter ranges reported.

The distribution of properties in Figure A is as reported in Graham and Redman (1993).

Figure 5.9

In the Synchronous model experiments the threshold shown in Table 5.2 relates to the soma threshold parameter(k_2) described in section 4.4. In these simulations the transfer of frequency into current is taken to be 1Na = 1Hz. Current thresholds for repetitive firing in alpha motorneurons typically range from 3-40Na (Kernell and Monster, 1981). There is a strong correlation between the soma threshold of a motor neuron and the maximum tentanic tension of its motor unit (Kernell and Monster, 1981). The range of tetanic tension reported in a cat MG muscle is 1-132g, Hence by linear regression the soma threshold can be calculated as follows:-

$$\text{Threshold(Na)} = 0.28 \times \text{tetanic force} + 2.72$$

The minimum and maximum firing rates of a motor neuron are strongly correlated with the twitch contraction times (Kernell, 1979), which range from about 10-110ms (Burke, 1981; Carew, 1981; Burke, 1990a). Using linear regression the minimum and maximum firing rates can be calculated as follows:-

$$\text{Min.firing rate(Hz)} = -0.15 \times \text{twitch contraction time} + 21.5$$

$$\text{Max.firing rate(Hz)} = -0.6 \times \text{twitch contraction time} + 86.0$$

Table 5.2 therefore has a linear distribution of twitch contraction times and maximum output force as already described and the other parameters are calculated from these.

Cat medial gastrocnemius(MG) motor pool simulation parameters

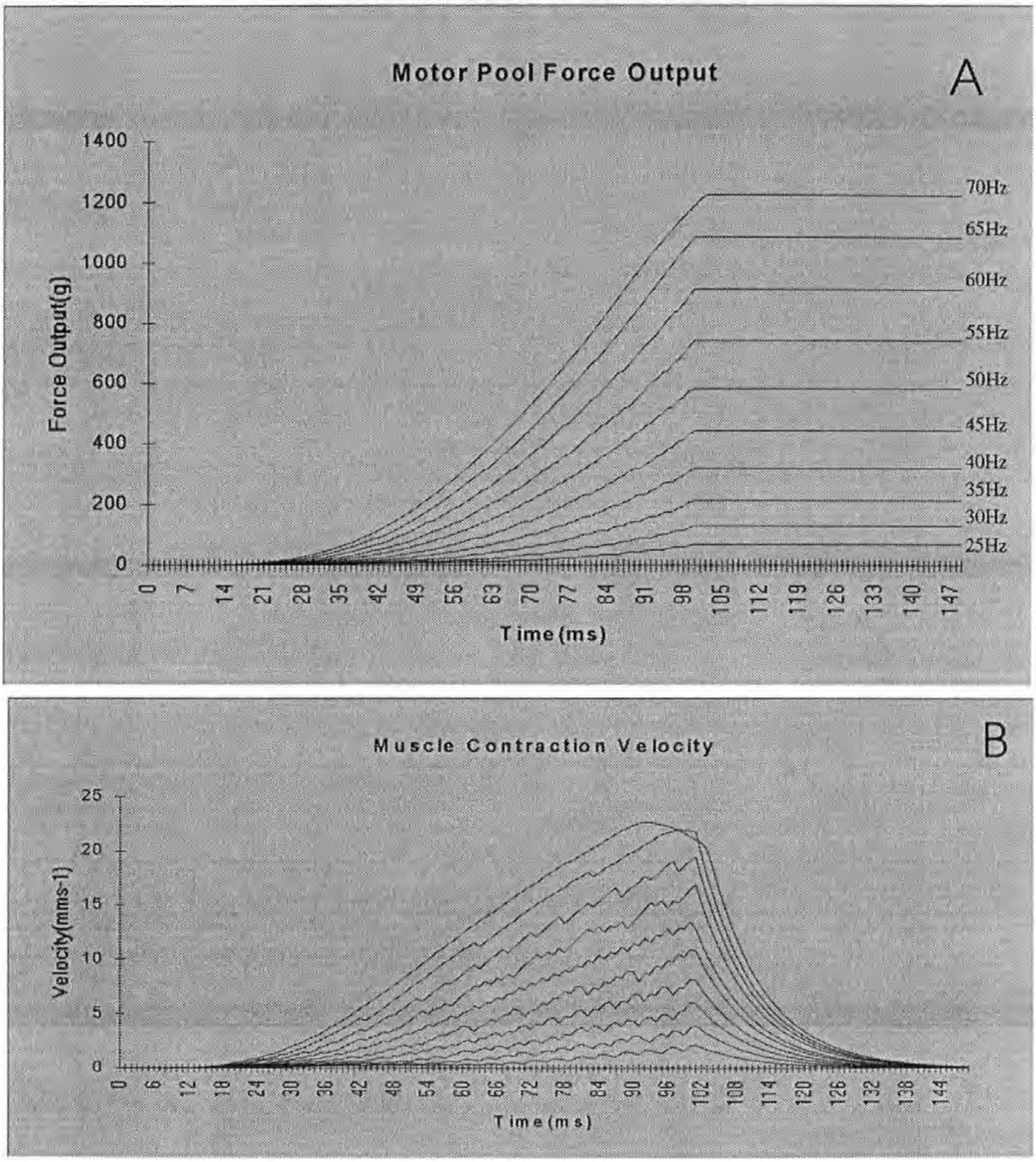


Figure A shows the force output from a motor pool with similar properties to that of a cat medial gastrocnemius motor pool.
The numbers to the right in Figure A indicate the input frequency to the motor pool.
Figure B depcits the resulting muscle contraction velocity in a simplified linear muscle model. The peaks correspond in the two graphs.

Figure 5.10

From Figure 5.10A it can be seen that the pool behaves in a similar way to the previous simulations shown in Figure 5.8a (to the left of the centre line). In Figure 5.10 it can be seen that the input stimulus was held for the duration of the simulation, whereas in Figure 5.8 it was allowed to decay back to a resting state. From Figure 5.10 it can also be seen that the output peaks are no longer proportionally linear to the input stimulus as in Figure 5.8. For example an input frequency of 35Hz produces an output of 200g, but an input of 70Hz produces an output of 1200g. Figure 5.10 shows that although the relationship is not as clear as in Figure 5.8 it could still be easily learnt by the higher motor control centers, because there is a clear peak separation.

By assuming that the target muscle has a simple linear relationship over the output range of the motor pool. The resultant muscle contraction velocity can be calculated. The plot of muscle contraction velocity can be seen in Figure 5.10B, where the velocity peaks proportionally correspond to the peak forces in Figure 5.10A.

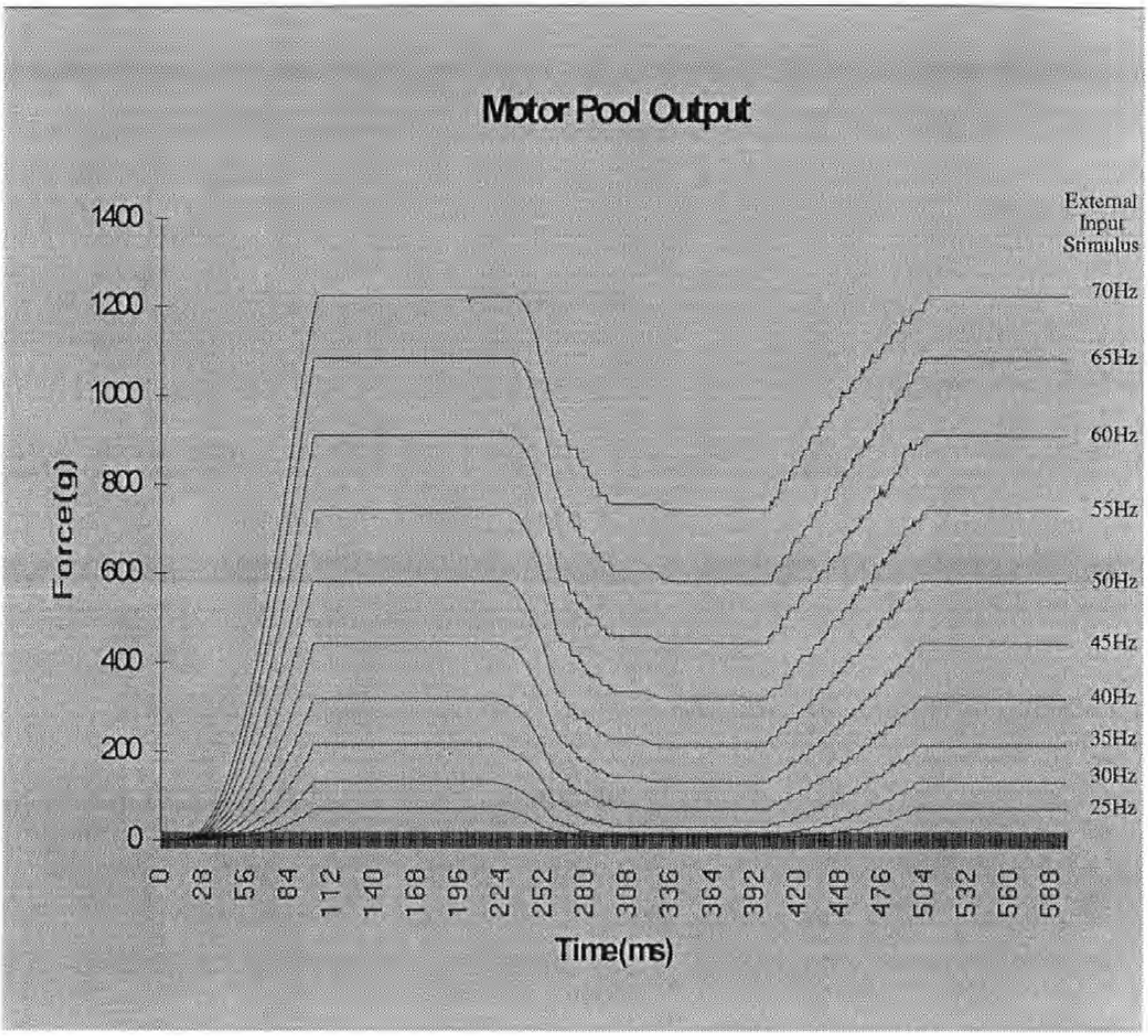
Figure 5.11 demonstrates how the network responds to successive input stimuli. At 200ms the input is dropped by 15Hz and is increased again at 400ms. The network was able to smoothly integrate between these points.

The properties in a Synchronous motor pool that can be controlled by single input signal can be summarised as follows:-

- clearly defined relationship between the input frequency and the elicited peak force output of the pool;
- smooth integration between successive input stimuli in real-time;

- duration invariance for varying targetted force levels;
- assymetric bell-shaped velocity profiles are observed for completed movements at varying targetted force levels, when a simple linear muscle model is assumed;
- transitions between successive stimuli occur within fixed periods of time defined by network parameters and connections. This temporal relationship could be utilised by other motor centers to simplify synergy formation and timing considerations.

Cat medial gastrocnemius(MG) motor pool



The figure shows the motor pool output for a given initial input. At 200ms the input drops by 15Hz, and 400ms it is increased again to the original level.
The simulations demonstrate that the network is able to integrate successive stimuli smoothly.

Figure 5.11

5.5 Conclusion

The experiments described in this chapter have shown that neurons arranged with properties similar to those observed in human motor pools are capable of producing commonly observed physiological properties. The results from these simulations have led to the hypothesis that the motor pools have important functional roles and should not be approximated by being modelled as a single neuron when bridging the gap between motor cortical output and muscles (Akazawa and Kata, 1990). Furthermore the Synchronous experiments have shown that the process of recruitment could also have an important temporal control aspect as well as an important spatial aspect. By utilising a connectivity and size based recruitment topology it has been shown that the speed, force and duration of the muscle response can be controlled through a single command signal. The Synchronous modelling experiments have shown that the spatial form or spiking of an input signal could be an important aspect for the control and timing of reaching movements, and is of particular relevance in considering synergy formation problems.

The process of "recruiting" nodes in a network on the basis of connections or parameter criterion is a computationally elegant solution for solving one-to-many mapping problems. It is also a paradigm that is suited to a distributed or parallel implementation, a rare event considering that almost all control models today are still described in sequential terms. The application of the "recruitment" process to a control model in general is discussed in more detail in chapter 6.

The simulations described in this study are not sufficiently detailed or accurate to claim biological plausibility, they are instead merely biologically inspired.

Considerably more experimentation would be required to apply these results to a full-scale control model. Many issues have been ignored, the most significant being the role of Renshaw Cells (RC) and in general the feedback and adaption of such networks. The neglect of these topics is in keeping with the focus of this study which is of the initial ballistic phase of motor action only. Graham and Redman (1993) noted the importance of RC cells in adapting the force output of a motor pool in handling dynamic loading effects, although Akazawa and Kato (1990) have pointed out that the orderly recruitment of motor neurons occurs in the absence of RC cells, which was a major factor in deciding to neglect the role of these cells in this study.

Wada and Kawato (1993) describe a model of trajectory formation using forward and inverse dynamic models based upon a minimum torque change criterion. Their results indicate that the production of profiles similar to those described above required between 30 to 60 smoothing operations and is a considerably more computationally expensive technique.

The motor pool studies have shown that these networks show promise for the reproduction of physiologically observed properties such as duration invariance and velocity profile invariance. There are still many questions left unanswered. The pools described are still not adequate in many areas. A more detailed model of muscle behaviour is a prerequisite for the production of more functional and accurate motor pool behaviour.

6. The Recruitment model of ballistic voluntary motor action

6.1 Introduction

The human skeletal frame is a complex mechanical device; it can move in almost any direction in any number of ways. Even when just the arm joints are considered it has many degrees of freedom along with redundancy i.e. it has more than one way of achieving a given reaching action. As a bone rotates about a joint various biomechanical interactions occur (such as angular acceleration, coriolis torques etc.) which cause further analytical and computational problems (Hollerbach J.M, Flash T. 1982). Bones are moved about joints by the contraction and de-contraction of muscle pairs (agonists and antagonists) which pull in opposite directions. The total summated net tension in a muscle pair causes a rotation in the joint about which they act. The direction of action of a single muscle pair with respect to the joint is often not the required direction of action. As a result another muscle pair is required to balance the action of the first and so on. This is a complex balancing process in which even for simple movements, the ease of use belies the complexity of interactions required to achieve a given movement. If a particular movement is required and the appropriate muscle pairs have been selected to achieve that movement, further calculations are needed to determine which pairs become active, by how much (i.e. tension level) and for how long. The human control process is therefore a complex timing problem which controls a myriad of spatiotemporal signals which in combination form an overall movement.

Given the complexity of these spatiotemporal signals, the intricacy of the timing process, and the failure of other artificial control processes to reach anything approaching the functionality of the human motor control system, the question:- how is this achieved? - becomes compelling.

The human motor system not only manages to produce the required calculations for each joint and every muscle, but manages to do this in real time. Furthermore, movements can be made extremely quickly, they can be pre-planned, learnt and reproduced with skill. The motor control systems therefore comprises of learning, prediction, planning and production sub-components.

6. 2 Overview of the recruitment model of motor control

The Recruitment model describes a mechanism that is capable of producing abilities similar to those of human motor system. The scope of this research is further reaching than many previous model proposals (see section 2.5) because it does not start from the premise that the device being controlled is a robotic arm. By rejecting the robot arm as the target device and assuming a mechanism more similar to that of the human arm (see chapter 3) the task initially appears to be needlessly complicated.

For example, muscles that act around joints act in pairs. The resultant levels of activation indicate not only the final position, but the stiffness of the muscle. As already mentioned, a particular required position is likely to require two or three cooperating muscle pairs. Muscles are delicate and need to be controlled with care to prevent tearing or ripping of the muscle. Muscles themsleves can have complex internal architectures and span multiple joints (Burke, 1991). A muscle cannot be utilised relentlessly like a robot actuator, it needs to have periods of rest built into its control program to prevent muscle fatigue. Muscles are therefore slower, less robust and more complex to control than a robot actuator. The reason why muscles are utilised so effectively in the human motor system must therefore be due to the control system that governs them. It is from this premise that the Recruitment Model of motor control is proposed as the basis of a mechanism for controlling a human like manipulator. The starting premise of this model is therefore distinct from that of many previous computational models. The Recruitment Model is therefore a response to the question:- How does the device being controlled affect the control mechanism that governs it? To begin to answer this question it is first necessary to reduce the scope of the question to a particular subsection of the human control process.

6.2.1 Ballistic phase of motor action

From the information reviewed in chapter 3 it is clear that one of the main motor pathways (the pyramidal tract) is a feed-forward process in the initial stages of any movement. The motor cortex (M1) targets by way of the pyramidal tract neurons (PTNs) directly onto the brain stem and spinal cord grey matter. The spinal cord grey matter is made up of motor neurons organised into motor pools (chapter 3.4.2) which in turn target onto individual parts of muscles. Information from the muscle is then fed back to the motor cortex by way of the extrapyramidal tracts to alter the descending motor cortical command. This subsequent altered movement is known as the corrected phase of movement. The initial movement made before feedback is known as the ballistic phase of movement. The ballistic phase of motor action therefore precludes the complexities of areas like the basal ganglia and cerebellum, and is known to directly involve M1, the motor pools and the muscles themselves, simply for the production of the movement (the planning side requires numerous other cortical and sub-cortical regions) (chapter 3).

6.2.2 Classes of Motor Action

There are a number of different types of motor action:- locomotion, voluntary and reflexive actions are some examples. Reflexive motor acts occur primarily in the spinal cord and brain stem and are low level stimulus response circuits. Reflexive circuits differ markedly from voluntary motor acts because they are overridden by descending higher motor commands. For example, if a person wanted to, they could keep their hand in a fire, even though their reflexive motor system would naturally withdraw it. In this case it is the higher motor centres which are directly inhibiting the behaviour of the reflexive motor centres. Reflexive motor circuits are not currently described in the Recruitment Model. Locomotion occurs usually without conscious effort, but it is a complex process that occurs in numerous brain regions, both cortical and sub-cortical. The Recruitment Model proposes a central role for an area such as M1 which has a neurophysiological basis (section 3.3.1). Voluntary movements and locomotion are both examples of motor actions that would utilise M1 for the "production" of the movement. The "production" of movements is described in the following section.

A voluntary motor act is an action which you consciously make. This class of motor act underlies many other types of motor action, for example writing letters, planning hand movements and recall of complex sequences such as typing or playing a piano. The Recruitment Model of motor control proposes that each of these "higher" or "complex" motor acts consists of two functional components: the "planned" or recalled component and the "production" component. The Recruitment Model makes a distinction between these and further proposes that a specialised "higher" or "complex" area for each class of motor act operates through a common "production" area. The "higher" areas are known as "higher motor centres" (HMC) in the Recruitment model and are described in section 6.4. The production process is described below.

6.3 Motor production

To understand how this division will operate further definitions are required. In the Recruitment Model the motor production areas are responsible for implementing "simple" movements. A "simple" movement is a movement that can be realised in a single step. A "complex" movement is therefore a temporal sequence of "simple" movements. Examples of simple movements are reaching out to pick up a cup or pointing to any position in space (within the person's immediate sphere of action). It is important to note that a so-called simple movement is in fact a very complex process. For example the majority of arm reaching movements will involve several joints moving over differing distances, through the operation of many muscles, contracting by varying amounts in equal time. This implies a complex spatial relationship as well as a complex temporal one. These aspects combined are commonly referred to as the synergy formation problem (section 2.4.3).

In the Recruitment Model, M1 is deemed to be the entry point from the higher motor centres (HMC) and is therefore the start of the production process. M1 is in an excellent position to mediate information flow from cortical and sub-cortical motor structures (see section 3.3.1.1). M1 has also been shown to receive a good deal of sensory and proprioceptive information. Many of the surrounding cortical areas have also been shown to possess "supra" as opposed to "sub" functionality (section 3.3.2) with relation to M1, confirming their possible role as higher motor centres in the Recruitment Model.

By hypothesing such a role fo M1 a major problem needs to be addressed. For M1 to carry out instructions from many other disparate areas, an assumption is made that each area "speaks a common language". Put another way, what is the evidence to suggest that there is a common communications protocol between motor areas?. The evidence turns out to be quite strong:- the somatotopic arrangement of M1 (Woolsey et. al. 1952), has subsequently been reported to exist in some form in many other motor areas. These include the premotor cortex (Caminiti et. al. 1991) the cerebellum (Fortier et. al. 1989) as well as parietal area 5 (Kalasaka et. al. 1983; Kalasaka, 1988). This fact alone is not sufficient to confirm a common protocol as it only confirms preservation of spatial information. Each of the studies mentioned also confirms that arm direction is encoded in a population code.

Given that there is this common communications protocol between areas, what is the exact information that is being exchanged? The Recruitment Model proposes for practical reasons that it is body-centered, joint-based information. The practical reason for this is one of reducing the number of variables under control. For example, if muscles are chosen then the connectivity would be massively higher. The information is body-centered or put another way based on relative angular information. This is because there is no reason to assume that the human motor system would adopt our Euclidean coordinate system. This hypothesis is also backed up by the above studies, since they report a correlation of movement direction with cell activity and not muscle activation. By use of this common communication highway it is possible to see how some of the amazing flexibility of the human motor system is achieved. The somatotopy preserves the same spatial parallelism with all cooperating areas. The common information exchange opens up the possibility of many specialised motor areas formulating specific plans. These plans could be evoked at a moment's notice through a change in intentionality. These specialised contingency plans have a neurophysiological basis also, as

Godschalk et. al. (1981,1985) have noted, as prepratory “set” related activity in the premotor cortex. The massive parallelism allows processing to be carried out extremely quickly as only a few steps are required to make complex transitions.

The Recruitment Model (Figure 6.1 - duplicate of figure 1.2 reproduced for ease of reading) breaks down the motor production process into three steps:- the Joint Recruitment Level (JRL), the Muscle Recruitment Level (MRL) and the motor pools. HMC’s would target onto the JRL, recruiting the appropriate populations, each of which would in turn recruit muscle related populations at the MRL. Each MRL population would then target the appropriate motor pools to produce the desired movement. Each area is discussed in more detail below. The JRL and the MRL are functionaly equivalent to M1 (see section 3.3.1.6). The JRL and the MRL therefore are responsible for the transition from joint related information to muscle related information (Kalasaka 1992). The MRL then targets the appropriate motor pool with correctly weighted connections (Fetz et. al. 1990); the details of these connections is described in the proceeding sections of this chapter.

The Recruitment Model Of Ballistic Voluntary Motor Action.

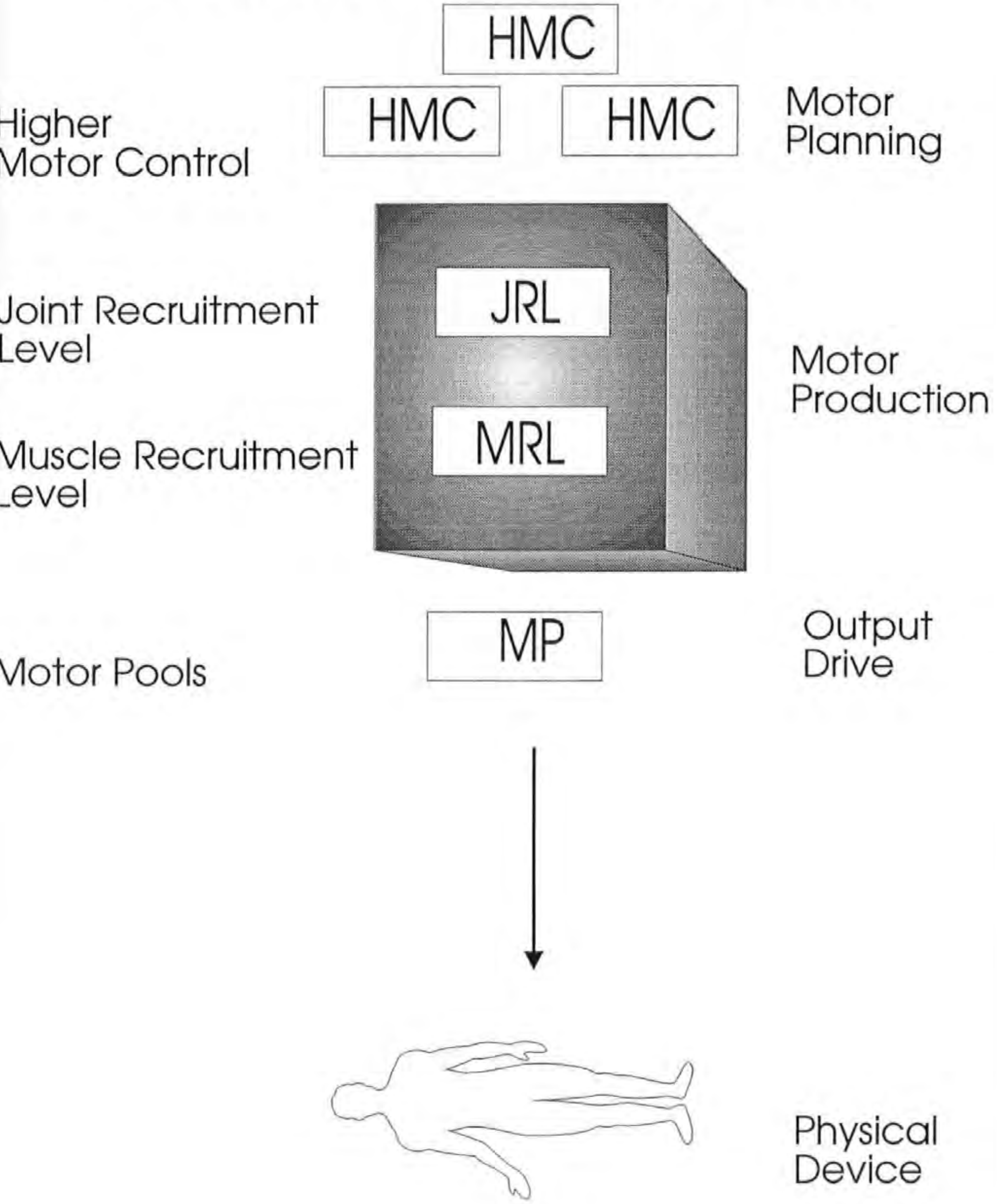


Figure 6.1

6.4 Higher motor centres (HMC)

Each HMC would target the appropriate area of the JRL with a desired movement amplitude. The spatial form of the spike train would be used to indicate the speed of the desired movement. This arrangement presupposes a strong temporal correlation between all motor cortical areas. The output from a HMC at any instant in time would be a request for a particular joint position. Each HMC would also broadly preserve the somatotopic organisation of the JRL. The somatotopy within the HMC's is likely to become distorted on the basis of the specialised function being encoded. The HMC would therefore request a "simple" movement be executed by the JRL at each instant in time. Over a period of time a series of "simple" movements would be requested, which would in combination form a "complex" movement. The timing of the release of "simple" movements to the JRL will be critical. Consider a series of three required positions, where the third position is released before the second position has had sufficient time to be implemented. In this case the second position is never realised, instead some other intermediate position is reached. The combination of positions that are reached, and those that are not would encode a virtual path or trajectory. This is in principle similar to the requirements of Spring To Endpoint (STE) models (Bizzi et. al. 1984).

The practicalities of implementing the timing of the release of such models is not currently explained any further in the Recruitment Model, except to note that a base-time for each movement would be known, by virtue of the motor pools being able to execute any movement amplitude in a similar period of time (duration invariance). In the Synchronous neuronal model the speed of contraction caused by a motor pool is controlled by the spatial spiking of the input signal i.e. slower movements are achieved by increasing duration between spikes, even though the same absolute frequency is achieved ultimately. The same concept applies to HMCs requesting joint rotations in the JRL.

An important point is that each HMC has no direct knowledge of which muscles are utilised to carry out the required movement. Neither is the force of the upcoming movement considered. These tasks are delegated to the motor production process, which is assumed to carry out movement requests correctly. The HMC's are therefore able to focus on the task of planning only, and are isolated from the task of implementation.

Feedback from the JRL would be required for the formulation of motor plans, and for the confirmation of action in subsequent corrected phases of motor action. It is also possible that the multiple (homonocular like) representations observed in M1 (Humphrey and Mitz, 1989, Sato and Tanji, 1989; Sanes and Donoghue, 1992) would be utilised by HMCs for predictive planning in parallel with a currently executing movement, essentially allowing planned movements to be tested or prepared before they are required.

6.4.1 Example of a simplified higher motor centre (HMC)

To demonstrate how a simplified version of the Recruitment model would function, a simulation of a "stick-man" running has been devised. The details of the simulation are described below in section 6.7. From the review of coordinate systems in section 2.2 and again in chapter 3.3 it was decided that the Recruitment Model would adopt a coordinate system based around relative body centered angular coordinates.

The function of this particular HMC is to release a sequence of positions sufficient for the "man" to move his limbs as if running. If we now consider the arm in isolation, it will have two well defined positions; a start position and an end position (denoted by the position of the hand). In this model the movement of the arm is defined in only two dimensions. In a more realistic version even in only two-dimensions the arm movement would require the release of at least three coordinate positions, even though the intermediate positions would probably never be reached. The current simulation could be easily adapted to account for these intermediate positions.

The position of the hand in space is defined by two parameters, the angle of the hand taken in a straight line from the shoulder relative to an upright vertical axis bisecting the shoulder, and a displacement from it. From this information it is possible to compute the angular displacement of the shoulder joint and of the elbow joint (the wrist joint is not modelled), utilising a biomechanical property of human physiology; namely, the elbow and shoulder joints have adjoining limbs of approximately equal length.

For a required angular displacement, α_l from the vertical axis of the shoulder and a total displacement l_l the joint positions θ_l and θ_2 (shown in Figure 6.2) are defined as follows:-

$$\theta_l = \mathrm{Sin}^{-1}(l_3/ l_l)$$

$$\theta_2=(\pi-2\theta_l)-(\pi-\alpha_l)= \alpha_l-2\theta_l$$

where:-

l_l is the displacement to arm endpoint

α_1 is the angular displacement of the first joint

l_3 is the limb length

The leg and hip joint limbs are analogous to the arm and shoulder joint limbs and are subject to similar treatment.

The target joint positions for each limb are then communicated to the relevant JRL utilising the known somatotopic organisation. The communication takes the form of a spike train indicating the required displacement. The spike train representing the required displacement could be modified to change the speed/duration of the movement, but in this simulation all the joints were required to begin and end the movement simultaneously. The HMC will then oscillate through the defined start and end positions for each joint of each limb. In the "running man" simulations the period of oscillation was fixed.

Simplistic model of a Recruitment Model Higher Motor Center (HMC)

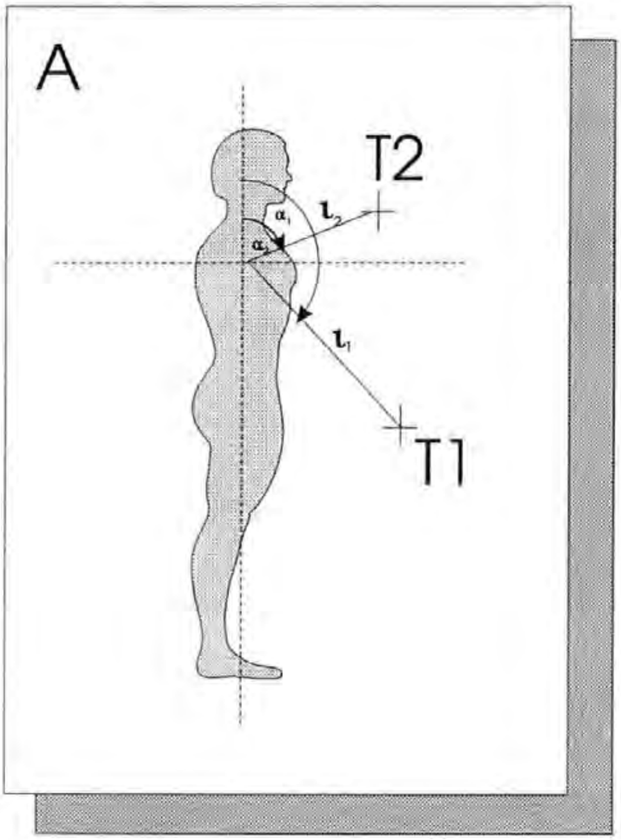


Figure A depicts the two positions (T1,T2) released from the simplified HMC during the "running man" simulations.

Endpoint for each limb is encoded as a displacement from the shoulder.

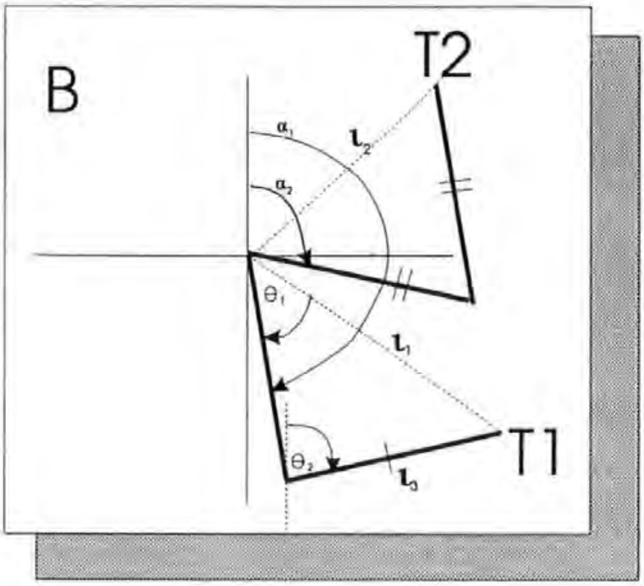
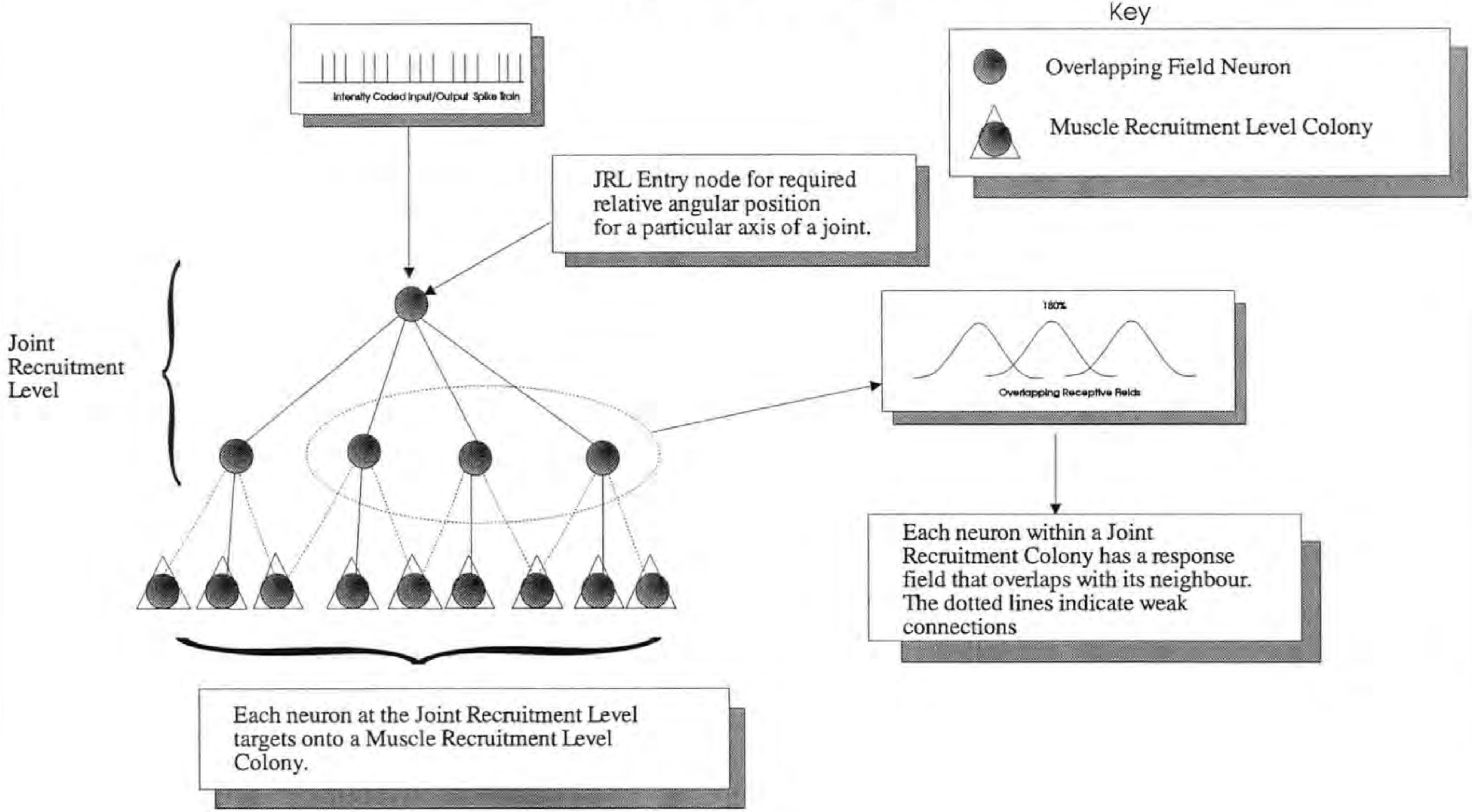


Figure B, shows the two positions T1 and T2 in more detail. The model utilises the biomechanical property of the arm joints being equal in length.

Figure 6.2

Interactions at the Joint Recruitment Level (JRL)



JRL Entry node for required relative angular position for a particular axis of a joint.

Each neuron within a Joint Recruitment Colony has a response field that overlaps with its neighbour. The dotted lines indicate weak connections

Each neuron at the Joint Recruitment Level targets onto a Muscle Recruitment Level Colony.

Figure 6.3

6.4 Joint Recruitment Level (JRL)

The JRL encodes the entire body position in terms of joint information. Each individual joint is encoded in terms of angular coordinates, relative to its nearest neighbouring joint. The relative merits of this choice has been described in the sections above.

A separate colony of cells (a Joint Recruitment Colony, JRC) is used to represent each degree of freedom for every joint. Each of the JRCs target directly onto the MRL (see Figure 6.3), and in particular onto specific Muscle Recruitment Colonies (MRC-see section 6.6.1) within the MRL. The somatotopic order observed in M1 is preserved in both JRL and MRL.

In the human control system this arrangement would work as follows:- each joint position would be encoded using up to three JRCs(one for each degree of freedom of the joint). Each of these JRCs would then target onto around two MRCs in a simple case, where only a single agonist and antagonist need be considered. In more complex cases where MRCs do not pull in exactly the correct line of action, further MRCs would be connected. The connection strength between the JRC and the targeted MRC indicates the efficiency of the MRC in that particular direction, for example an MRC with very little effect in a particular direction would be weakly connected. The additional complexities accounted for at the MRL are described below in section 6.6.

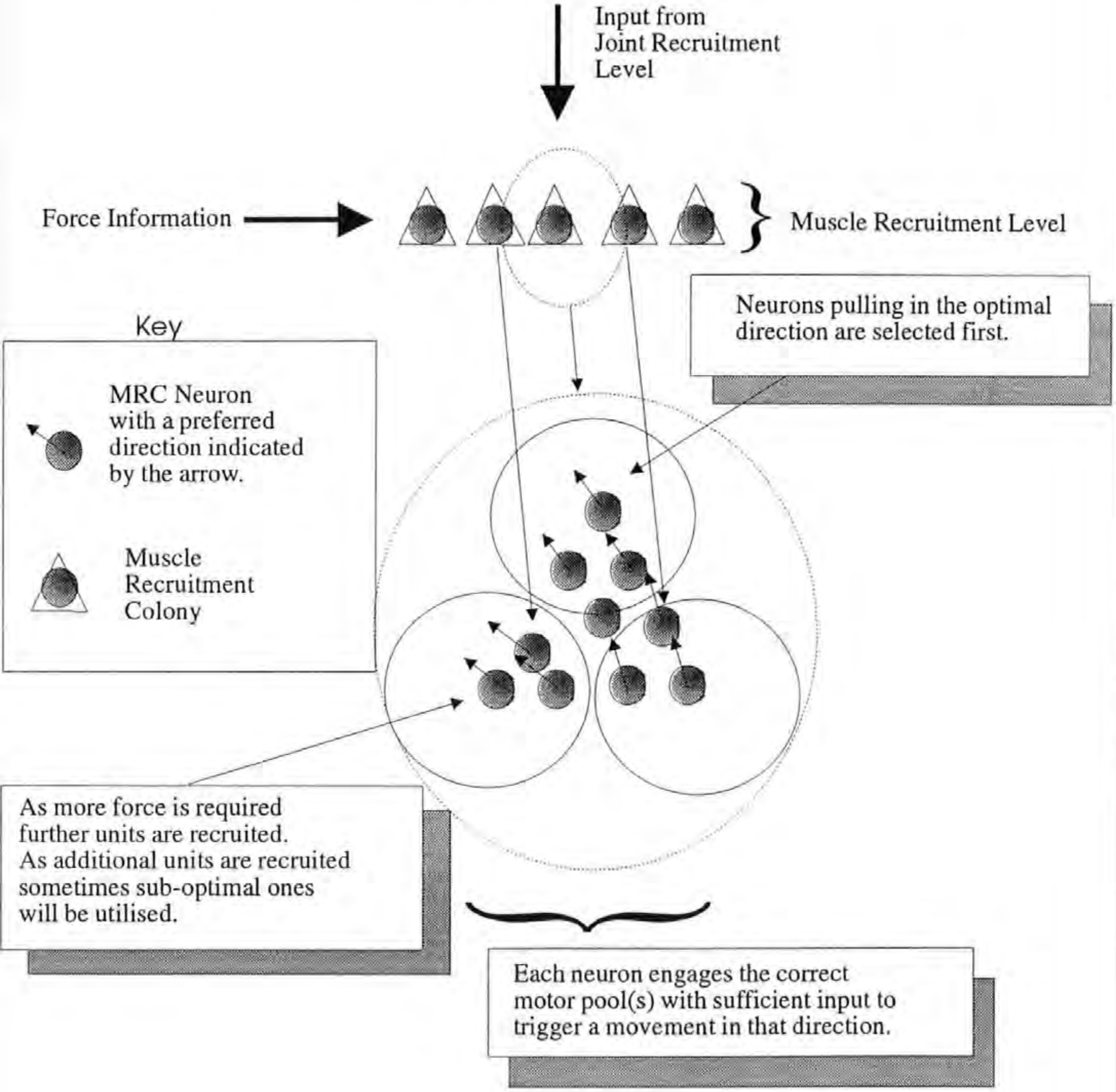
The timing information would be carried forward to the MRL unaltered, the amplitude of the reaching movement would become distributed through the number of active cells in each JRC. The activity of each cell within a JRC would effectively encode pre-learnt unit-force information. This information could be learnt through a motor babbling phase (Piaget, 1963). The JRL would receive angular proprioceptive feedback in subsequent corrected phases of movement to ensure correct adaption.

6.5.1 Joint Recruitment Colony (JRC)

A JRC represents a particular plane of rotation about a joint. A JRC consists of a number of neurons, each of which receives the integrated required relative angle. The neurons in the JRC have a Guassian response characteristic similar to that described by Schwartz et. al. 1988 (see section 3.3.1). This means that each neuron has a particular preferred angular rotation for which its output is strongest. Any inputs falling outside this range elicits outputs on a gradually declining scale dependant upon the difference between the preferred direction and the actual required direction. The benefit of distributing the input amongst various neurons within the JRC is that the most efficient MRC colonies can be targeted for particular angular rotations. Another benefit is that physical disturbances and interactions with neighbouring joints can also be accounted for at this level.

The JRCs collectively encode the transition from Euclidan gemoetry to high dimensional muscle geometry. Each JRC recruits cells within the population as successively higher levels of angular rotation is required. Each JRC is analagous to a motor pool neuron, except that the output connectivity is more complex, in that it connects to multiple targets with varying strength.

Interaction within the Muscle Recruitment Level (MRL)



Neurons pulling in the optimal direction are selected first.

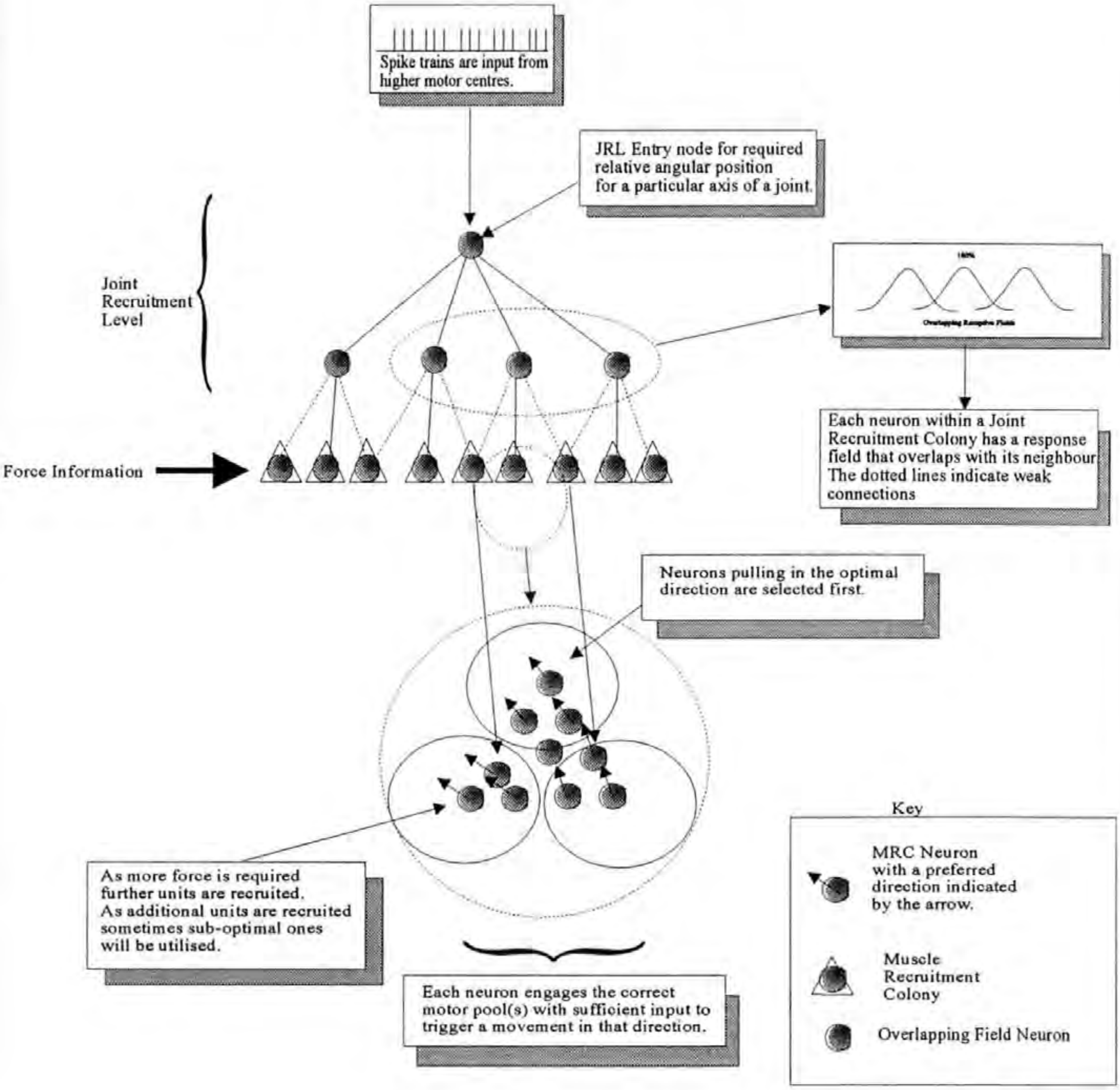
As more force is required further units are recruited. As additional units are recruited sometimes sub-optimal ones will be utilised.

Each neuron engages the correct motor pool(s) with sufficient input to trigger a movement in that direction.

Inputs from the Joint Recruitment Level are targetted onto the most efficient Muscle Recruitment Colony.
This signal is then integrated with another signal indicating any additional force requirements.
The combined signal is then propogated from the point of entry to adjoining neurons.

Figure 6.4

Interactions at the Joint Recruitment Level(JRL) and the Muscle Recruitment Level(MRL)



Spike trains are input from higher motor centres.

JRL Entry node for required relative angular position for a particular axis of a joint.

Each neuron within a Joint Recruitment Colony has a response field that overlaps with its neighbour The dotted lines indicate weak connections

Neurons pulling in the optimal direction are selected first.

As more force is required further units are recruited. As additional units are recruited sometimes sub-optimal ones will be utilised.

Each neuron engages the correct motor pool(s) with sufficient input to trigger a movement in that direction.

Inputs from the Joint Recruitment Level are targetted onto the most efficient Muscle Recruitment Colony.
This signal is then integrated with another signal indicating any additional force requirements.
The combined signal is then propagated from the point of entry to adjoining neurons.

Figure 6.5

6.6 Muscle Recruitment Level (MRL)

The layout at the MRL is more complex (Figure 6.4). Each neuron at this level will target on average three motor pools, although the range is between one and six (Fetz et. al., 1990).

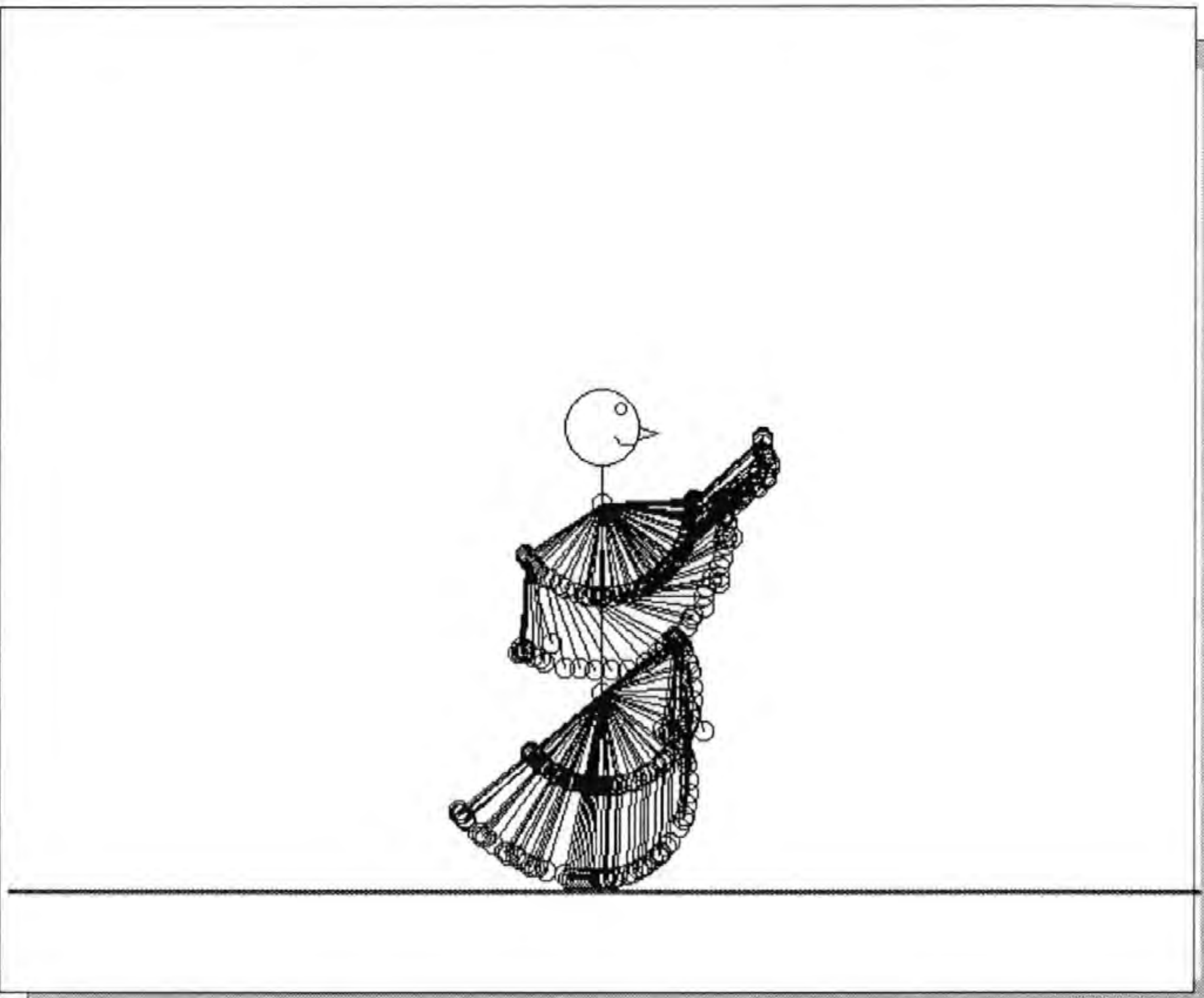
A neuron that targets only one motor pool obviously only acts in one direction, and will be the first choice of the relevant JRC if a movement is required in that direction. The second choice would be a neuron that only targets two motor pools and so on until neurons are reached that have no effect in the required direction. A neuron or population of neurons that has connectivity in more than one direction will be targeted by more than one JRC. It therefore must be able to integrate these signals. If that population is already active then a mechanism is required that allows the next best selection to be made and so on until either the required movement is elicited or a maximum physical output is reached. This mechanism is particularly important because it is at this level that loading effects must be considered. As the load increases, more and more MRL cells must be recruited but in a balanced fashion.

The MRL therefore essentially receives information about the timing of each muscle and exactly which muscle is to be used from the JRL. This information is then combined with loading requirements to give the total output requirement. MRL therefore encodes a mechanism that allows the most efficient cells to be targetted first, and successively higher levels of activation recruit adjoining cells. The process is supported by the somatotopy evident in M1. It is at this level that the quirks and complexities of the human motor system would be accounted for, for example multiple muscle compartments (Burke, 1991) could be modelled here which could serve to compensate for many physical interactions, by accurately controlling step changes in movement i.e. each small step within a movement trajectory would be modelled by a cell population that is specifically optimised for that position.

6.6.1 Muscle Recruitment Colony

Each neuron within a MRC has a preferred direction of action identical to those described in section 3.3.1. As the required force increases, additional units are recruited by allowing the signal to propagate to adjoining neurons. The last neurons to be recruited are those that pull in the required direction least efficiently. Each neuron within the MRC will target the appropriate neuronal pools with sufficient input to evoke a movement in the preferred direction. This means that connections from a single neuron to multiple pools would have connection strengths to those pools which indicate their contribution to that movement. Proprioceptive feedback from each muscle encoding muscle length would be required to ensure correct adaption at this level. The motor pools isolate the MRL from the problems of fatigue and impose a usefull timing relationship (chapter 5).

"Running Man" Simulation Using A Simplified Recruitment Model



Simulation Parameters:-

8 Motor Pool Networks (1 per joint)
28 Nodes Per Network.
224 Nodes.

Each limb oscillates with a fixed period between two positions released from a Higher Motor Center (HMC)

Figure 6.6

6.7 The "running man" simulation.

The "running man" simulation (Figure 6.6) has 8 separate motor pools, each with 28 neurons, giving a total of 224 neurons, all of which are simulated in parallel in real-time. The rotation about each joint is represented by a network (motor pool), which encodes both agonist and antagonist muscle activity.

Figure 6.7 shows how the Recruitment Model has been simplified for the purpose of this simulation. The model is defined in two dimensions only, and assumes that the muscles causing the rotations about the joint pull directly in the required line of action. The simulation takes no account of force or loading effects and completely ignores all physical interactions that would need to be solved in the "real world". As a result of these simplifications the HMC effectively passes the signal directly onto the appropriate motor pool. Output from a HMC model (described in section 6.4.1) is encoded into a spike train with appropriate temporal and spatial characteristics. This output is then targeted onto the correct motor pool (a simplification of the somatotopy). Each motor pool was defined as described in chapter 5 (Synchronous single tier motor pool experiments). The simulation also assumes that muscle stiffness is regulated by Renshaw Cells(RC) in the motor pool to give a linear characteristic (Nichols and Houk, 1976).

Two positions for each limb were defined such that the transitions between them would cause the simulated joints to contract at differing rates in equal time. All the intermediate positions were generated by the temporal nature and output profile characteristics of the motor pool. From the simulation it can be seen that the joints start the movement cycle together and end together, thus demonstrating how the intrinsic timing in the motor pool output can be utilised to help in the synchrony of movements and the formation of synergies.

An observation of note is that visually the movements appear “fluid”, i.e. they look quite human-like compared to the sort of movement that would be achieved if this animation was implemented using standard sequential animation techniques. One reason for the “fluid” nature of these movements is that the actual output of the motor pools is inaccurate compared to the display resolution, and it is the errors in the output that lead to a more flowing movement. The same is presumably true for human movements.

Simplified Recruitment Model For Real-Time Simulation

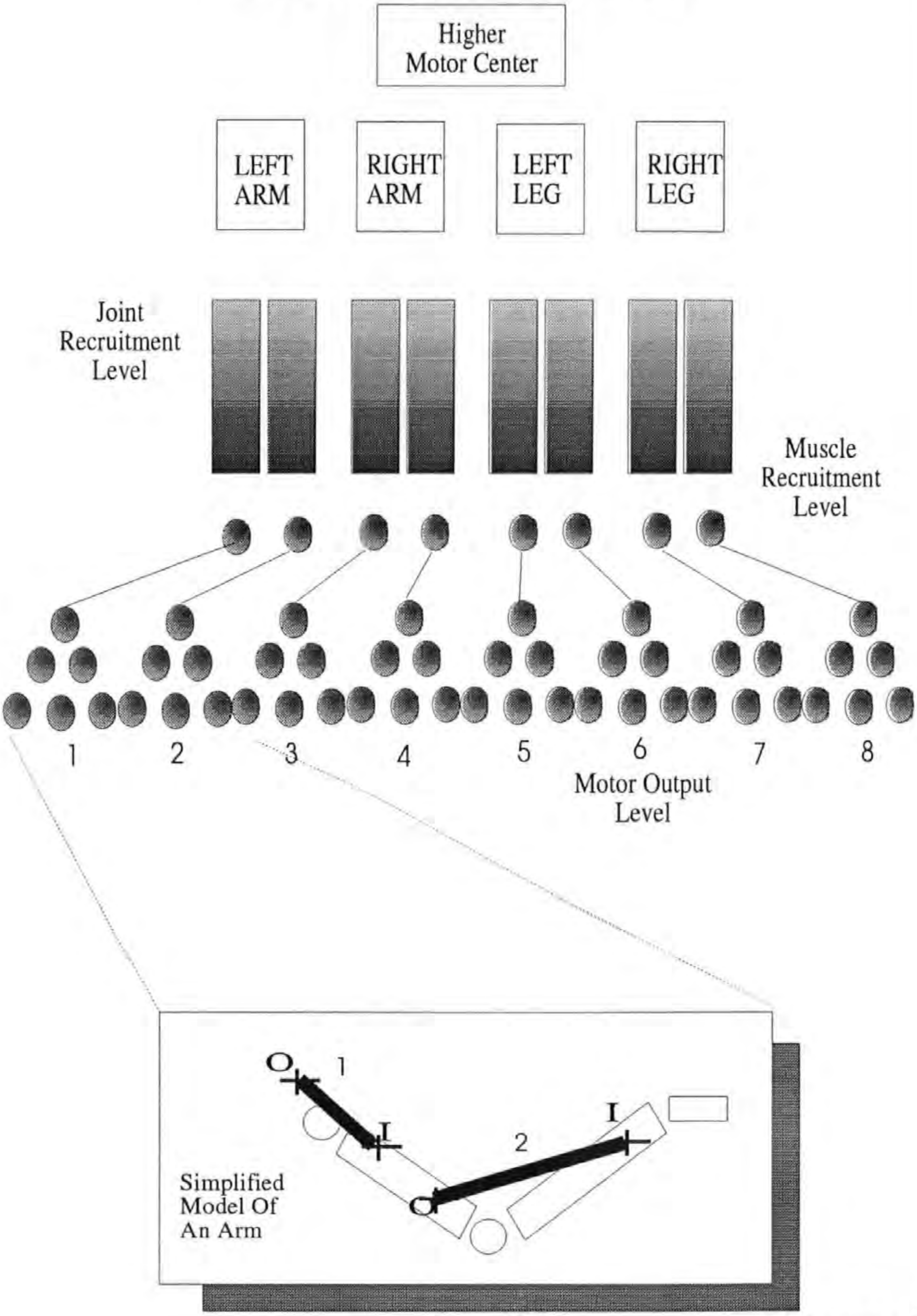


Figure 6.7

6.8 Summary

In this chapter the Recruitment Model of ballistic voluntary motor action was presented in detail. The Recruitment Model was described in terms of the neurophysiological evidence presented in chapter 3 and with particular reference to the experimental simulations presented in chapter 5.

The proceeding discussion has highlighted the complexity of the interaction that must occur within the motor production system in order to account for the flexability of the human motor system. It is highly distributed in nature and would require a "recruitment" mechanism to facilitate the distribution of the command signal to the appropriate units. The Recruitment Model draws upon detailed information presented in disparate fields (such as physiology, neurophysiology, control theory and biomechanics) and attempts to combine this information in a form that can accommodate a more computationally functional discussion on ballistic voluntary motor action.

In the current form the Recruitment Model is acknowledged to suffer from a number of severe shortcomings. The most significant omission from the model is the lack of feedback and the absence of any force or loading information. This is a major restriction in the further clarification of the details at both the MRL and JRL.

Considerable improvements would need to be made to the model before these simulations could be carried out, in particular the simulations of the physical device being controlled would need to be more realistic and incorporate at least some of the physical complications described by Hollerbach and Flash (1982) in order to test the effectiveness of the Recruitment Model.

The simulations of the "running man" demonstrated the Recruitment Model implemented in its simplest form. The simulation demonstrated what form a Higher Motor Center (HMC) might take and how it would interact with the motor pools to demonstrate the importance of duration invariance reported from the motor pool experiments in chapter 5 to at least partially solve the synergy formation problem.

7. Conclusions and Future Directions

In chapter 1 some general issues relating to motor control were introduced, and the goals and objectives of this research were defined. Some terms and definitions relating to the Recruitment model of ballistic voluntary motor action were introduced along with a description of each of the chapters in this thesis and a brief description as to how each related to this investigation.

In chapter 2 a more detailed overview of motor control issues was presented as well as some physiological observations made regarding human arm movements. The chapter concluded by looking at some models that presented novel approaches to motor control. The control and physiological concepts introduced in chapter 2 then provided a context for chapter 3, which took a detailed look at the neurophysiology relating to motor control. This detailed review of neurophysiology forms the basis for the majority of chapters that follow and is a key component to understanding the Recruitment Model and the motor pool experiments in chapter 5.

In chapter 4 two neuronal models were described, the Leaky Integrator neuronal model which was used during the early motor pool simulations and the Synchronous neuronal model which was used in all subsequent simulations. Chapter 4 gives details of how each model was simulated on computer and how each of the parameters affects the neuron's behaviour. The Synchronous neuronal model was developed as part of this study for a number of reason highlighted by deficiencies in the early Leaky Integrator motor pool studies described in chapter 5. The Synchronous neuronal model specifically models the input/output frequency relationships for each cell as well as the spatial form of spiking to and from each cell.

Chapter 5 introduced the features of the INSIGHT TOO software package and its application in the investigation of motor pool behaviour utilising both the Leaky Integrator and Synchronous neuronal models. The early Leaky Integrator simulations attempted to reproduce physiologically observed data using motor pool neurons organised to produce a spatiotemporal output sequence for a given input. The approach proved to be flawed; subsequent Synchronous simulations were more successful and were based on a more continuous and responsive network design. In particular the simulations showed that motor pool neurons could be responsible for the production of physiologically observed properties such as duration invariance and velocity profile invariance.

The observed characteristics of the motor pool experiments were then put into the broader context of a complete model of motor control based on the information presented in chapter 2 and chapter 3. The result was a theoretical model of ballistic voluntary motor action called the Recruitment Model which was described in detail in chapter 6. The Recruitment Model introduced the distinction between motor planning and motor production. It is hypothesised that M1 is responsible for motor production, and that multiple motor planning areas communicate with M1 using relative body centered angular coordinates transmitted between many cortical areas through a shared connectional topographic organisation known as somatotopy. The motor production areas are responsible for realising “simple movements”, defined as movements that can be realised in a single step. Whereas the motor planning areas release temporal sequences of "simple movements" to produce "complex movements".

The "simple movements" are integrated into groups of cells acting synergistically to perform the desired motor act. The Joint Recruitment Level(JRL) and the Muscle Recruitment Level(MRL) were introduced in chapter 6 as the structures required for the transformation of joint related information to muscle based information as part of the motor production process. The JRL and MRL both use "recruitment" as the process of achieving this transformation in a truly distributed manner. This contrasts with the inherently sequential architecture of most other control models.

A much simplified version of the Recruitment Model was simulated using the Synchronous neuronal model, and a simple HMC was developed to release a sequence of "simple movement" instructions. The resultant simulation was of a "running man" which simulated two joints for each of four limbs of a stick man.

The "running man" simulations proved to be of interest due to the fluid nature of the movements, which was due to the inaccuracies of the motor pool output. The simulation also demonstrated how certain aspects of the synergy formation problem might be solved by this model.

The motor pool simulations were not sufficiently detailed to claim biological plausibility, they were instead biologically inspired. To apply the motor pool results to a full model of motor control would require significantly more investigation. Motor pool behaviour would need to be studied for a variety of required force conditions under a variety of temporal conditions. The interaction between muscle pairs would also need to be investigated and the resultant affect of the "stiffness" caused in the muscle. As processing power becomes more readily available larger motor pool sizes will become practicable, facilitating greater experimental detail. A number of other tasks need to be performed in conjunction with this, namely a detailed study of muscle function and the compartmentalisation of muscles. This study needs to be assessed in the context of an accurate dynamics model. As the level of simulation detail progresses, more biomechanical complexity can also be considered. It is believed that the structural form of the Recruitment Model is sufficient to cater for this additional complexity but future investigation will be required to test this hypothesis. The detailed and thorough investigation of these low level functions is essential to accurately inform on the exact nature of the interactions with the higher levels of motor control.

A glaring and well acknowledged deficit of the Recruitment Model is the complete lack of any feedback or adaption. To understand the role of feedback a detailed look at many other cortical and subcortical areas in the brain will need to be undertaken. This study is likely to yield a relatively simple implementation in the Recruitment Model but its exact form will be critical in building to a truly adaptive model. There are many small local feedback circuits and special cases will also need to be considered, but before that, a large amount of work needs to be undertaken in understanding the primary motor areas.

As future work progresses, one of the most interesting areas of study will be the exact form of the higher motor centers (HMC) and how easily the wide variety of complex motor tasks can be simulated. The Recruitment Model predicts a strong need for temporal correlation between cortical and sub-cortical areas. This has not been the subject of any studies to date. The design of the Recruitment Model and the motor production interface with the motor planning centers should provide a good structure to integrate information from a number of disparate areas.

There is however, a clear critical path of analysis, and it starts with a more detailed look at muscle function in the context of more detailed physical modelling of parameters such as gravity and momentum.

The INSIGHT TOO software although very flexible suffered from not being as interactive as its predecessor, ideally a combination of both tools would be used in future investigations. A number of neuronal software simulation tools of sufficiently good quality are now readily available. These software tools negate the need for any further development of a general software simulation tool. There will most probably always be a role for specific programs tailored to a specific tasks. In order to push simulations to there limit these specialised tools will always be required. Considering the complexity and size of the brain it is not likely this requirement will diminish in the near future.

As far as neuronal modelling is concerned there are many models similar in behaviour to the Synchronous neuronal model (Chapter 4; Redman and Graham (1993)) developed as part of this study, but the results of this study have indicated that models that simulate individual spiking behaviour would be a positive asset in the future investigation of motor action.

The Recruitment Model at some point should be implemented on a "real" device, it is possible that the output model could be adapted to be implemented on a rigid robotic manipulator, but care should be taken not to lose the flexibility of the human design, and it certainly should be implemented in a distributed fashion. The ideal would be to continue this investigation using "smart materials" capable of behaving in a similar manner to that of a human muscle. At the time of writing this report such materials are not available, but their future availability looks promising. If this should prove not to be the case then 3D modelling technologies are becoming more accessible. Either way a tool capable of displaying and controlling a simulation of complex interacting control elements needs to be made available.

8.0 References

Abend, W., Bizzi, E., and Morasso, P (1982). Human arm trajectory formation. Brain,105, 331-348.

Akazawa K. and Kata, K., (1990). Neural network model for control of muscle force based on the size principle of motor unit. Proceedings of the IEEE, Vol. 78. No 9. September 1990

An C.H., Atkeson C.G. Hollerbach J.M.(1988) Model-based control of a robot manipulator. MIT Press. Cambridge.

Atkeson, C.G. and Hollerbach, J.M. (1985).Kinematic features of unrestrained vertical arm movements. Journal of Neuroscience, 5 (9), 2318-2330.

Barbas H, and Pandya D.N. (1987) Architecture and frontal cortical connections of the premotor cortex (area 6) in the rhesus monkey. J. Comp Neurol. 256:211-228.

Bigland-Ritchie B.R, Johansson R.S, Smith S, Woods, J.J. (1983) Changes in motorneurone firing rate during sustained maximal voluntary contractions. J.Physiol. (Lond.) 340, 335-436.

Bigland-Ritchie B.R, Dawson N.J, Johansson R.S. and Lippold O.C.J. (1986) Reflex origin for the slowing of motorneurone firing rates in fatigue of human voluntary contractions. J.Physiol. (Lond.) 379, 451-459.

Bizzi E, Accornero N, Chapple W, Hogon N. (1982) Arm trajectory formation during arm movment. J. Neurosci. 4: 2738-2744.

Bizzi E, Acconero N, Chapple W, Hogan N (1984) Posture control and trajectory formation during arm movement. J. Neurosci. 4:2738-2744.

Bizzi E, Mussa-Ivaldi, F. (1990) Muslce properties and the control of movement. In D. Osheron, S.Kossyln and J.Hollerbach (Eds),, Visual Cognition and Action (Vol 2. pp 213-242). Cambridge: MIT Press.

Bressloff P.C. and Taylor J.G. (1990) Proc Int. Neural Network Conf. Paris, 1009-1012

Brinkman C (1984) Supplementary motor area of the monkey's cerebral cortex: short-and longterm deficits after unilateral ablation and the effects of subsequent callosal section. J. Neurosci. 4: 918-929

Brodmann K.(1909) Vergleichende Lokalisationlehre der Grosshirnrinde in ihren Prinzipien dargestellt auf Grund des Zellenbaus. Barth, Leipzig.

Bullock D. and Grossberg S. (1988) Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation. 95, 49-90.

Bullock D. and Grossberg S, (1988a) The VITE model: A neural command circuit for generating arm and articular trajectories. In J.A.S. Kelso, A.J. Mandell, and M.F. Sclesinger (Eds.) Dynamic patterns in complex systems. Singapore: World Scientific Press, 305-326.

Burke, R.E. (1981): Motor units: anatomy, physiology, and functional organization. In : Handbook of physiology. Brooks, V.B. (ed.).Sect. 1:The nervous system, Vol, II. Motor control, Part 1, 345-422, Bethesda: Am. Physiol. Soc.

Burke, R.E. (1990a) Motor unit types: Some history and unsettled issues. In M.D. Binder & L.M. Mendell (Eds.), The segmental motor system (pp. 207-221). New York: Oxford University Press.

Burke, R.E. (1991) Selective recruitment of motor units. In: Motor control concepts and issues. Edited by D.R. Humphrey and H.J.Freund. John Wiley & Sons Ltd.

Burnod Y, Granguillaume P, Otto I, Ferraina S, Johnson P.B, Vaminiti R (1992) Visuo-motor transformations underlying arm movements toward visual targets: a neural network model of cerebral cortical operations. J. Neurosci.

Caminiti R, Johnson PB, Urbano A (1990) Making arm movements within different parts of space: dynamic aspects in the primate motor cortex. J. Neurosci. 10: 2039-2058.

Caminiti R, Johnson P.B, Galli C, Ferraina S, Burnod Y. (1991) Making arm movements in different parts of space: the premotor and motor cortical representation of a coordinate system for reaching visual targets. J. Neurosci. 11:1182-1197.

Caminiti R, Johnson P.B, Ferraina S, Burnod Y. (1992) Reaching toward visual targets. I. Neurophysiological studies. In: Control of arm movement in space. Neurophysiological and computational approaches. (Eds) Caminiti R, Johnson P.B, Burnod Y. Springer-Verlag.

Carew T.J. (1981) Spinal cord I: Muscles and muscle receptors. In E.R. Kandel & J.H. Schwartz (Eds.), Principles of neural science (pp. 284-292). New York: Elsevier North Holland.

Cheney P.D., Fetz E.E. (1985) Comparable patterns of muscle facilitation evoked by individual corticomotoneuronal (CM) cells by single intracortical microstimuli in primates: evidence for functional groups of CM cells. J Neurophysiol 53:786-804.

Darian-Smith C, Darian-Smith I, Cheema SS (1990) Thalamic projections to sensorimotor cortex in the macaque monkey: use of multiple retrograde fluorescent tracers. J. Comp. Neurology. 299:17-46

DeVito JL, Anderson ME (1982) An Autoradiographic study of efferent connentions of the globus pallidus in Macaca mulatta. Exp Brain Res 46: 107-117

Donoghue J.P, Levibovic S.L, Sanes J.N. (1989) Organization of muscle representations in primate forelimb motor cortex. In: Neurscience intgerative functions: a symposium hônoring Vernon B. Mountcastle, Book of Abstracts, abstract no. 44

Dubose L, Schelhorn T.B, Clamann H.P. (1987) Changes in contractile speed of cat motor units during activity. Muscle & Nerve. 10, 744-752.

Dum RP, Strick PL (1991) The origin of corticospinal projections from the premotor area on the frontal lobe. J. Neurosci. 11:667-689

English A. W. (1984). An electromyographic analysis of compartments in cat lateral gastrocnemius muscle during unrestrained locomotion. J. Neurophysiol. 52: 114-125.

English. A. W. and O. I. Weeks.(1984). Compartmentalization of single muscle units in cat lateral gastrocnemius. Exp. Brain Res. 56:361-368.

Enoka,R, M, and Stuart, D. G. (1984). Henneman's "size principle": Current issues. Trends Neurosci. 7. 226-228.

Evarts E.V. (1968) Relation of pyramidal tract activity to force exterted during voluntary movement. J. Neurophysiology. 31:14-27.

Evarts E.V, Shinoda Y, Wise SP (1984) Neurophysiological approaches to higher brain functions , Wiley, New York.

Favilla M, Hening W, Ghez C (1989). Trajectory control in targeted force impulses: VI. Independent specification of response amplitude and direction. Exp Brain Res 75:280-294.

Feldman A.G. (1966). Functional tuning of the nervous system during control of movement of maintenance of a steady posture-III. Mechanographic analysis of the execution by man of the simplest motor tasks. Biofizika 11:667-675.

Fetz E.E, Cheney PD (1980) Postspike facilitation of forelimb muscle activity by primate corticomotoneuronal cells. J Neurophysiol. 44:751-772.

Fetz.E.E, P.D Cheney. K. Mewes, and S. Palmer. (1990). Control of forelimb muscle activity by populations of corticomotoneuronal and rubromotoneuronal cells. In: Afferent Control of Posture and Locomotion, J.H.J, Allum and M. Hulliger. Prog. Brain Res.

Fitts P.M. (1954) The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology, 1954, 47(6), 381-391

Flash, T. and Hogan, N, (1985) The coordination of arm movements: An experimentally confirmed mathmatical model. Journal of Neuroscience, 5(7), 1688-1703

Fortier P.A., Kalaska J.F., Smith A.M. (1989) Cerebellar neuronal activity related to whole-arm reaching movements in the monkey. J. Neurophysiol. 62:198-211

Freund, H.J. and Budingen, H.J. (1978). The relationship between speed and amplitude of the fastest voluntary contractions of human arm muscles. Experimental Brain research. 31,1-12.

Fulton J.F (1934). Forced grasping in relation to the syndrome of the premotor area. Arch Neural Psychiatr 31:221-235.

Georgopoulos A.P., Caminiti R., Kalasaka J.F. and Massey J.T. (1982) On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. J.Neurosci. 2: 1527-1537.

Georgopoulos A.P. Camimiti R. Kalaska J.F. Massey J.T. (1983) Spatial coding of movement: a hypothesis concerning the coding of movement direction by motor cortical populations. Exp Brain Res (Suppl.) 7:327-336.

Georgopoulos A.P., Taira, M., Lukashin A.V. (1993) Cognitive neurphysiology of the motor cortex. Science, 260, 47-52

Georgopoulos, A.P.,. and J.T. Massey (1985) Static versus dynamic effects in motor cortex and area 5: Comparison during movement time. Behav. Brain Res. 18:159-166.

Georgopoulos A.P, Kettner RE, Schwartz AB (1988) Primate motor cortex and free arm movements to visual targets on three-dimensional space. II. Coding of the direction of movement by a neuronal population.J Neurosci. 8:2928-2937.

Giszter, S.F, Mc Intyre J, Bizzi E (1989) Kinematic strategies and sensorimotor transformations in the wiping movements of frogs. Journal of Neurophysiology, 62: 750.

Godschalk M, Lemon R.N., Nijs H.G.T., Kuypers H.G.J.M. (1981) Behavior of neurons in monkey peri-arcuate and precentral cortex before and during visually guided arm and hand movements, Exp. Brain Res. 44:113-116.

Godschalk M, Lemon R.N., Kuypers H.G.J.M., Ronday H.K. (1984) Cortical afferents and efferents of monkey postarcuate area: and anatomical and electrophysiological study. Exp. Brain Res. 56:410-424.

Godschalk M, Lemon RN , Kuypers H.G.J.M., van der Steen J (1985) The involvement of monkey premotor cortex neurones in preparation of visually cued arm movements. Behav. Brain Res. 18:143-157.

Goldberg G (1985). Supplementary motor area structure and function: review and hypotheses. Behav. Brain Sci. 8:567-616.

Graham B. and Redman S. (1993) Dynamic behaviour of a model of the muscle stretch reflex. Neural Networks. Vol 6, 947-962.

Hamada I, Kubota K (1979) Monkey pyramidal tract neurons and changes of movement parameters in visual tracking. Brain Res. Bull. 4:249-257.

Henneman, (1977). Functional organization of motoneuron pools: The size principle. Proc, Int. Union Physiol. Sci. 12,50.

Henneman, E. (1957). Relation between size of neurons and their susceptibility to discharge. Science 126, 1345-1346.

Henneman, E,. and Olson, C. B. (1965). Relations between structure and function in the design of skeletal muscle. J. Neurophysiol. 28, 581-598.

Helms Tillery S.I, Flanders M, Soechting J.F. (1991) A coordinate system for the synthesis of visual and kinesthetic information. J. Neurosci. 11:770-778

Hepp-Raymond M.C., Diener R (1983). Neural coding of force and rate of force change in the precentral finger region of the monkey. Exp. Brain Res. (Supp) 7:315-326.

Hestenes, D (1994) Invariant body kinematics:II. Reaching and Neurogeometry. Neural Networks, Vol 7, No. 1. pp79-88.

Hocherman S, Wise SP (1991). Effects of hand movement path on motor cortical activity in awake, behaving rhesus monkeys. Exp. Brain Res. 83:285-302.

Hodgkin A. L. and Huxley A. F. (1952) Physiol (London). 117 500-544.

Hoffer, F. A., G. E. Loeb, N. Sugano, W. B. Marks, M. J. O Donovan and C. A. Pratt. (1987). Cat hindlimb motoneurons during locomotion. I I I . Functional segregation in sartorius. J. Neurophysiol. 57:554-562.

Hogan N. (1984) An organizing priciple for a class of voluntary movements. Journal of Neuroscience, 4(11), 2745-2754

Hogan N (1988). Planning and execution of multi-joint movements. Can J Physiol Pharmacol 66:508-517.

Hollerbach J.M (1982). Computers,Brains and the control of movement. Trnds Neurosci 5:189-192.

Hollerbach J.M , Atkeson C.G. (1987) Deducing planning variables from experimental arm trajectories: pitfalls and possibilities. Biol. Cybern. 56:279-292

Hollerbach J.M, Flash T (1982). Dynamic interactions between limb segments during planar arm movement. Biol. Cybern. 44: 67-77.

Houk J.C. and Rymer, W.Z.(1981). Neural control of muscle length and tension. In Handbook of physiology: The nervous system II.Bethesda, MD: American Physiological Society, pp.257-322.

Houk J.M, Atkeson C.G (1987). Deducing planning variables ftom experimental arm trajectories: pitfalls and possibilities. Biol. Cybern. 56:279-292.

Hultborn H. and Illert M, (1991) How is motor behaviour relected in the organization of spinal systems? In: Motor control concepts and issues. Edited by D.R. Humphrey and H.J. Freund. John Wiley & Sons Ltd.

Humphrey D.R, Schmidt E.M, Thompson W.D. (1970) Predicting measures of motor performance from multiple cortical spike trains. Science. 170:758-762.

Humphrey D.R. and Mitz A.R. (1989) Multiple, joint weighted representations of the arm with the precentral (MI) motor cortex of the macaque monkey. In: Neuroscience integrative functions: a symposium honoring Vernon B. Mountcastle. Baltimore.

Johnson P.B. (1992) Toward an understanding of the cerebral cortex and reaching movements: A review of recent approaches. In: Control of arm movement in space. Neurophysiological and computational approaches. (Eds) Caminiti R, Johnson P.B, Burnod Y. Springer-Verlag.

Kalaska J.K, (1988). The representation of arm movements in postcentral and parietal cortex. Can. J. Physiol. Pharamacol. 66: 455-463.

Kalaska J.F, Caminiti R. Georgopoulos A.P (1983). Cortical mechanisms related to the direction of two-dimensional arm movements: relations in parietal area 5 and comparison with motor cortex. Exp. Brain Res. 51:247-260.

Kalaska J.F, Cohen D.A.D. Hyde M.L, Prud'homme M (1989). A comparison of movement direction-related vs.load direction-related activity in primate motor cortex,using a two-dimensional reaching task. J. Neurosci. 9:2080-2102.

Kalaska J.F. (1992) What parameters of reaching are encoded by discharges of cortical cells. In: Motor control concepts and issues. Edited by D.R. Humphrey and H.J. Freund. John Wiley & Sons Ltd.

Keizer K, Kuypers H.G.I.M (1989). Distribution of corticospinal neurons with collaterals to the lower brain stem reticular formation in monkey (Macaca fascicularis). Exp. Brain Res. 74: 311-318.

Kernell D. (1979) Rythmic properties of motorneurons innervating muscle fibres of different speed in m.gastrocnemius medialis of the cat. Brain Research, 160, 159-162.

Kernell D, and Monster A.W, (1981) Threshold current for repetitive impulse firing in motorneurons innervating muscle fibers of different fatigue sensitivity in the cat. Brain Research, 229, 193-196.

Kernell D, (1983) Functional properties of spinal motorneurons and gradation of motor force. In J.E. Desmedt (Ed.), Motor control mechanisms in health and disease (pp. 213-226). New York: Raven Press.

Kernell D, (1984) The meaning of discharge rate: Excitation-to-frequency transduction as studied in spinal motorneurons. Archives Italiennes de Biologie, 122, 5-15

Kossev A.R, Robinson G.A, Enoka R.M. (1987) Fatigue induced changes in the threshold forces of recruitment and derecruitment for low threshold motor units of first dorsal interosseus. Soc. Neursci. Abstr. 13, 873.

Kurata K, Wise S.P (1988a). Premotor cortex of rhesus monkeys: set-related activity during two conditional motor tasks. Exp. Brain Res. 69: 327-343.

Lacquaniti F, Soechting J.F (1982). Coordination of arm and wrist motion during a reaching task. J. Neurosci. 8:2938-2947.

Lacquaniti F, Soechting J.F (1986). Responses of mono-and bi-articular muscles to load perturbations of the human arm. Exp. Brain Res. 65:135-144.

Lemon R.N, Mantel G.W.H, Muir R.B (1986). Cortiospinal facilitation of hand muscles during voluntary movement in the conscious monkey. J Physiol (Lond) 381:497-527.

Liddell, E.G.T, and C.S. Sherrington.(1925). Recruitment and some other factors of reflex inhibition. Proc. R. Soc. Lon. B. 97: 488-518.

Loeb G.E., W.F.Yee, C.A.Pratt, C.M.Chanaud, and F.JRichmond. (1987). Cross-correlation of E.M.G. reveals widespread synchronization of motor units during some slow movements in intact cats. J. Neurosci. Meth. 21:239-249.

Luria AR (1980). Higher cortical functions on man. Basic, New York.

Miles, T. S,. and Turker, K. S. (1986). Does reflex inhibition of motor units follow the "size principle "?. Exp. Brain Res. 62.443-445.

Morasso P.(1981) Spatial control of arm movements. Exp. Brain Res. 42: 223-227.

Murphey J.T., Kwan H.C., MacKay W.A, Wong, Y.C. (1982) Precentral unit activity correlated with angular components of a compund arm movement. Brain Res. 246: 141-145.

Nelson W.L (1983). Physical principle for economies of skilled movements. Biol Cybern 46: 135-147.

Nichols,T.R & Houk,J.C. (1976). Improvement in linearity and regulation of stiffness that results from actions of stretch reflex. Journal of Neurophysiology, 39, 119-142.

Norman M.P. (1991) Investigation of space-time neural networks with application to sensory-motor control in adaptive robotics. Msc Dissertation. University of Plymouth.

Okano K, Tanji J (1987). Neuronal activties in the primate motor fields of the agranular frontal cortex preceding visually triggered and self-paced movement. Exp. Brain Tes. 66:155-166.

Orgogozo J M, Larsen B (1979). Activation of the supplementary motor area during voluntary movement in man suggests it works as a supramotor area. Science 206:847-850.

Orin D.E, McGhee R.B, Vukobratovic M, Hartoch G. (1979). Kinematic and kinetic analysis of open-chain linkages utilizing Newton-Euler methods. Math Biosci 43: 107-130.

Passingham RE (1987). Two cortical systems for directing movement. In: Motor areas of the cerebral cortex, Ciba Foundation Symposium 132. Wiley, Chichester.

Piaget J. (1963) The origins of intelligence in children. New York. Norton.

Polit, A. and Bizzi, E.(1978). Processes contolling arm movements in monkeys. Science, 201, 1235-1237.

Rack P (1981). Limitations of somatosensory feedback in control of posture and movement. In: Brooks VB (ed) Handbook of physiology: the nervous system. II. Motor control. American Physiological Society, Bethesda.

Redman S.J, (1976) A quantitative approach to integrative function of dendrites. In R. Porter(Eds), International review of physiology--Neurophysiology II, Volume 10 (pp. 1-35). Baltimore MD: University Park Press.

Riehle A, Requin J (1989). Monkey primary motor and premotor cortex: single unit activity related to prior information about direction and extent of an intended movement. J. Neurophysiol. 61:534-549.

Rizzolatti G., Scandolara C., Matelli M., and Gentilucci M.(1981) Afferent properties of pericurate neurons in macaque monkeys. I Somatosensory responses. Beh. Brain Res. 2:125-146.

Roland P.E. (1987). Metabolic mapping of sensorimotor integration if the human brain. In: Motor Areas of the Cerebral cortex. John Wiley and Sons, Chichester.

Russell. C.J.D.B.Dunbar, D.S.Rushmer, J.M.Macpherson, and J.O.Phillips. (1982). Differential activity of innervation subcompartments of cat lateral gastrocnemius during natural movements. Soc. Neurosci. Abst. 8: 948.

Saltzman E (1979). Levels of sensorimotor representation. J. Math. Psychol. 20:91-163.

Sanes J.N. and Donoghue (1992) Organization and adaptability of muscle representation in primary motor cortex. In: Control of arm movement in space. Neurophysiological and computational approaches. (Eds) Caminiti R, Johnson P.B, Burnod Y. Springer-Verlag.

Sato K.C, Tanji J. (1989). Digit-muscle responses evoked from multiple intracortical foci in monkey precentral motor cortex. J. Neurophysiol. 62:959-970.

Schell G.R, Strict P.L (1984). The origin of thalamic inputs to the arcuate premotor and supplementary motor areas. J. Neurosci. 4:539-560.

Schmidt, E.M., Jost, R.G. and Davis, K.K.(1975). Reexamination of the force relationship of cortical cell discharge patterns with conditioned wrist movements. Brain Research, 83, 213-223.

Schwartz A.B, Kettner R.E, Georgopoulos A.P (1987). Relations between the amplitude of 2-dimensional arm movements and single cell discharge in primate motor cortex. Soc. Neurosci. Abs 13:244.

Sherrington, C.S (1910). "Flexion-reflex of the limb, crossed extension reflex, and reflex stepping and standing " J. Physiol, vol. 40, pp 28-121.

Smith A.M. Hepp-Reymond M C. Wyss UR (1975). Relation of activity in precentral cortical neurons to force and rate of force change during isometric contractions of finger muscles. Exp. Brain Res. 23:315-332.

Soechting J.F, Lacquaniti F. (1981). Invariant characteristics of a pointing movement in man. J. Neurosci. 1: 710-720.

Soechting J.F., Ross B (1984). Psychophysical determination of coordinate representation of human arm orientation. Neuroscience. 13:595-604.

Soechting J.F.,Flanders M. (1989b) Sensorimotor representations for pointing to targets in three-dimensional space. J. Neurophysiol. 62:595-608

Stuart, D.G,. and Enoka, R.M.(1983). Motoneurons, motor units and the size principle. In The Clinical Neurosciences. Neurobiology (ed. R . N Rosenburg). Churchill Livingstone, New York, pp.471-517.

Stuart, D.G,.Enoka, R.M,. and Gordon, D.A.(1988). Implications of cat motor-unit studies for the study of human muscle fatigue. ENA Abstr. 11,148.

Tanji J, Kurata K. (1985). Contrasting neuronal activity in supplementary and precentral motor cortex of monkeys. I. Responses to instructions determining motor responses to forthcoming signals of different modalities. J. Neurophysiol. 53:129-141.

Thach, W. T. (1978) Correlation of neuronal discharge with pattern and force of muscular activity, joint position, and direction of intended next movement in motor cortex and cerebellum. J.Neurophysiol. 41: 654-676.

Uno, Y, Kawato, M., and Suzuki, R. (1989). Formation and control of optimal trajectory in human arm movement-minimum torque change model. Biological Cybernetics. 61, 89-101.

Vogt O., and Vogt C.(1919) Ergebnisse unserer Hirnforschung. J. Psychol Neurol 25:277-462.

Von Bonin G, and Bailey P.(1947) The neocortex of the Macaca mulatta. University of Illinois Press, Urbana.

Wada, Y and Kawato,M (1993) A neural network model for arm trajectory formation using forward and inverse dynamics models. Neural Networks, Vol 6, 919-932.

Weeks. O. I. and A.W English (1985). Compartmentalization of the cat lateral gastrocnemius motor nucleus. J Comp. Neural. 255:255-267.

Wise S.P.(1984) The primate premotor cortex: past, present and preparatory. Ann Rev. Neurosci 8:1-19

Wise S.P., di Pellergrino G. (1991) Primate frontal cortex: visuospatial vs. visuomotor activity. Soc. Neurosci. Abs. 17:1112

Wise S.P., Alexander G.E., Altman J.S., Brooks V.B, Freund H.J., Fromm C.J., Humphrey D.R. Sasaki K., Strick P.L., Tanji J. and Weisendanger (1991) Group report: What are the specific functions of the different motor areas? In: Motor control concepts and issues. Edited by D.R. Humphrey and H.J. Freund. John Wiley & Sons Ltd.

Woolsey, C. N. (1952). Patterns of localization in sensory and motor areas of the cerebral cortex. In The Biology of Mental Health and Disease. New York: Milbank Memorial Fund Hoeber.

Woolsey C.N, Settlage P.H, Meyer D.R, Sencer W, Hamuy T.P, Travis A.M (1952). Patterns of localization in precentral and "supplementary" motor areas and their relation to the concept of a premotor area. Res Publ Assoc Res Nerv Ment Dis 30:238-264.

Zajac, F. E,. and Faden, J,S.(1985). Relationship among recruitment order, axonal conduction velocity, and muscle unit properties of type-identified motor units in cat plantaris muscle, J. Neurophysiol, 53,1323-1344.

Appendix A

The appendix contains sample code demonstrating the INSIGHT source code files, that were used during motor pool simulations in chapter 5. The following sample files are included:-

1. Synchronous neuron simulation code.
2. Leaky Integrator simulation code
3. Sample INSIGHT simulation control file.

1. Synchrous neuron simulation code.

```
rem ****************************************************************************
rem * Program:          SYNC2.INS
rem * Description:
rem *                   The synchronous neuron takes an input stream of pulses and calculates
rem *                   their input frequency over a period (Tbase). The longer the period the
rem *                   more accurate the calculation but the slower the neuron is to react
rem *                   to change. The value of the incoming pulses are added together for
rem *                   each time step if the input is above some threshold (THRESHi) then
rem *                   the pulse is used in the current frequency calculation. The output
rem *                   frequency is the same as the input frequency multiplied by some factor
rem *                   Rf, as long as the input frequency is within the frequency limits
rem *                   FREQmin and FREQmax. The output frequency is updated in accordance
rem *                   with the UPDATErate parameter. When look at input pulses it is
rem *                   possible to assume some "persistance" of impulses and look back
rem *                   several impulses before average, the number of steps to look
rem *                   back is determined by DECAYwindow parameter. The impulses tail of
rem *                   exponentially from the current impulse, these are then added together
rem *                   and thresholded. This allows some degree of synchrony between
rem *                   multiple inputs to be explored.
rem *
rem *                   FREQmin             -       Minimum Input Frequency
rem *                   FREQmax             -       Maximum Input Frequency
rem *                   Rf              -   Refractory Factor (Scales Input/Output)
rem *                   UPDATErate      -   Output Reaction Rate
rem *                   UPDATEcount     -   Output Reaction variable
rem *                   THRESHi             -       Thresholding Variable
rem *                   DECAYwindow     -   Number of steps to average input over
rem *                   Tbase           -   Number of steps to calculate frequency over
rem *                                       this affectively determines response speed
rem *
rem *                   CONNECTvalue    -   Value of input
rem *                   CONNECTdelay    -   How long it takes to get there
rem *                   CONNECTmin      -   Minimum connection firing frequency
rem *                   CONNECTmax      -   Maximum connection firing frequency
rem *
rem *                   connect_source[] -  Incoming pulse log ((t - Tbase) -> t)
rem *                   connect-from[]  -   Outgoing pulse log ((t - Tbase) -> t)
rem *                   Fire_Log[]      -   Incoming future pulse log (t -> (t+ Tbase))
rem,*
rem * Written:          Mark Norman (11th July 1993)
rem ****************************************************************************

iter = i


rem *** Calculate The Input Frequency FREQin

gosub CalcInFrequency

rem *** Calculate The Output Frequency FREQout
rem *** Scale the output with the Rf parameter

FREQout = FREQin * Rf

rem gosub ChangeFreqRate
```

```
gosub CalcCurFrequency

FireNode = 0
rem if FREQin < FREQmin or FREQin > FREQmax then goto clear_output


if FREQcurrent <= FREQout then FireNode = 1

gosub SendPulseOut

finish:


print "[";n;"] Iter: "; iter; " Input: ";FREQin, "Output: "; FREQout,"Current: ";FREQcurrent; " - ";
UPDATEcount
if FireNode = 1 then print "*** FIRED ***"
if FireNode = 0 then print "*** DID NOT FIRE ***"

if FireNode = 1 then write out, "5,";FREQcurrent;",";
if FireNode = 0 then write out, "0,";FREQcurrent;",";
if FireNode = 1 then total = total + 1
endchain

rem -----------------------------------------------------------------------------------------

rem *** change frequency rate
rem *** FREQrate determines how many step go by before
rem *** The output frequency is updated


ChangeFreqRate:


UPDATEcount = UPDATEcount + 1
if UPDATEcount < UPDATErate then FREQout = oldFREQ
if UPDATEcount < UPDATErate then return

rem *** update frequency ***

oldFREQ = FREQout
UPDATEcount = 0
return


rem *** Clear the relevant slot in the output window

clear_output:

pos = (iter - (trunc(iter/Tbase)*Tbase)) + 1
if pos > Tbase then pos = 1
connect_from[pos] = 0

goto finish

rem *** Calculates FREQin - which is the input frequency
rem *** Threshold THRESHi determines how many input units are required to count as
rem *** and input pulse
```

```
CalcInFrequency:

count = 0

rem *** Get any due impulses out of the FireLog

pos = (iter - (trunc(iter/Tbase)*Tbase)) + 1

rem print "Fire_Log[n]: ", Fire_Log[FireCount], "Fire Count: ", FireCount
connect_source[pos] = Fire_Log[FireCount]

if Fire_Log[FireCount] > 0 then last = iter

Fire_Log[FireCount] = 0
FireCount = FireCount + 1
if FireCount > Tbase then FireCount = 1

rem *** Add up all the impulses

step = 0
fire = 0
factor = 0
FREQin = 0

if iter - last > Tbase then return


for f = 1 to Tbase

        fire = connect_source[f]
        rem if sync = 1 and fire = 2 then fire = 0
        goto skip_window
again:

        x = f + (step+1)
        if x < 1 then x = Tbase + x

        rem factor = (DECAYwindow - step)/DECAYwindow
        rem fire = fire + (connect_source[x] * factor)


        factor = connect_source[x] / (DECAYwindow - step)

skip_zero:
        fire = fire + (connect_source[x] - factor)
        step = step + 1

        rem print "[";f;"] Step: ", step, "Factor: ", factor, "Fire: ", fire, "x: ", connect_source[x], " T:
", THRESHi
        if step < DECAYwindow then goto again

skip_window:

        if fire >= THRESHi then count = count + 1
        step = 0
        fire = 0

next
```

```
rem *** Divide through by the time base to get the frequency

rem base = Tbase
rem if iter < Tbase then base = iter+1
FREQin = count /Tbase

return

rem *** Calculates FREQcurrent - which is the previous frequency rate plus one extra
rem *** output pulse to determine if the node needs to fire again


CalcCurFrequency:

pos = (iter - (trunc(iter/Tbase)*Tbase)) + 1
connect_from[pos] = 0

count = 0

for f = 1 to Tbase
        fire = connect_from[f]
        if fire >= 1 then count = count + fire
next

rem base = Tbase
rem if iter < Tbase then base = iter+1
FREQcurrent = ((count+1) /Tbase)

rem pos = ((iter-1) - (trunc((iter-1)/Tbase)*Tbase)) + 1
rem connect_from[pos] = 0


return




SendPulseOut:


rem *** Calculate position of pulse in output window

pos = (iter - (trunc(iter/Tbase)*Tbase)) + 1

rem *** Adjust output pulse window

connect_from[pos] = 0
if fireNode=1 then connect_from[pos] = 1

result = saveneuron(stream1, n)

if result = -1 then print "Neuron Save File Error."

for j = 1 to connect_count
        con = connect[j]
        if con > -1 then gosub SendPulseExternOut
```

```
next

return

SendPulseExternOut:

result = loadconnect(stream2, con)
if result = -1 then print "Connection File Read Error."

rem print "Connection To: ", destination

result = loadneuron(stream1, destination)

if result = -1 then print "Neuron File Read Error."

rem *** Calculate pos in target nodes input window

if CONNECTdelay > Tbase then print "ERROR: CONNECTdelay param greater than Tbase",
CONNECTdelay, "Node: ", destination, "Tbase: ", Tbase
pos = FireCount + CONNECTdelay

rem *** Adjust for end of output window


if pos > Tbase then pos = pos - Tbase

rem *** Put pulse in target nodes fire log

rem if FREQout < CONNECTmin then FireNode = 0
rem if FREQout > CONNECTmax then FireNode = 0

if FireNode = 1 and Fire_Log[pos] = -1 then Fire_Log[pos] = 0;
if FireNode = 1 then Fire_Log[pos] = Fire_Log[pos] + CONNECTvalue

result = saveneuron(stream1, destination)
result = loadneuron(stream1, n)
return
```

2. Leak Integrator simulation code

```
rem ******************************************************************
rem * Name:             Leake.ins
rem * Description:      Standard Leaky Integrator With Modifications
rem *                   to handle external connections.
rem * Date:             26th Febuary 1993
rem ******************************************************************

rem *** Check if any pulses have arrived at the membrane

rem print "Node: ", n, "fire count: ", fire_count

for j = 1 to fire_count
        fire = fire_log[j]
        if fire > -1 then gosub find_value
next

rem *** Apply Decay ***

membrane = membrane - (membrane / decay)

current = last_fired+refractory
rem *** Check To See If Node Has Fired ***

hasfired = 0
if membrane>threshold and current<=i then gosub send_output

print "Iter: ",i,"Node: ", nodenum, "Membrane: ", membrane, "Last Fired: ", last_fired

write out, membrane;",";hasfired;
if last_fired=i and i > 0 then total=total+1
if n < number_neurons then write out, ",";

rem print "Node: ", n, "Fire Log[1]: ", fire_log[1]

endchain


rem *** Update All The Nodes That Have  Received An Input ***

send_output:

rem *** Send Output Along All Connections ***

hasfired = 1

rem print "*** Fired *** - Node ", n, "Number Connections: ", connect_count

result = saveneuron(stream1, n)

if result = -1 then print "Neuron Save File Error."

for j = 1 to connect_count
        con = connect[j]
        if con > -1 then gosub slot
next
```

```
return

rem *** update the target connections slot ***

slot:

extern = -1
result = loadconnect(stream2, con)
if result = -1 then print "Connection File Read Error."
if result = -2 then gosub open_extern
result = loadneuron(stream1, destination)
if result = -1 then print "Neuron File Read Error."

rem *** find next free slot ***


pos = -1

for h = 1 to fire_count
        fire = fire_log[h]
        if fire < 0 then pos=h
next


if pos=-1 then gosub new_slot

rem print "pos: ", pos, "fire_count: ", fire_count

rem *** set the current  slot position to the current iteration ***

fire_log[pos] = i

if extern=1 then connect_source[pos] = old_project
if extern=-1 then connect_source[pos] = -1

connect_from[pos] = con

rem print "will come off on: ", (i+delay), "For Node: ", destination, "pos: ", pos

result = saveneuron(stream1, destination)

if result = -1 then print "Neuron File Save Error."


rem *** reload node information ***

if extern = 1 then gosub close_extern

result = loadneuron(stream1, n)
if result = -1 then print "Error Reloading Node Data."
last_fired = i
return


new_slot:
rem print "(before) fire: ", fire_count
pos=fire_count+1
```

```
fire_count=fire_count+1
rem print "(after) fire: ", fire_count
return

find_value:

rem *** membrane = membrane + (MRC - membrane) x conductivity ***

iter = delay + fire_log[j]
rem print "Due To Fire On: ", iter
if iter > i then return

extern = -1
con = connect_from[j]
source = connect_source[j]
if source > -1 then gosub change_network
result = loadconnect(stream2, con)
if result = -1 then print "Error Loading Connect Data."
rem if result = -2 then print "*** External Connection ***"
if extern = 1 then gosub restore_network
value = (mrc - membrane) * conductivity
mem = membrane
membrane = membrane + value
fire_log[j] = -1
rem print "--------------------------------------------------------------------------------"
rem print "delay: ", delay, "mrc: ", mrc, "conductivity: ", conductivity, "old membrane: ", mem
rem print "Node: ", n, "[";j;"] From: ", source, " Value: ", value
rem print "--------------------------------------------------------------------------------"

return


open_extern:
old_project = findproject(proj,sourcenet$)
if old_project = -1 then print "Project File Error",proj
new_project = findproject(proj,destnet$)
if new_project = -1 then print "Project File Error",proj
close(stream1)
close(stream2)
loadproject(proj,new_project)
extern = 1
return

close_extern:

close(stream1)
close(stream2)
loadproject(proj,old_project)
extern=-1
return

change_network:

close(stream1)
close(stream2)
loadproject(proj,source)
extern=1
return
```

```
restore_network:

close(stream1)
close(stream2)
loadproject(proj,k)
extern = -1
return
```

3. Sample INSIGHT simulation control file.

```
rem ***********************************************************************
rem * Name:              Main4.ins
rem * Description:       Standard main module template with modifications
rem *                    to handle project files, and to have a dual output
rem *                    for motor pool simulations.
rem * Date:              28th Febuary 1993
rem ***********************************************************************

proj = open("c:\source\insight\mp.prj", "rb+", 4)
result = initproject(proj)
if result = -1 then print "Error Initialising Project File"

stream1 = 1
stream2 = 2
stream3 = open("c:\source\insight\output1.txt","w+",3)
stream5 = open("c:\source\insight\output2.txt","w+",5)
stream6 = open("c:\source\insight\output3.txt","w+",6)

cls

iterations = 700
number_neurons = 28
number_connections = 5
square = iterations/20

sx = 40-(((number_neurons/2)*2)+2)
sy = 6

gotoxy(31,1):print "Leaky Integrator"
gotoxy(31,2):print "ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ"



result = 0

write stream3, " ,node 0, fire0, node 1, fire 1, node 2, fire 2, node 3, fire 3"
write stream5, " ,node 0, fire0, node 1, fire 1, node 2, fire 2, node 3, fire 3"
write stream6, "Total Output"

for k = 1 to projcount-1
        loadproject(proj,k)
        resetneuron(stream1)
        resetconnect(stream2)
        close(stream1)
        close(stream2)
next

for i = 0 to iterations
        gtotal = 0

        for k = 1 to projcount-1
                if k=1 then out=stream3
                if k=2 then out=stream5

                write out, i;",";
```

```
                print "-----------------------------------------------------"
                print "Project File: ",project$[k]
                print "-----------------------------------------------------"
                loadproject(proj,k)

                n = 0
                total = 0

                main_loop:

                        rem print "Iteration: ", i, "Node: ", n, "Total: ", iterations
                        result = loadneuron(stream1,n)
                        if result = -1 then goto output
                        if nodenum = -1 then goto skip_code
                        chain "leake.ins"
                        result = saveneuron(stream1,n)
                        if result = -1 then print "File Save Error"
                skip_code:
                        n = n + 1
                        goto main_loop

                output:

                write out, total

                if i>1 and i<10 and k = 1 then gosub external

                close(stream1)
                close(stream2)

                gtotal = gtotal + total
                print "Total Fired: ", total
        next

        write stream6, "Total", ",", gtotal
        print "Grand Total: ",gtotal
next

gotoxy(25,22):print "**** SIMULATION END ****"
close(stream1)
close(stream2)
close(stream3)
close(proj)
close(stream5)
close(stream6)
end

external:
rem print "*** External Input ****"
result = loadneuron(stream1, 16)
if result = -1 then print "File Load Error"
membrane = membrane + 2.0
result = saveneuron(stream1, 16)
if result = -1 then print "File Save Error"

return
```

Appendix B

This appendix contains the detailed data for experiments described in chapter 5. The following experiments are covered:-

- Two tier Leaky Integrator motor pool
- Single tier Leaky Integrator motor pool
- Synchronous neuron motor pool

Two tier Leak Integrator mtor pool.

Network Topology

Each network layer has a connectional topology similar to that shown in Figure 5.4.
The topology of each layer is identical with the following exceptions:-
Output from the second ouput pool has been scaled to be larger by a factor of 2 when compared with the first.
The decay factors in the second pool have been set to 50 (decays twice as quickly).
Intra layer connections are made from all nodes in the first layer to the root node on the second layer. The parameters are as follows:- MRC = 7.0 and CONDUCTIVITY = 0.5.

Simulations

Simulations were carried out using the INSIGHT TOO software with output calculated using EXCEL 4.0. The following files were used:-

MAIN4.INS	- Controlling simulation code
LEAKE.INS	- leaky integrator code with extension to handle external inputs/outputs.
MP4.INF MP4.ICF	- Layer 1 Node data file & Connect File
MP5.INF MP5.ICF	- Layer 2 Node data file & Connect File

[1] The external stimulas input was 1.0 for iterations 1 -> 10. MP9.XLS -> MP11.XLS
[2] The external stimulas input was 0.5 for iterations 1 -> 10. MP12.XLS ->MP14.XLS
[3] The external stimulas input was 1.5 for iterations 1 -> 10. MP15.XLS ->MP17.XLS
[4] The external stimulas input was 2.0 for iterations 1 -> 10. MP18.XLS ->MP20.XLS
[5] [1] -> [4] combined into a single chart MP21.XLS.

Node Data

Layer 1

```
Node: 0, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 1, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 2, Layer: 0, Params: MEMBRANE = 1.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 3.70, DELAY = 10.00, LAST_FIRED = 0
Node: 3, Layer: 0, Params: MEMBRANE = 1.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 3.70, DELAY = 10.00, LAST_FIRED = 0
Node: 4, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 3.95, DELAY = 10.00, LAST_FIRED = 0
Node: 5, Layer: 0, Params: MEMBRANE = 1.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 3.95, DELAY = 10.00, LAST_FIRED = 0
Node: 6, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 3.95, DELAY = 10.00, LAST_FIRED = 0
Node: 7, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 8, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 9, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 10, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 11, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 12, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 13, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 14, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 15, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 16, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 150.00,
                           THRESHOLD = 3.50, DELAY = 10.00, LAST_FIRED = 0
Node: 17, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 18, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 19, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 20, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 21, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 22, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 23, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 24, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 25, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 26, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
```

```
Node: 27, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 100.00,
                            THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
```

Layer 2

```
Node: 0, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 1, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 2, Layer: 0, Params: MEMBRANE = 1.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 3.70, DELAY = 10.00, LAST_FIRED = 0
Node: 3, Layer: 0, Params: MEMBRANE = 1.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 3.70, DELAY = 10.00, LAST_FIRED = 0
Node: 4, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 3.95, DELAY = 10.00, LAST_FIRED = 0
Node: 5, Layer: 0, Params: MEMBRANE = 1.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 3.95, DELAY = 10.00, LAST_FIRED = 0
Node: 6, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 3.95, DELAY = 10.00, LAST_FIRED = 0
Node: 7, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 8, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 9, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 10, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.25, DELAY = 10.00, LAST_FIRED = 0
Node: 11, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 12, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 13, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 14, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 15, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 4.60, DELAY = 10.00, LAST_FIRED = 0
Node: 16, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 6.00, DELAY = 5.00, LAST_FIRED = 0
Node: 17, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 18, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 19, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 20, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 21, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 22, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 23, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 24, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 25, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
Node: 26, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                           THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
```

```
Node: 27, Layer: 0, Params: MEMBRANE = 3.00, REFRACTORY = 10.00, DECAY = 50.00,
                            THRESHOLD = 5.05, DELAY = 10.00, LAST_FIRED = 0
```

Connection Data

Layer 1

```
Source Node: 1, Destination Node: 0,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 1, Destination Node: 22,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 17, Destination Node: 22,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 17, Destination Node: 23,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 2, Destination Node: 5,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 18, Destination Node: 23,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 2, Destination Node: 4,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 3, Destination Node: 6,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 3, Destination Node: 5,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 18, Destination Node: 24,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 4, Destination Node: 10,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 19, Destination Node: 24,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 4, Destination Node: 9,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 5, Destination Node: 8,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 19, Destination Node: 25,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 5, Destination Node: 9,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 6, Destination Node: 8,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
```

```
Source Node: 6, Destination Node: 7,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 20, Destination Node: 25,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 7, Destination Node: 11,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 7, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 20, Destination Node: 26,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 8, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 8, Destination Node: 13,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 21, Destination Node: 26,
        Source Network: CURRENT, Destination Network:
        CURRENT: Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 21, Destination Node: 27,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 9, Destination Node: 13,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 9, Destination Node: 14,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 3, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 10, Destination Node: 14,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 10, Destination Node: 15,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 2, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 11, Destination Node: 21,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 4, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 11, Destination Node: 20,
        Source Network: CURRENT,
        Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
```

```
Source Node: 12, Destination Node: 20,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 12, Destination Node: 19,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 5, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 13, Destination Node: 19,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 13, Destination Node: 18,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 6, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 14, Destination Node: 18,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 10, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 14, Destination Node: 17,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 15, Destination Node: 17,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 9, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 15, Destination Node: 1,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 16, Destination Node: 2,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 16, Destination Node: 3,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 8, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 7, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 15, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
```

```
Source Node: 14, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 13, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 12, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 11, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 1, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 17, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 18, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 19, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 20, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 21, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 0, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 22, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 23, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 24, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 25, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
```

```
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF: Params:
        MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 26, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 27, Destination Node: 16,
        Source Network: C:\SOURCE\INSIGHT\MP4.INF,
        Destination Network: C:\SOURCE\INSIGHT\MP5.INF:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 1, Destination Node: 0,
        Source Network: CURRENT,
        Destination Network: CURRENT:
        Params: MRC = 8.0, CONDUCTIVITY = 0.7
Source Node: 28, Destination Node: 36,
        Source Network: CURRENT,
        Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 28, Destination Node: 35,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
```

Layer 2

```
Source Node: 1, Destination Node: 0,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 1, Destination Node: 22,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 17, Destination Node: 22,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 17, Destination Node: 23,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 2, Destination Node: 5,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 18, Destination Node: 23,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 2, Destination Node: 4,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 3, Destination Node: 6,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 3, Destination Node: 5,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 18, Destination Node: 24,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 4, Destination Node: 10,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
```

```
Source Node: 19, Destination Node: 24,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 4, Destination Node: 9,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 5, Destination Node: 8,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 19, Destination Node: 25,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 5, Destination Node: 9,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 6, Destination Node: 8,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 6, Destination Node: 7,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 20, Destination Node: 25,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 7, Destination Node: 11,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 7, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 20, Destination Node: 26,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 8, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 8, Destination Node: 13,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 21, Destination Node: 26,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 21, Destination Node: 27,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 9, Destination Node: 13,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 9, Destination Node: 14,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 7.00
Source Node: 10, Destination Node: 14,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 10, Destination Node: 15,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 11, Destination Node: 21,
```

```
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 11, Destination Node: 20,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 12, Destination Node: 20,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 12, Destination Node: 19,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 13, Destination Node: 19,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 13, Destination Node: 18,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 14, Destination Node: 18,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 14, Destination Node: 17,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 6.00
Source Node: 15, Destination Node: 17,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 15, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.50, MRC = 7.00
Source Node: 16, Destination Node: 2,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 16, Destination Node: 3,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 28, Destination Node: 36,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.30, MRC = 6.00
Source Node: 28, Destination Node: 35,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: CONDUCTIVITY = 0.30, MRC = 6.00
```

Single Tier Leaky Integrator Motor Pool

Network topology

The network has a connection topology similar to that shown in Figure 5.4. but a considerably larger pool size (65 Nodes). Within this network all the nodes have the same properties and the connection parameters were varied.

Simulations

Simulations were carried out using the INSIGHT TOO software with output calculated using EXCEL 4.0. The following files were used:-

MAIN4.INS - Controlling simulation code
LEAK1.INS - Same as LEAKE.INS but with modification to implement the MIN & MAX variables to put extents on the membrane range, this has the advantage of not allowing nodes to drop beneath a pre-defined level which allows the response to an input to be more readily predictable. The upper limit variable is not particularly usefull since the equations impose an upper limit anyway. This is a way of reducing the limit without affecting that particular mechanism.
MP12.INF - Nodedata file & connect file after connection
changes
MP12.ICF
MP2.PRJ - Project definition file.

External Inputs:-

[1] The external stimulas input was 0.5 for iterations 1 -> 10. MP27.XLS
[2] The external stimulas input was 1.0 for iterations 1 -> 10. MP26.XLS
[3] The external stimulas input was 1.5 for iterations 1 -> 10. MP29.XLS
[4] The external stimulas input was 2.0 for iterations 1 -> 10. MP28.XLS

Nodes

Node: 0, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 1, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 2, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 3, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 4, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 5, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
DELAY = 10, DECAY = 150, LAST_FIRED=0

```
Node: 6, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
                           DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 7, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
                           DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 8, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
                           DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 9, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10,
                           DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 10, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 11, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 12, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 13, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 14, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 15, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 16, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 17, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 18, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 19, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 20, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 21, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 22, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 23, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 24, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 25, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 26, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 27, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 28, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 29, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 30, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 31, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 32, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 33, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 34, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
```

```
Node: 35, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 36, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 37, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 38, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 39, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 40, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 41, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 42, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 43, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 44, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 45, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 46, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 47, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 48, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 49, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 50, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 51, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 52, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 53, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 54, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 55, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 56, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 57, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 58, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 59, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 60, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 61, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 62, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 63, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                           , DELAY = 10, DECAY = 150, LAST_FIRED=0
```

```
Node: 64, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                              , DELAY = 10, DECAY = 150, LAST_FIRED=0
Node: 65, Layer: 0, Params: MEMBRANE = 4.0, THRESHOLD = 6.0, REFRACTORY = 10
                              , DELAY = 10, DECAY = 150, LAST_FIRED=0
```

Connections

```
Source Node: 0, Destination Node: 1,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 1.0
Source Node: 0, Destination Node: 2,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 1, Destination Node: 3,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 1, Destination Node: 4,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 2, Destination Node: 4,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 2, Destination Node: 5,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 3, Destination Node: 6,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.8
Source Node: 3, Destination Node: 7,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 4, Destination Node: 7,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 4, Destination Node: 8,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 5, Destination Node: 8,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 5, Destination Node: 9,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.9
Source Node: 6, Destination Node: 10,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 6, Destination Node: 11,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 7, Destination Node: 11,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 7, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 8, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
```

```
          Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 8, Destination Node: 13,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 9, Destination Node: 13,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 9, Destination Node: 14,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.7
Source Node: 10, Destination Node: 15,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 10, Destination Node: 16,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 11, Destination Node: 16,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 11, Destination Node: 17,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 12, Destination Node: 17,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 12, Destination Node: 18,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 13, Destination Node: 18,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 13, Destination Node: 19,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 14, Destination Node: 19,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 14, Destination Node: 20,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.6
Source Node: 15, Destination Node: 21,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 15, Destination Node: 22,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 16, Destination Node: 22,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 16, Destination Node: 23,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 17, Destination Node: 23,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 17, Destination Node: 24,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.5
```

```
Source Node: 18, Destination Node: 24,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 18, Destination Node: 25,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 19, Destination Node: 25,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 19, Destination Node: 26,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 20, Destination Node: 26,
        Source Network: CURRENT, Destination Network:
        CURRENT: Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 20, Destination Node: 27,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.5
Source Node: 21, Destination Node: 28,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 21, Destination Node: 29,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 22, Destination Node: 29,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 22, Destination Node: 31,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 23, Destination Node: 31,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 23, Destination Node: 32,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 24, Destination Node: 32,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 24, Destination Node: 33,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 25, Destination Node: 33,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 25, Destination Node: 30,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 26, Destination Node: 30,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 26, Destination Node: 34,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 27, Destination Node: 34,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 27, Destination Node: 35,
```

```
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.4
Source Node: 28, Destination Node: 36,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 28, Destination Node: 37,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 29, Destination Node: 37,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 29, Destination Node: 38,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 31, Destination Node: 38,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 31, Destination Node: 39,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 32, Destination Node: 39,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 32, Destination Node: 40,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 33, Destination Node: 40,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 33, Destination Node: 41,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 30, Destination Node: 41,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 30, Destination Node: 42,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 34, Destination Node: 42,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 34, Destination Node: 43,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 35, Destination Node: 43,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 35, Destination Node: 44,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.3
Source Node: 36, Destination Node: 45,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 36, Destination Node: 46,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 37, Destination Node: 46,
          Source Network: CURRENT, Destination Network: CURRENT:
```

```
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 37, Destination Node: 47,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 38, Destination Node: 47,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 38, Destination Node: 48,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 39, Destination Node: 48,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 39, Destination Node: 49,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 40, Destination Node: 49,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 40, Destination Node: 50,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 41, Destination Node: 50,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 41, Destination Node: 51,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 42, Destination Node: 51,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 42, Destination Node: 52,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 43, Destination Node: 52,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 43, Destination Node: 53,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 44, Destination Node: 53,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 44, Destination Node: 54,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.2
Source Node: 45, Destination Node: 55,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 45, Destination Node: 56,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 46, Destination Node: 56,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 46, Destination Node: 57,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 0.1
```

```
Source Node: 47, Destination Node: 57,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 47, Destination Node: 58,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 48, Destination Node: 58,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 48, Destination Node: 59,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 49, Destination Node: 59,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 49, Destination Node: 60,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 50, Destination Node: 60,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 50, Destination Node: 61,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 51, Destination Node: 61,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 51, Destination Node: 62,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 52, Destination Node: 62,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 52, Destination Node: 63,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 53, Destination Node: 63,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 53, Destination Node: 64,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 54, Destination Node: 64,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 54, Destination Node: 65,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 0.1
Source Node: 0, Destination Node: 1,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
```

```
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
Source Node: 0, Destination Node: 1,
          Source Network: CURRENT, Destination Network: CURRENT:
          Params: MRC = 6.5, CONDUCTIVITY = 1
```

Synchronous neuron motor pool

Network topology

Similar pyramidal topology to that shown in Figure 5.4 but with only 14 nodes.

Simulations

Simulations were carried out using the INSIGHT TOO software with output calculated using EXCEL 4.0. The following files were used:-

MAIN8.INS - Controlling simulation code
SYNC2.INS - Synchronous Neuron Code

SYNC2.INF - Node data file & Connect File
SYNC2.ICF

External Input

[1] External stimulus level was 100% for iterations 65 -> 96. MPS1.XLS
[2] External stimulus level was 100% for iterations 51 -> 105. MPS2.XLS
[3] External stimulus level was 100% for iterations 51 -> 110. MPS3.XLS
[4] External stimulus level was 50% for iterations 51 -> 110. MPS4.XLS
[5] External stimulus level was 25% for iterations 51 -> 110. MPS5.XLS
[6] External stimulus level was 33% for iterations 51 -> 110. MPS6.XLS
[7] Combined output chart. MPS7.XLS
[8] External stimulus level was 75% for iterations 51 -> 110. MPS8.XLS

Network data was smoothed using a exponential smooth function (damping factor 0.9) and moving average (window = 5).

Nodes

Node: 0, Layer: 0, Params: RF = 1.0, FREQMIN = 0.00, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 1, Layer: 0, Params: RF = 1.0, FREQMIN = 0.10, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 2, Layer: 0, Params: RF = 1.0, FREQMIN = 0.20, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 3, Layer: 0, Params: RF = 1.0, FREQMIN = 0.10, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 4, Layer: 0, Params: RF = 1.0, FREQMIN = 0.20, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 5, Layer: 0, Params: RF = 1.0, FREQMIN = 0.20, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 6, Layer: 0, Params: RF = 1.0, FREQMIN = 0.30, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0

Node: 7, Layer: 0, Params: RF = 1.0, FREQMIN = 0.30, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 8, Layer: 0, Params: RF = 1.0, FREQMIN = 0.30, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 9, Layer: 0, Params: RF = 1.0, FREQMIN = 0.30, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 10, Layer: 0, Params: RF = 1.0, FREQMIN = 0.40, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 11, Layer: 0, Params: RF = 1.0, FREQMIN = 0.4, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 12, Layer: 0, Params: RF = 1.0, FREQMIN = 0.40, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 13, Layer: 0, Params: RF = 1.0, FREQMIN = 0.40, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0
Node: 14, Layer: 0, Params: RF = 1.0, FREQMIN = 0.40, FREQMAX = 2.0, TBASE = 15,
THRESHI = 1, FIRECOUNT = 1, OLDFREQ = 0, UPDATERATE = 5,
UPDATECOUNT = 0, DECAYWINDOW = 5, LAST = 0

Connections

Source Node: 0, Destination Node: 1,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTDELAY = 10, CONNECTVALUE = 1
Source Node: 0, Destination Node: 3,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 1.0, CONNECTDELAY = 5
Source Node: 1, Destination Node: 2,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 1, CONNECTDELAY = 5
Source Node: 1, Destination Node: 4,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 2, Destination Node: 6,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 1, CONNECTDELAY = 5
Source Node: 2, Destination Node: 7,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 3, Destination Node: 4,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 3, Destination Node: 5,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 1.0, CONNECTDELAY = 5
Source Node: 4, Destination Node: 7,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 4, Destination Node: 8,
Source Network: CURRENT, Destination Network: CURRENT:
Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 5, Destination Node: 9,

```
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 1.0, CONNECTDELAY = 5
Source Node: 5, Destination Node: 8,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 6, Destination Node: 11,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 1, CONNECTDELAY = 5
Source Node: 6, Destination Node: 10,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 7, Destination Node: 10,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 7, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 8, Destination Node: 12,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 8, Destination Node: 13,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 9, Destination Node: 13,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 0.5, CONNECTDELAY = 5
Source Node: 9, Destination Node: 14,
        Source Network: CURRENT, Destination Network: CURRENT:
        Params: CONNECTVALUE = 1, CONNECTDELAY = 5
```