DX

171569

# Dynamic Modelling of Articulated Figures suitable for the purpose of Computer Animation

A thesis submitted to the
Council for National Academic Awards
in partial fulfilment of the requirements
for the Degree of Doctor of Philosophy

by
Nickos A Vasilonikolidakis

The Polytechnic of North London

August 1991

# Acknowlegements

I would like to thank my supervisor Dr G.J.Clapworthy for his encouragement, support and invaluable help for the last three years.

I also like to thank Dr Alexandrou for his help with the provision of various software that helped the presentation of this thesis.

Last but by no means least, I would like to thank Nick Glazzard and 5D Solutions for their help to produce the animation sequences.

I like to dedicate this work to my parents for their enormous support throughout my studies.

Nick Vasilonikolidakis

## Abstract

The animation of articulated bodies presents interest in the areas of biomechanics, sports, medicine and the entertainment industry. Traditional motion control methods for these bodies, such as kinematics and rotoscoping are either expensive to use or very laborious. The motion of articulated bodies is complex mostly because of their number of articulations and the diversity of possible motions.

This thesis investigates the possibility of using dynamic analysis in order to define the motion of articulated bodies. Dynamic analysis uses physical quantities such as forces, torques and accelerations, to calculate the motion of the body. The method used in this thesis is based upon the inverse Lagrangian dynamics formulation, which, given the accelerations, velocities and positions of each of the articulations of the body, finds the forces or torques that are necessary to generate such motion. Dynamic analysis offers the possibility of generating more realistic motion and also of automating the process of motion control. The Lagrangian formulation was used first in robotics and thus the necessary adaptations for using it in computer animation are presented.

An analytical method for the calculation of ground reaction forces is also derived, as these are the most important external forces in the case of humans and the other animals that are of special interest in computer animation. The application of dynamic analysis in bipedal walking is investigated. Two models of increasing complexity are discussed. The issue of motion specification for articulated bodies is also examined. A software environment, Solaris, is described which includes the facility of dynamic and kinematic motion control for articulated bodies. Finally, the advantages and problematics of dynamic analysis with respect to kinematics and other methods are discussed.

# Contents

# Chapter 1

# Prologue

## 1.1 Introduction

Computer graphics and animation have been used over the last fifteen years or so for many diverse applications, sometimes out of necessity and sometimes as a product of art. The necessity of computer graphics is not always apparent, especially when it is used to represent real objects. As the sophistication of computer graphics increases, so do the applications. It is still used by companies in the advertising sector to promote a high technological profile, but it is increasingly used for visualization of scientific data, where the assimilation of huge realms of numbers can only be done visually. In the case of visualization of scientific data, computer graphics has to exhibit speed and precision.

The animation of human beings and other articulated bodies is a typical example where applications range from the advertising and entertainment industry to a variety of scientific applications. Human animation is one of the most challenging and difficult areas in computer graphics - there is a number of reasons to make this claim, but the most important is that we try to create images of ourselves. During this task we face all the prejudices that surround the process of creating an icon identical to a human being. Another reason that makes this task challenging is the limited hardware and software resources that seem to be available in order to animate human figures. Our knowledge about the modelling and motion of the human body seems to be very limited and we are far from finding an analytic solution to the problem. Thus, only approximations to the real problem can be used. The modelling of the human body has attracted much research because of its potential in the entertainment industry. Special emphasis has been given to the modelling and expressions of the face by using sophisticated methods such as muscular models, and by using the elastic properties of the skin.

The process of animation is divided into three steps. First the objects are described in a suitable form, and in human animation that means that the shape of the human body has to be defined. Unfortunately the human body has irregular and variable shape with the extreme case of the head and the face which are probably the most difficult parts of the human body to model.

The second step is to define the motion of the body. This is extremely difficult since the human body has many articulations and links, and for every link at each frame, its position has to be defined explicitly. One of the major problems with the motion of human bodies is that of realism. It is difficult to move a human figure around without having some reference footage or alternatively without any knowledge of the mathematics involved. Finally the third step is rendering where colour and texture mapping are added to produce the final animation sequence.

In spite of these difficulties ~~there~~ is a lot of research has been undertaken in this area, and the main reason is the potential of applications of human animation. The main consumer for the time being is the entertainment industry, but interest is increasing in other areas such as ergonomics where man-environment interaction is crucial, in prosthetics where artificial joints are used to replace damaged ones, in accident prevention and finally it can be used in the computer simulation of new designs of robotics manipulators.

The purpose of this thesis is to concentrate on the second stage of the animation process, the motion control for articulated bodies. The term ***articulated bodies*** or ***figures*** needs to be defined with more clarity. For the purpose of this thesis an articulated body will be one that consists of rigid bodies (bodies with no elastic properties) called links, which are connected together by joints of multiple degrees of freedom. It is also assumed that the links of the body remain connected when the body starts to move. In that sense a large class of living creatures and objects can be classified as 'articulated'. Mammals, insects and other animals are articulated bodies, as are many robots and satellites. On the other hand, a pile of bricks stacked up one on top of another is not described as an articulated body.

The techniques presented in the thesis apply to this class of objects. There is, however, a special emphasis on human motion, as different bodies within this classification exhibit considerably different patterns of motion behaviour, and a need for specialisation is necessary.

## 1.2 Mathematical modelling of motion

One way of modelling the motion of articulated bodies is to forget about the real world where they exist and the semantics of their motion, and concentrate on the motion as an abstract notion. In this way the animator thinks in terms of individual movements of the various links which combined together can produce the desired coordinated motion. This technique has the advantage that the animator has fine control of the motion but unless he is very familiar and and has a deep knowledge about the type of motion, it is difficult to achieve the requested result. Another disadvantage of the method is the separation of the model with the real world which implies that the model has no physical properties and thus cannot interact with its virtual environment.

Although articulated bodies can have very divergent forms, their motion has a common denominator, its complexity. It is inconceivable for anyone to attempt a complete mathematical model for describing the motion of a human or a dog. Humans coordinate their motion through a complex system of nerves, sensors and electric discharges between brain cells. Human limbs have complicated elastic properties, non-uniform mass distribution and a sophisticated motion behaviour which, aside of the mathematical and physical factors, depends upon inherited customary habits that have been acquired over thousands of years of experience.

All these aspects of the problem, and many more, have to be included in an ideal model, but this is impossible because of our limited understanding of the motion. But even in the case we could understand human movement, the solution of its complex equations of motion would require enormous computational time even to the most powerful computers. Therefore, the only possibility of simulating human motion is to simplify the model by excluding some of the complex factors of the problem. One approach towards this direction is to exploit the findings from research in athropomorphic robots.

There are many methods in robotics that deal with the problem of motion control of articulated bodies and a number are described in subsequent chapters of the thesis. Although these methods try to simplify the modelling by considering only selected aspects of the problem, the mathematics is quite complicated and computationally intensive. The objective of these methods is to find the dynamic quantities (forces or accelerations) that will move a degree of freedom from one point in space to another, but they are not concerned about the style of motion, a factor that is of paramount importance in computer animation. Since most robots are rather simplistic in comparison to humans, these methods need major modifications before they can be applied to the motion control of humans or other animals.

The attractive property of dynamic analysis is the possibility of creating a more realistic model for motion within which the body can interact with its virtual environment and obey physical rules.

## 1.3 Outline of the thesis

This thesis examines the issues of modelling the motion of articulated bodies. The objective is to investigate the advantages and disadvantages of dynamic analysis and its application to human locomotion.

Chapter 2 gives a survey of the current methods in motion control for articulated bodies including traditional methods as well as those that are still at the research stage. Chapter 3 describes the derivation of inverse Lagrangian dynamics as applied to the motion control of articulated bodies. The calculation of the forces is based on recursive formulae to reduce the computational cost and thus speed up the method. Chapter 4 introduces the adaptations of the Lagrangian dynamics that make the method suitable for use in computer animation. Chapters 5 and 6 describe the lower-level-motion-control layer of the software system. Chapter 5 discusses the kinematic control component of the system that can be used for describing kinematic profiles for the degrees of freedom of the body, while Chapter 6 describes the design and implementation of the dynamic component of the system and its capabilities. In this chapter the derivation of a direct method for the calculation of ground reaction forces is described.

Chapter 7 describes the second layer of the dynamic control which embodies human locomotion. In this chapter, a detailed description of the analysis and implementation of a model for bipedal walking is given. Chapter 8 introduces a more realistic and complicated model for walking which is an extension of the model described in Chapter 7. Finally, Chapter 9 summarizes the work and draws some conclusions about the behaviour of the dynamic model, its capabilities in locomotion and other types of motion as well as the motion specification aspects of the method.

# Chapter 2

# A Survey of Methods In Motion Control

## 2.1 Introduction

A number of methods tackle the problem of motion control and these can be classified as conventional methods such as rotoscoping, kinematic control and inverse kinematics, and as experimental methods that originate from current research such as synergic, algorithmic and behavioural control. In the latter class of methods belongs dynamics analysis which has its own taxonomies. The issue of motion control is closely coupled with the way motion is described i.e motion specification. In an attempt to define the problem of motion control and survey previous work, this chapter starts with the efforts that have been made in order to find a description language for human motion and continues with an investigation to the methods of motion control of articulated bodies. The survey then concentrates on the classifications and considerations for building animation systems based on dynamics methods. In this context, peripheral issues are addressed such as collision detection, use of parallel computers and use of knowledge base systems in dynamics. General descriptions of the animation process can be found in Wyvill [89] and Lansdown [85], while Tost & Pueyo [88] offer a survey on the modelling and animation of human models.

We need to introduce some terminology first. An articulated body such as a human, a horse or a robot is assumed to consist of a number of rigid links connected together by joints which have multiple degrees of freedom. A *degree of freedom* determines the direction along which a joint can move. Degrees of freedom can be rotational or translational. This means that a joint can have a maximum of six degrees of freedom (three rotational and three translational). There is a number of different kinds of joints that we can use for connecting links together, but we can classify them as rotational or translational joints. A translational joint is one that allows linear motion, while a rotational joint allows angular motion. The type of joint is determined by the type of degrees of freedom it allows (ball or hinge joint).

## 2.2 Representation of human motion

Badler & Smoliar [87] and Badler [83] claim that the main problem in representing human motion is the lack of agreement on how the motion should be described. Humans can easily understand natural language as a form of movement description. The Soviets use it for classical ballet choreography. The use of natural language embodies the problem that extensive knowledge about human motion is required and the use of classical ballet idioms makes description for the common animator even more difficult. In general human motion can be described using five modes :

- direction signs which refer to translational motion of joints.
- revolution signs refers to types of motion such as twisting, pivoting etc.
- facial signs refer to the orientation of the body
- contact signs indicate surface points of body parts in contact with other bodies.
- shape description refer to the formation of a shape.

The most common notations used in human motion include the *Eshkol-Wachmann* notation and *Labanotation*. The Eshkol-Wachmann notation considers the human body as a set of limbs connected by joints, whereas Labanotation considers it as a set of joints connected by limbs.

*Effort/Shape Analysis* is concerned with how and where the body applies effort in order to accomplish a motion. It is not a complete notation in the sense that it does not describe any positional information but only indications of energy consumptions and is more suitable for the muscular model of the human body. The information about the effort of energy that a muscle applies for a motion is derived from electromyography. Labanotation and Energy/Shape Analysis are often used as complementary notation systems.

## 2.3 Conventional methods in motion control

This section presents the three most commonly used methods for motion control of articulated bodies. The section starts with rotoscoping, the oldest method of the three, and continues with the investigation of parametric keyframing and inverse kinematics. All these methods are used widely by commercial companies and employ rigorously defined and tested procedures.

### 2.3.1 Rotoscoping

Max Fleischer invented and patented rotoscoping in 1917. It has been used in computer animation successfully for the last fifteen years. Rotoscoping is the process of projecting live action film of characters and objects onto the animation cells and can be used in three different ways McGovern [87]. Firstly, when exact reproduction of real motion is required, rotoscoping is used to project the motion without any changes onto the animation cells. Secondly when changes to the initial real motion are required in order to produce the final animation sequence. Finally it can be used as a reference tool, where the animator uses rotoscoping to enhance his or her knowledge about the objects involved in the animation.

*i. Exact reproduction of motion from film footage* : In this case, actors are used to perform the required scenes. The acts are recorded from more than one cameras (usually three or four cameras, a top, front, and one or two sideways views). The cameras have be synchronized so that they are all filming the same exact fraction of a second, frame by frame. A cut of the action film is made afterwards according to the script. In order to simplify later processing all views from the various cameras are pulled together to one film. The Sexy Robot in the short advert 'Brilliance' was animated using this method.

A prototype of the figure is then encoded into the computer. For each frame and for each view, the animator sets up the camera in the animation package according to the information recorded during the filming. The animator then, chooses the keyframes (frames that are important to the motion) and sets the correct position of the cameras at each keyframe for the body, for each different view. Once this process has finished, an interpolation technique can be applied to fit a curve of position against time, using the positional information about keyframes. The technique is known as keyframe interpolation and we shall study it in detail in the next section. At this stage the animator can evaluate his/her work so far. If the motion is not satisfactory then a number of measures can be taken, including changing the position curves or introducing more keyframes to the motion.

In general this kind of rotoscoping produces realistic motion when it can be applied. It is costly though, because of the use of many cameras, renting the actors and film processing. The task of rotoscoping is tedious because it is done for each camera. This can be simplified by using one film frame for each animation frame (using the image from one camera only).

*ii. Using modified film footage*: In the second approach, the action cannot be recorded using an actor because the object of the animation is not human but some articulated animal like a cat or a dog. Hiring trained animals can be very costly and the results are probably not satisfactory. The animator in this case has to rely on studying existing footage of the animal in question. This can be done by finding illustrations in books. It is very likely that a model of the real animal with all the necessary articulations will be needed in order to position the model against the reference footage. The pose of the model is hand traced onto animation paper with the pivot points of the articulations clearly marked. This produces a stick figure for each keyframe which is encoded into the computer. The next step is to convert the figure to 3-D models. A similar process to the first case is followed where interpolation can take place again until the right result is produced. One of the major problems in this approach of rotoscoping is that it might be from difficult or even impossible to find literature with the necessary footage. A rich collection of different human and animal gaits can be found in Muybridge [55/1, 55/2].

*iii. Using rotoscoping as a reference tool* : In the case where rotoscoping is used as a reference, the live action footage is recorded on a cheap medium, Black and White rather than colour film. The film is then cut to contain the scenes necessary to the scenario. Stick figures can be created by the live action and encoded into the computer. The stick figures are positioned using the live action footage in a way that satisfies the animator. The difference between this and the first approach is, although they look much the same, that the first case suits clearly defined motion where the animator knows what is required beforehand, where in this case the animator has not a clear idea about how the final sequence would look like, and uses the live footage as the reference to decide about the form of the animation sequence.

### 2.3.2 Kinematic control and parametric keyframing

Parametric keyframing is probably the most popular animation technique. In the previous section we saw that is used as part of the rotoscoping process. Although in many cases it is used as a complementary technique to other methods, it is also used as the main method. The various motion parameters such as positions, velocity and acceleration are determined at keyframes, i.e, frames that are important to the motion, and the values of these parameters at the in-between frames are calculated using some mathematical interpolation technique as in Steketee & Badler [85].

In parametric keyframing the motion of the articulated bodies is specified by using parametric keyframes. The basic idea is that a motion parameter is specified at a set of times during the course of the motion. The choice of set of times depends on the animator. The requirements for an interpolation system that uses kinematic control are described in Steketee and Badler [85]. These requirements include continuous motion, local control, kinetic control, continuous transitions and phrased transitions. Local control and phrased transitions are not present in every interpolating system. One of the fundamental requirements of keyframe interpolation is that the value of the parameter with respect to time must be continuous. This is often sufficient to produce realistic motion. First and second derivative continuity is a desirable feature. When the parameter in question is displacement, then the first and second derivatives are velocity and acceleration respectively and in this case first and second derivative continuity is necessary since discontinuity of velocity or acceleration will give rise to unnatural and jerky motion.

Local control means that the animator can change the interpolant between two keyframes without changing it in any other section of the curve. One of the main disadvantages of the interpolation methods used in parametric keyframing is that it is difficult to achieve local control. If the animator changes the value of the parameter at keyframe C, we should expect interpolants $B_1$ and $C_1$ to change but the rest of the curve to remain unchanged (Fig 2.1). The kinetic control of the curve is also very important. We may end up with a smooth curve that fits the motion perfectly but with incorrect initial and final velocities. In order to have a smooth kinetic control and transitions from one motion to another, boundary conditions concerning the velocity have to be incorporated.

The next requirement is that the transitions between two successive motions have to be continuous. When an object changes its type of motion, then continuity of its acceleration and velocity curves is necessary in order to achieve realistic motion. The concept of phrased transitions in the context of keyframe interpolation is also introduced in Steketee and Badler, where two successive motions blend smoothly into one. This may affect only neighbour portions of the curve to the transition point.

The central idea to keyframe interpolation is to decide what kind of interpolation technique to use. It is widely accepted that piece-wise interpolation is the most suitable method. Use of a single polynomial for the entire function requires the polynomial to be of high degree which produces undesirable oscillations of the interpolating function. Cubic splines are the most popular amongst these techniques because the number of coefficients that has to be found is low and usually equal to the number of the boundary conditions. In Badler's work, third degree B-splines are used because they give second derivative continuity and easier local control. Using keyframe interpolation, it may be necessary to interpolate more than one parameter. Therefore for each keyframe there is a set of data which represents the values of the parameters at this keyframe. Two different interpolants can be used for each parameter. A kinetic B-spline which gives the keyframe number as a function of the time, and a position B-spline which gives the value of the parameter as a function of time.

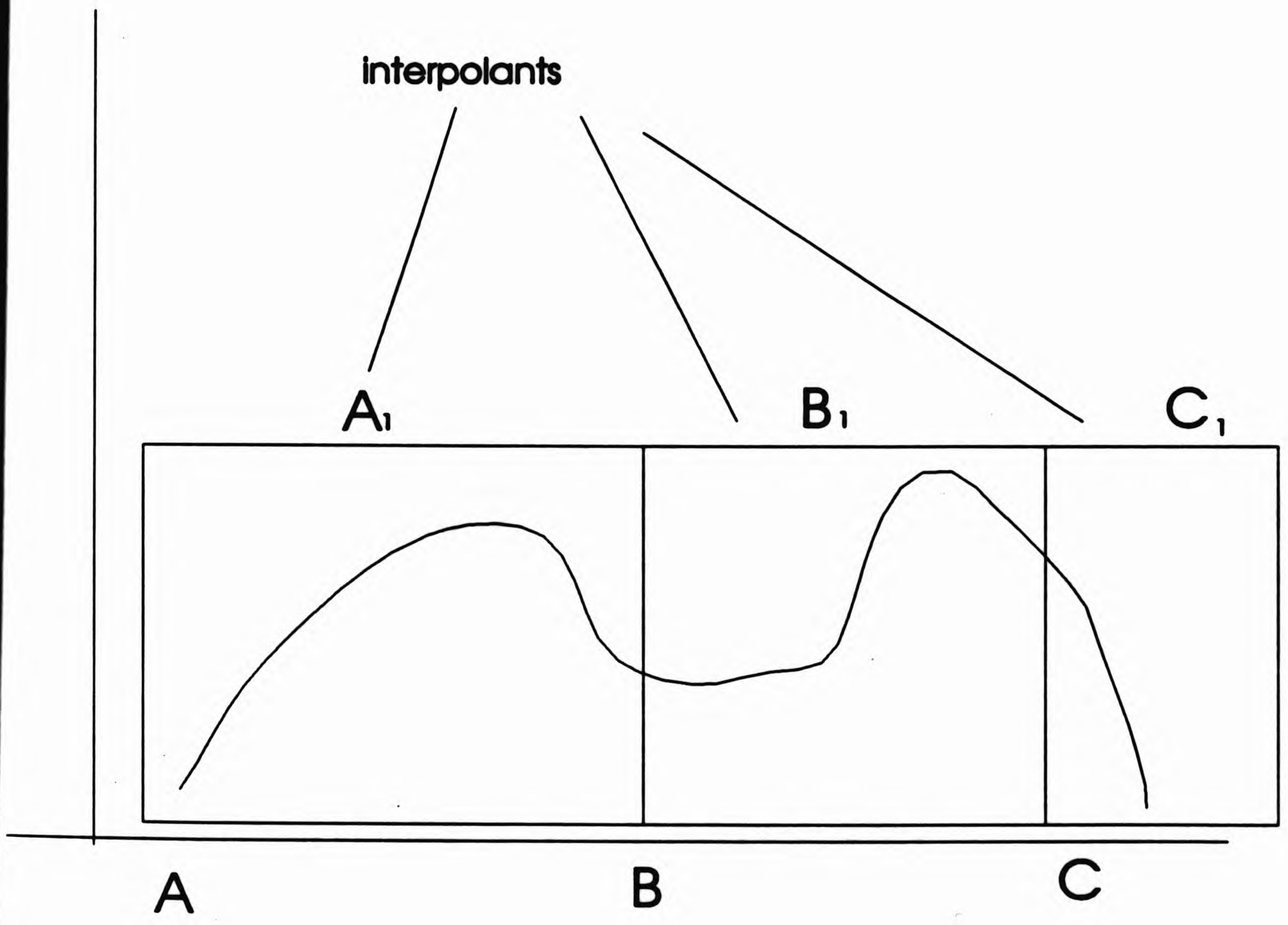


**Fig 2.1 : Parametric keyframing**

Since all parameters are recorded at the same keyframe times, the kinetic B-splines are identical for all the parameters. The position B-spline will be different for each parameter. Any boundary conditions regarding velocity and acceleration can be entered at this stage. The use of B-splines is inexpensive because it results in a triangular system which can be solved in time which increases linearly with the number of points. Kochanek splines, Kochanek [84], offer an attractive alternative to B-splines because they are computationally inexpensive and they offer a high degree of local control by using three parameters, namely ***bias***, ***continuity*** and ***tension***. Kochanek splines will be described in detail in Chapter 5.

The keyframe interpolation technique can produce realistic motion but it is subject to numerical problems. Continuous transitions can be difficult for the animator to arrange and they certainly cannot be arranged automatically by the system. The animator has to progress in a trial and error fashion until a satisfactory fit for all the parameters is found. Another problem is that the keyframes are not necessarily apparent to the animator from the beginning. This may result in a re-iteration process where new keyframes might need to be added. However, the most difficult problem is the task of motion specification, and that includes the determination of the position, velocity and acceleration curves that give reasonably realistic motion and smooth transitions and hence this problem is inherent in parametric keyframing and does not depend upon the interpolating method.

### 2.3.3 Positional constraints and inverse kinematics

The method of inverse kinematics is the newest of the three and the least established. One difference between inverse kinematics and the other two methods is that it is the only one which was not devised originally not for the purpose of computer animation but for defining positions of robotic mechanisms. The method uses the concept of an articulated body, which implies a sequence of rigid bodies (links) connected by joints which can either rotate or translate the links depending upon the type of joint.

The central idea of inverse kinematics is to find the appropriate angles for each link in order to position some of the links in the body to particular positions. For example, moving the lower leg from one position to another, may require changing the positions of the upper leg, lower body and the torso in such a way that the motion is accomplished. The technique originated in robotics, where the problem has been solved geometrically. The problem in animation is more difficult, though, because of the large number of degrees of freedom and the complicated configuration of the human body. In the context of animation Girard [87,91], Badler [87] and Thalmann [89] have given partial solutions to this problem based on iterative algorithms.

In the system designed by Thalmann [89] the animator can impose constraints on the pelvis, hands and feet. A constraint according to Badler is a desired relationship between entities but the method of achievement is not known a priori. PODA is the animation system designed by Girard and the positioning is solved using either inverse of direct kinematics. It uses pseudo Jacobian control that linearizes the kinematic equations about a point. Inverse kinematics control takes the form :

$$\theta \;=\; F(C,\, P)$$

where C is the cartesian space position of the goal, P is the desired joint space position, $\theta$ is the actual joint position and F is the kinematic function. The equation must be solved in terms of C such that the deviation of the limb from P is minimum. PODA uses tables to record postures that the limb has to pass through. The animator can interactively edit these posture tables. The interpolation method used for the in-between frames is based on B-spline techniques. Leg motion adaptations to body motion and periodic gait specification are supported by PODA.

In the animation system design by Thalmann the positional information regarding the feet and the hands can be specified. In the best tradition of animation systems this is done in the local coordinate system of the limb. The user may express the constraints as functions of parameters related to the environment around the figure. The system solves the inverse kinematics problem based on information about the pelvis and trunk angles. It then starts working outwards until the extreme limbs are reached. A solution to the problem may not exist. On the other hand an infinite number of solutions may exist for a position within the reach of the limb in question. Minimizing the angular variation between limbs is a constraint that picks a solution out of the infinite set. There are other decisions that can affect the trajectory of the limb such as avoiding collisions with other objects. In Badler, a mixture of positional constraints and inverse kinematics is also used. In order to achieve a positional goal many constraints may need to be satisfied. Instead of solving for one constraint at the time, many goals are set up in the software system simultaneously and the best position that satisfies all is found.

In order to rank the goal in a hierarchy of importance, each goal carries a weight according to how important its achievement is. The ranking system is assigned by the user. The POSIT animation system designed by Badler operates in a multiple constraint mode. One of the main advantages of POSIT is the use of Polhelmus, a 3-D digitizer which operates on six degrees of freedom, within a cubic metre of space. A reach-tree hierarchy is used by the software to depict the way that the limbs of the body are ordered, starting from the lower body (usually, but not always) and branching out towards the extreme limbs. Each node in a reach tree has a weight that depends upon the distance of the limb from the goal and its goal weight. Using these two quantities the weight of the node is calculated and the weight of each subtree is then computed. A balanced reach tree is one in which all subtrees branching from its root either have weight zero or their weight cannot be decreased if the root limb is not moved. The algorithm checks whether the tree is balanced; if so, it then stops.

## 2.4 Experimental motion control methods

This section presents four motion control methods that are at the research stage and promise features, such as ease of motion specification and enhanced realism which will be present in future animation systems. The ideas embodied in these methods were not originally derived for computer animation, but for other disciplines ranging from psychology to cybernetics. These methods often use complicated mathematics and are thus more computationally expensive than the methods presented in Section 2.3, but they offer the possibility of producing more realistic motion. The section starts with synergic control which has been used solely for articulated bodies and continues with algorithmic and stochastic control. The latter two methods are not offered as alternatives to motion control of articulated bodies, but they can be used for specific cases of animation sequences. Finally, the behavioural control model, which can be used when many articulated bodies have to be animated, is discussed.

### 2.4.1 Synergic control

Zeltzer [82/1, 82/2] introduced the idea of motor control in human animation. The motor-control problem consists of the identification and implementation of the programs needed to coordinate and control the actions of the joints in order to move a human skeleton. In biological motor control systems, a ***distributed problem solving*** approach is followed. The motor control system is simplified into a set of subsystems called ***functional synergies***. These subsystems include sets of muscles and joints that are needed to perform a particular type of motion. Each subsystem requires its own set of local motor programs. The problem is simplified by the use of local motor programs (LMP) that are combined to produce task-level motor programs in an analogous way to that in which reusable software is applied in software engineering.

Human movement is represented in a hierarchical way, where we move from high-level to low-level motor programs as we move from the top of the hierarchy to the bottom. This is similar to ***procedural control***. A ***task description*** is a specific movement sequence such as 'turn right and walk for 50 yards'. A set of movement functions are extracted for each task description called the ***skills***, for example running, walking etc. Finally, the skills are represented by motor programs which, in turn, invoke LMP to control the motion of each subsystem. An executive piece of software sits on top called the ***task manager*** which accepts the task descriptions and decomposes them into component skills. This list of component skills forms the ***movement queue***. The task manager has to know only what skills must be invoked for each task description.

The motor programs are sited just below the task manager. Each of them invokes a set of LMP which are procedures that access a fixed set of links and joints in order to change the skeleton's database. Zeltzer uses finite state machines (FSM) to implement LMP. He uses a four-state FSM to implement a joint controller which is called a ***cybernetic actuator***. Based on this principle Zeltzer has built his control walker which controls 22 degrees of freedom. Parameterization of the motor control programs will allow variations on the same theme of motion. It will also allow the animator to add new types of motion. Although this can be simple for motions that involve few joints, it can be complicated when rotational goals are non-linear and discontinuous. The extension to new types of motion needs detailed knowledge of the skeleton model and substantial programming skills. Central to the approach by Zeltzer is the idea of

***potentiation*** and ***depotentiation***. The activation of one behavioural unit may enable other units to become activated but not necessarily active. The current state of the environment may make these potentiated units active. These ideas come from research in neurophysiology and psychology.

### 2.4.2 Algorithmic control

In algorithmic control, motion is described algorithmically. Laws of physics are applied to parameters which describes the position of the human figures. The animator can control these laws by programming in a language such as ASAS and MIRA or in a director-oriented language such as MIRANIM Thalmann [87]. The physical laws are made by the animator, e.g. do not apply gravity force. The user might suggest parameters for controlling the algorithm but does not control the specific motion at the lowest level, that is, specifying the values of individual degrees of freedom. This kind of motion control is suited to repetitive motion.

A part of algorithmic control is ***stochastic control***. Random algorithms are used to generate waves, clouds and other physical objects. Particle systems are also based on this principle to generate phenomena such as the simulation of a fire Glazzard [87]. The fractal algorithms refer to the physical modelling of these objects. However, the motion of these objects is intrinsically bound to the manner they are physically modelled. The motion also has a degree of randomness which fits perfectly the modelling process used in their creation. Therefore their motion is usually designed by using statistical models which embody an element of randomness. Stochastic control has not yet been applied to human figure animation but sequences of motion of a non-deterministic nature can be modelled.

### 2.4.3 Behavioural control and Learning

Behavioural control requires that interactions with the environment are taken into account during the motion generation. It is sometimes called ***stimulus-response control*** because the object receives the stimuli from the external environment and responds according to the circumstances. This process, firstly, has to have information about the external environment and secondly, to know how to react to it. Reynolds [87] use a distributed behavioural model to animate flocks. A flock is a group of objects that their motion exhibits ***polarization***, ***non-colliding*** and ***aggregation***. Polarization means alignment of animal groups.

In Reynolds, the motion of a flock is the result of the interaction between the behaviour of individual birds. The term 'bird' is used as a generic object-member of a flock. The perceptual mechanism of birds and aspects of the physics of aerodynamic flight must be incorporated. If the simulated bird has the correct flock-member behaviour then different instances of it can create a flock. The time that is needed in order to perform the algorithm used for this distributed model is proportional to the square of the number of birds involved in the motion. This is because each bird has to consider the behaviour of the flock. This approach is probably over-complicated because it is known that real birds do not consider the behaviour of all the members of the flock in order to decide what to do. Therefore, it is possible to speed the algorithm up, either by sorting the birds into lattices and examining neighbouring lattices only, or by using collision detection techniques as described later in the chapter. The initial naive algorithm required 95 seconds for each frame of animation of a flock of 80 birds. Important issues for distributed behavioural control is the synchronization of the current state of the flight dynamic model with the amplitude and frequency of the wing motion cycle. Obviously the model has little mental state but it provides possibilities for interesting future work. The term mental state is used as a quantifier to indicate of the amount of information of the program about the real model.

A step forward and probably the highest degree of sophistication is ***learning control***. The main obstacle is that we do not understand the mechanism of learning. A trial and error system can be used where every time a system makes an error it is registered into a database. An approach to this direction is examined with the use of Expert Systems later in this chapter.

## 2.5 Dynamic analysis

This section presents another experimental method for motion control of articulated bodies, however, because of its importance in this work and its potential, it is presented in a separate section.

### 2.5.1 Background and rationale

The realistic animation of three-dimensional scenes containing human beings conscious of their environment, where object-environment interactions can be done in an intelligent way, will be one of the most interesting and challenging areas of computer animation, Thalmann [89], Tost & Pueyo [88]. It is believed by many researchers that future animation systems for human animation will be based on motion control systems directly derived from principles from robotics, AI, biomechanics, and cybernetics Thalmann [89], Armstrong [87/1], Wilhelms & Barsky [85], Ginsberg [83]. The animator will be able to define the motion at task level and not by specifying low level parameters such as angles.

Many of the studies of the dynamics of human motion have originated in physiology, medicine and biomechanics, Bendal [69], Dagg [77], McMahon [84, 87] and Alexander [83]. Research has also been done on dynamics for animating living creatures, other than articulated bodies, such as snakes and worms, Miller [88] and objects, such as the Luxor lamp, Witkins & Kass [88] and Baraff [89].

Dynamic analysis refers to the study of the relationship between torques (for rotational motion) and forces (for translational motion) that apply to a body and their effect to the motion of the body. All dynamics methods originate from Newton's second Law :

$$F = m\alpha$$

where F is the force acting on a particle with constant mass m and $\alpha$ its acceleration. Although some dynamic formulations may look mathematically very different, the central idea is based on the above formula. For realistic animation, 3-D models are needed and the complexity of the relationships increases with the geometrical complexity. The limbs of the body are also considered as being extended bodies rather than points, and this introduces the complexity of considering the mass distribution and inertia tensors. In all dynamics methods, the number of equations increases with the number of degrees of freedom, in some cases linearly, in some cases not.

**Although Wilhelms simplifies the human body to 24 degrees of freedom (18 internal and six external to describe the position and orientation of the body in space, three translational and three rotational), it is known that that the human body contains over 200 articulations, Zeltzer [82/1]. Until now, only very simplistic views of the human body have been simulated dynamically, the reason being that the speed of most algorithms will not even approach the real-time barrier. An important feature of dynamic analysis is that the problem cannot been solved locally to the limb where the force applies, but the entire configuration must be considered.**

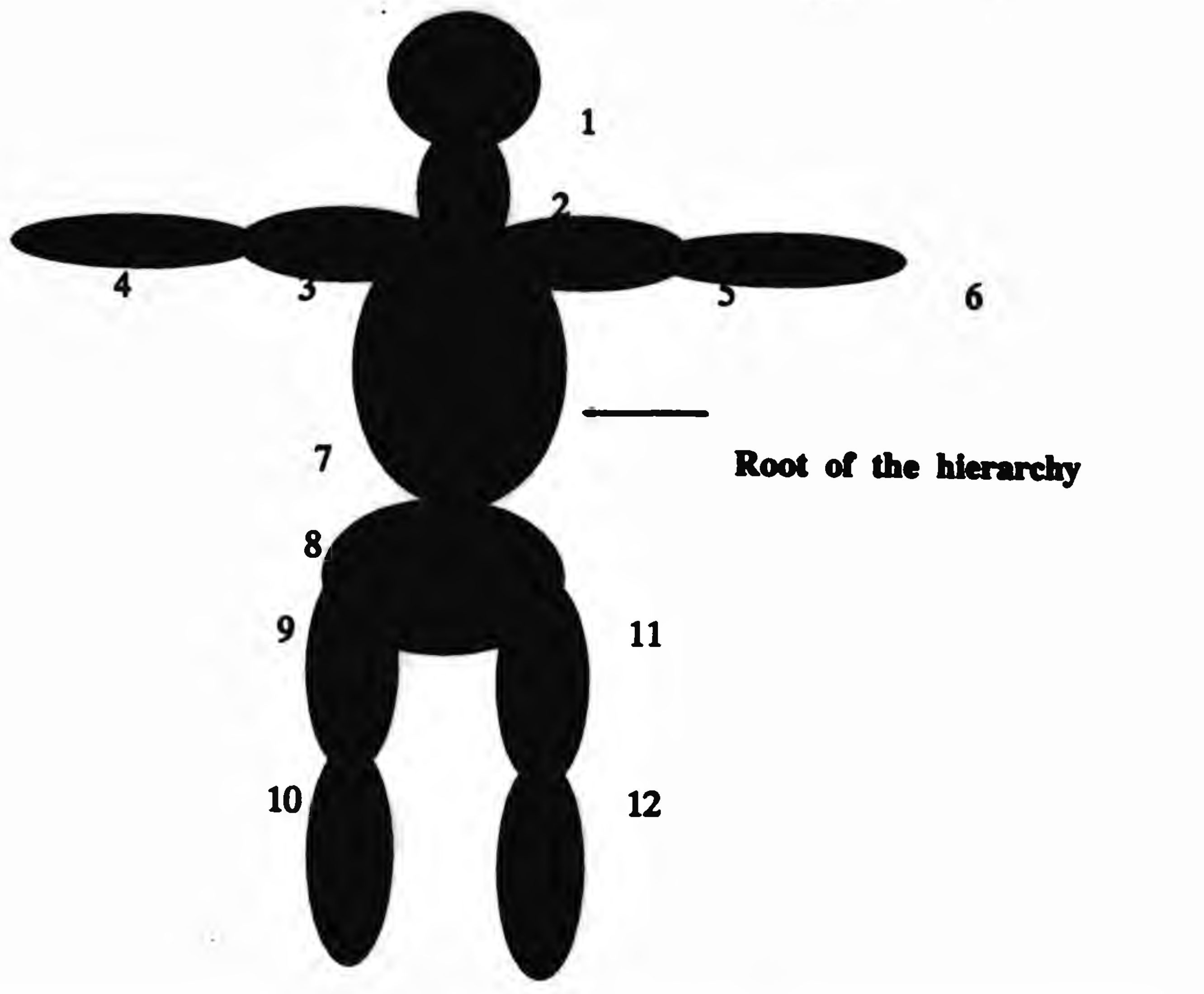
**In all dynamics methods the human body can be schematically represented by a tree structure. One link serves as the root of the tree and in most cases this link is the torso (Fig 2.2). Depending upon the type of motion to be accomplished the root link can be other links as well such as the hands or the legs. An example where the root link is other than the torso is when athletes exercise on the parallel bars. In this case the root might be the hand or the lower arm. However, current dynamics methods universally use the torso as the root.**

**A *kinematic chain* is a set of consecutive links connected together with joints of multiple degrees of freedom. Such a chain can be *open* or *closed*. An open kinematic chain is one for which the start and finish links do not coincide. A closed kinematic chain is one that forms a loop. The human body does not contain loops and therefore has open kinematic chains only. A kinematic chain may have branches, in which case it is called *branched*, or it may be *branch-free*.**

**Finally the term *method* refers to the expression of the relationships between forces, torques and accelerations and not to the solution of the relationships. The term *formulation* refers to the solution of the method.**

## 2.5.2 Taxonomies of dynamic methods

**There are two important categories of dynamics formulations, namely, *direct dynamics* and *inverse dynamics*. In direct dynamics, the various external forces that are applied to each of the links have to be specified by the user, and the system can calculate the corresponding accelerations, velocities and positions of each link of the body.**



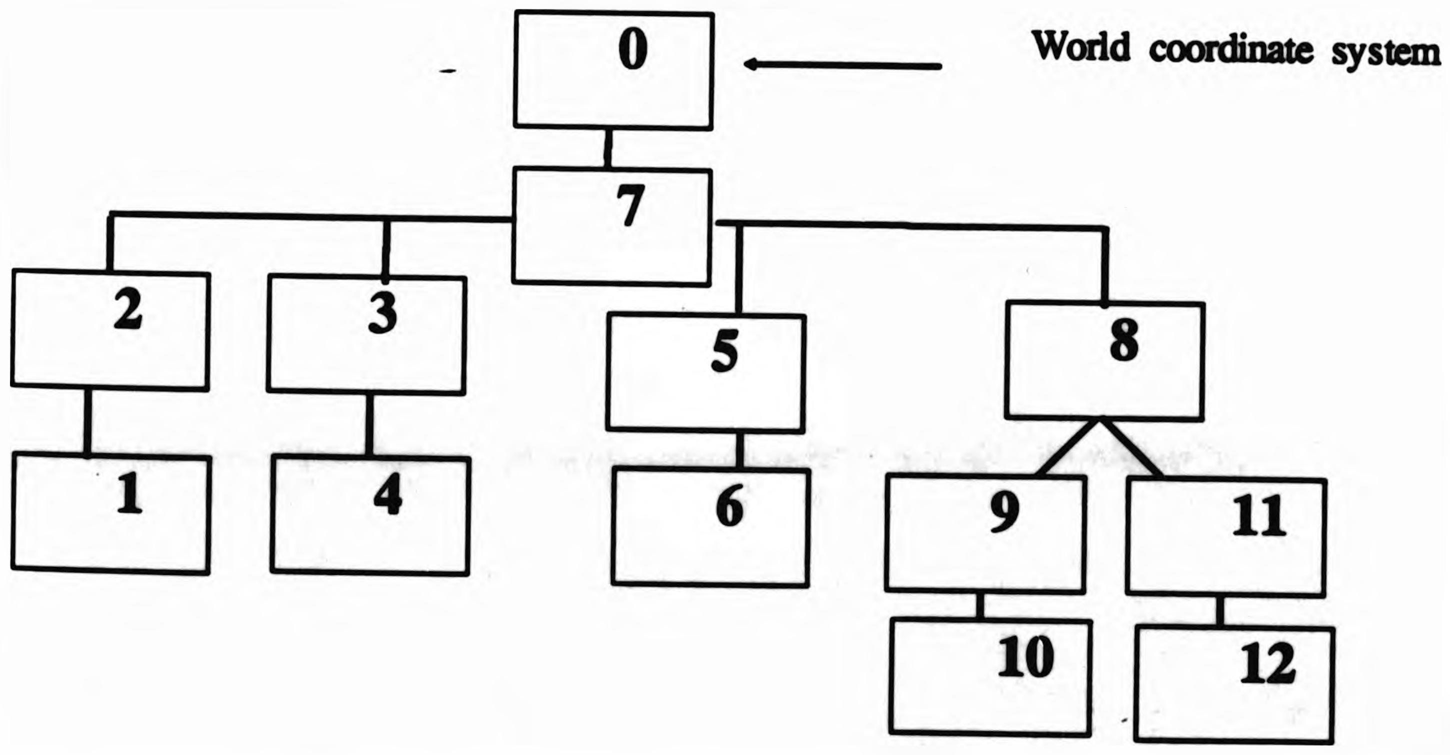


**Fig 2.2 : Hierarchical tree structure of the body.**

In inverse dynamics, the accelerations, velocities and positions are supplied and the system calculates the forces required in order to accomplish the desired motion. Most formulations in human animation apply direct dynamics, whereas applied robotics uses inverse dynamics. However, this distinction is not clear-cut because inverse dynamic formulations can also be applied in human animation. In any dynamics method, there is a relationship between accelerations, positions, velocities, forces and torques. If the forces are known then the accelerations, velocities and positions can be found (direct dynamics). If the accelerations, velocities and positions are known then the forces can be found (inverse dynamics). This means that any dynamics method can be solved either as an inverse or direct dynamics problem.

Although this classification is quite universal in the area of animation and robotics, Wilhelms refers to direct dynamics as the *indirect dynamics problem* and to inverse dynamics as the *direct dynamics problem* and although this can be confusing it is the only inconsistency that has appeared in the literature concerning these two terms.

The second classification is that of high-level versus low-level systems. This classification is analogous to high-level and low-level languages and it refers to the software implementation of the method rather than the mathematical modelling. In high-level systems, commands such as 'walk' or 'turn left' can be issued and then translated into sequences of low-level commands. Low-level systems deal directly with individual degrees of freedom, and the user has to supply data about forces, accelerations etc. A comprehensive discussion of high-level and low-level systems can be found in Wilhelms [87/1] and Zeltzer [91]. Interface problems in motion control are also discussed by John [89], Badler et al [91] and Calvert [91].

There are standard methods in computer science that are used to convert high-level specification to low-level specification. *Parameterization* uses parameters to define the motion of the object. Parameters are adequate for complex motions because they need only few commands to describe them. Parameterization can be implemented easily but fine control can be sacrificed for ease of control. A *finite state machine* can also be used to convert high-level commands to low-level, Zeltzer [82/1, 82/2]. *Command libraries* and *hierarchies* such as tree structures can also be used. In command libraries a command 'walk' could be interpreted as 'move right leg', 'move left leg' where both commands may invoke other commands. In hierarchical structures, as we move down in the hierarchy, less generic and more detailed descriptions of the motion can be be found. One of these methods, or combinations of them, can be used to convert from high-level to low-level commands.

Dynamics methods use mathematical techniques that try to approximate the problem of motion in reality and they use different mathematical conversions such as Armstrong [85], Luh [79], Walker [82], Hahn [88] and Barzel & Barr [88] use Newton-Euler dynamics, whereas Wilhelms [85, 87/2] uses the Gibbs-Appell formulation, Hollerbach [80, 89] and Cotsaftis [89] uses Lagrangian dynamics, Featherstone [83] uses articulated body inertias, Raibert [78] uses the space configuration method and Stepanenko [88] uses a compound vector approach based on Newton-Euler dynamics. Silver [81] has also shown that the computational complexity of Lagrangian and Newton-Euler dynamics is the same.

On the other hand, some methods use a mixture of dynamics with kinematics. This is because the implementor of the animation system is faced with a dilemma central to human animation: keyframe interpolation is suited to portrait drama, expression, etc, but might produce unrealistic animation, whereas dynamic analysis gives rise to realistic motion but it is difficult to model. Brotman [88] has designed an animation system in which the animator defines the keyframes and the in-between frames are constructed using dynamic analysis. Vubobratovic & Stokic [77] also use dynamics with prescribed kinematics. Wilhelms' [86], animation system can be used for dynamic analysis as well as kinematics.

### 2.5.3 Considerations in formulating dynamics

When designing an animation system for articulated bodies based using any of the control methods mentioned above, there is a number of considerations that the implementer has to take into account which can simplify the problem and tailor the method to his or her needs:

*i. Coordinate systems* : Because the problem is formulated in three dimensional space, the resulting mathematics can be very complicated and the choice of an appropriate coordinate system can simplify and speed up the formulation. A local coordinate system is assigned to each link and moves along with the link; this approach has the advantage that quantities such as the moment of inertia of the link with respect to its local frame will be constant throughout the motion. In addition to the local coordinate systems, a fixed global world frame must be assigned. The global coordinate system is associated with the root of the hierarchical structure e.g the torso. In order to find the space orientation of a limb, transformation matrices from the root to that link are used to convert from joint coordinates to Cartesian coordinates.

*ii. Choosing link origins* : Each local coordinate system must have its own local origin which is considered to be the origin of the link. Most formulations assume the origin of a link to be at the proximal joint of the link, as shown in Fig 2.3.

*iii. Determining the type of joint* : All methods allow single-degree-of-freedom joints only. Since we may have more than one degree of freedom in a joint, as is the case with human figure animation, we split up multiple-degree-of-freedom joints into sequences of single-degree-of-freedom joints (Fig 2.4). Wilhelms' formulation allows any combination of rotational and translational joints in any sequence, whereas Armstrong's method allows six degrees of freedom for the root link (link 1 as shown in Fig 2.2), three translational and three rotational, while for any other joint it allows only the three rotational degrees of freedom. Hollerbach's method allows both kinds of degrees of freedom. Usually we need only three translational degrees of freedom for the whole body and these degrees of freedom are associated with the root link.

*iv. Consideration of constraints* : The most appropriate formulation for an application can depend upon the kind of constraints that are imposed on the motion. For instance, the knee joint can rotate only about the x-axis (Fig 2.5) - this is just an approximate axis because in reality the axis of rotation of the knee for the female, forms an angle with the x-axis. Rotations in other directions will not be permitted. With Armstrong's method, such restrictions are impossible because the efficiency and correctness of the method depends upon the fact that any link (except the root link) must have exactly three degrees of freedom. On the other hand, the methods of Wilhelms and Hollerbach can incorporate such constraints. Armstrong's method overcomes the problem by greatly increasing the moments of inertia about the restricted axes of rotation, since the moment of inertia is a measure of the resistance of the body to rotational motion, and as the inertia increases, the resistance of the body to motion is increased as well.

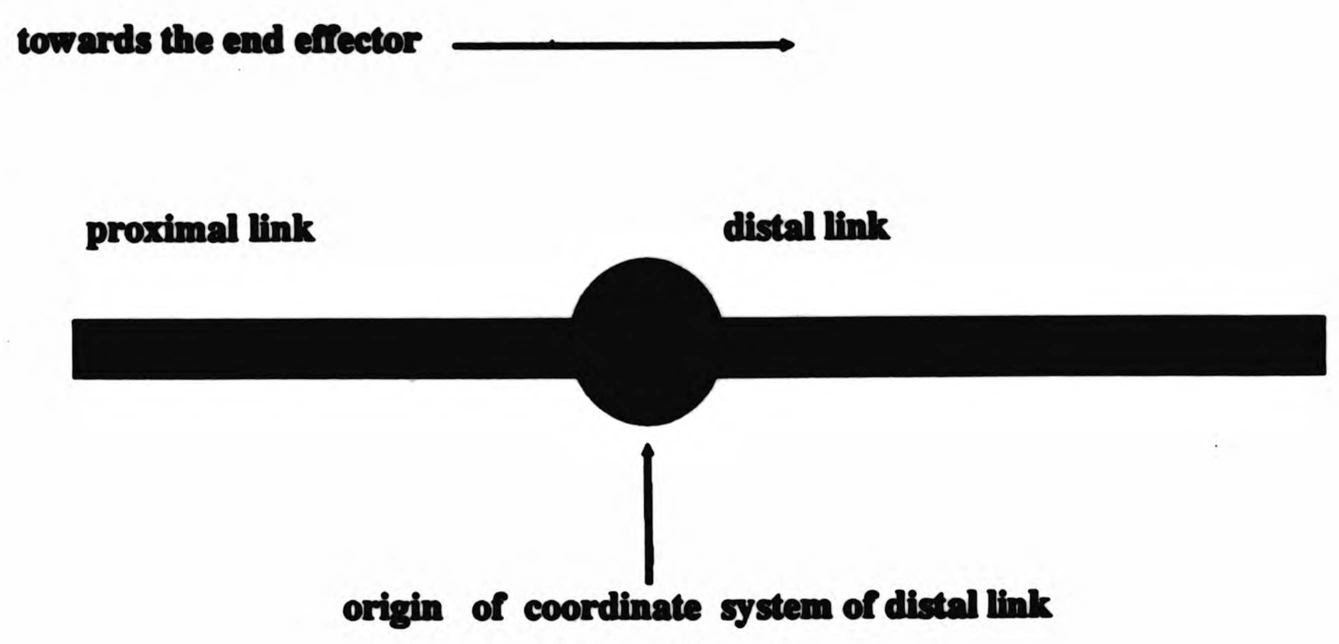


**Fig 2.3 : Proximal and distal links.**

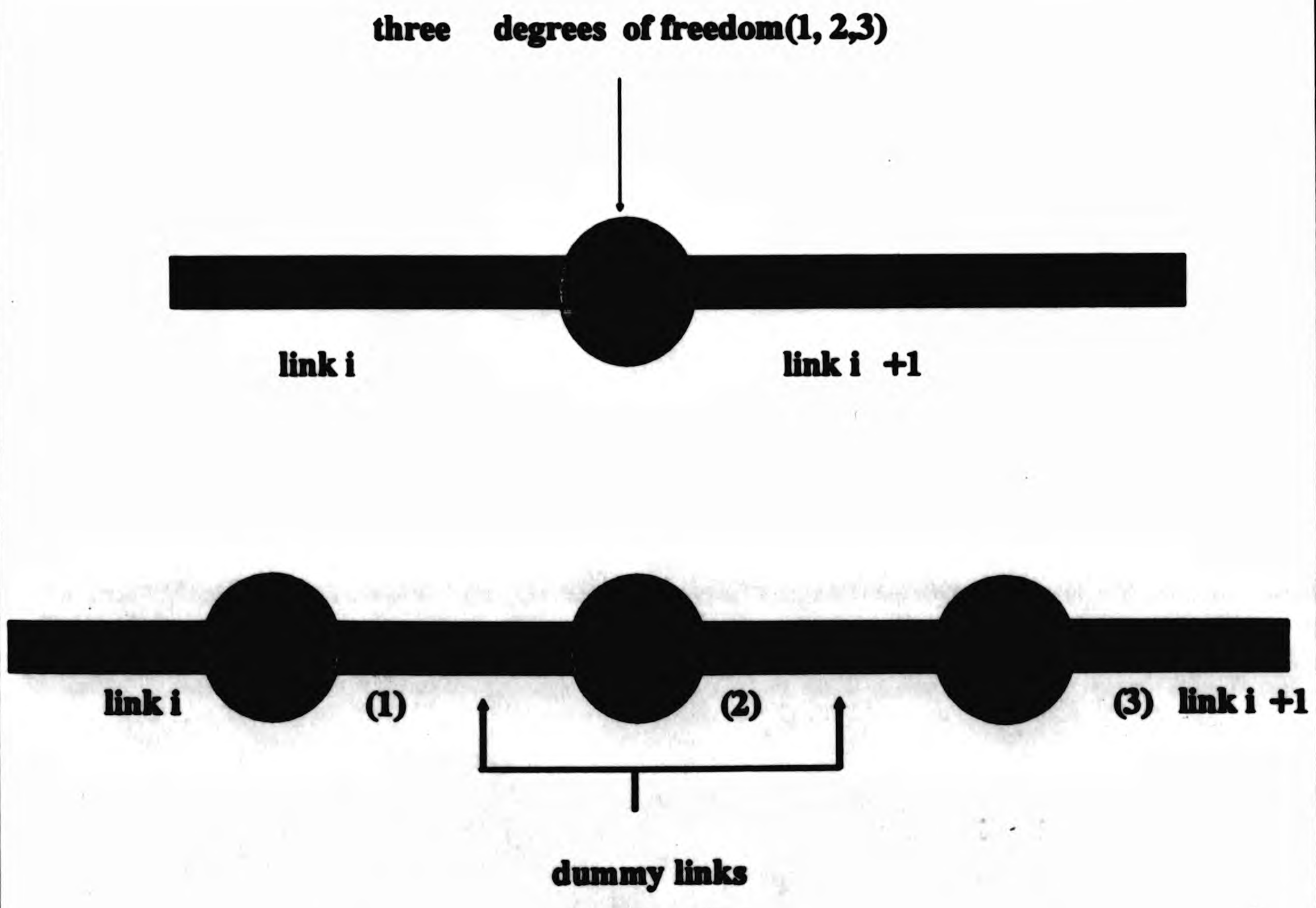


**Fig 2.4 : Decomposition to single-degree-of-freedom joints.**

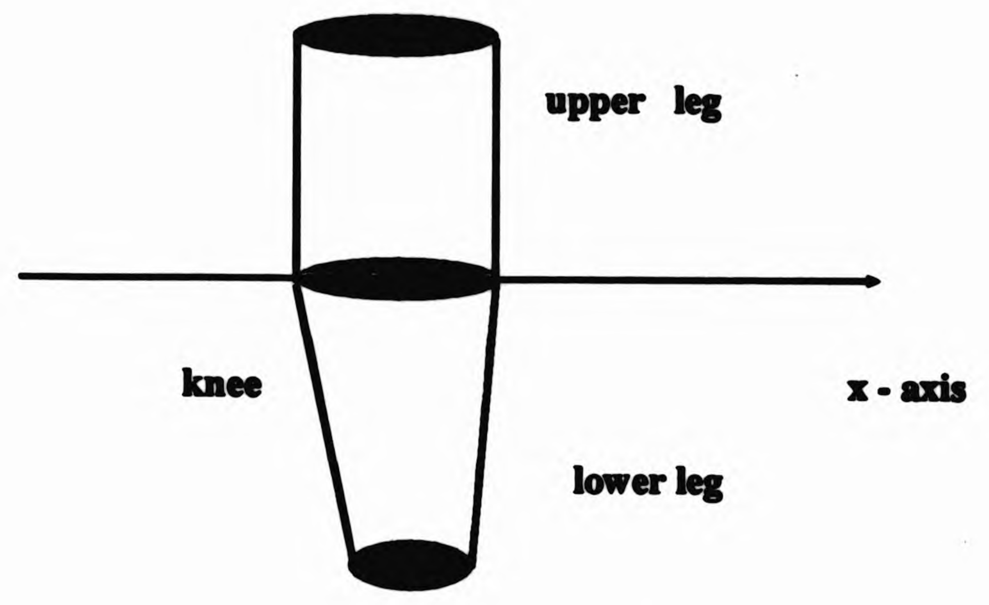


**Fig 2.5 : x - axis rotation of the knee**

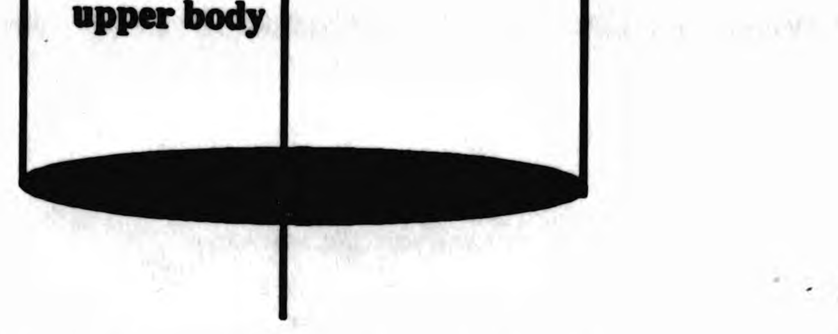


**Fig 2.6 : 90° y-axis rotation of the neck**

*v. Calculating constants* : There are quantities involved in the dynamics formulation that have to be calculated beforehand, such as moments of inertia, the original positions of the links, etc. Sometimes, it may be necessary to provide an initial estimate which is later adjusted as a result of experience gained in running the system. This is especially true for quantities such as masses and inertia tensors. From research in anthropometry, average values of masses and dimensions of limbs can be found, Croney [71, 80], but we may wish to amend them, in order to achieve the desired result.

*vi. Formulating the problem* : The generalized solution of the dynamics can now be solved and implemented. Armstrong and Wilhelms solve the problem for finding accelerations, velocities and positions given the external forces that apply to the body, while Hollerbach solves for the forces given accelerations, velocities and positions. The way that a dynamics method solved, depends upon two decisions. The first one involves the speed of the method of solution. For instance, if the inverse dynamics formulation of a method involves matrix inversion of large matrices, we may decide to go for the direct solution. On the other hand, we have to consider the information that is available: accelerations or forces. One of the major obstacles in dynamic analysis is the total lack of any information about the magnitudes of acceleration and forces required to move the body from one position to another for a particular model.

*vii. Graphics and interface design* : Finally the system interface and graphics output is designed. In most animation systems two computers are used, one for calculating the dynamic equations of the motion and the other for displaying the graphical output. There is a number of choices for representing the models graphically, which depend upon the requirements regarding the speed and quality of output:

*α. Stick figures* : These are oversimplified models but have the advantage of being very fast to draw and manipulate. Depths are difficult to judge since body parts cannot occlude one another. It is often easy to confuse clockwise and anti-clockwise rotation and twisting of limbs is extremely difficult to see. Stick figures are used to represent the connectivity of the skeleton rather than exterior form of the human body.

*β. Solid cylinder and box models* : In these, links are represented by simple 3-D geometric primitives such as cylinders or boxes. These add to the reality of the simulations but produce cluttered and confusing diagrams unless hidden-line removal is used. This produces acceptable images but adds to the computing time involved.

$\gamma$. ***Surface models*** : In this case, figures are drawn using accurate surface models for the human body. The body surface may be modelled by combining planar or curved patches. These methods are highly intensive in computation time because of the large number of polygons involved, and are recommended for the final animation sequence only.

$\delta$. ***Spherical and Volume models*** : In this case, links are designed as intersecting spheres or other volume primitives such as ellipsoids and cylinders. They can be computationally expensive, but can be realistic for raster displays. These models require shading and hidden surface algorithms in order to be displayed. The main problem with cylinders is that the planar end caps have to be smoothed at the joints and surface removal for cylinders can be extremely time consuming. For this reason, spherical primitives are used more frequently. By increasing the number of spheres a more realistic model can be achieved. By using spheres, joint deformation problems disappear because the surface of a sphere at a joint is always defined no matter what orientation the adjacent body parts assume.

### 2.5.4 Issues in dynamics methods

This section concentrates on the peripheral issues that need to be addressed in the design of an animation system. The perspective view here is to see how dynamics methods can tackle these issues. However, discussion of the motion planning and collision detection is deferred until Section 2.6.

In order to achieve adequate dynamic analysis for the model, we have to take into consideration the external and internal forces that are applied to the body - gravity is an example of an external force, whereas the force applied by anactuator to move a muscle is internal. These forces determine the state of the body and can be classified as ***automatic*** or ***heuristic***. Automatic forces, such as gravity, can be calculated automatically by the software system, while heuristic forces can be given an initial estimate which is rectified as the dynamics analysis proceeds. An example of a heuristic force is the force that has to be applied to the foot in order to prevent it from descending below the floor surface.

*i.Ground reaction forces* : These belong to the second class (heuristic) and represent the forces that the ground exerts on the body when they are in contact. Wilhelms tackles the problem of ground reaction forces by considering them as external forces produced by a spring and damper combination which is placed between the foot and the ground. As the foot approaches the ground a force is exerted from the ground to the foot proportional to the length of the appropriate spring. Vertical and tangential ground reaction forces must be considered. Vertical forces are applied to avoid collisions with the ground and tangential forces are used to apply friction in a direction opposite to the direction of motion. Collisions with the ground are detected by checking the boundary coordinates of a link against the coordinates of the current ground level. We have to make sure that the coordinates of any link are always above, or at least equal to, the ground coordinates. This method has the inherent problem that the rate of sampling coordinates has to be high enough to prevent any link descending below the ground. Armstrong uses a similar technique where spring forces are used to keep the feet above the ground.

*ii.Obeying angle limits* : Another problem is to prevent a joint from rotating by an unnatural angle, for instance we can not allow the neck to rotate more that $90^0$ about the Y-axis as shown in Fig 2.6. Wilhelms solves the problem by applying a force, dependent upon the angle of rotation, to counteract the motion. This has the effect that the closer the angle of rotation is to its limiting value, the greater is the counteracting force.

Armstrong uses a different approach. Let assume that we want to rotate the neck as in Figure 2.6, from its initial position by an angle of $45^0$ about the y-axis. If we keep accelerating the neck until it attains that value then the motion will be unnatural because it will come to a standstill very suddenly. Armstrong proposes a technique for overcoming this problem. He sets the maximum torque to occur at the midpoint of the motion i.e at $22.5^0$. After that point the torque decreases until it becomes zero and the link stops moving. In other words he applies acceleration for the first half of the motion and deceleration for the second half. Another difference between the two approaches is that Wilhelms calculates the counteracting force automatically using the masses of the links, whereas Armstrong requires the user to input a number of constants associated with each joint.

In our implementation of Hollerbach's method a similar model to Armstrong's was used. Each link starts with zero acceleration and increases as a function of the angle of rotation until it gets halfway to its limiting value. It then starts decelerating until it returns to zero, when the limiting value of the degree of freedom has been achieved. Additionally, the user does not have to supply any constants as it happens in Armstrong's approach.

*iii.Balance* : Humans keep their balance subconsciously - we are not aware of the effort required to keep our balance, except in extreme cases. It would be nice to be able to apply this idea in human animation, but we do not know the actual mechanism of keeping the balance. An approximation to the real situation is to find the difference in positions between the highest and lowest link of the body and apply a force proportional to this difference to both feet. Armstrong employs this method by considering the torso as always being the highest link and the feet as being always the lowest links. This approach is very simplistic and not always correct because the feet are not always the lowest links in the body. A more sophisticated method would be to find the lowest link in the body for the particular motion and then calculate the height difference.

*iv.Internal collisions* : The problem of how to prevent links from intersecting one another has not yet been addressed by these methods, but possible approaches would be the use of either volume intersection or springs as for ground reaction forces. For example, in order to prevent the upper arm intersecting the torso, we might apply a spring force to the arm as it approaches the torso. We can also use the idea of gravity fields where a safety distance limit exists between any two limbs. If this limit is violated, the spring force is then activated.

*v.Processes or states* : In the software implementations of Armstrong and Wilhelms, there is a number of predefined states for the model. Armstrong calls these processes and classifies them as local or global processes. Local processes are called *limb processes* and deal with individual links while *global processes* deal with the state of the entire body. For instance, swinging the forearm is a local process, whereas keeping of the body balance is a global process. Wilhelms uses a similar technique but employs the term 'states' instead of 'processes'. For instance, freezing a degree of freedom to a constant value, or orientating the figure to a specified direction are predefined states in Wilhelms' system.

Wilhelms' system supports four states. The simplest is *relaxation*, where a degree of freedom is left loose to react to external forces and *freezing*, where a degree of freedom is fixed to a certain position while the rest of the body can move freely. *Worldspace orientation*, stabilizes a limb to a designated world orientation, in which case the user provides a vector and a degree of 'play' - if this range of 'play' is exceeded, an external torque is applied in order to keep it within the play limits. The fourth state is *hybrid positional control*, which is used for degrees of freedom under internal control. Armstrong also supports four local processes, *free swing* which is the equivalent to Wilhelms' relaxation state, *friction* which generates a velocity-dependent torque to damp the motion, *maintenance process* which is equivalent to Wilhelms' 'freezing' state and finally the *move process* which moves a limb from one position to another.

*vi.Speed and efficiency* : Wilhelms' method is $O(n^4)$ where n is the number of degrees of freedom, while Hollerbach's and Armstrong's methods are $O(n\ )$. There is available software for Wilhelms' method (*Deva*) and Armstrong's method (*Dynatree*). Wilhelms claims the method can be improved to $O(n^3)$ although this is not a great improvement when compared with O(n) of Armstrong or our implementation of Hollerbach's method. A linear method is not always faster than an $O(n^4)$ for few degrees of freedom, but when n is large (say 50 or more) then this is the case.

*vii.Accuracy and realism* : Wilhelms' and Hollerbach's methods are more general and flexible in the sense that they allow the links to be connected with rotational or translational joints, whereas Armstrong's method allows only rotational joints except for the root link. In the case of human figure animation, however, this is not restrictive since all joints are rotational. Hollerbach's formulation originates from mechanics and robotics and, as such, it is suited to configurations without branches. However, human figures contain branches, such as the two shoulders branching out of the torso as shown in Figure 2.2 so the method has to be adapted to incorporate branches for the purpose of animating articulated figures.

Because Armstrong's method was devised from work done for the space shuttle arm, the objective was directed towards efficiency and speed. This makes his method less general that those of Wilhelms and Hollerbach. One important point, is that the addition of an extra link to the configuration requires an extra set of three degrees of freedom, even if not all of them are required. This restriction does not exist in the other two methods. However, the computational cost incurred by choosing

Wilhelms' method may offset this advantage. Because of the intensive calculations, Wilhelms recommends that the method is used only for figures containing up to 24 degrees of freedom, although she has tested it with systems having up to 36 degrees of freedom. Considering that the human body has more than 200 degrees of freedom, these methods cannot solve the problem in real time. Hollerbach's method combines the speed of Armstrong's method and part of the efficiency of Wilhelms' method. This means that when an extra link is added we can associate one, two or three degrees of freedom.

## 2.6 Motion planning and object collision detection

None of the motion control methods that are presented in this chapter are capable of addressing the problem of path planning and collision detection. However, these control methods may be possible to be used with the algorithms and ideas presented in this section in order to address the problem.

The trajectory and path-planning problem is well addressed in Robotics by Canny [87], Perez [84], Cameron [85], Kant & Zucker [86] and Faugeras [86]. It seems feasible to apply some techniques of robotics in animation. The path-planning problem in Robotics is called the *generalized movers'* problem which is the problem of moving a general robot in three dimensional space filled with obstacles Canny [87]. The first general solution to the problem was derived by Schwartz & Shariz [87] with a solution of order O(n) where n is the number of degrees of freedom. The solution involves the exact computation of large constants (multiple precision) which increases the computation time. Canny improved the solution by introducing the *roadmap algorithm* based on the use of multivariant resultants to solve systems of polynomials. The roadmap algorithm can either be used directly on a set of collision-free configurations or on a subset of maximal clearance configurations. A new type of Voronoi diagram is used that are directly derived from functions based on the boundary surfaces in configuration space.

In the context of human animation, the problem of external collision detection is more complicated than in robotics, because of the number of articulations and the branching of the human body. For articulated bodies, coordinates are defined either in Cartesian form or in joint form. Cartesian coordinates describe the position of the body in world coordinate space, whereas joint coordinates describe the position of a joint in its local coordinate system. In path planning, the animator can define the trajectory of the body in Cartesian coordinates and use inverse kinematics to find the joint positions Wilhelms [87/1]. Collision detection is an integral part of path planning and the simplest solution will be, naturally, a straight line between the current position of the body and its goal position. When an obstacle is encountered, a new path is designed until the body reaches its goal position. Obviously, such a simplistic algorithm will result in unnatural and jerky motion. To avoid that and to decide upon the path before departure, more information about the environment must be known. The *Lozano-Perez* algorithm is probably the most well-known algorithm to solve the problem for stationary obstacles Perez [79]. It uses graph theory to construct a directed graph that has as nodes the start and finish positions, as well as the possible in-between points. The edges between nodes are weighted according to their distance between the nodes that they connect, and a minimal path algorithm such as those of Dijkstra or Kruskal can be used to find the minimum path between start and finish positions, Price [71].

Cameron presents three simple methods for collision detection. A body is considered to be a set of points contained within the model such as :

$$S = ((x, y, z) : f(x), g(y), w(z))$$

where functions $f(x)$, $g(y)$ $w(z)$ represent the volume distributions of the body along the x, y and z axis respectively. A *location* function gives the position of each point at time t :

$$L = ((\vec{x}\ , t)\ :\ \vec{x} \in S)$$

A collision between $n$ bodies $S_1$, $S_2, \ldots, S_n$ does not occur when their location functions $L_1(t)$, $L_2(t), \ldots, L_n(t)$ are disjoint sets i.e :

$$L_1(t) \cap L_2(t) \cap \ldots \cap L_n(t) \ = \ \emptyset$$

In ***multiple interference detection*** the shapes of the bodies are tested for interference at selected times. The method performs well if the time steps are selected correctly. In ***four-dimensional intersection detection*** a set of four dimensional points (space-time) are generated. These sets form abstract notions of the motion of the body. In ***sweeping***, a three dimensional object is used to generate a new three dimensional object that contains the volume swept by the object. In Perez, the method for achieving constrained motion in the presence of position uncertaincy is by the use of ***controlled compliance***. Compliant motion meets the constraints of the motion by specifying how the robot motion should be modified in response to forces generated when constraints are violated. Compliant motion requires the knowledge of the geometric constraints imposed by the task. A common feature in path planning is that the uncertaincy about the motion is so large that the programmer cannot know precisely which geometric constraint holds at any instance in time. A ***fine-motion strategy*** is a method that combines the force and position control in order to achieve the goal.

In animation, work on collision detection has been carried out by Moore & Wilhelms [88]. They present a number of algorithms for collision detection and response for articulated bodies. The first algorithm deals with articulated bodies with tree-like linkage structures with revolute joints. For a collision which involves n rigid bodies, the solution of a system of 9n linear equations is required. The sparsity of the solution matrix increases with n and this implies that the solution is approximately of order O(n). The information about all the objects is transformed into a common inertial frame from their local coordinate frames. Once the system of equations has been solved we have to transform back to local coordinate frames. They also offer an alternative algorithm for non-tree-like linkage structures and an algorithm for sliding joints. Although current animation systems do not provide collision detection, considerable benefits could be gained by including automatic collision detection especially when dynamic analysis is used.

## 2.7 Parallelism and dynamic analysis

With the introduction of parallel machines and transputer technology, it is natural to pursue the re-organisation of dynamics formulations so they run on many processors, thus accelerating the whole process of motion control. Because dynamics formulations involve much number-processing, they are ideal candidates for running on parallel machines. Another reason for pursuing parallelism is that in few years time it is anticipated that most machines will be parallel, so the software should take advantage of this fact whenever possible.

One such experiment was attempted by Olafsson & Marsland [85], who describe an environment designed and built for investigating distributed processing using standard equipment and the services of the UNIX operating system. The actual realization of the theoretical model uses stand-alone Motorola 68000 processors and the system does not support more than one process per physical processor.

Armstrong's method requires around 9 to 17 minutes of CPU time in order to produce ten seconds of animation on various machines (Vax 11/780, Iris, Sun) which is not acceptable for near real-time animation. Armstrong [87/2] collaborated with Olafsson and Marsland to show that the solution of motion equations can be accelerated by the use of a virtual tree machine (VTM) connected by a local network. The designer of the system had the choice to use either a single link per process approach or a multiple link per process approach in which each process was mapped to a processor and both approaches were tried. Since Armstrong's formulation is such that we work the equations from the tips of the model (legs, hands) towards the root link (torso), a processor can be assigned to each set of equations associated with a link. Once the processor has finished with this link, it can start working the equations of the next one towards the torso. Fig 2.7 shows how this link-processor assignment can be done. There are two problems with this method :

*i. Synchronization of processors* : The tree structure has branches like link 1 in Fig 2.7, which means in order to start calculating the dynamics equations for a branch link, the solution of the equations of all immediately subordinate links (links 2,4,8,12 in Fig 2.2) must have been calculated in order that the information can be passed to the processor assigned to the branch link. If the synchronization of the processors is poor then the processor utilization rate is low. For example, if the processor assigned to 2 has finished but the processor assigned to 4 is still busy, then processor 2 will remain idle until processor 4 finishes.

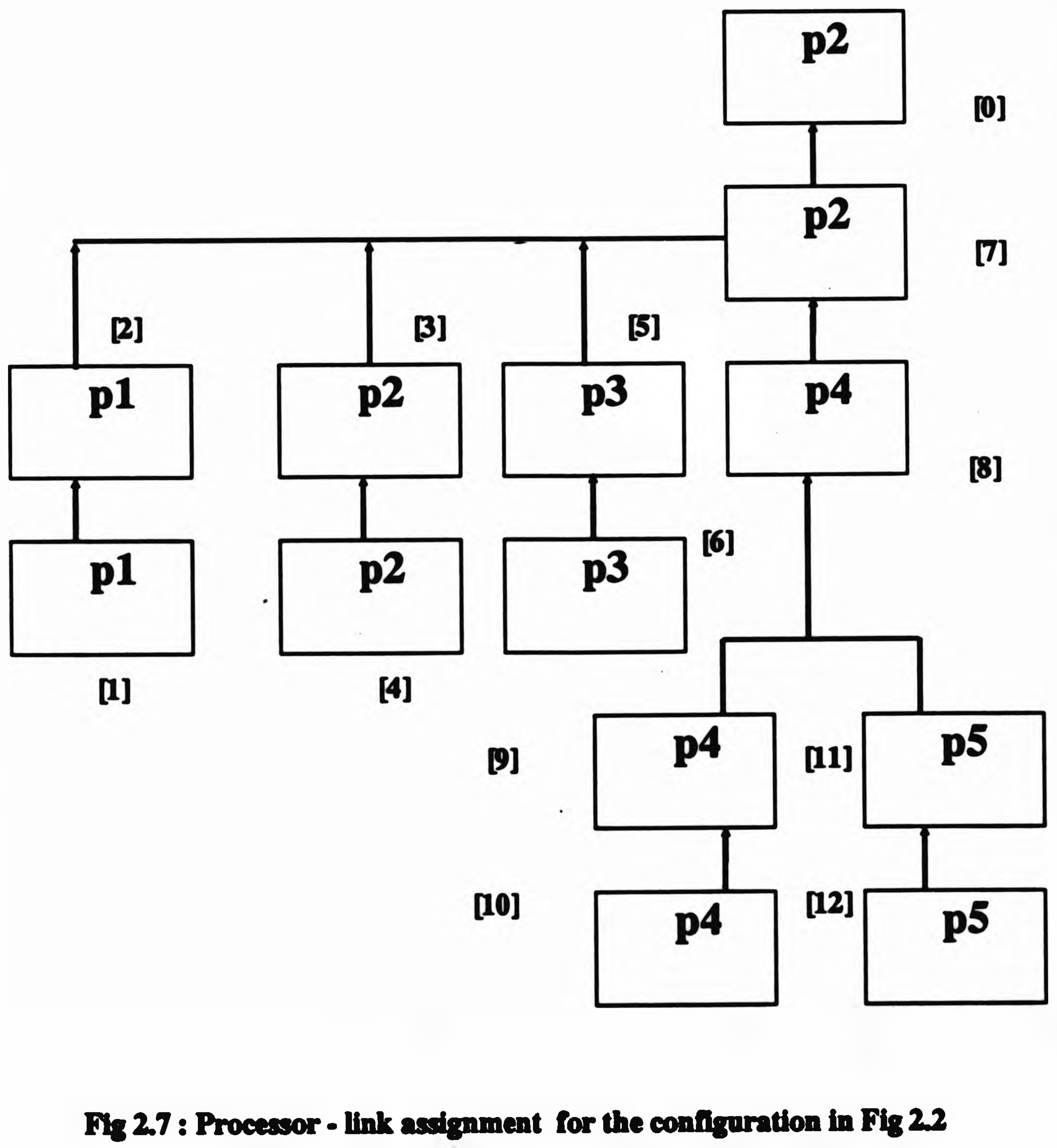


**Fig 2.7 : Processor - link assignment for the configuration in Fig 2.2**

**The numbers in [] correspond to the link numbers of the hierarchy.**

*ii. Communication overheads* : Information has to be passed from one processor to another. If the communication overheads are high, then the method will slow down considerably.

As can be seen from Fig 2.7, five processors are required for this particular tree structure. Experimental results showed that communication overheads, rather than synchronization were the main problem. However a technique for speeding up the method is described by Armstrong [87/2] and Olafsson & Marsland [85]. It can be seen from Fig 2.7 that there is a limit as to the number of processors that can be used for a certain configuration and in the case of Fig 2.2 this limit is five. This is because we cannot start the next frame if the positions of the previous frame have not been calculated and fed back into the system.

It was found that the VTM approach with five processors produces poor speed up for animating a single figure but for simulating many figures simultaneously, it is three times faster. To reduce synchronization overheads and achieve maximum speedup, several figures can be simulated out of phase, so that idle times for one can be used by others.

In robotics, Hashimoto & Kimura [89] have managed to devise a parallel computational scheme for Newton-Euler dynamics which implementation is suitable for VLSI design.

## 2.8 Enhancing the intelligence of the system

The ultimate goal for a human figure animation system is to achieve the ability to learn from the virtual reality of its environment and use this information in order to simplify the task of motion specification. Current trends in software engineering suggest that the right direction for achieving this goal is the use of Expert Systems, AI and Neural Networks.

The intelligence of any human animation system can be increased by using standard software engineering techniques such as libraries, parameterization, finite state machines and hierarchies. Wilhelms [87/2] gives a good explanation of these methods.

However the use of dynamics in articulated figure animation opened the way to new technologies that can make human animation systems behave in an intelligent way.

One approach is the use of *Hybrid Numerical Expert Systems* to produce systems that are able to learn as they are operated. Mohamed has carried out substantial work in this area, which could not have been done without the development of dynamics formulations. Mohamed [88/1, 88/2, 89] use Armstrong's method to develop a hybrid knowledge base motion control system for a multi-legged articulated robot. His *Autonomous Intelligent System* (AIS) starts with a weak knowledge base containing the rules that govern the motion control of the robot, as well its interaction with the external environment. As the user operates the system, a trial and error technique is used by the program to rectify these governing rules as the robot makes mistakes.

The rectification of the knowledge base is aided by the help of two software components : the *numerical controller* (NC) and the *learning controller* (LC). The numerical controller deals with the low aspects of the motion. This means that the NC gives an intelligent estimate of the values of the various torques and forces involved in the motion and passes these values on to the dynamics program. These values can be considered as an initial estimate. The LC can rectify the rules in the knowledge base, according to the resulting motion. The system uses different levels of rules in terms of generality and precision.

## 2.9 Conclusions

This Chapter has dealt with existing motion control techniques such as rotoscoping, kinematic control, synergic control and also algorithmic and stochastic control. The method of dynamic analysis was then described together with considerations associated with building and implementing an animation system based on dynamic analysis. Peripheral issues of motion control were discussed such as external collision detection, parallelism and the use of Expert Systems in motion control.

The variety of methods for motion control in animation leads to a need for some classification and possible unification. There are two important results from the survey of these methods. Firstly, very few of them can be considered suitable for articulated bodies and secondly, many of these methods can be used only as complementary to one another.

Rotoscoping has produced some good animation sequences with articulated bodies but it is not offered as a method for building future motion control systems, mainly because of the difficulty in motion specification and its low cost/effectiveness ratio.

Parametric keyframing, on the other hand, can be used as an aiding tool for other methods of motion control but it lacks the ability of high-level motion specification.

Inverse kinematics can provide only a partial solution to the problem of motion control of articulated figures, since only heuristic algorithms exist.

Behavioural and stochastic control are able to contribute to particular cases of motion control for articulated models, but cannot yield a general solution.

Synergic control has been used in articulated body animation with some successful results and, although it might be one of the approaches for building future systems, it does not address problems such as collision detection and path planning.

Dynamic analysis has gained many followers during the last few years because of its ability to model the problem as accurately as possible. Current solutions present large computational complexity but, with the development of hardware, this may not be a major problem. The main advantage of dynamics is its close coupling with parallel computing and expert systems.

A future motion control system for articulated body animation will combine dynamic analysis with some elements of algorithmic and synergic control and parametric keyframing. These approaches will act in a complementary manner to tackle the entire problem.

# Chapter 3

# Inverse Lagrangian Dynamics For Articulated Bodies

## 3.1 Introduction

In the previous chapter a number of motion control methods were introduced. It was shown that there is a large variety of dynamics methods which originate from robotics. These methods are broadly classified into direct and inverse dynamics. Both Armstrong's and Wilhelms' methods are based on direct dynamics i. e by providing the forces that apply to the body, solutions are obtained for the acceleration, velocities and positions. This chapter introduces a method that was developed for the dynamic control of robot manipulators which is based on inverse Lagrangian dynamics, i. e by providing the accelerations, velocities and positions, solutions are obtained for the forces. The method which was derived by Uicker [65] for solving general linkage problems in the automobile industry and then modified by Hollerbach [80, 89] to solve the dynamics of robot manipulators.

The chapter starts by explaining the way in which coordinate systems have to be constructed in order to comply with the method. The need for the use of local coordinate systems is also explained. The formulation is then derived. The derivation is quite long and therefore an effort is made to split it into small logical steps. Several identities that are used in the derivation and their proof is also included. Once the basic formulae for the method have been found, expressions are given for calculating terms involved in the formulae, such as the inertia tensors and the derivatives of transformation matrices. Therefore, this chapter deals solely with the derivation of the dynamics method as it is used in robotics. In the next chapter, the necessary modifications for adapting the method for articulated body animation will be discussed. The rest of the introduction is dedicated to the explanation of the conventions for the vector quantities that are used in this chapter.

Articulated figures consist of rigid bodies linked together by joints of multiple degrees of freedom. The links of the body are described by two quantities, their position and their orientation, and these are defined by using a coordinate system, called a *frame*, which is attached to the link. The position and orientation of a link can then be described by examining the state of the frame with respect to some fixed reference coordinate system. When a joint is rotational, the displacements of the links from their rest position are called *joint angles*. In the case where a joint is translational, the relative displacement between links is called the *joint offset*. The human body consists, as do most articulated bodies, of a number of kinematic chains, and in each chain the first link is called the *root of the chain* and the last link is called the *end-effector*.

The world coordinate system is a fixed non-moving coordinate system and the positions of the frames of the links are described with respect to this coordinate system. In the rest of this work, parameters in lower case describe 4x1 vectors in uniform coordinates. For example ${}^{i}a$ means that vector a is expressed in frame coordinates $i$ (coordinate system of link $i$). If the leading superscript is omitted then the vector is expressed in world coordinates. In addition a subscript is used to indicate the frame to which the vector points, e.g ${}^{i}a_{j}$ means a vector quantity relating to link j, but expressed in terms of coordinate frame i. Quantities in capital letters refer to 4x4 matrices and the same superscript and subscript notation is used.

## 3.2 Frame and link description

This section describes the construction of local and global coordinate systems for articulated bodies in the context of inverse Lagrangian dynamics.

A simple kinematic chain is considered where both end links of the chain, the root and the end-effector are free to move in space. Suppose that the kinematic chain is open and that there are n links in the chain. The root is labelled link 1 and the end-effector as link number n. Consecutive links are linked with joints of single degrees of freedom. The world coordinate frame is labelled link 0 by convention. For the purpose of dynamic analysis, a link is considered to be a rigid body which defines the relationship between two neighbouring joint axes of the articulated configuration. A joint connects two neighbouring links and the joints are numbered such that joint i connects links i-1 and i. This means that for links $0, 1, \ldots, n$ we have joints $1, 2, \ldots, n$.

In order to define the position of a link relative to its neighbouring links we need four numbers. This approach is well known in robotics and is called the *Denavit-Hartenberg* notation, Devanit & Hartenberg [55]. For any two axes in 3-D space, there is a well-defined distance measure between them, called the *link length* (denoted by quantity $a$) which is the distance measured along a line which is perpendicular to both axes. The second number is the *link twist*, which is evaluated by constructing a plane whose normal is perpendicular to both axes and measuring the angle, $\alpha$, between the projections of the axes on to the plane. This angle can be measured in the left or right sense as long as the direction is consistent among all joint axes. A coordinate system associated with each link can now be defined. This coordinate system is fixed at some point on the link, moves along with it and is defined as follows (see Fig 3.1):

$z_i$ : is directed along the axis of joint $i+1$

$x_i$ : is directed along the common normal between $z_{i-1}$ and $z_i$

$y_i$ : completes the right coordinate system

The link length and twist can now be defined as follows :

$a_i$ : distance between the origins of coordinate systems $i-1$ and $i$, measured along $x_i$

$\alpha_i$ : angle between the $z_{i-1}$ and $z_i$, measured in the right hand sense about $x_i$

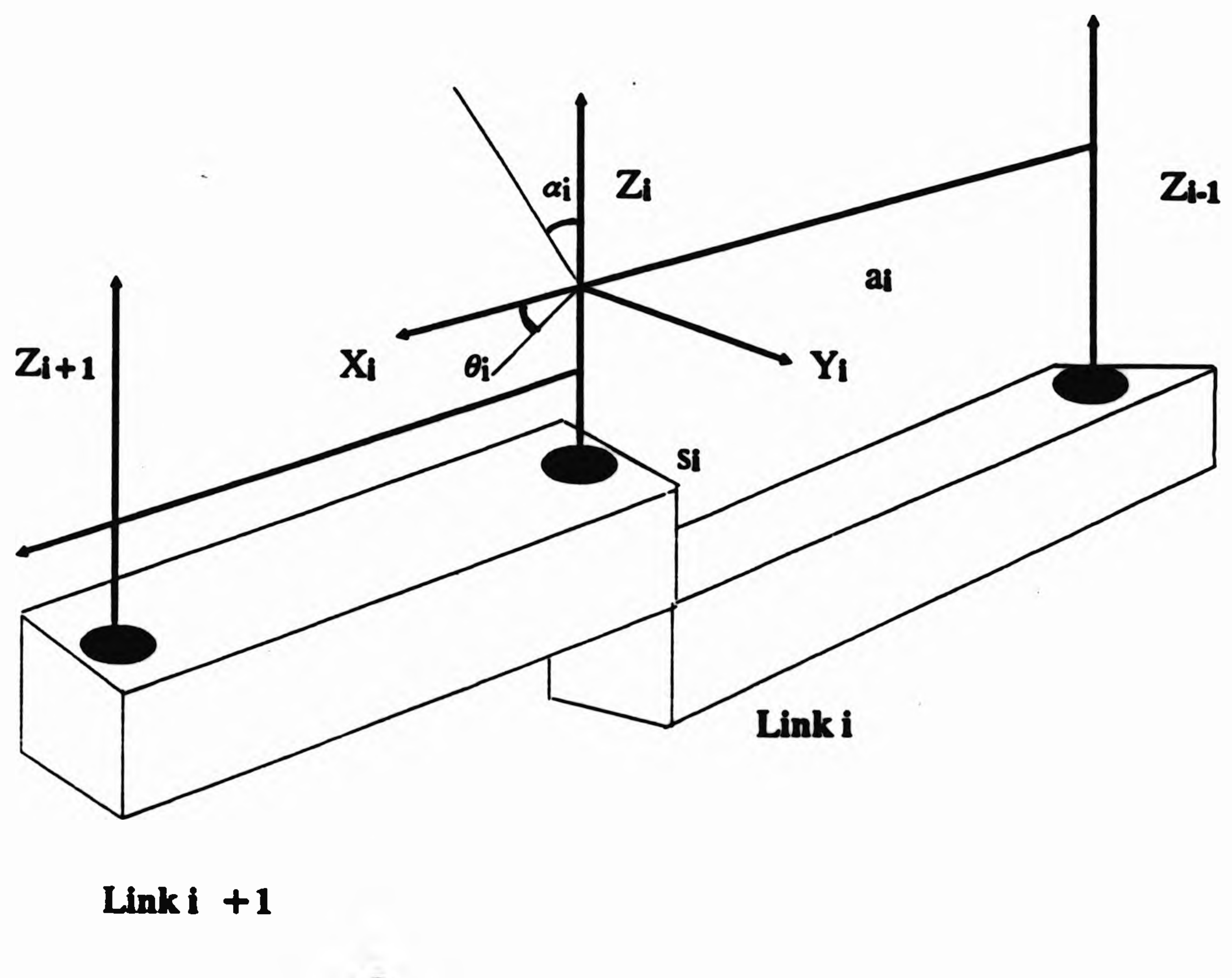


**Fig 3.1 : Construction of local coordinate systems.**

Two other quantities have to be defined to complete the notation :

$s_i$ : distance between $x_{i-1}$ and $x_i$ measured along $z_{i-1}$

$\theta_i$ : angle between $x_{i-1}$ and $x_i$ axes measured in the right hand sense about $z_{i-1}$

In the Lagrangian formulation, $s_i$ and $\theta_i$ are the joint variables - if joint i is rotational, then the variable is $\theta_i$, and if the joint is translational, the variable is $s_i$. Given the configuration of the particular articulated body, variables $a_i$ and $\alpha_i$ are invariant and known. We also define $\theta_i^0$ and $s_i^0$ to be the offset initial values of the joint variables when the figure is in the rest or neutral position and therefore $\theta_i$ and $s_i$ are the dispositions of the link from that rest position.

If ${}^i v_j = (1xyz)^T$ is a vector quantity relating to link j, but expressed in terms of coordinate frame i, then :

$$ {}^{i-1}v_j = A_i\, {}^i v_j \tag{3.1}$$

where $A_i$ is the transformation matrix between coordinate systems i-1 and i and is defined in the following way :

$$A_i = \begin{pmatrix} 1 & 0 & 0 & 0 \\ a_i\cos(\theta_i^0+\theta_i) & \cos(\theta_i^0+\theta_i) & -\sin(\theta_i^0+\theta_i)\cos(\alpha_i) & \sin(\theta_i^0+\theta_i)\sin(\alpha_i) \\ a_i\sin(\theta_i^0+\theta_i) & \sin(\theta_i^0+\theta_i) & \cos(\theta_i^0+\theta_i)\cos(\alpha_i) & -\cos(\theta_i^0+\theta_i)\sin(\alpha_i) \\ s_i^0+s_i & 0 & \sin(\alpha_i) & \cos(\alpha_i) \end{pmatrix}$$

We define $A_0 = I,\quad {}^iW_i = I,\quad W_i = A_1A_2\ldots A_i,$

$$ {}^iW_j = A_{i+1}A_{i+2}\ldots A_j \;\; for \;\; i < j \tag{3.2}$$

and ${}^i v_k = {}^iW_j\, {}^j v_k$. Therefore W reflects the compound effect of transformation matrices and can be used to change coordinates from one frame to another.

The choice of numbers for the D-H notation is not unique. For example we have to choose the sense of the joint axis, although which sense is selected is not important. When two adjacent joint axes intersect one another, we define axis x to have the direction of the normal to the plane formed by the joint axes. Again there are two choices as to which direction to take. Finally, when two adjacent joint axes are parallel, the position of the common normal can be arbitrarily defined, although it would usually be chosen such that the offset value of s is zero.

## 3.3 Derivation of inverse Lagrangian dynamics

This section covers the derivation of the Lagrangian formulation as it is used for robotics. The method is based on the principle of energy conservation. The coordinates systems are defined as described in the previous section - the reason for using local coordinate systems is to try to keep the inertia tensors invariant during the motion. The assumptions that were stated in the previous section are again true i. e the links are solid extended masses connected with single-degree-of-freedom joints, and the kinematic chains contain no loops or branches.

In Lagrangian dynamics, the calculation of the forces that apply to a given system requires knowledge of the configuration of the system, the accelerations and velocities, and assumes that the dynamic system has n independent variables which can be expressed as an n-vector $q = (q_1 q_2 \ldots q_n)$. In the case of articulated bodies, q is the displacement of the ith degree of freedom from its idle position. If the degree of freedom is rotational then $q_i$ represents $\theta_i$, and if it is translational then $q_i$ represents $s_i$. Hence, in the following derivation we assume :

$$q_i = \left[ \begin{array}{ll} \theta_i & \text{if degree of freedom is rotational} \\ s_i & \text{if degree of freedom is translational} \end{array} \right.$$

If L represents the Lagrangian, which is defined as the difference between the kinetic energy, K, and potential energy, P, i. e L = K - P, then the Lagrangian equations that give the forces are :

$$F_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \quad for \ i = 1, \ldots, n \tag{3.3}$$

Thus the first step is to derive the potential and kinetic energy of the system.

Let $h_i$ be a vector from the origin of the world coordinate system (link 0) to a point fixed in link i and let ${}^i h_i$ be a vector from the origin of frame i to the same point, then :

$$h_i = W_i \, {}^i h_i \tag{3.4}$$

By differentiating (3.4) w.r.t time, the velocity of the point is :

$$\dot{h}_i = \dot{W}_i \, {}^i h_i + W_i \, {}^i \dot{h}_i = \dot{W}_i \, {}^i h_i \tag{3.5}$$

The vector ${}^i h_i$ is directed to a point in link i w.r.t the origin of the coordinate system of link i and is therefore constant. Thus ${}^i \dot{h}_i$ is zero. The kinetic energy of the point (derived from the formula $K = \frac{1}{2} mv^2$) is then given by :

$$k_i = \frac{1}{2} tr \left( \dot{h}_i \dot{h}_i^T \right) dm \tag{3.6}$$

where dm is the mass of the point. From (3.5) and (3.6) :

$$k_i = \frac{1}{2} tr \left( \dot{W}_i \, {}^i h_i \, {}^i h_i^T \, \dot{W}_i^T \right) dm \tag{3.7}$$

Integrating (3.7) gives the kinetic energy of the entire link :

$$K_i = \int \frac{1}{2} tr \left( \dot{W}_i \, {}^i h_i \, {}^i h_i^T \, \dot{W}_i^T \right) dm \Rightarrow$$

$$k_i = \frac{1}{2} tr \left( \dot{W}_i \left[ \int {}^i h_i \, {}^i h_i^T \, dm \right] \dot{W}_i \right) \tag{3.8}$$

This is because $\dot{W}_i$ does not depend upon the mass distribution of the link. Let

$$J_i = \int \left( {}^i h_i \, {}^i h_i^T \right) dm$$

where $J_i$ is the inertia tensor with respect to frame i. Now (3.5) can be written as :

$$K_i = \frac{1}{2} tr \left( \dot{W}_i \, J_i \, \dot{W}_i^T \right) dm \tag{3.9}$$

which gives the kinetic energy of the ith link. The kinetic energy of the entire system is the sum of the kinetic energies of all links :

$$K = \sum_{j=1}^{n} K_j = \frac{1}{2} \sum_{j=1}^{n} tr \left( \dot{W}_j \, J_j \, \dot{W}_j^T \right) \tag{3.10}$$

The potential energy of the system depends upon the gravitational force only and is equal to the work required to displace the centre of mass of each link from the horizontal reference plane. If $g$ is the gravitational vector $g = (0 \; g_x \; g_y \; g_z)$ and ${}^i r_i$ is the coordinate position of the centre of mass of link i expressed in link i coordinates, then the potential energy of the system is :

$$P = C - \sum_{j=i}^{n} m_j \, g^T \, W_j \, {}^j r_j \tag{3.11}$$

where C is a constant. From the equations of Lagrange we have :

$$F_i = \frac{d}{dt} \left( \frac{\partial}{\partial \dot{q}_i} (K - P) \right) - \frac{\partial}{\partial q_i} (K - P) \tag{3.12}$$

But the potential energy as shown in (3.11) does not depend upon the velocity therefore:

$$\frac{\partial P}{\partial \dot{q}_i} = 0$$

and (3.12) can be simplified to :

$$F_i = \frac{d}{dt} \left( \frac{\partial K}{\partial \dot{q}_i} \right) - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i} \tag{3.13}$$

We now start calculating the terms on the RHS of (3.13).

### 3.3.1 Calculation of the term $\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right)$

From (3.10) we have :

$$\frac{\partial K}{\partial \dot{q}_i} = \frac{1}{2}\sum_{j=i}^{n} \frac{\partial}{\partial \dot{q}_i}\left(tr\left(\dot{W}_j J_j \dot{W}_j^T\right)\right) \tag{3.14}$$

We notice that the index of the summation changes to include only the links from i to n. This is because only these links contribute to the kinetic energy with respect to $\dot{q}_i$. This is because ${}^iW_j$ is not defined for $i > j$ and therefore the derivatives of the terms $\dot{W}_j$ and $\ddot{W}_j$ w.r.t $q_i$ and $\dot{q}_i$ are defined for only $j \geq i$.

The order of the trace and the partial derivative in (3.14) can be interchanged to give

$$\frac{\partial K}{\partial \dot{q}_i} = \frac{1}{2}\sum_{j=i}^{n} tr\left(\frac{\partial}{\partial \dot{q}_i}\left(\dot{W}_j J_j \dot{W}_j^T\right)\right)$$

By also using that $tr(A + B) = tr(A) + tr(B) \Rightarrow$

$$\frac{\partial K}{\partial \dot{q}_i} = \frac{1}{2}\sum_{j=i}^{n}\left[tr\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \dot{W}_j^T\right) + tr\left(\dot{W}_j \frac{\partial J_j}{\partial \dot{q}_i} \dot{W}_j^T\right) + tr\left(\dot{W}_j J_j \frac{\partial \dot{W}_j^T}{\partial \dot{q}_i}\right)\right] \tag{3.15}$$

But $\frac{\partial J_i}{\partial \dot{q}_j} = 0$ since $J_j$ does not depend upon the velocity of the link and $tr(A) = tr(A^T)$ gives :

$$tr\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \dot{W}_j^T\right) = tr\left(\left[\frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \dot{W}_j^T\right]^T\right) = tr\left(\dot{W}_j J_j^T \frac{\partial \dot{W}_j^T}{\partial \dot{q}_i}\right) = tr\left(\dot{W}_j J_j \frac{\partial \dot{W}_j^T}{\partial \dot{q}_i}\right)$$

since $J_j$ is symmetric. Thus

$$\frac{\partial K}{\partial \dot{q}_i} = \sum_{j=i}^{n} tr\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \dot{W}_j^T\right) \tag{3.16}$$

Hence, using the fact that $dJ/dt = 0$

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) = \sum_{j=i}^{n}\left[tr\left(\frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right)J_j\dot{W}_j^T\right) + tr\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}J_j\ddot{W}_j^T\right)\right] \tag{3.17}$$

**We also seek to prove that :**

$$\frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right) = \frac{\partial \dot{W}_j}{\partial q_i}$$

**but we need two results first.**

**Lemma 1** *: Prove that*

$$\frac{\partial \dot{W}_j}{\partial \dot{q}_i} = \frac{\partial W_j}{\partial q_i} \; for \; i \le j.$$

**Proof : By definition $W_j = A_iA_2 \ldots A_j$. Since $q_1 \ldots q_n$ are independent $\frac{\partial A_i}{\partial q_j} = 0$ for $i \ne j$ and hence :**

$$\frac{\partial W_j}{\partial q_i} = A_1 \ldots \frac{\partial A_i}{\partial q_i} \ldots Aj \; for \; i \le j \; [= 0 \; for \; i > j\,] \tag{3.18}$$

**Also :**

$$\dot{W}_j = \dot{A}_1A_2\ldots A_j + A_1\dot{A}_2\ldots A_j + \ldots\ldots + A_1A_2\ldots A_{j-1}\dot{A}_j \tag{3.19}$$

**Differentiating (3.19) with respect to $\dot{q}_i$ yields :**

$$\frac{\partial \dot{W}_j}{\partial \dot{q}_i} = A_1 \ldots \frac{\partial \dot{A}_i}{\partial \dot{q}_i} \ldots Aj \; for \; i \le j \tag{3.20}$$

**Also :**

$$\dot{A}_i = \frac{\partial A_i}{\partial q_i}\dot{q}_i \tag{3.21}$$

**which implies that :**

$$\frac{\partial \dot{A}_i}{\partial \dot{q}_i} = \frac{\partial}{\partial \dot{q}_i}\left(\frac{\partial A_i}{\partial q_i}\right)\dot{q}_i + \frac{\partial A_i}{\partial q_i} \tag{3.22}$$

**but $\frac{\partial}{\partial \dot{q}_i}(\frac{\partial A_i}{\partial q_i}) = 0$ because $\frac{\partial A_i}{\partial q_i}$ is independent of $\dot{q}_i$ so from (3.22) $\Rightarrow$**

$$\frac{\partial \dot{A}_i}{\partial \dot{q}_i} = \frac{\partial A_i}{\partial q_i} \tag{3.23}$$

Substituting (3.23) int (3.20) and comparing with (3.18) gives :

$$\frac{\partial \dot{W}_j}{\partial \dot{q}_i} = \frac{\partial W_j}{\partial q_i} \tag{3.24}$$

**Lemma 2** *: Show that*

$$\frac{\partial \dot{W}_j}{\partial q_i} = \frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right)$$

Proof : From (3.19) we have :

$$\frac{\partial \dot{W}_j}{\partial q_i} = \dot{A}_1 \ldots \frac{\partial A_i}{\partial q_i} \ldots A_j + \ldots + A_1 A_2 \ldots \frac{d}{dt}\left(\frac{\partial \dot{A}_i}{\partial q_i}\right) \ldots A_j + \ldots + A_1 \ldots \frac{\partial A_i}{\partial q_i} \ldots \dot{A}_j$$

and also from (3.20) :

$$\frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right) = \dot{A}_1 \ldots \frac{\partial \dot{A}_i}{\partial \dot{q}_i} \ldots A_j + \ldots + A_1 A_2 \ldots \frac{d}{dt}\left(\frac{\partial \dot{A}_i}{\partial \dot{q}_i}\right) \ldots A_j + \ldots + A_1 \ldots \frac{\partial \dot{A}_i}{\partial \dot{q}_i} \ldots \dot{A}_j \tag{3.25}$$

Comparing these equations, it can been seen from (3.23) that :

$$\frac{\partial \dot{W}_j}{\partial q_i} = \frac{d}{dt}\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i}\right) \tag{3.26}$$

Using the two lemmas, (3.17) becomes :

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) = \sum_{j=i}^{n} tr\left[\frac{\partial W_j}{\partial q_i} J_j \ddot{W}_j^T + \frac{\partial \dot{W}_j}{\partial q_i} J_j \dot{W}_j^T\right] \tag{3.27}$$

### 3.3.2 Calculation of term $\frac{\partial K}{\partial q_i}$

From eqn (3.10) this term can be written as:

$$\frac{\partial K}{\partial q_i} = \frac{1}{2}\sum_{j=i}^{n} tr\left(\frac{\partial \dot{W}_j}{\partial q_i} J_j \dot{W}_j^T + \dot{W}_j J_j \frac{\partial \dot{W}_j^T}{\partial q_i}\right) = \sum_{j=i}^{n} tr\left(\frac{\partial \dot{W}_j}{\partial q_i} J_j \dot{W}_j^T\right) \qquad (3.28)$$

### 3.3.3 Calculation of term $\frac{\partial P}{\partial q_i}$

From eqn (3.11) :

$$\frac{\partial P}{\partial q_i} = -\sum_{j=i}^{n} m_j\, g^T \frac{\partial W_j}{\partial q_i}\, {}^j r_j \qquad (3.29)$$

However, $W_j = W_i\, {}^iW_j$ and since ${}^iW_j$ does not depend upon $q_i$,

$$\frac{\partial W_j}{\partial q_i} = \frac{\partial W_i}{\partial q_i}\, {}^iW_j \qquad (3.30)$$

Therefore this term is written as :

$$\frac{\partial P}{\partial q_i} = -g^T \frac{\partial W_i}{\partial q_i} \sum_{j=i}^{n} m_j\, {}^iW_j\, {}^j r_j \qquad (3.31)$$

### 3.3.4 Combining the terms for evaluating the forces

Substituting back in (3.13) from eqn (3.27), (3.28) and (3.31) :

$$F_i = tr\left[\frac{\partial W_i}{\partial q_i} \sum_{j=i}^{n} {}^iW_j J_j \ddot{W}_j^T\right] - g^T \frac{\partial W_i}{\partial q_i} \sum_{j=i}^{n} m_j\, {}^iW_j\, {}^j r_j \qquad (3.32)$$

### 3.3.5 Recursive calculation of forces

The calculation of the various terms of (3.32) can be simplified by using recursive formulae which calculate the terms from the end-effector towards the root. Let :

$$c_i = \sum_{j=i}^{n} m_j\, {}^iW_j\, {}^jr_j \quad and \quad D_i = \sum_{j=i}^{n} {}^iW_j J_j \ddot{W}_j^T \tag{3.33}$$

then (3.32) yields :

$$F_i = tr\left(\frac{\partial W_i}{\partial q_i} D_i\right) - g^T \frac{\partial W_i}{\partial q_i} c_i \tag{3.34}$$

$c_i$ can be recursively calculated :

$$c_i = \sum_{j=i}^{n} m_j\, {}^iW_j\, {}^jr_j = m_i\, {}^iW_i\, {}^ir_i + \sum_{j=i+1}^{n} m_j\, {}^iW_j\, {}^jr_j =$$
$$m_i\, {}^ir_i + A_{i+1} \sum_{j=i+1}^{n} m_j\, {}^{i+1}W_j\, {}^jr_j$$

i.e

$$c_i = m_i\, {}^ir_i + A_{i+1} c_{i+1} \tag{3.35}$$

A similar treatment for D leads to

$$D_i = {}^iW_i J_i \ddot{W}_i^T + \sum_{j=i+1}^{n} {}^iW_j J_j \ddot{W}_j^T$$

and since ${}^iW_i = I$

$$D_i = J_i \ddot{W}_i^T + A_{i+1} D_{i+1} \tag{3.36}$$

Therefore the forces can be calculated from (3.34) and terms c and D from (3.35) and (3.36) respectively.

## 3.4 Calculation of the transformation matrices and their derivatives

The calculation of the term D in eqn (3.36) involves the quantity $\ddot{W}_i$. This section produces expressions for the evaluation of $\ddot{W}_i$. The derivation of the term yields a recursive expression which involves the terms $W_i$ and $\dot{W}_i$ ( which in turn are evaluated recursively).

By definition $W_j = A_1 A_2 \ldots A_j$ which can be written as :

$$W_j = W_{j-1} A_j \tag{3.37}$$

in which form it can be used to calculate $\ddot{W}_j$. By differentiating (3.37) w.r.t time we have :

$$\dot{W}_j = \dot{W}_{j-1} A_j + W_{j-1} \dot{A}_j \tag{3.38}$$

and from (3.21), this becomes :

$$\dot{W}_j = \dot{W}_{j-1} A_j + W_{j-1} \frac{A_j}{q_j} \dot{q}_j \tag{3.39}$$

By differentiating (3.40) we get :

$$\begin{aligned} \ddot{W}_j = {} & \ddot{W}_{j-1} A_j + \dot{W}_{j-1} \dot{A}_j + \dot{W}_{j-1} \frac{\partial A_j}{\partial q} \dot{q}_j + \\ & W_{j-1} \frac{d}{dt}\left(\frac{\partial A_j}{\partial q_j}\right) \dot{q}_j + W_{j-1} \frac{\partial A_j}{\partial q_j} \ddot{q}_j \end{aligned} \tag{3.40}$$

By using :

$$\frac{d}{dt}\left(\frac{\partial A_j}{\partial q_j}\right) = \frac{\partial^2 A_j}{\partial q_j^2} \dot{q}_j \tag{3.41}$$

together with (3.21), (3.40) can be written :

$$\ddot{W}_j = \ddot{W}_{j-1}A_j + 2\dot{W}_{j-1}\frac{\partial A_j}{\partial q_j}\dot{q}_j + \dot{W}_{j-1}\frac{\partial^2 A_j}{\partial q_j^2}\dot{q}_j^2 + W_{j-1}\frac{\partial A_j}{\partial q_j}\ddot{q}_j \tag{3.42}$$

Finally :

$$W_j = A_1A_2\ldots A_j \Rightarrow$$

$$\frac{\partial W_j}{\partial q_j} = \frac{\partial}{\partial q_j}(A_1A_2\ldots A_j) \textit{ and since } q_1\ldots q_n \textit{ are independent } :$$

$$\frac{\partial W_j}{\partial q_j} = A_1A_2\ldots A_{j-1}\frac{\partial A_j}{\partial q_j} = W_{j-1}\frac{\partial A_j}{\partial q_j} \tag{3.43}$$

Expressions (3.37), (3.42) and (3.43), together with (3.35) and (3.36), can be used to calculate the parameters required for finding the generalized forces. Section 3.6 deals with the evaluation of the derivatives of transformation matrices A.

## 3.5 Converting from global coordinates to joint coordinates

The centres of masses of the links would usually be given in global coordinates because it is easier for the user. In this case, the local coordinates can be found with the help of the inverses of the transformation matrices as follows :

$$r_j = W_j\,{}^jr_j \Rightarrow {}^jr_j = W_j^{-1}r_j.$$

Hence the $W_j^{-1}$ have to be calculated. Since $W_j = W_{j-1}A_j$, it follows :

$$W_j^{-1} = A_j^{-1}W_{j-1}^{-1} \tag{3.44}$$

where $W_1^{-1} = A_1^{-1}$.

Matrices $A_j$ are constructed, by successive transformations as follows :

- x-rotation by angle $\alpha$, represented by matrix $M_1$

- z-rotation by angle $\theta$, represented by matrix $M_2$
- translation by $(a\cos\theta, a\sin\theta, s)$, represented by matrix $M_3$ .

Therefore $A = M_1 M_2 M_3$ and hence the inverse of A is given by :

$$A^{-1} = M_3^{-1} M_2^{-1} M_1^{-1}$$

The inverse of matrix $M_1$ can be found by replacing $\alpha$ by $-\alpha$ in the matrix. This will give a rotation of the same magnitude but opposite direction to the original one. Similarly, in $M_2$ , $\theta$ is replaced by $-\theta$. Finally, the inverse of $M_3$ is a translation by $(-a\cos\theta, -a\sin\theta, -s)$. Therefore the inverse of A can be found to be :

$$A_i^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -a_i & \cos(\theta_i^0 + \theta_i) & \sin(\theta_i^0 + \theta_i) & 0 \\ -(s_i^0 + s_i)\sin\alpha_i & -\sin(\theta_i^0 + \theta_i)\cos\alpha_i & \cos(\theta_i^0 + \theta_i)\cos\alpha_i & \sin\alpha_i \\ -(s_i^0 + s_i)\cos\alpha_i & -\sin(\theta_i^0 + \theta_i)\sin\alpha_i & -\cos(\theta_i^0 + \theta_i)\sin\alpha_i & \cos\alpha_i \end{pmatrix}$$

## 3.6 Evaluation of the derivatives of transformation matrices

The derivatives of $A_i$ with respect to $q_i$ are given by :

- for a translational degree of freedom i :

$$\frac{\partial A_i}{\partial q_i} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

-for a rotational degree of freedom i :

$$\frac{\partial A_i}{\partial q_i} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -a_i \sin(\theta_i^0 + \theta_i) & -\sin(\theta_i^0 + \theta_i) & -\cos(\theta_i^0 + \theta_i)\cos\alpha_i & \cos(\theta_i^0 + \theta_i)\sin\alpha_i \\ a_i \cos(\theta_i^0 + \theta_i) & \cos(\theta_i^0 + \theta_i) & -\sin(\theta_i^0 + \theta_i)\cos\alpha_i & \sin(\theta_i^0 + \theta_i)\sin\alpha_i \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

The second derivatives of $A_i$ are :

- for a translational degree of freedom i :

$$\frac{\partial^2 A_i}{\partial q_i} = 0$$

-for a rotational degree of freedom i :

$$\frac{\partial^2 A_i}{\partial q_i^2} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -a_i \cos(\theta_i^0 + \theta_i) & -\cos(\theta_i^0 + \theta_i) & \sin(\theta_i^0 + \theta_i)\cos\alpha_i & -\sin(\theta_i^0 + \theta_i)\sin\alpha_i \\ -a_i \sin(\theta_i^0 + \theta_i) & -\sin(\theta_i^0 + \theta_i) & -\cos(\theta_i^0 + \theta_i)\cos\alpha_i & \cos(\theta_i^0 + \theta_i)\sin\alpha_i \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

## 3.7 Calculation of inertia tensors

The calculation of the inertia tensors of a body depends upon two factors :

- the shape and
- the mass distribution of the body.

The shape of the links of the human body can be approximated by using cylinders or boxes - we shall use boxes. More sophisticated shapes can be considered at the expense of more complicated calculations for the inertia tensors. For the sake of simplicity, we also assume uniform mass distribution. The inertia tensors are calculated w.r.t their local coordinate systems and, are therefore constant throughout the motion.

The pseudo 4x4 inertia tensors, are defined in terms of the proper 3x3 inertia tensors, masses of the body and the position of the centres of mass as being :

$$J_i = \begin{pmatrix} m_i & m_i\,{}^i r_i^T \\ m_i\,{}^i r_i & {}^i I_i \end{pmatrix}$$

Where ${}^iI_i$ is the 3x3 inertia tensor of link i w.r.t its origin :

$${}^iI_i = \begin{pmatrix} a_i & {}^iI_{ixy} & {}^iI_{ixz} \\ {}^iI_{ixy} & b_i & {}^iI_{iyz} \\ {}^iI_{ixz} & {}^iI_{iyz} & c_i \end{pmatrix} \quad and$$

$$a_i = \frac{{}^iI_{iyy} + {}^iI_{izz} - {}^iI_{ixx}}{2}$$
$$b_i = \frac{{}^iI_{ixx} + {}^iI_{izz} - {}^iI_{iyy}}{2}$$
$$c_i = \frac{{}^iI_{ixx} + {}^iI_{izz} - {}^iI_{izz}}{2}$$

where for a given link i :

$$I_{xx} = \int (y^2 + z^2) dm, \; I_{yy} = \int (x^2 + z^2) dm, \; I_{zz} = \int (x^2 + y^2) dm$$
$$I_{xy} = \int xy\, dm, \; I_{xz} = \int xz\, dm, \; I_{yz} = \int yz\, dm$$

Consider the shape in Fig 3.2. The volume of the box is $\alpha\beta\gamma$. Let the mass of the link be M, and consider a small element of the box with dimensions $\delta x, \delta y$ *and* $\delta z$. The mass of this element is :

$$\delta m = \frac{\delta x \delta y \delta z}{\alpha\beta\gamma} \quad and\ therefore$$

$$\int x^2\, dm = \frac{M}{\alpha\beta\gamma} \int_{-\frac{\gamma}{2}}^{\frac{\gamma}{2}} dy \int_{-\frac{\beta}{2}}^{\frac{\beta}{2}} dz \int_0^{\alpha} x^2\, dx = \frac{2M\alpha^2}{3}, \; and$$

$$\int y^2\, dm = \frac{M}{\alpha\beta\gamma} \int_{-\frac{\gamma}{2}}^{\frac{\gamma}{2}} y^2\, dy \int_{-\frac{\beta}{2}}^{\frac{\beta}{2}} dz \int_0^{\alpha} dx = \frac{M\gamma^2}{12}, \; and\ similarly$$

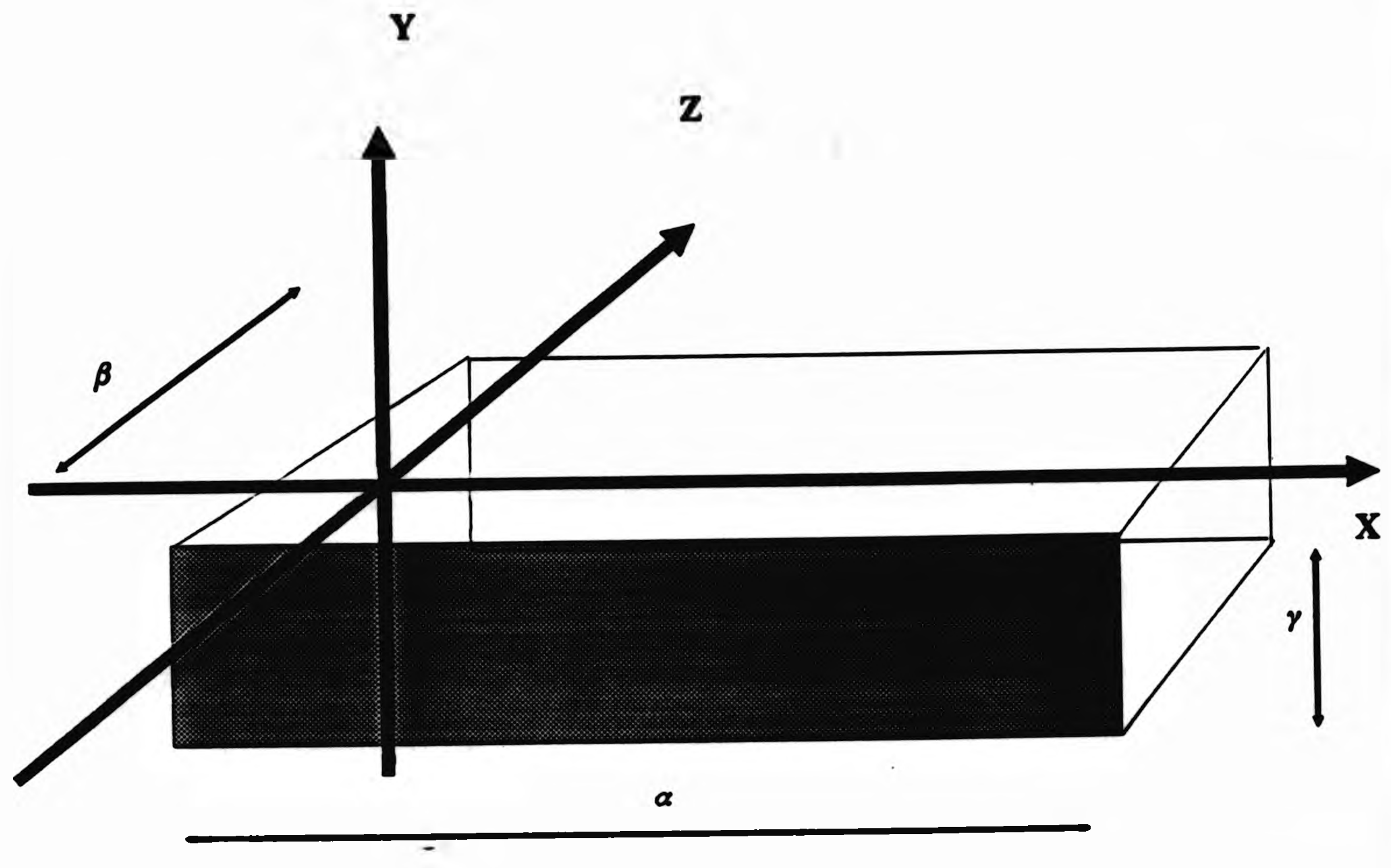


**Fig 3.2 : Calculation of inertia tensors.**

$$\int z^2\,dm = \frac{M\beta^2}{12}$$

Hence :

$$I_{xx} = \frac{M(\beta^2+\gamma^2)}{12},\ I_{yy} = \frac{M(8\alpha^2+\beta^2)}{12},\ \text{and } I_{zz} = \frac{M(8\alpha^2+\gamma^2)}{12}\ \text{so}$$

$$a = \frac{2M\alpha^2}{3}\quad b = \frac{M\gamma^2}{12}\quad c = \frac{M\beta^2}{12}$$

It can also be easily shown that :

$$Ixy = \int xy = \frac{M\alpha\gamma}{4},\ Ixz = \int xz = \frac{M\alpha\beta}{4},\ Iyz = \int zy = \frac{M\beta\gamma}{16}.$$

This defines fully the pseudo 4x4 inertia tensors since the masses and the positions of the centre of masses are clearly defined.

## 3.8 Boundary conditions

When the various quantities involved in the formulation are calculated, some consideration has to be taken for the calculation of the terms in the root and end effector of each chain. There are two types of boundary conditions, one for the quantities that are calculated using forward recursion, and one for the quantities which are calculated using backward recursion.

- *Forward recursion*

$$W_0 = I\ \text{and so}\ \dot{W}_0 = 0\ \text{and}\ \ddot{W}_0 = 0.$$

Thus since $W_j = W_{j-1}A_j$, we see that $W_1 = W_0A_1 \Rightarrow W_1 = A_1$.

Also

$$\dot{W}_j = \dot{W}_{j-1}A_j + W_{j-1}\frac{\partial A_j}{\partial q_j}\dot{q}_j \Rightarrow \dot{W}_1 = \frac{\partial A_1}{\partial q_1}\dot{q}_1\ \text{and}$$

$$\ddot{W}_j = \ddot{W}_{j-1}A_j + \frac{\partial A_j}{\partial q_j}\dot{q}_j + W_{j-1}\frac{\partial^2 A_j}{\partial q_j^2}\dot{q}_j^2 + W_{j-1}\frac{\partial A_j}{\partial q_j}\ddot{q}_j \quad \textit{thus}$$

$$\ddot{W}_1 = \frac{\partial^2 A_1}{\partial q_1^2}\dot{q}_1^2 + \frac{\partial A_1}{\partial q_1}\ddot{q}_1$$

- ***Backward recursion***

**These boundary conditions refer to the evaluation of terms c and D, and they occur at the end-effectors.**

$$c_i = m_i\,{}^i r_i + A_{i+1}\,c_{i+1} \Rightarrow c_n = m_n\,{}^n r_n + A_{n+1}\,c_{n+1}$$

**if we define $c_{n+1} = 0$ then $c_n = m_n\,{}^n r_n$. This is because the last rationalised link in the chain will always carry mass.**

**$D_i = J_i\ddot{W}_i^T + A_{i+1}D_{i+1}$. We define $D_{n+1} = 0$ and thus $D_n = J_n\ddot{W}_n^T$.**

**Backward recursion transmits geometrical information while the forward recursion transmits information about the forces, although admittedly the distinction is blurred.**

## 3.9 Conclusions

**In this chapter, the original inverse Lagrangian formulation developed by Hollerbach was presented. This solves for the forces, given the accelerations, velocities and positions of the links. It was derived from first principles by using the law of energy conservation and expressions for evaluating the forces were produced. We then introduced two terms, $c_i$ and $D_i$, which were used to yield recursive formulae for the calculation of forces. Techniques for the evaluation of the derivatives of the transformation matrices were also given. The way of transforming from global to local coordinate systems was presented since most physical quantities are given - by the user - in global coordinates but the method works in local frame coordinates. Finally, we showed a way of calculating the inertia tensors expressed in local frames using box-like shapes for the links.**

**The original inverse Lagrangian formulation by Uicker [65], required approximately :**

$$32n^4 + 86n^3 + 171n^2 + 53n - 128 \text{ multiplications and}$$

$25n^4 + 66n^3 + 129n^2 + 42n - 96$ additions.

This means that the method is $O(n^4)$ which makes it highly inefficient for computational purposes. Hollerbach's inverse Lagrangian formulation used forward and backward recursion to achieve computational savings and arrives at the same equations with :

$830n - 592$ muliplications and

$675n - 464$ additions

Li [88] presented an even faster Lagrangian formulation for robotic mechanisms but its suitability for the purpose of articulated body animation has not been tested as yet.

The method presented in this chapter is based on two assumptions :

- the kinematic chains are open and contain no branches
- the links are connected by single-degree-of-freedom joints.

These two assumptions are very restrictive and prohibit the application of the formulation to articulated figures in computer animation, since most articulated bodies in the real world contain both kinematic chains with branches and links which are connected by multiple-degree-of-freedom joints. It is, therefore, our task to modify the original formulation to deal with such models. In the next chapter, these adaptations are discussed.

# Chapter 4

# Adaptations For The Purpose Of Computer Animation

## 4.1 Introduction

In the previous chapter, the formulation of inverse Lagrangian dynamics for single-degree-of-freedom joints and branchless kinematic chains was presented. Clearly, in this form the formulation can not be applied directly to human figure animation and accordingly it has to be modified to accommodate kinematic chains with both branches and multiple-degree-of-freedom joints. This chapter deals with these adaptations to the Lagrangian formulation.

This is the first time that such an effort has been made as the method was previously used only for robotic mechanisms. Therefore our aim here is to produce the rigorous mathematical expressions that extend the method to articulated body animation. When adjusting the formulation to meet our needs, the basic principle of energy conservation should not be violated. The adaptations to the formulation do not require its derivation from first principles but it introduces changes as to how the recursive formulae for calculating the forces are used.

This chapter also presents the final algorithm that can be used to solve for forces for any articulated body which contains kinematic chains without loops (open kinematic chains). We first present the algorithm as it is used for robotic mechanisms and then its modified version for articulated bodies using structured pseudo code. Finally, we apply the algorithm in an experimental model. By doing this we illustrate how local coordinate systems are constructed and how to tackle multiple-degree-of-freedom joints and branches. The next two sections deal with the adaptations to the formulation for articulated bodies.

## 4.2 Inverse Lagrangian dynamics for kinematic chains with branches

Articulated bodies usually contain branches and the method described above applies only to kinematic chains without branches. The formulation can be extended to include branches by considering equation (3.10):

$$K = \sum_{j=1}^{n} tr\left(W_j J_j \dot{W}_j^T\right)$$

In order to find the term $\frac{\partial K}{\partial \dot{q}_i}$, equation (3.16) yielded :

$$\frac{\partial K}{\partial \dot{q}_i} = \sum_{j=i}^{n} tr\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \dot{W}_j^T\right)$$

Consider the model in Fig 4.1 where there is a branch at link 2. For link 1 :

$$\frac{\partial K}{\partial \dot{q}_1} = \sum_{j=1}^{8} tr\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \dot{W}_j^T\right)$$

which means that the kinetic energies of links $2, 3, \ldots, 8$ will be considered, but notice that for link 3 :

$$\frac{\partial K}{\partial \dot{q}_3} = \sum_{j=3}^{8} tr\left(\frac{\partial \dot{W}_j}{\partial \dot{q}_i} J_j \dot{W}_j^T\right)$$

which is not correct. Although the expression is correct the numbering convention is wrong, i.e links 6 to 8 should not be included in the calculation.

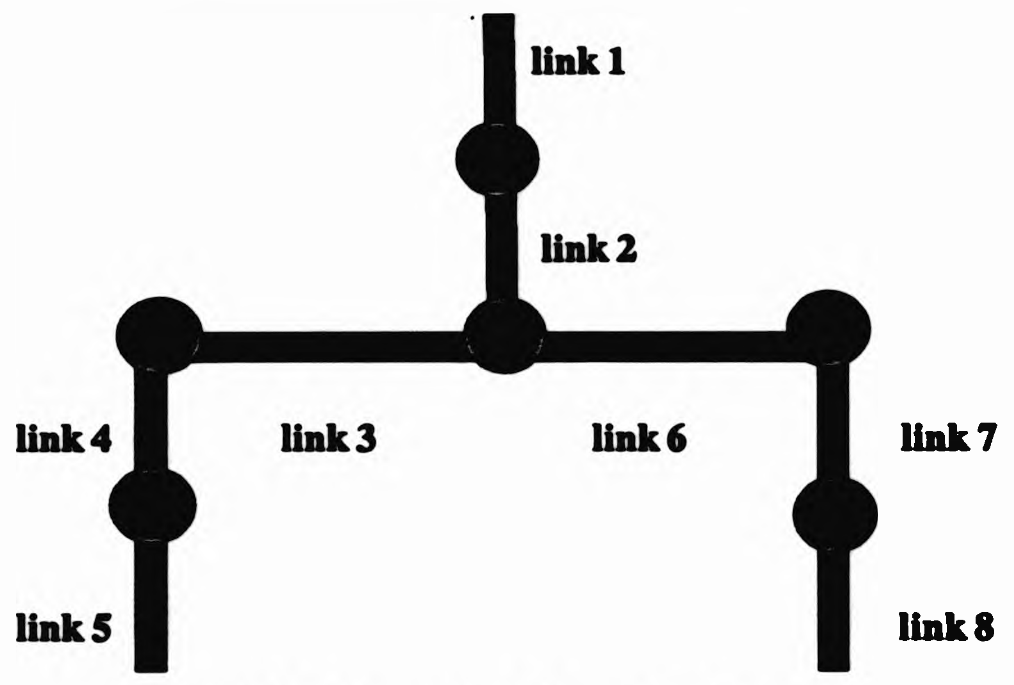


**Fig 4.1 : A branched kinematic chain.**

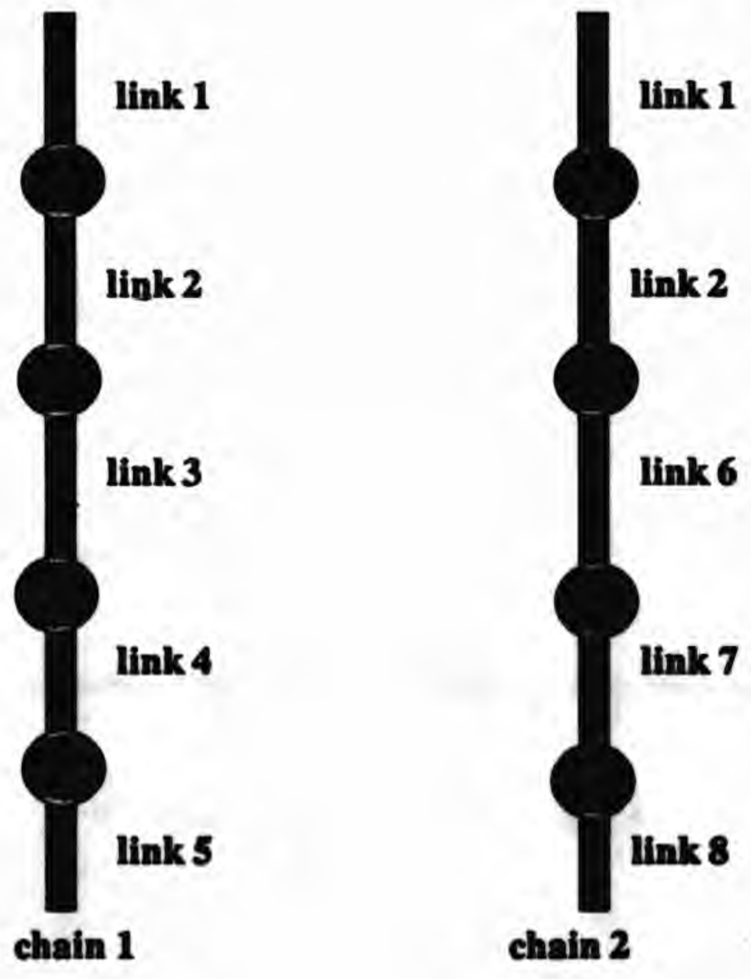


**Fig 4.2 : Decomposition of the chain in fig 4.1 into branchless kinematic chains.**

We can deduce a simple result from this, that the partial derivatives of the kinetic and potential energy with respect to the velocity and position vectors depend upon links that are situated at lower levels at the tree hierarchy and within the same kinematic chain. Therefore the formulation holds if the links are numbered in such a way that this is true. However as can be seen from Fig 4.1, links at the same level in the hierarchy can not be numbered in a way that suits this convention. The solution to that problem is to break the original branched kinematic chain into two kinematic chains without branches which start from the root and work towards the end-effectors. Hence there has to the same number of kinematic chains as end-effectors, unless the root is an end effector in which case there is one fewer. Thus the kinematic chain in Fig 4.1 can be split into two kinematic chains as in Fig 4.2.

In order to find the force that is applied to link 1 we shall calculate the forces in chains 1 and 2, ${}^1F_1$ and ${}^2F_1$ respectively, and then calculate the total force:

$$F_1 = {}^1F_1 + {}^2F_1.$$

However, in calculating this we consider links 1 and 2 twice which is not correct. Hence care has to be taken to include only once the links that are in both chains. Consider the first chain, which consists of links 1,2,3,4 and 5. In calculating c we have :

$$\begin{aligned}
c_5 &= m_5\,{}^5r_5 \\
c_4 &= m_4\,{}^4r_4 + A_5c_5 \\
c_3 &= m_3\,{}^3r_3 + A_4c_4 \\
{}^1c_2 &= m_2\,{}^2r_2 + A_3c_3 \\
{}^1c_1 &= m_1\,{}^1r_1 + A_2\,{}^1c_2
\end{aligned}$$

For the second chain (links 1,2,6,7 and 8) :

$$\begin{aligned}
c_8 &= m_8\,{}^8r_8 \\
c_7 &= m_7\,{}^7r_7 + A_8c_8 \\
c_6 &= m_6\,{}^6r_6 + A_7c_7 \\
{}^2c_2 &= A_6c_6 \\
{}^2c_1 &= A_2\,{}^2c_2
\end{aligned}$$

Hence $c_1 = {}^1c_1 + {}^2c_1$ and $c_2 = {}^1c_2 + {}^2c_2$.

The same approach is followed for the calculation of D as well. In this way the masses are included only once.

## 4.3 Multiple degrees of freedom

The extension to multiple degrees of freedom can be done by breaking the multiple-degree-of-freedom joints into as many single-degree-of-freedom joints as there are degrees of freedom. These single-degree-of-freedom joints are connected with links of which only one will have mass and dimensions (see Fig 4.3). The root has six degrees of freedom, three rotational and three translational, whereas all the other links have three rotational degrees of freedom.

In order to avoid confusion, we shall refer to the kinematic chains that contains links connected with single-degree-of-freedom joints - after the decomposition has taken place - as *rationalised chains*. We shall also use the term *prototype chain* to refer to kinematic chains before decomposition i.e with multiple degrees of freedom and branches. The terms 'rationalised' and 'prototype' can refer to links or joints that are part of a rationalised or a prototype kinematic chain. Therefore, for n prototype links, there are, at most, 3(n+1) degrees of freedom with as many rationalised links but only n rationalised links will have mass.

The problem that arises from such a configuration is which of the rationalised links carries the mass. In order to understand the problem consider a single link with six degrees of freedom, as shown in Fig 4.4. The link is decomposed into six links connected by single-degree-of-freedom joints, and theoretically, any of these links can carry the mass (Fig 4.5). Assume that the mass is assigned to any link other than the last one, then, the last link has no mass and hence : $c_6 = A_7 c_7$, but $c_7 = 0$ and $A_7$ does not exist, because there is no seventh link and therefore there is no transformation matrix relating link number 6 with link number 7, so $c = 0$ no matter what the position of the link. This is certainly incorrect and it means that the link that carries the mass should always be the last one further from the root. This determines the way the terms $c_i$ and $D_i$ are calculated. Therefore, according to this convention the kinematic chain in Fig 4.4 is decomposed into the rationalised chain in Fig 4.5.

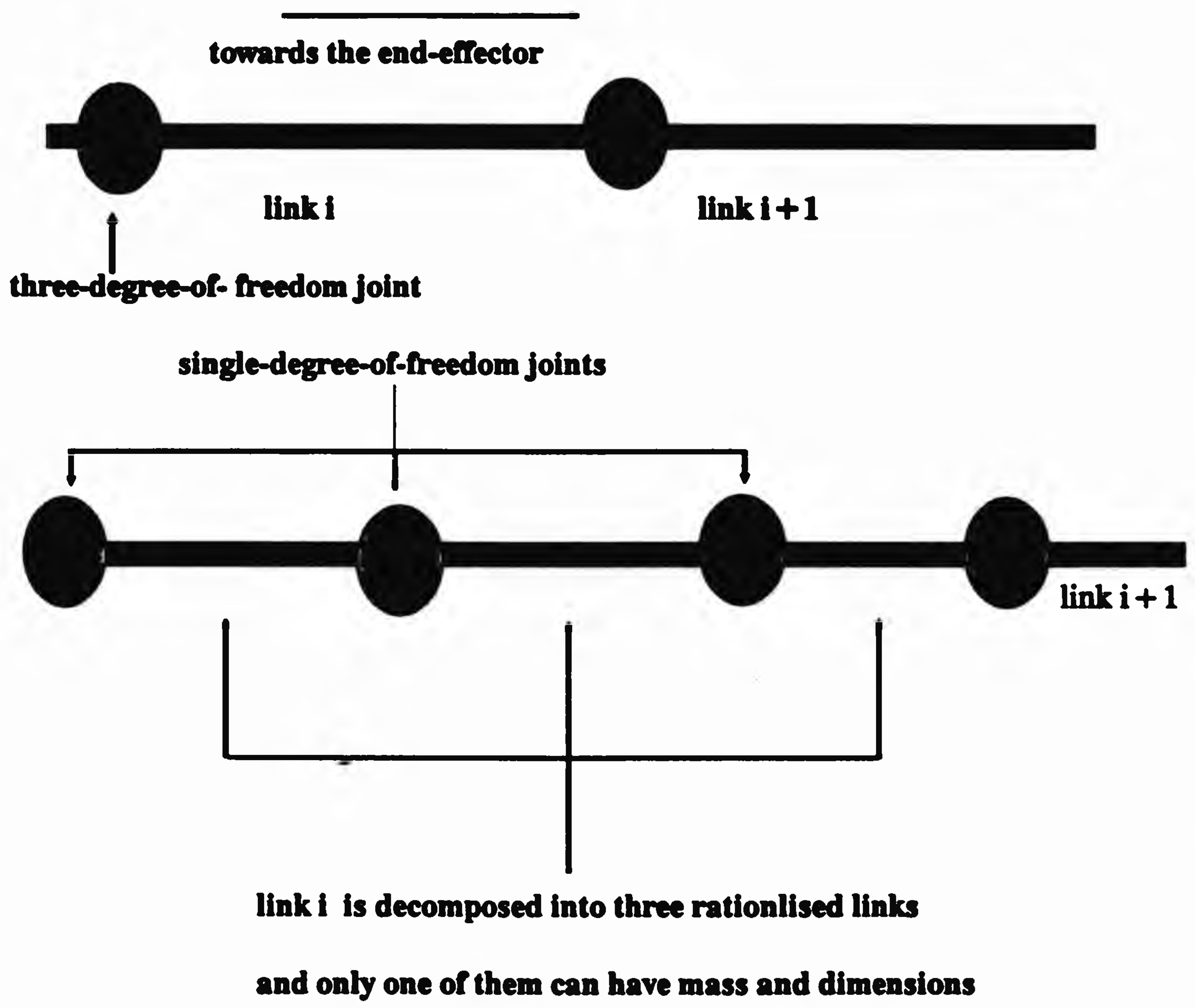

towards the end-effector
link i
link i + 1
three-degree-of- freedom joint
single-degree-of-freedom joints
link i + 1
link i is decomposed into three rationlised links
and only one of them can have mass and dimensions

**Fig 4.3 : Splitting multiple-degree-of-freedom joints into single-degree-of-freedom joints.**

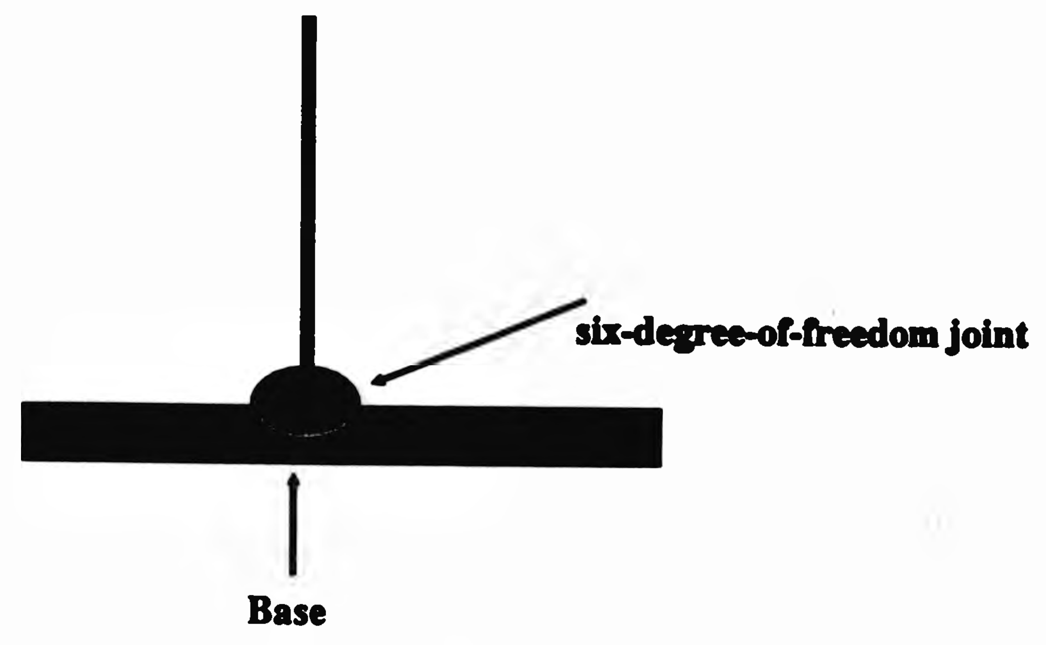


**Fig 4.4 : A kinematic chain with a single link .**

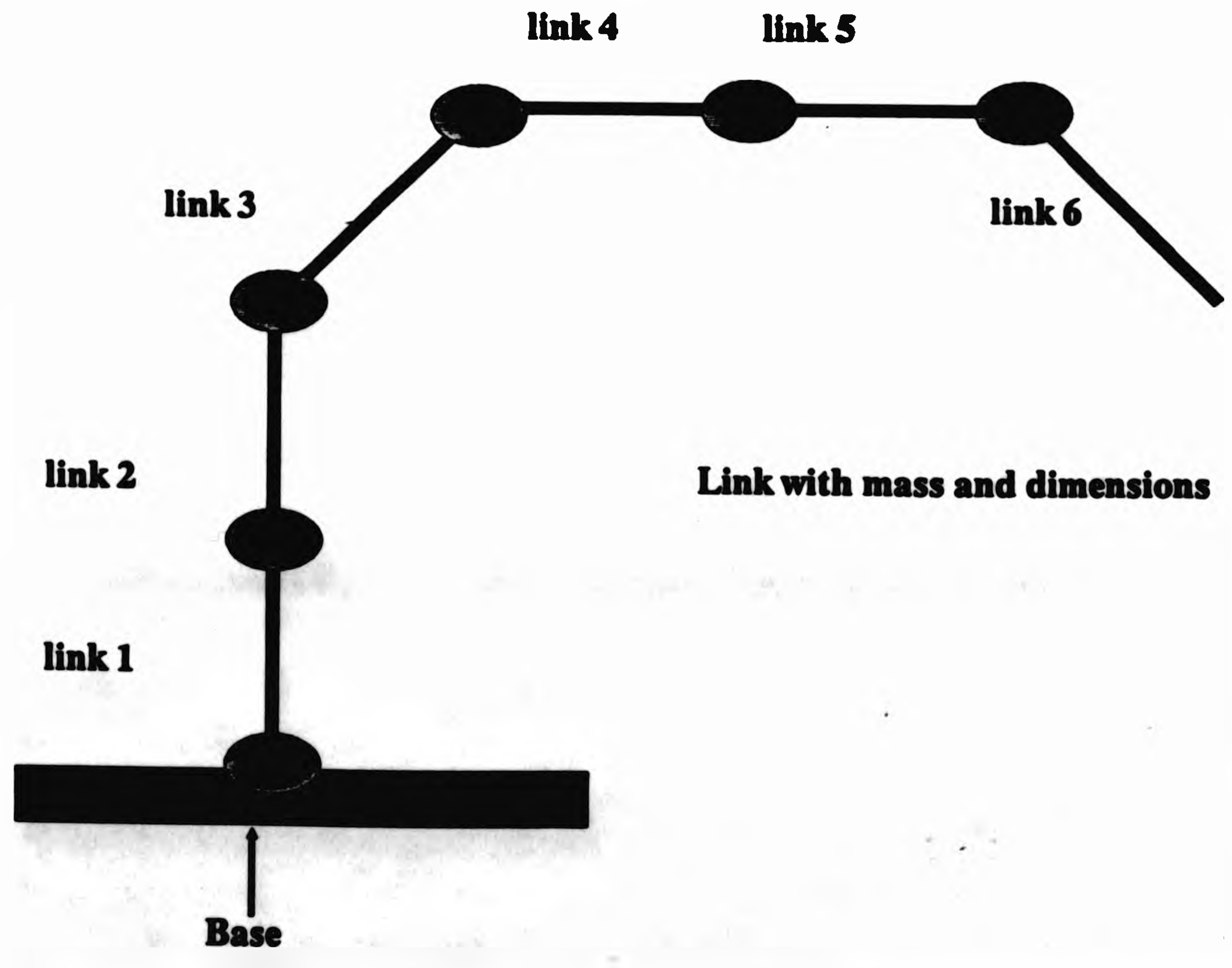


**Fig 4.5 : The mass of the link is allocated to the last rationlised link of the chain.**

## 4.4 Summary of formulae

This section summarizes the expressions that were derived in the previous chapter for calculating the forces as they are used in the subsequent section for the description of the algorithm.

$$A_i = \begin{pmatrix} 1 & 0 & 0 & 0 \\ a_i\cos(\theta_i^0+\theta_i) & \cos(\theta_i^0+\theta_i) & -\sin(\theta_i^0+\theta_i)\cos\alpha_i & \sin(\theta_i^0+\theta_i)\sin\alpha_i \\ a_i\sin(\theta_i^0+\theta_i) & \sin(\theta_i^0+\theta_i) & \cos(\theta_i^0+\theta_i)\cos\alpha_i & -\cos(\theta_i^0+\theta_i)\sin\alpha_i \\ s_i^0+s_i & 0 & \sin\alpha_i & \cos\alpha_i \end{pmatrix} \tag{4.1}$$

$$A_i^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -a_i & \cos(\theta_i^0+\theta_i) & \sin(\theta_i^0+\theta_i) & 0 \\ -(s_i^0+s_i)\sin\alpha_i & -\sin(\theta_i^0+\theta_i)\cos\alpha_i & \cos(\theta_i^0+\theta_i)\cos\alpha_i & \sin\alpha_i \\ -(s_i^0+s_i)\cos\alpha_i & \sin(\theta_i^0+\theta_i)\sin\alpha_i & -\cos(\theta_i^0+\theta_i)\sin\alpha_i & \cos\alpha_i \end{pmatrix} \tag{4.2}$$

The first derivative of $A_i$ w.r.t $_i$ are given by the following two formulae:
For a translational degree of freedom :

$$\frac{\partial A_i}{\partial q_i} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

For a rotational degree of freedom :

$$\frac{\partial A_i}{\partial q_i} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -a_i\sin(\theta_i^0+\theta_i) & -\sin(\theta_i^0+\theta_i) & -\cos(\theta_i^0+\theta_i)\cos\alpha_i & \cos(\theta_i^0+\theta_i)\sin\alpha_i \\ a_i\cos(\theta_i^0+\theta_i) & \cos(\theta_i^0+\theta_i) & -\sin(\theta_i^0+\theta_i)\cos\alpha_i & \sin(\theta_i^0+\theta_i)\sin\alpha_i \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.3}$$

The second derivative of $A_i$ w.r.t $q_i$ are given by the following two formulae:
if i is translational :

$$\frac{\partial^2 A_i}{\partial \dot{q}_i^2} = 0$$

if i is rotational :

$$\frac{\partial^2 A_i}{\partial q_i^2} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -a_i\cos(\theta_i^0+\theta_i) & -\cos(\theta_i^0+\theta_i) & \sin(\theta_i^0+\theta_i)\cos\alpha_i & -\sin(\theta_i^0+\theta_i)\sin\alpha_i \\ -a_i\sin(\theta_i^0+\theta_i) & -\sin(\theta_i^0+\theta_i) & -\cos(\theta_i^0+\theta_i)\cos\alpha_i & \cos(\theta_i^0+\theta_i)\sin\alpha_i \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.4}$$

Matrices $W$ and their derivatives w.r.t time are evaluated recursively :

$$W_j = W_{j-1}A_j \tag{4.5}$$

$$W_j^{-1} = A_j^{-1}W_{j-1}^{-1} \tag{4.6}$$

$$\dot{W}_j = \dot{W}_{j-1}A_j + W_{j-1}\frac{\partial A_j}{\partial q_j}\dot{q}_j \tag{4.7}$$

$$\ddot{W}_j = \ddot{W}_j A_j + 2\dot{W}_{j-1}\frac{\partial A_j}{\partial q_j}\dot{q}_j + W_{j-1}\frac{\partial^2 A_j}{\partial q_j^2}\dot{q}_j^2 + W_{j-1}\frac{\partial A_j}{\partial q_j}\ddot{q}_j \tag{4.8}$$

$$\frac{\partial W_j}{\partial q_j} = A_1A_2\ldots A_{j-1}\frac{\partial A_j}{\partial q_j} = W_{j-1}\frac{\partial A_j}{\partial q_j} \tag{4.9}$$

The terms $c_i$ and $D_i$ are calculated by :

$$c_i = m_i\,{}^i r_i + A_{i+1}c_{i+1} \tag{4.10}$$

$$D_i = J_i\ddot{W}_i^T + A_{i+1}D_{i+1} \tag{4.11}$$

Finally the forces are given by :

$$F_i = tr\left(\frac{\partial W_i}{\partial q_i}D_i - \right)g^T\frac{\partial W_i}{\partial q_i}c_i \tag{4.12}$$

## 4.5 Description of the algorithm for robotic mechanisms

This section presents the inverse Lagrangian dynamics algorithm as it is applied to robotic mechanisms i.e it does not accommodate kinematic chains with branches and multiple-degree-of-freedom joints. In the next section, the modified algorithm will be presented but in order to highlight the differences between the two approaches we start with the algorithm used in robotics.

**Step 1** : For $i = 1, \ldots, n$ input quantities $\alpha_i$, $\theta_i^0$, $a_i$, $s_i^0$, $J_i$, ${}^i r_i$ , $q_i$, $\dot{q}_i$ and $\ddot{q}_i$.

**Step 2** : Calculate $A_i, A_i^{-1}$, $\frac{\partial A_i}{\partial q_i}$ and $\frac{\partial^2 A_i}{\partial q_i}$ using (4.1), (4.2), (4.3) and (4.4) for i=1...n.

**Step 3** : Set $W_0 = I, \dot{W}_0 = 0$ and $\ddot{W}_0 = 0$.

**Step 4** : For $i = 1$ *to* $n$ do

- Calculate $W_i$ and $W_i^{-1}$ using (4.5), (4.6).
- Calculate $\dot{W}_i$ and $\frac{\partial W_i}{\partial q_i}$ using (4.7) and (4.9).
- Calculate $\ddot{W}_i$ using (4.8).

**Step 5** : Set $c_{n+1} = 0$ and $D_{n+1} = 0$.

**Step 6** : For $i = n$ down to 1 do

- Calculate $c_i$ and $D_i$ using (4.10), (4.11).
- Calculate $F_i$ using (4.12)

## 4.6 Outline of the algorithm for articulated bodies

The algorithm presented here is a version of the algorithm presented in section 4.5 modified so that it can deal with multiple-degree-of-freedom joints and branched kinematic chains. Assume that the configuration of the articulated body is given and that the body has been decomposed into rationalised kinematic chains as shown in sections 4.2 and 4.3. The following algorithm is a complete description of the procedures that have to take place in order to arrive at the forces. In the description of the algorithm the following assumptions are made :

- The model has n prototype links and N rationalised kinematic chains
- Rationalised kinematic chain $p$ has $L(p)$ rationalised links
- $k_p(i)$ represents the link numbers in the prototype chain, for the ith link in the pth rationalised chain
- A prototype link $j$ has $d(j)$ degrees of freedom
- The generic dynamic quantity ${}^pQ_j$ refers to some variable in rationalised kinematic chain p and rationalised link j
- A dual notation is used for the forces, if the notation refers to a force on the ith link in some rationalised kinematic chain p then it is written as ${}^pF_i$, whereas if it refers to the resultant force from all rationalised kinematic chains for a prototype degree of freedom j and link i, then it is written as $F_i(j)$.

We can now present the algorithm in pseudo code :

```
• Step 1 : Input Dynamic quantities.
        For i = $1, \ldots, n$
        input quantities $\alpha_i$, $\theta_i^0$, $a_i$, $s_i^0$, $J_i$, ${}^ir_i$ , $q_i$, $\dot{q}_i$ and $\ddot{q}_i$.

• Step 2 : Initialize forces.
        For m = 0 to n do
        For s = 1 to d(m) do
            F (s) = 0
```

*Now steps (3) to (5) are repeated for all rationalised chains.*

**For p = 1 to N do**

- **Step 3 :** ***Calculate geometrical data.***

$$\mu_p = \sum_{i=0}^{N(p)} d(k_p(i))$$

```
For j = 0 to $\mu_p$ do
 begin
  Evaluate ${}^pA_j$, ${}^pA_j^{-1}$, $\frac{{}^p\partial A_j}{\partial q_j}$ and $\frac{{}^p\partial^2 A_j}{\partial q_j^2}$
  using (4.1), (4.2), (4.3) and (4.4)

  Evaluate ${}^pW_j$, ${}^pW_j^{-1}$, ${}^p\ddot{W}_j$ $\frac{{}^p\partial A_j}{\partial q_j}$
  using (4.5), (4.6), (4.7) and (4.8)
 end
```

- **Step 4 :*****Recursively evaluate c and D.***

```
j = $\mu_p$ and ${}^pc_{j+1} = 0$, ${}^pD_{j+1} = 0$

For $i = L(p)$ down to 1 do
 For $k = d(k_p(i))$ down to 1 do
  begin
   if ($k$ is equal to $d(k_p(i))$)
   and ($d(k_p(i))$ does not belong to any
   kinematic chains from 1 to p-1) then

      ${}^pc_j = m_{k_p(i)}\,{}^jr_j + {}^pA_{j+1}\,{}^pc_{j+1}$ and

      ${}^pD_j = J_{k_p(i)}\,{}^p\ddot{W}_j^T + {}^pA_{j+1}\,{}^pD_{j+1}$

   else

      ${}^pc_j = {}^pA_{j+1}\,{}^pc_{j+1}$ and
      ${}^pD_j = {}^pA_{j+1}\,{}^pD_{j+1}$
      $j = j + 1$

  end
```

- Step 5 : *Evaluate forces.*

```
$j = 1$
  for $i = 0$ to $L(p)$ do
  $\lambda = k_p(i)$
    for $k = 1$ to $\lambda$ do
      begin
        Evaluate ${}^pF_j$ using (4.12)
            $F_\lambda(k) = F_\lambda(k) + {}^pF_j$
            $j = j + 1$
      end
```

The 'if' condition at step 4, ensures that the last link in the sequence has mass and that the mass of each link is included only once. The complexity of the algorithm depends upon the number of numerical operations it requires. The following calculations are based on the assumption that the body has N rationalised kinematic chains with $n_i$ degrees of freedom in each chain for i = 1 to N.

| Step number | Multiplications | Additions |
|---|---|---|
| 3 | $\sum_{i=1}^{N} 624(n_i - 1) + 112$ | $\sum_{i=1}^{N} 432(n_i - 1) + 48$ |
| 4 | $\sum_{i=1}^{N} 102\frac{2}{3}(n_i - 1) + 68$ | $\sum_{i=1}^{N} 76(n_i - 1) + 48$ |
| 5 | $\sum_{i=1}^{N} 84n_i$ | $\sum_{i=1}^{N} 66n_i$ |

The total number of operations is :

| Multiplications | Additions |
|---|---|
| $\sum_{i=1}^{N} 810\frac{2}{3}(n_i - 1) - 546\frac{2}{3}$ | $\sum_{i=1}^{N} 574(n_i - 1) - 412$ |

So although the method is linear, its coefficients are quite large and the efficiency of the method is apparent only for large number of degrees of freedom. Therefore for complex articulated bodies (humans), the method is fairly efficient. The above calculations are based on the assumption that the calculations of the geometrical data as they are performed in step 3, are done using 4x4 matrix multiplication routines. The numerical complexity of the algorithm can be reduced by using 3x3 multiplication routines. The following figures show the calculations using 3x3 matrix routines whenever possible :

| Step number | Multiplications | Additions |
|---|---|---|
| 3 | $\sum_{i=1}^{N} 270(n_i - 1) + 54$ | $\sum_{i=1}^{N} 162(n_i - 1) + 18$ |
| 4 | $\sum_{i=1}^{N} 82\frac{2}{3}(n_i - 1) + 68$ | $\sum_{i=1}^{N} 61(n_i - 1) + 48$ |
| 5 | $\sum_{i=1}^{N} 63n_i$ | $\sum_{i=1}^{N} 51n_i$ |

This brings the total number of calculations to :

| Multiplications | Additions |
|---|---|
| $\sum_{i=1}^{N} 415\frac{2}{3}(n_i - 1) - 234\frac{2}{3}$ | $\sum_{i=1}^{N} 274(n_i - 1) - 157$ |

These reductions can be achieved by re-adjustments in the software implementation and represent savings of approximately 50% in the number of calculations (for $n_i >$ 10). Some of the quantities are calculated more than once since they might appear in more than one chain. If these calculations are restricted to the necessary minimum ( calculate the quantity only once), some savings can be made. We can not quote any theoretical figures as to what savings one should expect as this depends upon the particular body configuration. However, for a realistic human model with 66 degrees of freedom, we can achieve a 12% saving in the number of calculations.

## 4.7 Applying the formulation to an experimental human model

This section gives a description of the construction of a human model. This includes the definition of kinematic chains, the construction of the hierarchy tree and the arrangement of kinematic chains and their frames. The model in Fig 4.6 will be considered to have the torso acting as the root and is depicted as a hierarchical structure in Fig 4.7.

The formulation deals with single degrees of freedom and our model is considered to have multiple degrees of freedom, requiring the introduction of dummy links as described in section 4.3. For simplicity of description it is assumed that every joint has three rotational degrees of freedom. We also need three translational degrees of freedom to track the linear motion of the body. A reference coordinate system is attached to the root which will be called link 0. The tree structure can be expanded into five rationalised kinematic chains which are shown in Fig 4.8.

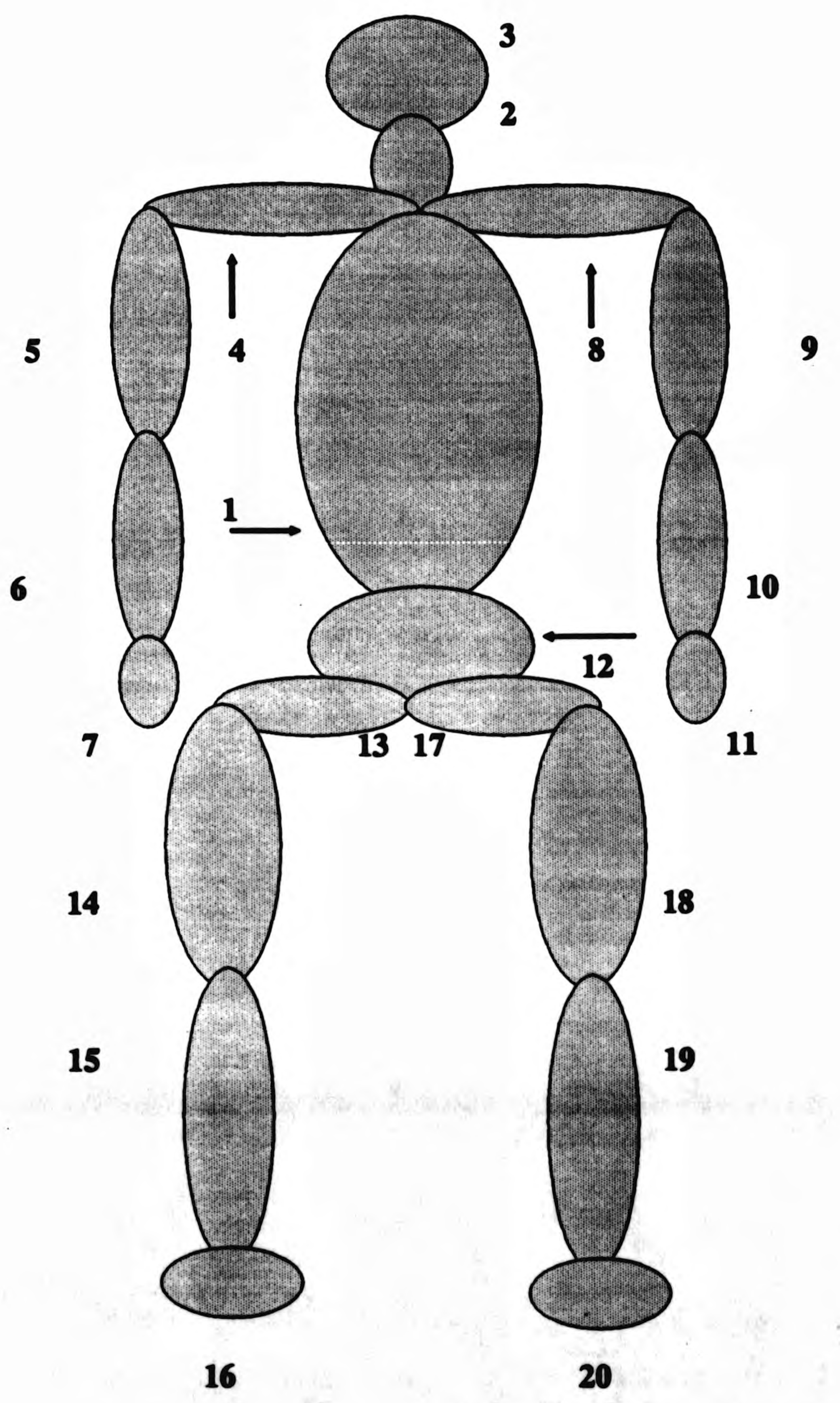


**Fig 4.6 : Experimental human figure.**

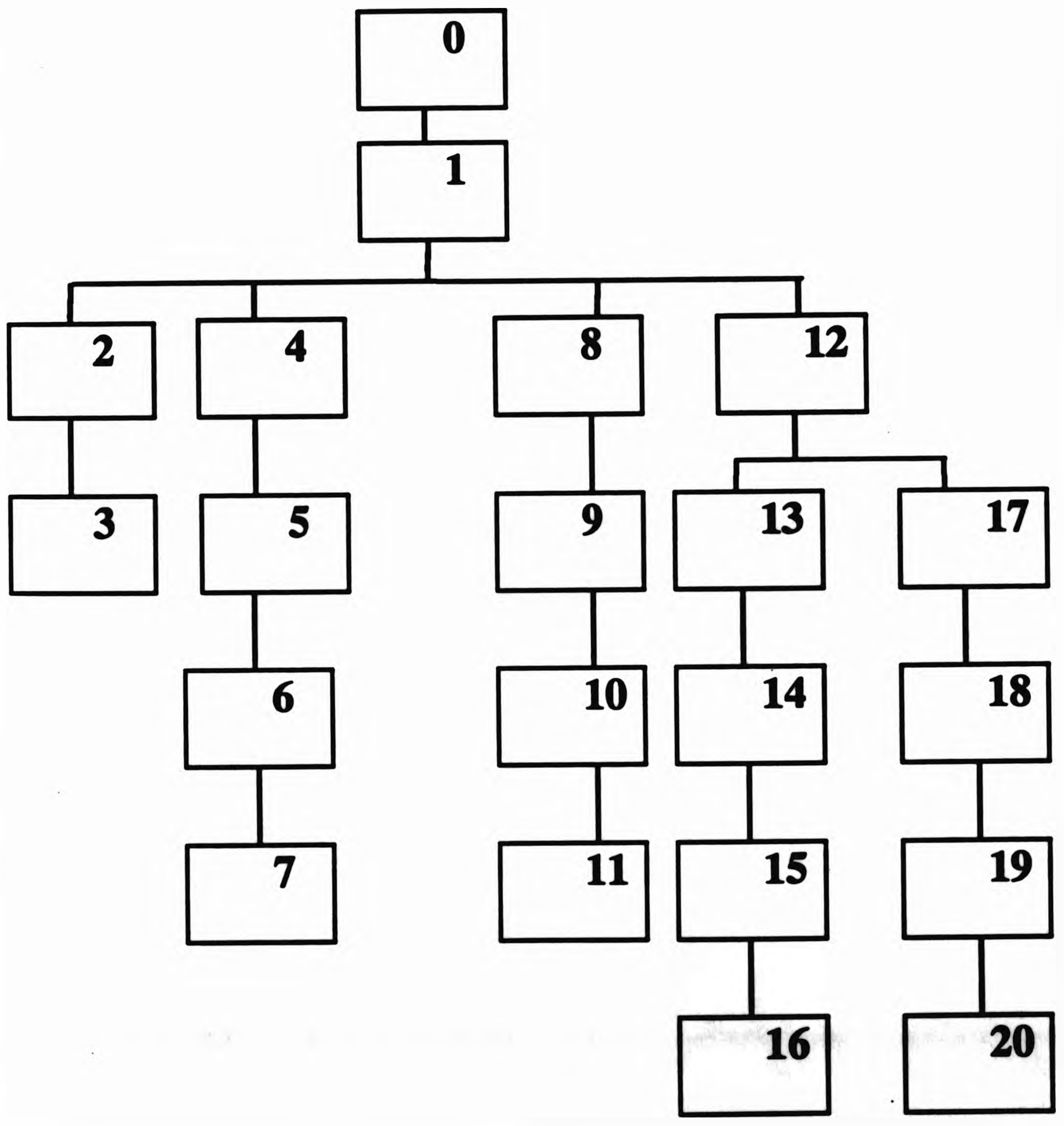


**Fig 4.7 : Hierarchical structure of the human figure in fig 4.6. The numbers in the boxes represent link numbers. Link 0 represents the world coordinate system.**

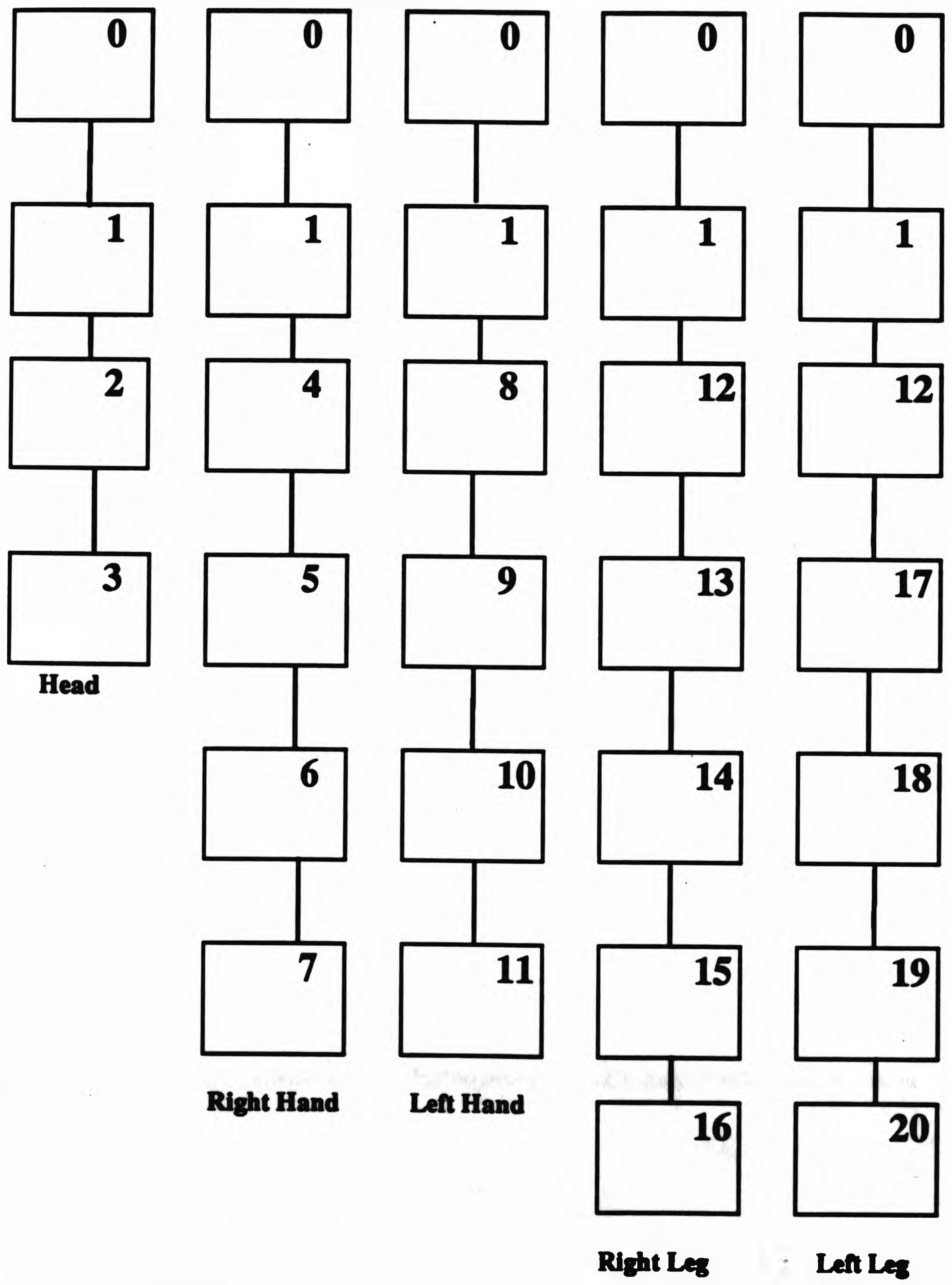


**Fig 4.8 : Decomposed open kinematic chains derived from structure in Fig 4.7.**

The first of these chains has three translational degrees of freedom and nine rotational, twelve in total. In fig 4.9 the twelve links are depicted (including the dummy ones). The shaded links are the ones that have mass and dimension. It is obvious that for the first seven links (link 0 and links 1-6 for the first six degrees of freedom, three translational and three rotational), only one should have mass, and after that only one every three links. The coordinate systems of the local frames can be set up as follows :

- The origin of each frame is at the beginning of each link, situated on the axis of symmetry which runs along the length of the link.
- Each limb except the torso has been split into three rationalised links with joint axes lying respectively in the Z-Y-X directions w.r.t the global coordinate system. This means that every prototype joint has been split into three rationalised joints, each of which is allowed to rotate along the direction of a single axis, with the first joint performing a Z rotation, the second joint a Y rotation and the third joint an X rotation.
- For the root, the translational degrees of freedom are successively in the Z-Y-X directions, and the rotational degrees are ordered according to the convention in the previous point.
- The origins of the frames lie initially in the X-Z plane.
- The first frame (frame for link 1) has its axes parallel to those of the world coordinate system.

The transformation matrices A can be found from eqn (4.1) if angles $\alpha$ and $\theta$ and distances $s$ and $a$ are known. The matrix in eqn (4.1) follows from the way the local coordinate systems are defined in Section 3.2. Table (4.1) gives the various quantities needed for the construction of the transformation matrices $A_i$.

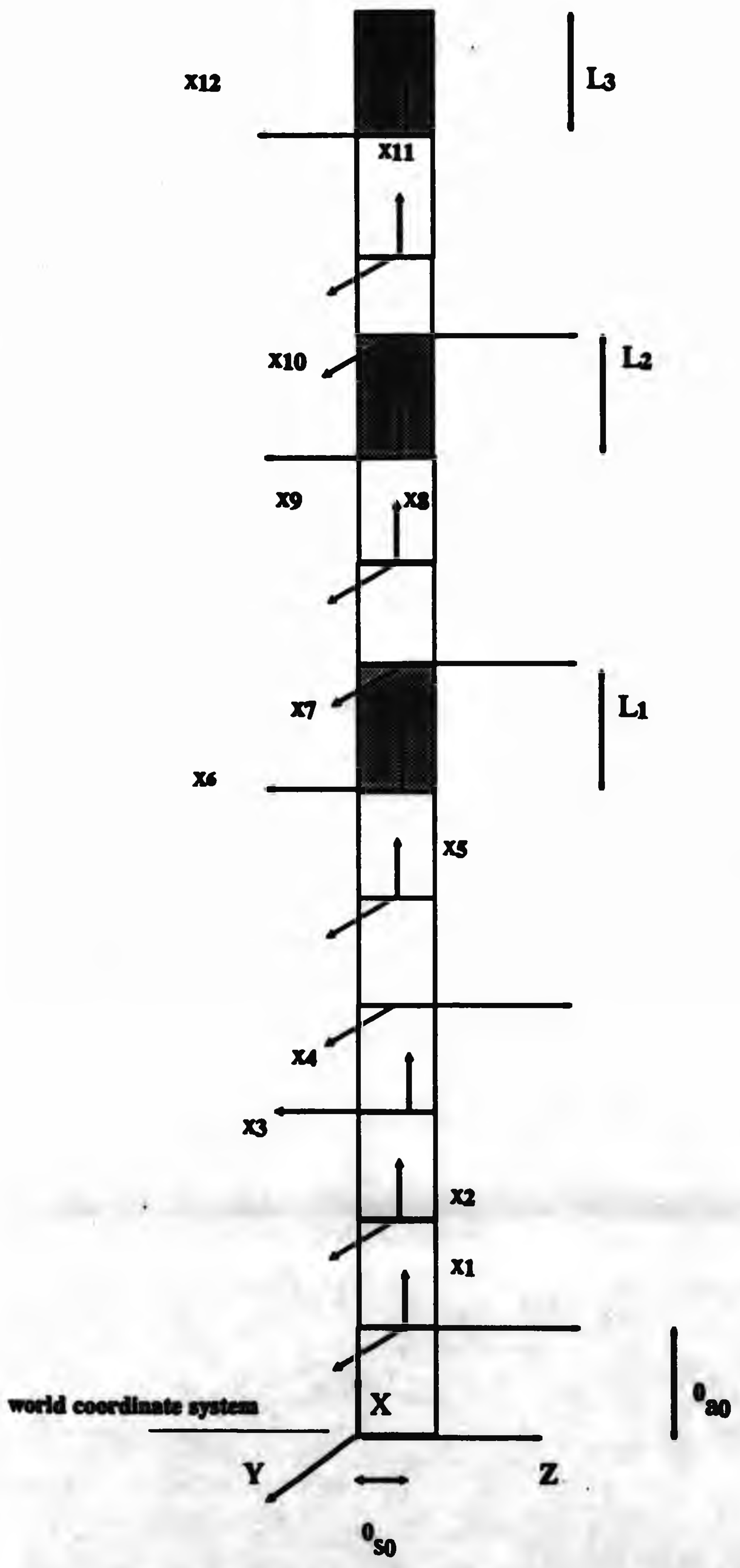


**Fig 4.9 : Rationlised kinematic chain 1 as shown in fig 4.8. Only shaded links contain mass.**

| Rationalised link number | $a$ distance | $s$ distance | $\alpha$ angle | $\theta$ angle |
|---|---|---|---|---|
| 1 | ${}^0a_0$ | $s_0 + {}^0s_0$ | 0 | 0 |
| 2 | 0 | $s_1$ | -90 | 0 |
| 3 | 0 | $s_2$ | 90 | 90 |
| 4 | 0 | 0 | -90 | $90 + \theta_3$ |
| 5 | 0 | 0 | -90 | $90 - \theta_4$ |
| 6 | 0 | 0 | -90 | $90 - \theta_5$ |
| 7 | 0 | $L_1$ | -90 | $90 - \theta_6$ |
| 8 | 0 | 0 | -90 | $90 - \theta_7$ |
| 9 | 0 | 0 | -90 | $90 - \theta_8$ |
| 10 | 0 | $L_2$ | -90 | $90 - \theta_9$ |
| 11 | 0 | 0 | -90 | $90 - \theta_{10}$ |
| 12 | 0 | 0 | -90 | $90 - \theta_{11}$ |

Table 4.1 : Head-Neck, Chain 1

It can be seen that there is a pattern in the way that the values of $\alpha_i$ and $\theta_i$ occur. This pattern also holds for the other four kinematic chains as long as the local coordinate systems are constructed using the same convention. Hence the construction of the transformation matrices can now be illustrated. For the first degree of freedom it will be :

$$A_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ a_0^0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ s_0^0 + s_0 & 0 & 0 & 1 \end{pmatrix}$$

with joint variable $s_0$ (translational). The next two matrices are :

$$A_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ L_1 & 1 & 0 & 0 \\ s_1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ s_2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

with joint variables $s_1$ and $s_2$ respectively. Notice that the order of the rows must be changed in order to correspond to axes Z-Y-X. An important point is that angles $\alpha$ and $\theta$ for the first rotational degree of freedom have to found with respect to the world coordinate system.

## 4.8 Conclusions

In this chapter, we produced the adaptations necessary for applying the method to articulated bodies for the purpose of computer animation. Hollerbach's formulation was designed for kinematic chains without branches and for single-degree-of-freedom joints. We extended the method to kinematic chains with branches and multiple-degree-of-freedom joints. We then introduced the algorithm for inverse Lagrangian dynamics as it is used in robotics, which was followed by the extended version of the algorithm which accommodates the changes made for the purpose of computer animation. The method was applied to an experimental model. The application of the Lagrangian approach to that experimental model concentrated on the construction of local coordinate frames and the way of constructing the rationalised kinematic chains.

The modification of the method showed that it still remains linear, (order O(n)), but the coefficients are large. It was found that the numerical complexity of the method depends upon two factors:

- the number of branches in the prototype kinematic chain
- the position of the root.

Both factors influence the number of end-effectors in the hierarchy and the number of rationalised kinematic chains. The number of rationalised kinematic chains is equal to the number of the end-effectors, unless the root is an end-effector, in which case it is one fewer. Therefore the more end-effectors there are in the prototype chain the more rationalised chains we have.

The speed of the algorithm is linearly dependent upon the number of rationalised kinematic chains, and the coefficients are quite large. However, by considering the way which the geometrical matrices are constructed, we can use (in many cases) 3x3 matrix routines instead of 4x4, and this results in savings of approximately 50%. The number of numerical operations can be further reduced if identical dynamic quantities that appear in more than one rationalised chain are calculated only once. For our experimental model this resulted in further savings of around 12%, although one has to say that this figure depends directly upon the configuration of the body and varies from one body to another.

It was shown when a prototype link is split into rationalised links, only the rationalised link furthest from the root contains mass and has dimensions. The reason for this was that if the mass is placed at any other link, the gravitational component of the forces will always be zero regardless of the position of the link, as explained in Section 4.3.

An experimental model with exactly three rotational degrees of freedom for rotational joints and six for the root - three rotational and three translational - was described. From the examination of the experimental model it was found that the values of angles $\alpha$ and $\theta^0$ follow a pattern when the coordinate systems are constructed in a consistent way. This simplifies the task of defining these values. The order of rotation for the joints was Z-Y-X and the pattern holds for this type of rotation joint, but it would be helpful to find patterns for other types of rotation as well. These issues are discussed in the chapter 7 where the lowest level dynamic control is described.

# Chapter 5

# First Level Kinematic Motion Control

## 5.1 Introduction

The motion of humans is not usually described in terms of forces and torques but in terms of directions and space paths. Therefore, the design of an animation system for human motion control should allow the animator to specify motions in directional terms i.e. as a director controls his actors. The designer of such an animation system should incorporate the facility of using kinematics to specify motions that cannot be accomplished by a purely dynamical system.

The major problem in defining motions for articulated bodies is that either the final positions for the various links of the body are not known or, if they are, the method of moving from the starting position to the end position is not known. If the latter is the case, kinematics instead of dynamic analysis can be used to specify the motion and the user must have a number of ways of specifying the behaviour of the motion between the start and end positions. However in the first case, dynamic analysis can be used so the user can get an initial idea about the profile of the motion. In both cases some input from the user is required, regarding the final positions of the links. In order to take this burden from the user, the animation system should decide upon these input values.

This chapter describes the kinematic control of the lowest level layer (kernel) where motions are described by specifying positions of individual degrees of freedom. It also deals with the various numerical schemes that are used to construct the torque-time profiles for kinematic control. A large variety of profiles is presented which covers a wide variety of behaviours.

The profiles are classified into three major classes : profiles for displacement, velocity and acceleration. Most profiles have a number of constraints imposed on them which are used to calculate the constants that are involved in the construction of the curves. This chapter also presents a method for blending profiles with one another in order to generate more complicated behaviours. It deals, finally, with the problem of specifying free-hand drawing curves and presents one solution using Konachek splines. Their subsequent use in dynamics is described in Chapter 6.

## 5.2 Assumptions for kinematic mode profiles

When the motion of a degree of freedom can be fully defined by specifying its initial and final positions, and the behaviour of the motion is known, then the motion can be described using kinematic control. Therefore in order to have a degree of freedom under kinematic control, three conditions must be satisfied :

- The starting and ending boundary conditions for displacement, velocity and acceleration must be known i.e ($\theta_0$, $\dot{\theta}_0$, $\ddot{\theta}_0$) and ($\theta_1$, $\dot{\theta}_1$, $\ddot{\theta}_1$).
- The simulation interval T for the motion must be known.
- The behaviour of the displacement, velocity and acceleration curves must be defined at any time instance within the simulation interval.

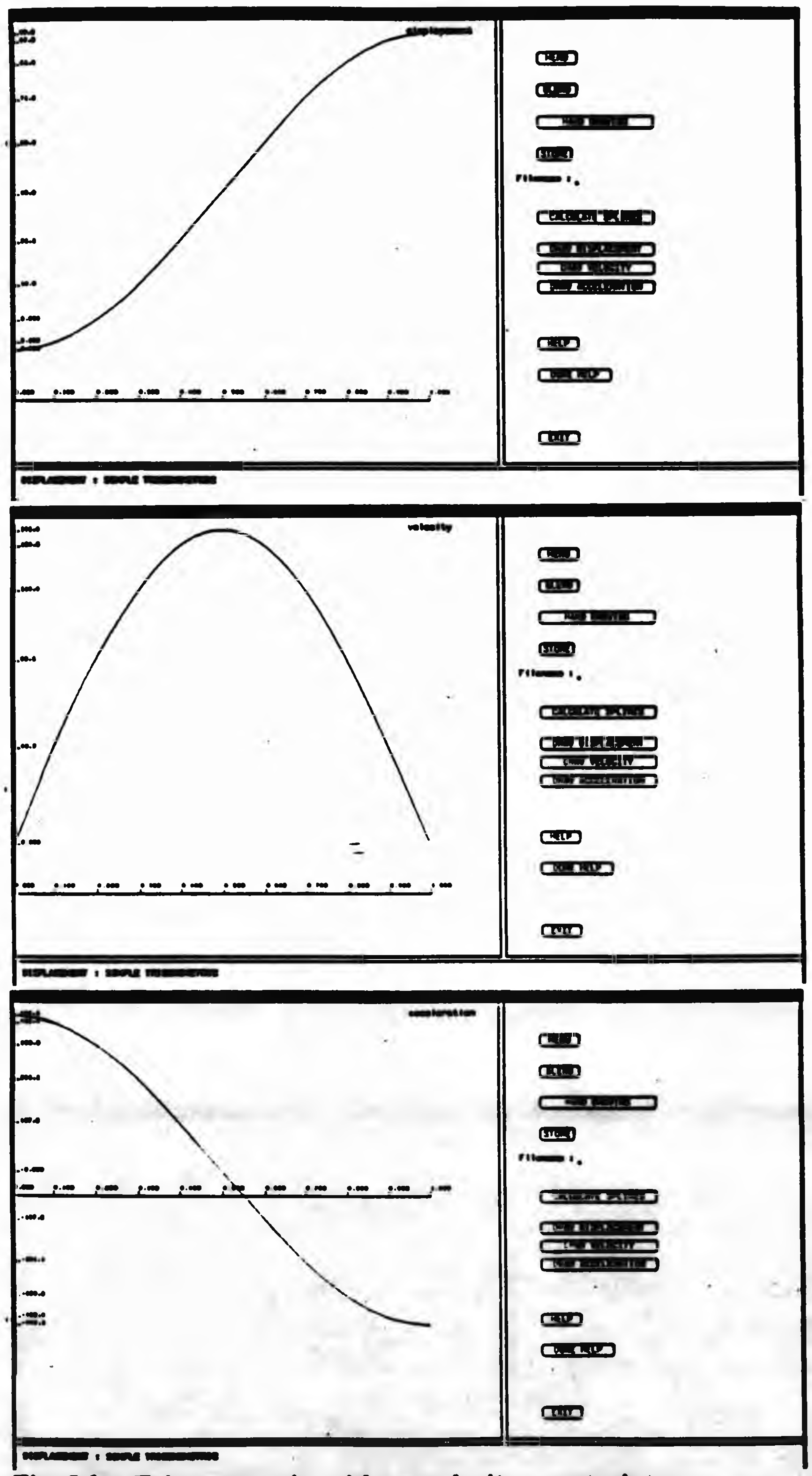


**Fig 5.1 : Trigonometric without velocity constraints.**

These conditions are general and may not apply to every profile presented in this chapter. In most cases, some of the boundary conditions are satisfied because the underlying philosophy is to keep the number of parameters minimal and to avoid the presence of local maxima and minima due to high order polynomials. On the other hand if one of the curves is analytically defined within the simulation interval then the other two can be deduced by straightforward integration or differentiation.

The approach used for choosing the profiles covers a large spectrum of different motion behaviours. The approach is hierarchical in the sense that once one curve is defined the other two are calculated automatically by the system i.e given the displacement profile, the velocity and acceleration profiles are automatically calculated. In the following sections, a description of the profiles is given and the evaluation of the associated parameters is given in Appendix A. In the following sections $T$ represents the simulation interval and the simulation takes place for $0 \leq t \leq T$.

## 5.3 Curves for displacement

### 5.3.1 Trigonometric

#### Without velocity constraints

This profile satisfies the displacement constraints $\theta(0) = \theta_0$ and $\theta(T) = \theta_1$ and does not satisfy any constraints for the velocity and the acceleration. This profile ensures that there are no stationary points for the displacement curve. The velocity curve has a maximum at $t = T/2$, when $\theta_1 > \theta_0$ , or a minimum when $\theta_1 < \theta_0$. The acceleration curve has no stationary points. Fig 5.1 shows a sample of the shape the three curves for this profile can take.

#### With velocity constraints

This profile satisfies the displacement constraints $\theta(0) = \theta_0$, $\theta(T) = \theta_1$ and the velocity constraints $\dot{\theta}(0) = \dot{\theta}_0$, $\dot{\theta}(T) = \dot{\theta}_1$. The profile can, however, produce undesirable stationary points in the displacement curve. The implementation of the profile in the kinematics component can trace the existence of stationary points and produce either a set of initial velocities (for a prescribed final velocity), or a set of final velocities (for a prescribed initial velocity), that result in displacement profiles without stationary points. If, for example, the user can prescribe the initial velocity and a starting value for the final velocity $\dot{\theta}_1^*$ , and a search range $R$. The system then searches the interval $[\dot{\theta}_1^*, \dot{\theta}_1^* + R]$ to find the values of the final velocities that produce displacement curves without stationary points. Both values, $\dot{\theta}_1^*$ and $R$ are defined arbitrarily by the user in order to fulfill the requirements of the particular profile. The same process can be performed by prescribing the final velocity and $\dot{\theta}_0^*$ together with a range. Fig 5.2 shows a sample of the three curves this profile can produce.

## 5.3.2 Exponential

There are three profiles with exponential behaviour which describe the style of motion for displacement. All three profiles are polynomials.

### Simple exponential without velocity constraints

This profile satisfies the displacement constraints $\theta(0) = \theta_0$ and $\theta(T) = \theta_1$ . The behaviour of the profile depends upon the degree of the polynomial $n$ ( $\geq$ 3). The higher the degree of the polynomial the flatter the curve becomes at the beginning of the motion and the more dramatically the slope increases towards the end of the motion. Fig 5.3 shows the profile for $n = 3$ and Fig 5.4 for $n = 9$ for the same angles and velocities. None of the curves with this profile produces stationary points.

### Exponential without velocity constraints

This profile also satisfies the displacement constraints $\theta(0) = \theta_0$ and $\theta(T) = \theta_1$ . The profile depends upon two parameters, n the degree of the polynomial and a constant s. The constant $s$ is a crucial factor for the shape of the displacement curve.

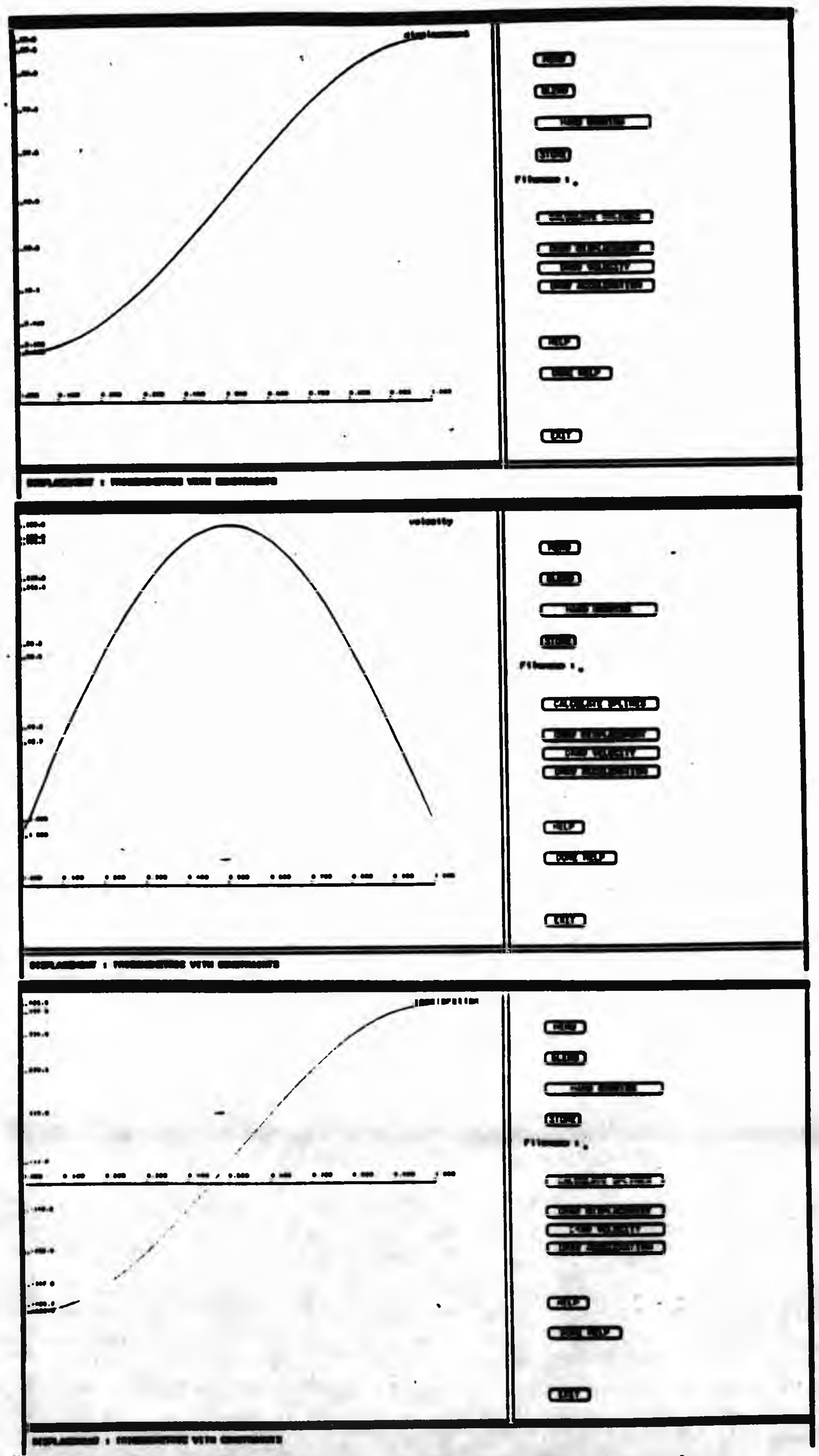
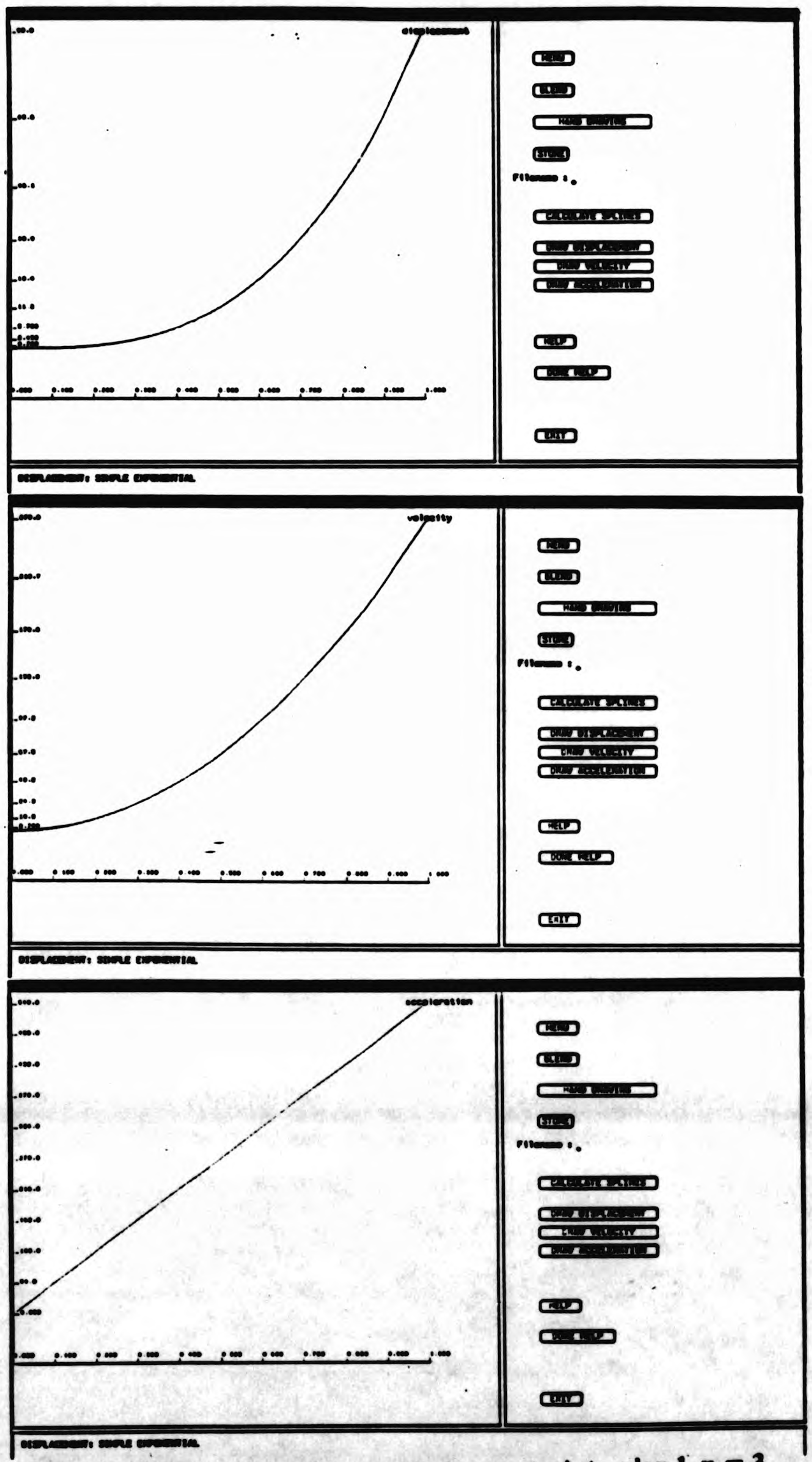


**Fig 5.2 : Trigonometric with velocity constraints.**

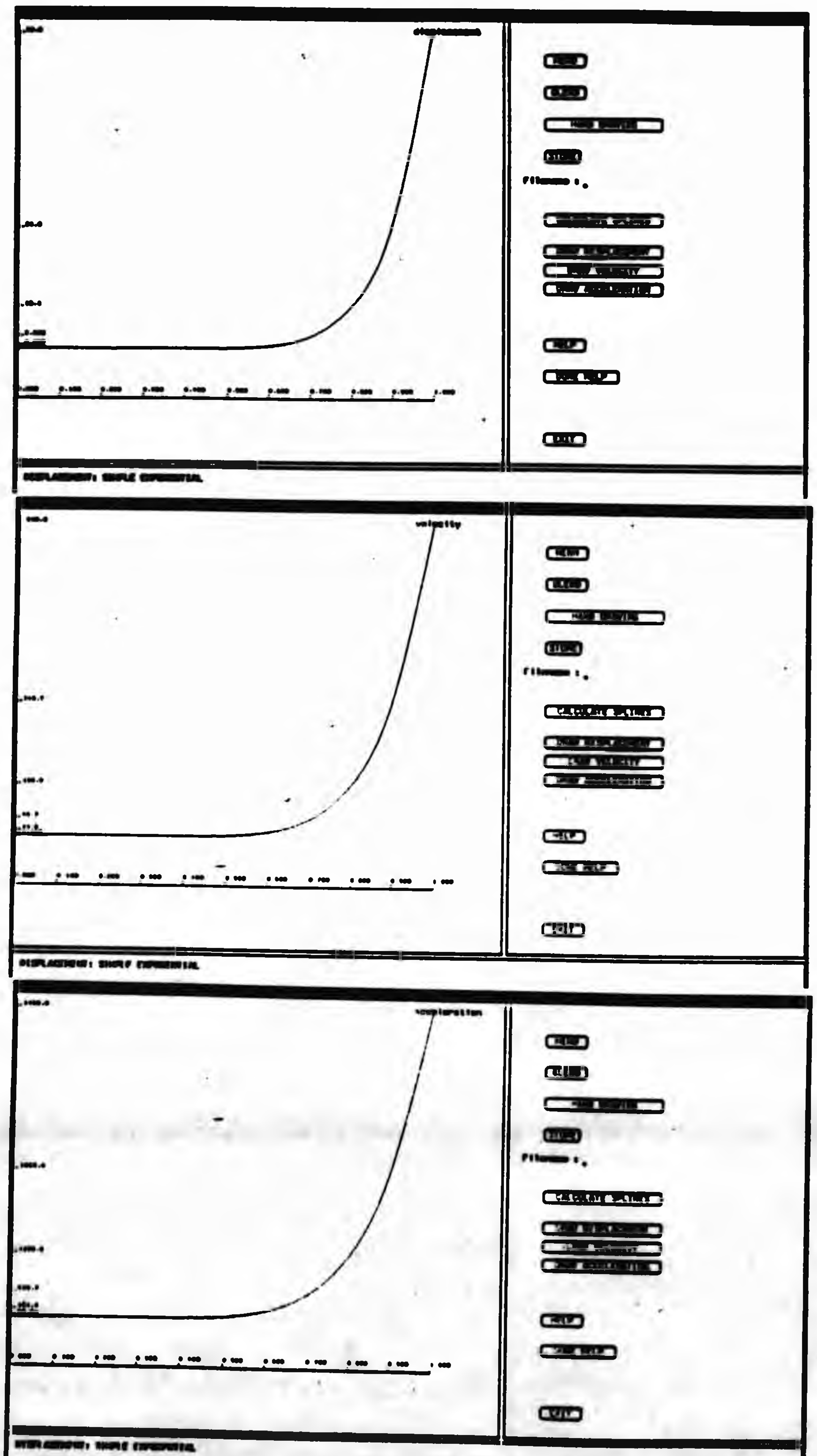


DISPLACEMENT: SIMPLE EXPONENTIAL

DISPLACEMENT: SIMPLE EXPONENTIAL

DISPLACEMENT: SIMPLE EXPONENTIAL

**5.3 : Simple Exponential without velocity constraints, plan 1, n = 3.**



**5.4 : Simple Exponential without velocity constraints, plan 1, n = 9.**

When $s$ is small ($< 0.3$), the displacement is almost a straight line and so are the velocity and acceleration, as shown in Fig 5.5. When s is between 1 and 3 the curve has a shape as shown in Fig 5.6. Finally for $s > 3$, the curve has the same behaviour as the first exponential profile as shown in Fig 5.7. Obviously these ranges can change for different angles and velocities but the principle remains the same.

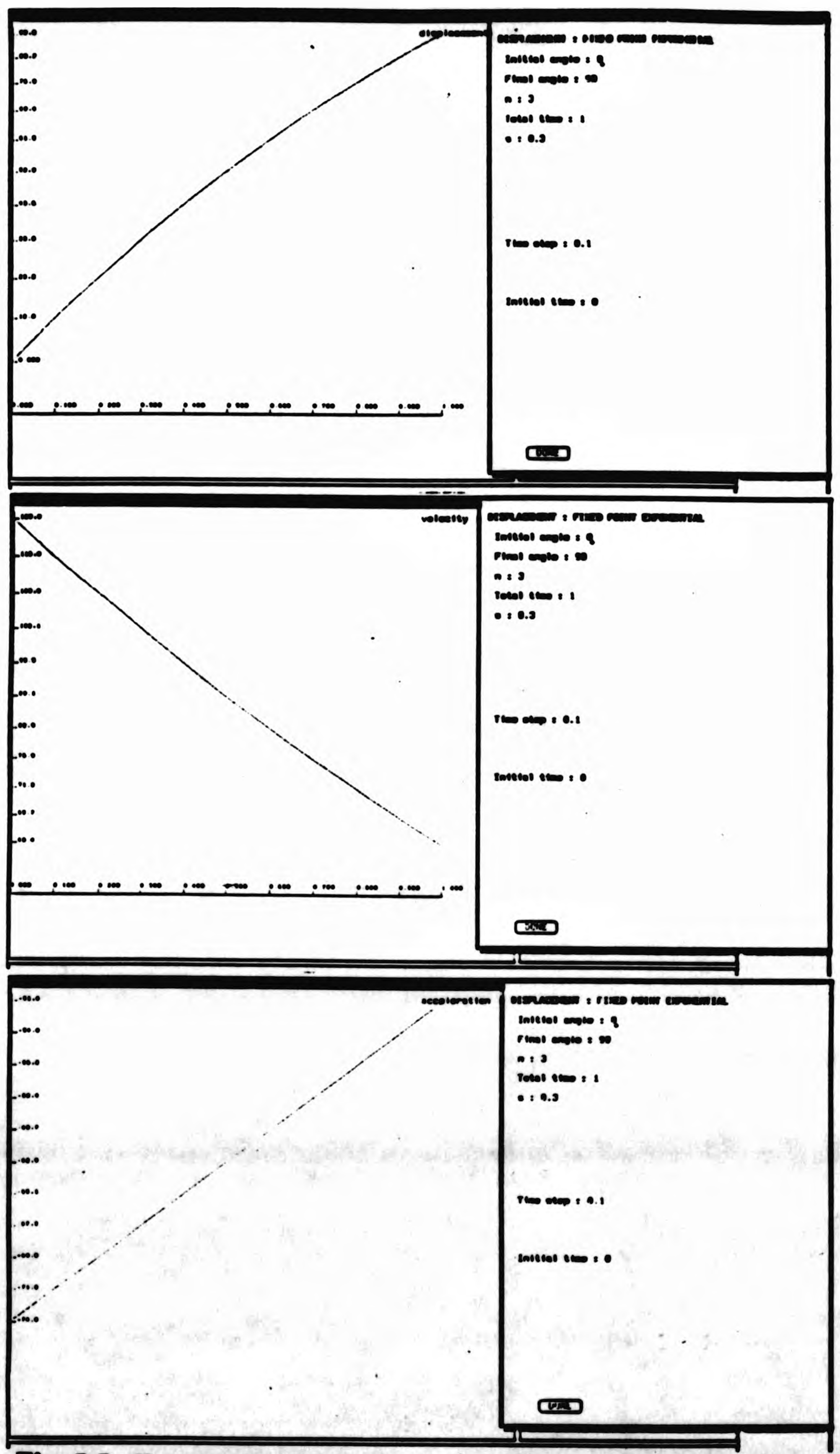
#### Exponential with velocity constraints

This profile satisfies the displacement constraints, $\theta(0) = \theta_0$, $\theta(T) = \theta_1$ and the velocity constraints $\dot{\theta}(0) = \dot{\theta}_0$ and $\dot{\theta}(T) = \dot{\theta}_1$. The displacement is expressed as a polynomial of powers of $(t - T/s)$ and it is very sensitive to changes of the value of $s$. As the profile requires the solution of a system of four linear equations, it is possible that there are sets of values for the two velocities and $s$ that produce unsolvable systems of equations. For this reason, the system prompts the user to make the right choice of $s$. For example the following set of values produce an unsolvable system of equations $\theta(0) = 0$, $\theta(1) = 1$, $\dot{\theta}(0) = 0$, $\dot{\theta}(T) = 1$, $n = 3$ and $s = 2$. The system decides, given the information available to it, that the value of s has to be changed. If $s$ is changed to 3, the system prompts the user to change $s$ to a negative value. For $s = -2$ the software detects that the system of equations is solvable and generates the profile. Fig 5.8 shows a typical example of this profile for $n = 3$. The transition from initial to final displacement is quite smooth for small values of $n$, whereas for large values ($n > 7$) this transition is more abrupt. In the latter case the velocity and acceleration profiles have stationary points, as shown in Fig 5.9.

### 5.3.3 Hyperbolic

#### Without velocity constraints

This profile satisfies the displacement constraints $\theta(0) = \theta_0$, and $\theta(T) = \theta_1$ and is expressed in terms of $\sinh(t/T)$. The resulting displacement, velocity and acceleration curves are smooth and have no stationary points, as shown in Fig 5.10.

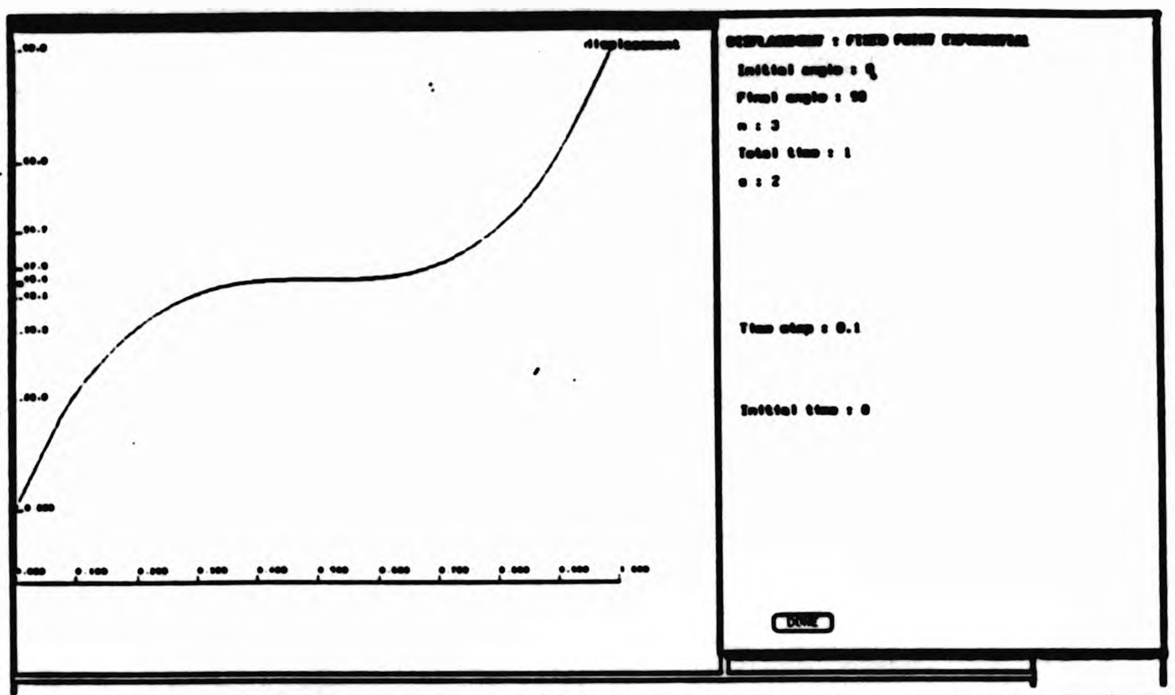


DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 0.3
Time step : 0.1
Initial time : 0

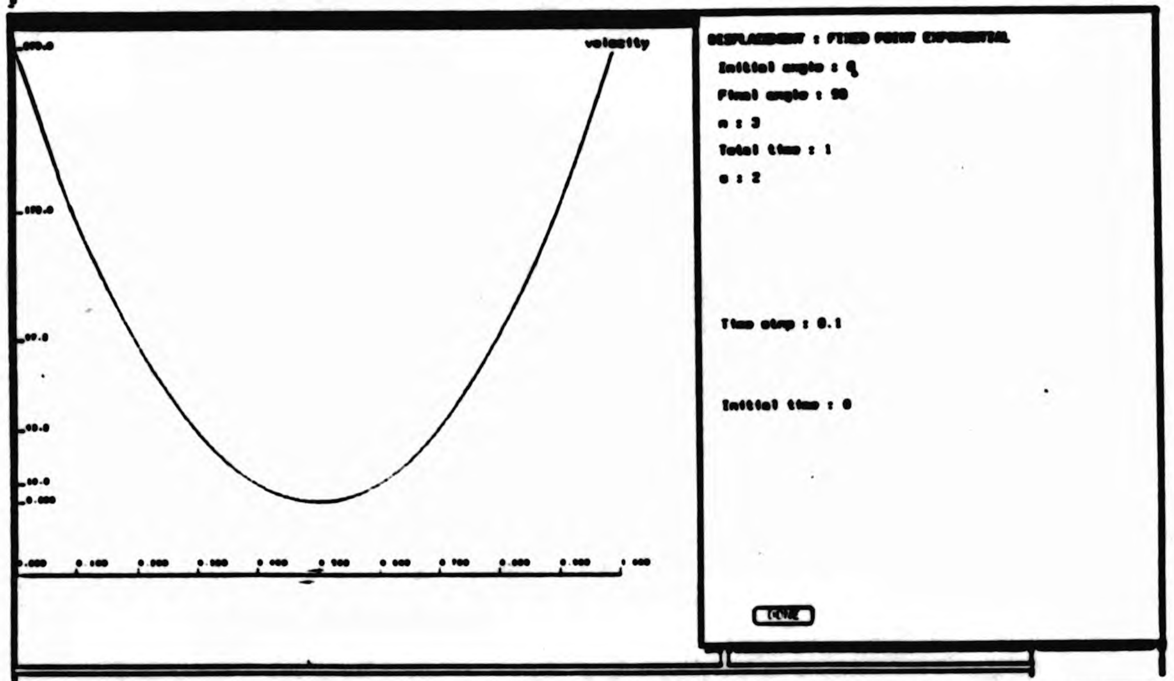
DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 0.3
Time step : 0.1
Initial time : 0

DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 0.3
Time step : 0.1
Initial time : 0

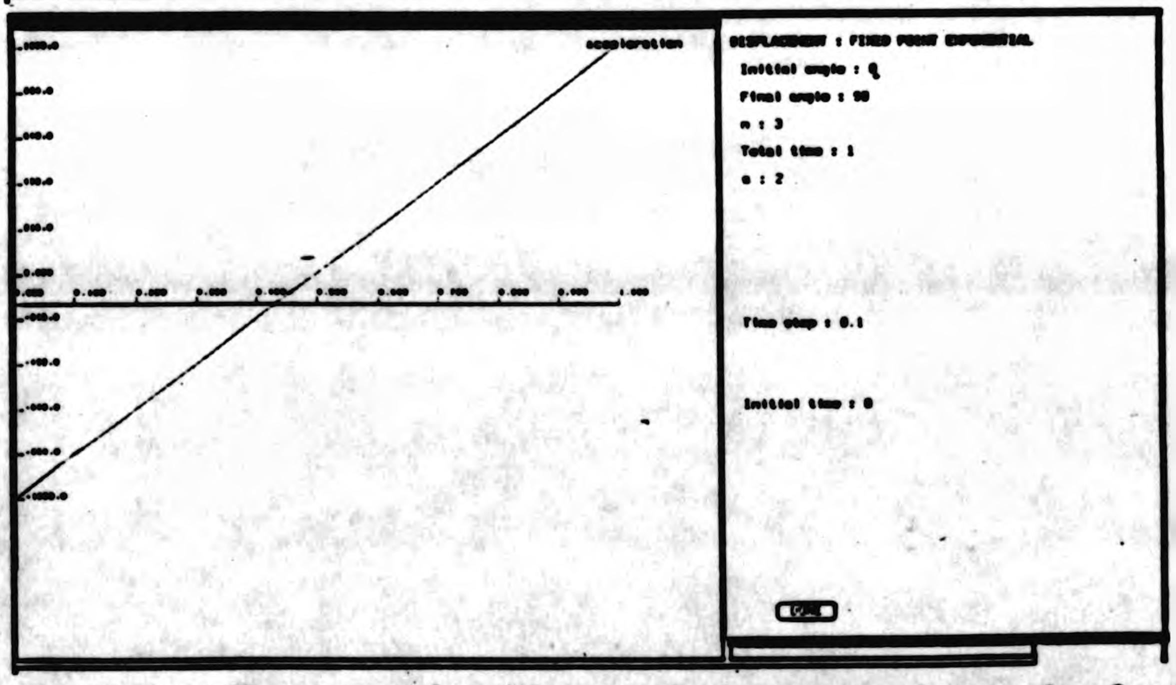
**Fig 5.5 : Exponential without velocity constraints, plan 2, s <= 0.3.**



DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 2
Time step : 0.1
Initial time : 0

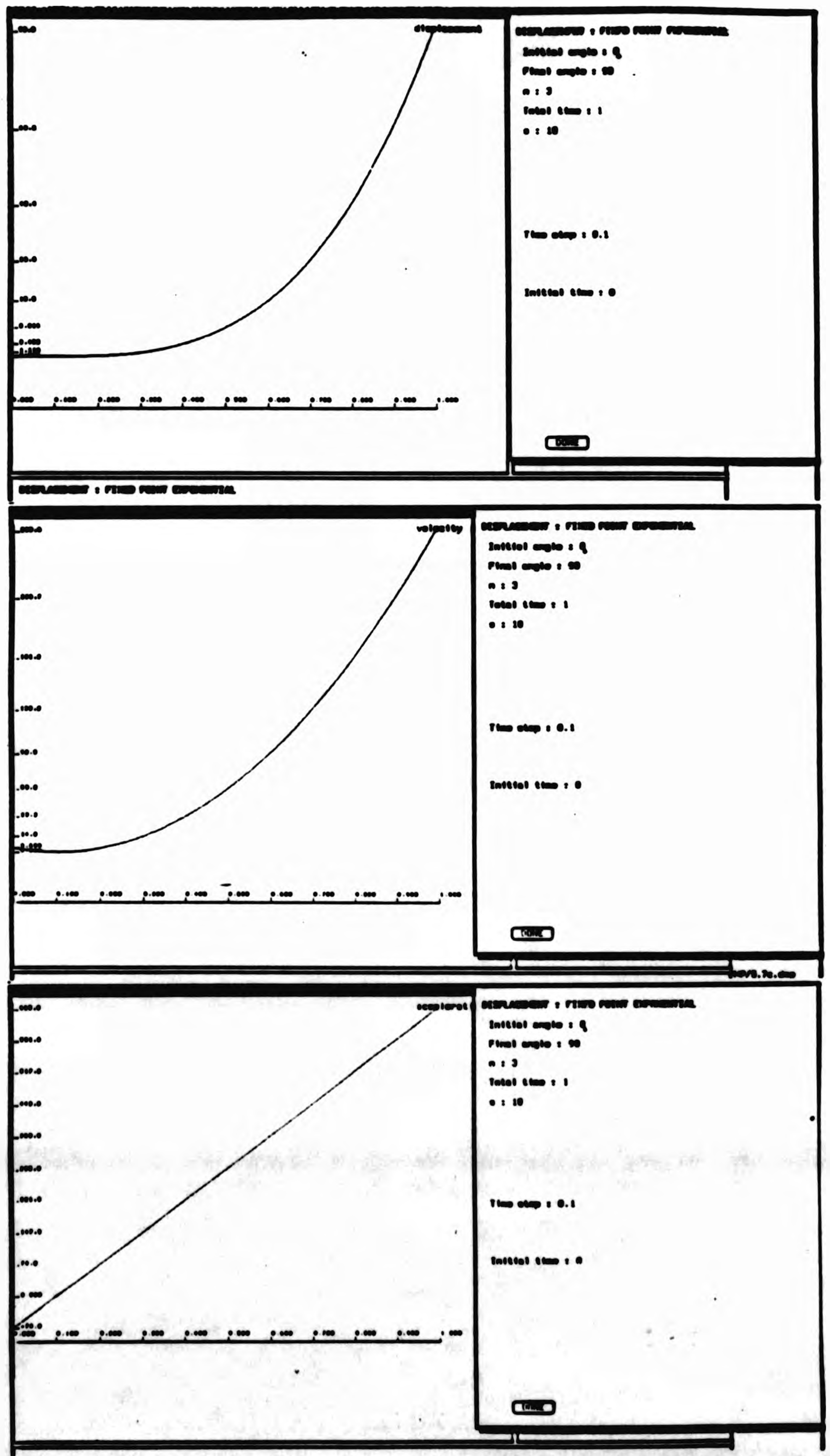


DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 2
Time step : 0.1
Initial time : 0



DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 2
Time step : 0.1
Initial time : 0

**Fig 5.6 : Exponential without velocity constraints, plan 2, $1 < s < 3$.**



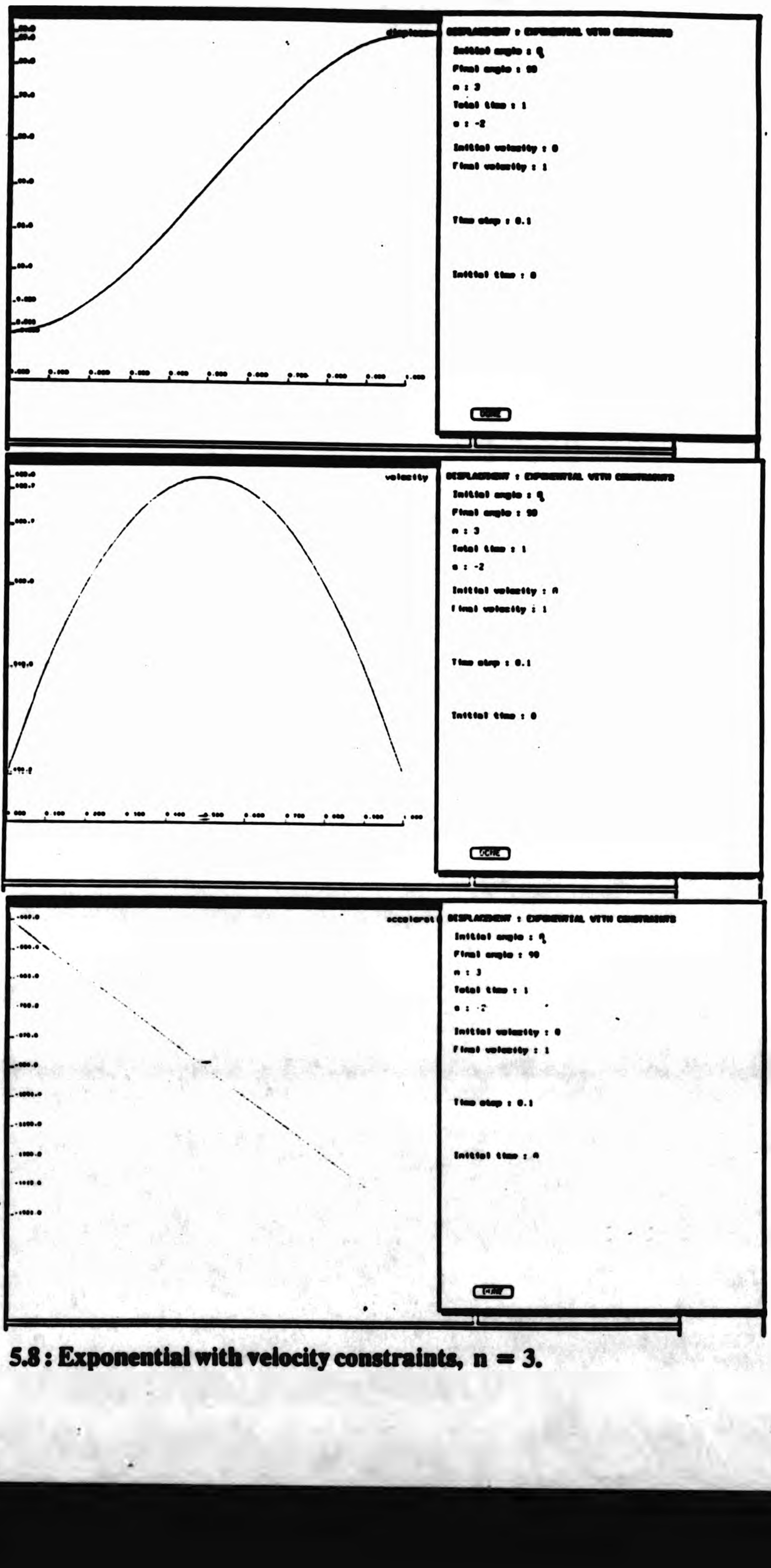
DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 10
Time step : 0.1
Initial time : 0

DISPLACEMENT : FIXED POINT EXPONENTIAL

DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 10
Time step : 0.1
Initial time : 0

DISPLACEMENT : FIXED POINT EXPONENTIAL
Initial angle : 0
Final angle : 90
n : 3
Total time : 1
s : 10
Time step : 0.1
Initial time : 0

**Fig 5.7 : Exponential without velocity constraints, plan 2, s > 3.**



DISPLACEMENT : EXPONENTIAL WITH CONSTRAINTS

Initial angle : 0
Final angle : 90
n : 3
Total time : 1
a : -2
Initial velocity : 0
Final velocity : 1
Time step : 0.1
Initial time : 0

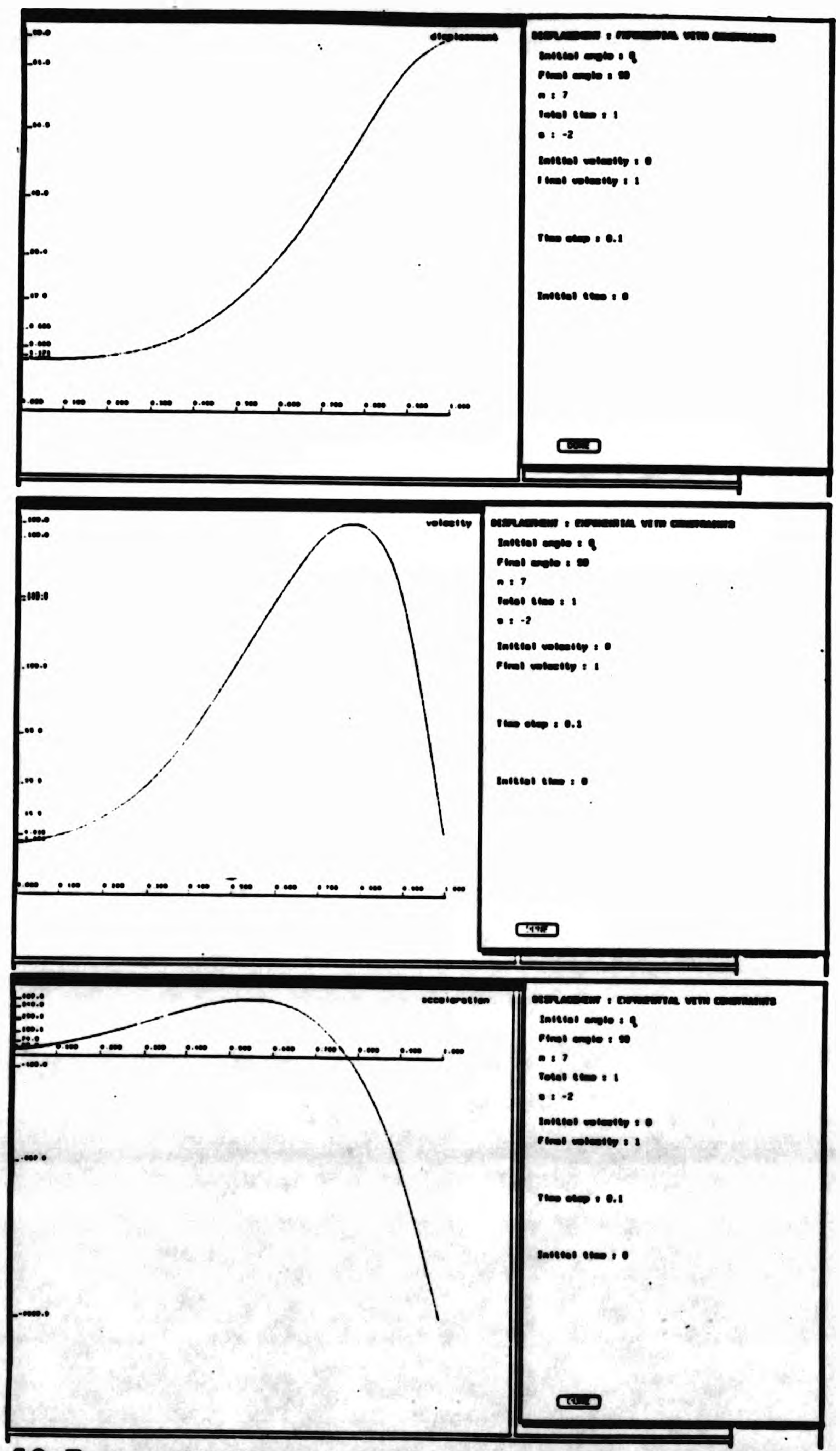
DISPLACEMENT : EXPONENTIAL WITH CONSTRAINTS

Initial angle : 0
Final angle : 90
n : 3
Total time : 1
a : -2
Initial velocity : 0
Final velocity : 1
Time step : 0.1
Initial time : 0

DISPLACEMENT : EXPONENTIAL WITH CONSTRAINTS

Initial angle : 0
Final angle : 90
n : 3
Total time : 1
a : -2
Initial velocity : 0
Final velocity : 1
Time step : 0.1
Initial time : 0

**5.8 : Exponential with velocity constraints, n = 3.**



DISPLACEMENT : EXPONENTIAL WITH CONSTRAINTS
Initial angle : 0
Final angle : 90
n : 7
Total time : 1
a : -2
Initial velocity : 0
Final velocity : 1
Time step : 0.1
Initial time : 0

DISPLACEMENT : EXPONENTIAL WITH CONSTRAINTS
Initial angle : 0
Final angle : 90
n : 7
Total time : 1
a : -2
Initial velocity : 0
Final velocity : 1
Time step : 0.1
Initial time : 0

DISPLACEMENT : EXPONENTIAL WITH CONSTRAINTS
Initial angle : 0
Final angle : 90
n : 7
Total time : 1
a : -2
Initial velocity : 0
Final velocity : 1
Time step : 0.1
Initial time : 0

**5.9 : Exponential with velocity constraints, n > 7.**

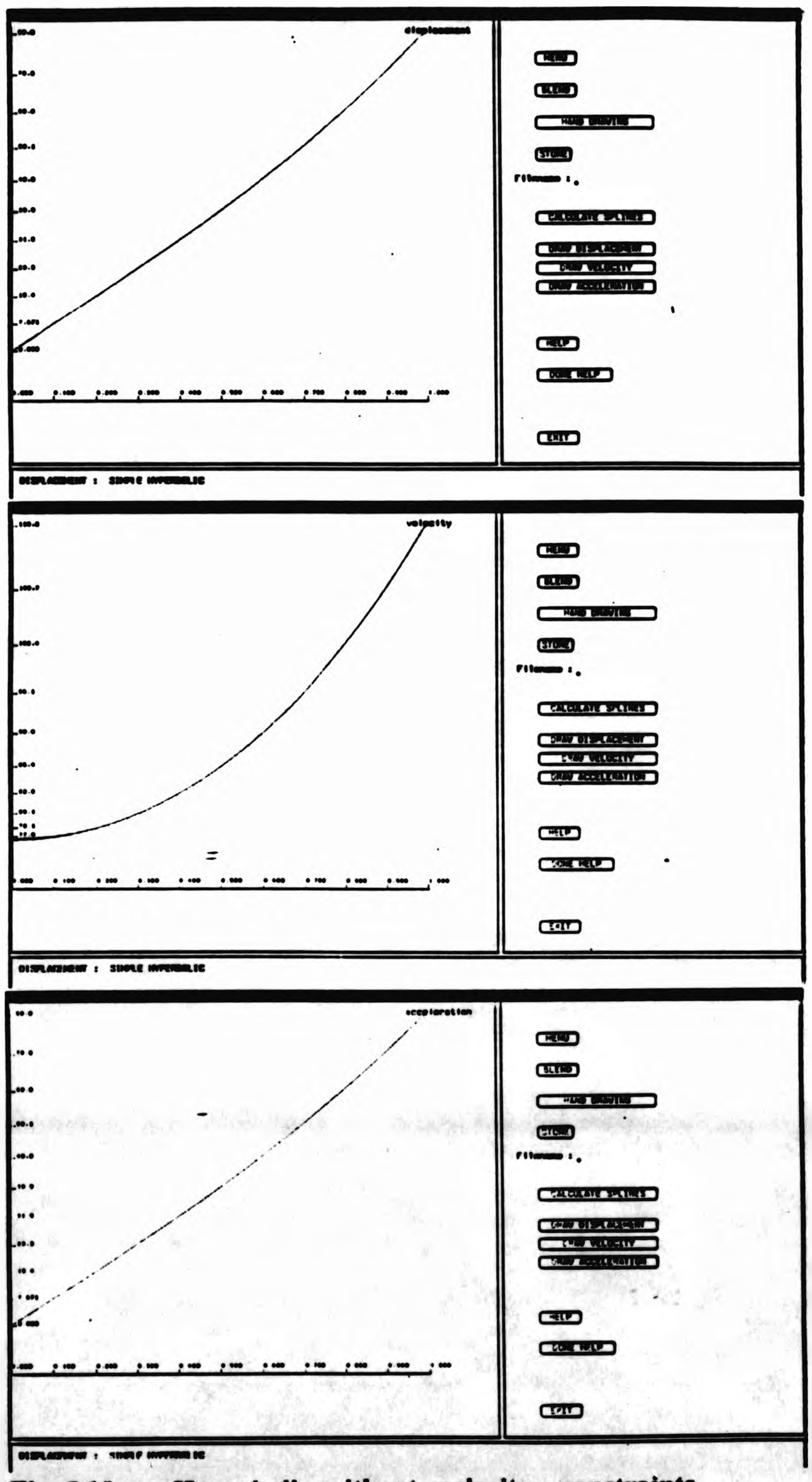


**Fig 5.10 : Hyperbolic without velocity constraints.**

**With velocity constraints**

This profile satisfies the displacement constraints $\theta(0) = \theta_0$, and $\theta(T) = \theta_1$ and the velocity constraints $\dot{\theta}(0) = \dot{\theta}_0$ and $\dot{\theta}(T) = \dot{\theta}_1$. It requires the solution of a system of four linear equations. It is possible that the system can be unsolvable and in this case the system prompts the user to change the input values as necessary. The factor that affects the solution here is the simulation time $T$. However, even when the system has a solution, there might be stationary points in the displacement curve. In this case the system works much like in the case of the trigonometric profile with velocity constraints. One of the velocities - either the initial or final - is prescribed and the system searches within a given range to find possible alternatives for the other. For example the following set of data has stationary points : $\theta(0) = 0$, $\theta(2) = 5$, $\dot{\theta}(0) = -1$, $\dot{\theta}(2) = -5$. By fixing $\dot{\theta}(2) = -5$, the system suggests 6 values between 0 and 1 i.e $\dot{\theta}(0) = (0.1, 0.3, 0.4, 0.5, 0.6, 0.7\ )$. For any of these values the curve has no stationary points. The displacement has the shape of $\sinh(x)$ whereas the velocity and acceleration curves have the shape of a quadratic polynomial with the velocity curve slightly skewed to the left as shown in Fig 5.11.

### 5.3.4 Straight Line

This profile satisfies the displacement constraints $\theta(0) = \theta_0$, and $\theta(T) = \theta_1$. The velocity curve is a line parallel to the x-axis (constant velocity) and the acceleration is zero throughout the motion as shown in Fig 5.12.

## 5.4 Curves for acceleration

### 5.4.1 Constant

This profile has constant acceleration $\ddot{\theta}(t) = \ddot{\theta}_1$ for $0 \leq t \leq T$ and satisfies the displacement constraint $\theta(0) = \theta_0$ and the velocity constraint $\dot{\theta}(0) = \dot{\theta}_0$. The displacement is a quadratic curve and the velocity a straight line. The displacement has no stationary points. Fig 5.13 shows an example of this profile.

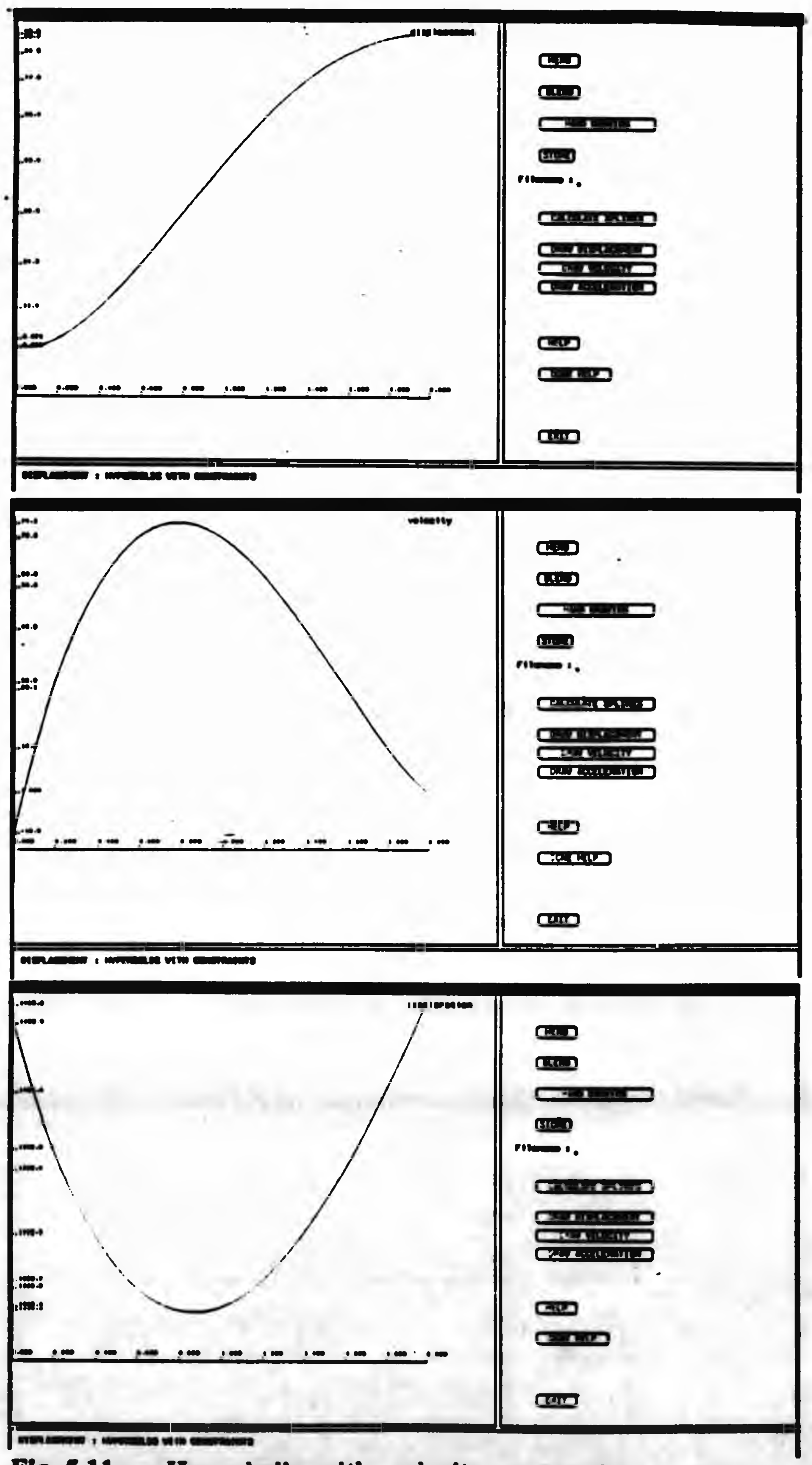
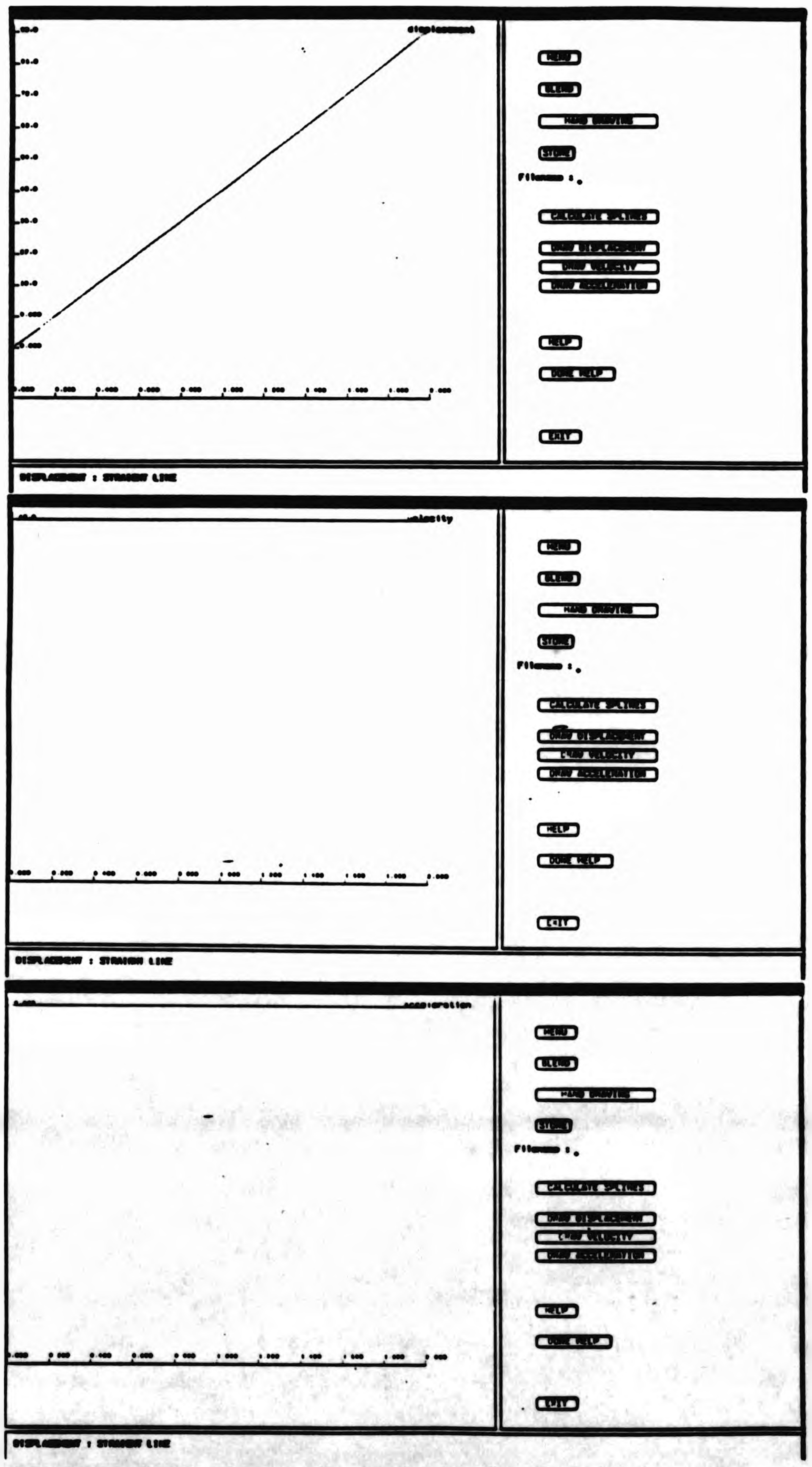

displacement
Filename :
DISPLACEMENT : HYPERBOLIC WITH CONSTRAINTS
velocity
Filename :
DISPLACEMENT : HYPERBOLIC WITH CONSTRAINTS
acceleration
Filename :
DISPLACEMENT : HYPERBOLIC WITH CONSTRAINTS

**Fig 5.11 : Hyperbolic with velocity constraints.**

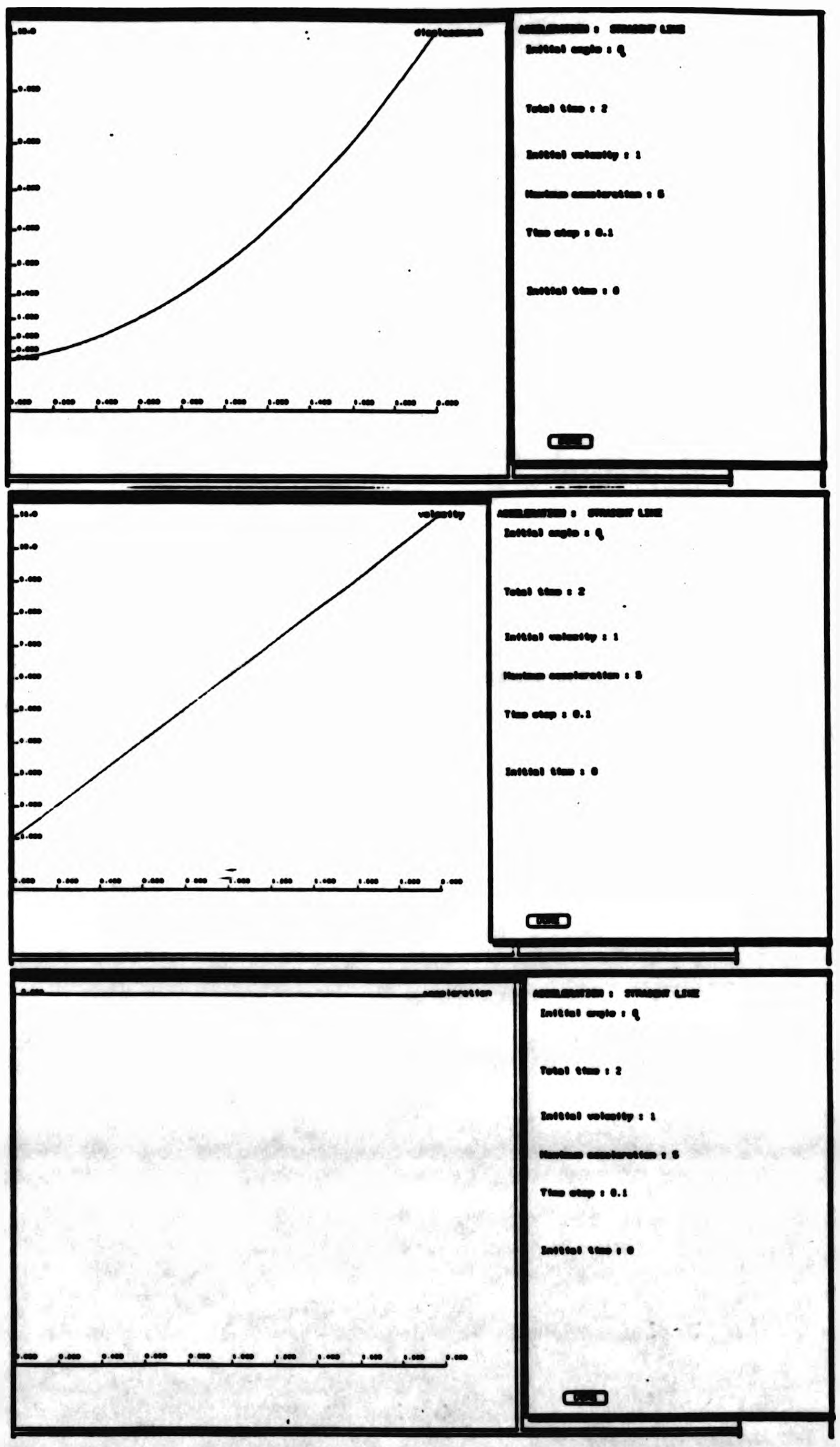


DISPLACEMENT : STRAIGHT LINE

DISPLACEMENT : STRAIGHT LINE

DISPLACEMENT : STRAIGHT LINE

**Fig 5.12 : Displacement straight line.**



```
ACCELERATION :  STRAIGHT LINE
Initial angle : 0

Total time : 2

Initial velocity : 1

Maximum acceleration : 5

Time step : 0.1

Initial time : 0
```

```
ACCELERATION :  STRAIGHT LINE
Initial angle : 0

Total time : 2

Initial velocity : 1

Maximum acceleration : 5

Time step : 0.1

Initial time : 0
```

```
ACCELERATION :  STRAIGHT LINE
Initial angle : 0

Total time : 2

Initial velocity : 1

Maximum acceleration : 5

Time step : 0.1

Initial time : 0
```

**Fig5.13 : Acceleration - constant.**

### 5.4.2 Trigonometric

This profile satisfies the displacement constraint $\theta(0) = \theta_0$, the velocity constraint $\dot{\theta}(0) = \dot{\theta}_0$ and acceleration constraints $\ddot{\theta}(0) = 0$ and $\ddot{\theta}(T) = 0$. In this profile the acceleration is expressed as a function of $\sin(\frac{\pi t}{T})$. The acceleration achieves its maximum amplitude at $t = \frac{T}{2}$. The velocity and displacement curves are monotonically increasing functions for positive maximum acceleration and monotonically decreasing for negative maximum acceleration. Fig 5.14 shows an example of this profile.

### 5.4.3 Ellipse

The elliptic curve for acceleration satisfies the displacement constraint $\theta(0) = \theta_0$, the velocity constraint $\dot{\theta}(0) = \dot{\theta}_0$ acceleration constraints $\ddot{\theta}(0) = \ddot{\theta}_0$, $\ddot{\theta}(T) = \ddot{\theta}_1$ and $\ddot{\theta}(\frac{T}{2}) = \ddot{\theta}_0$. The curve is a semi-ellipse (the upper half of the ellipse in the positive y-plane) and the maximum acceleration occurs when the slope of the ellipse changes sign. Fig 5.15 shows an example of this profile.

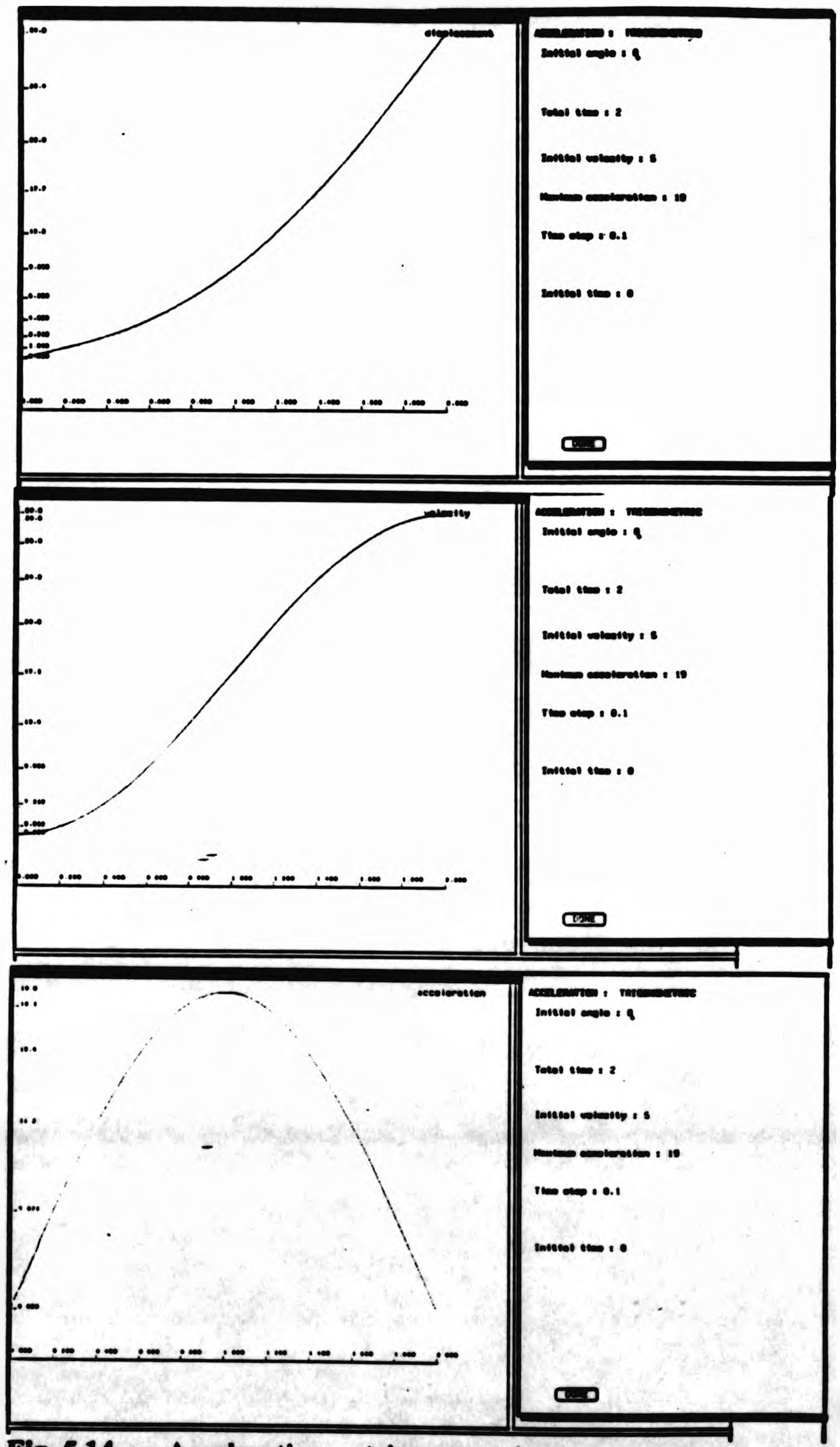
## 5.5 Curves for velocity

### 5.5.1 Straight line

This profile satisfies the velocity constraints $\dot{\theta}(0) = \dot{\theta}_0$ and $\dot{\theta}(T) = \dot{\theta}_1$ as well the displacement constraint $\theta(0) = \theta_0$. The displacement curve is quadratic without a stationary point and the acceleration is constant. The velocity and displacement curves have gradients of the same sign. Fig 5.16 shows an example of this profile.

### 5.5.2 Circle

This profile satisfies three velocity constraints : $\dot{\theta}(0) = \dot{\theta}_0$, $\dot{\theta}(T_1) = \dot{\theta}_m$ and $\dot{\theta}(T) = \dot{\theta}_1$ for $0 \leq T_1 \leq T$. It also satisfies the displacement constraint $\theta(0) = \theta_0$.



ACCELERATION : TRIGONOMETRIC
Initial angle : 0

Total time : 2

Initial velocity : 5

Maximum acceleration : 10

Time step : 0.1

Initial time : 0

DONE

ACCELERATION : TRIGONOMETRIC
Initial angle : 0

Total time : 2

Initial velocity : 5

Maximum acceleration : 10

Time step : 0.1

Initial time : 0

DONE

ACCELERATION : TRIGONOMETRIC
Initial angle : 0

Total time : 2

Initial velocity : 5

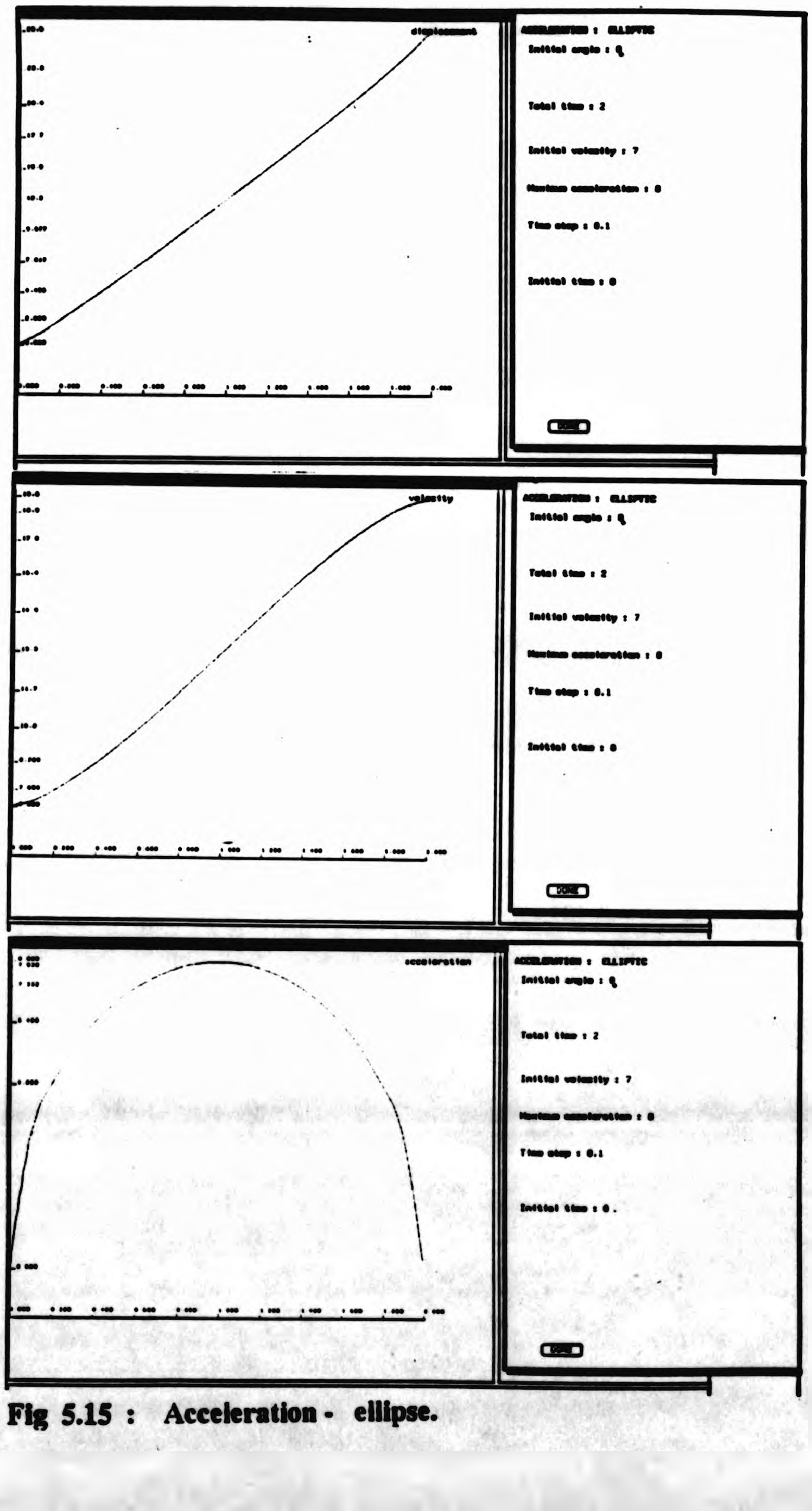
Maximum acceleration : 10

Time step : 0.1

Initial time : 0

DONE

**Fig 5.14 : Acceleration - trigonometric.**

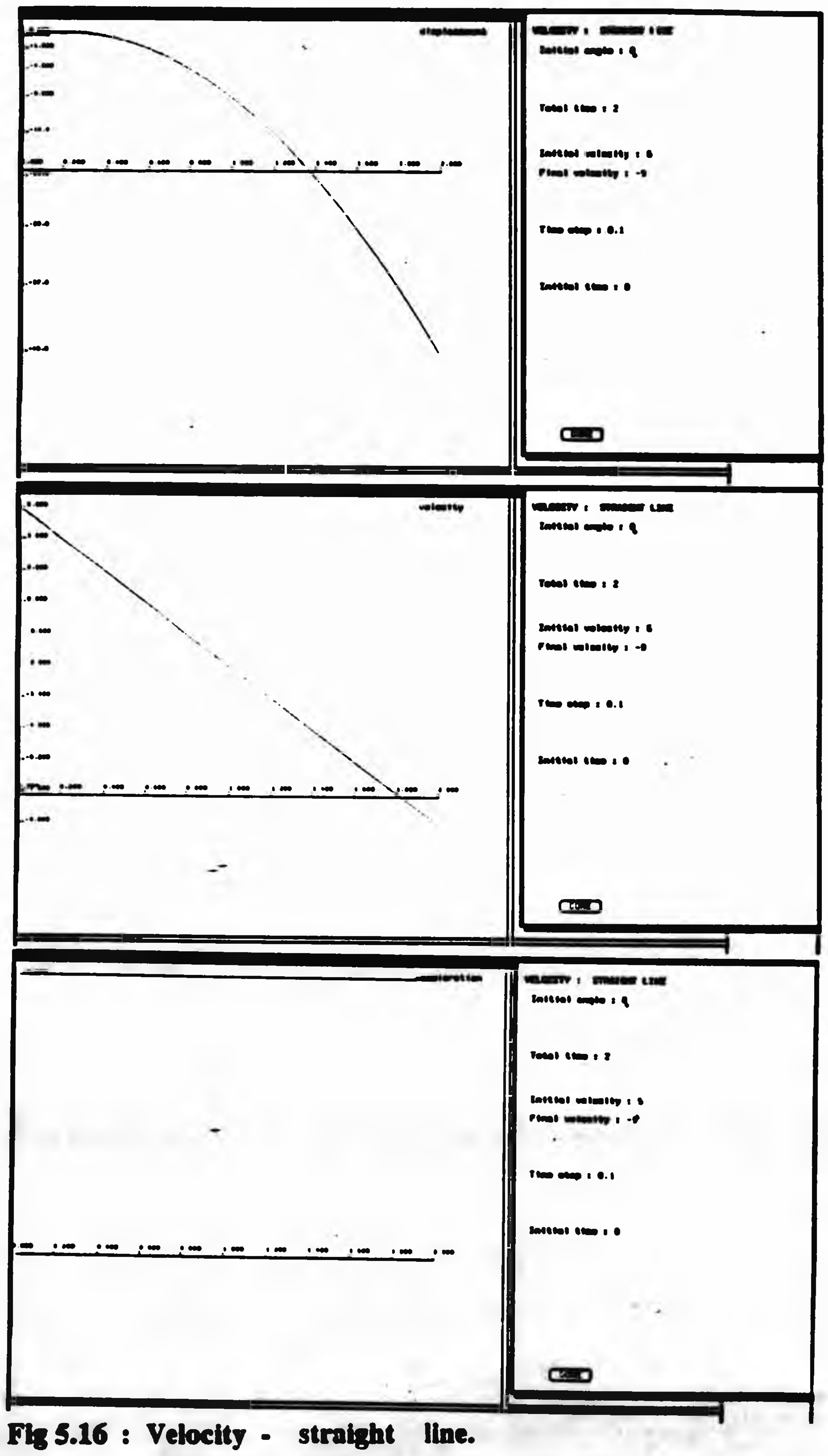


ACCELERATION : ELLIPTIC
Initial angle : 0
Total time : 2
Initial velocity : 7
Maximum acceleration : 8
Time step : 0.1
Initial time : 0

ACCELERATION : ELLIPTIC
Initial angle : 0
Total time : 2
Initial velocity : 7
Maximum acceleration : 8
Time step : 0.1
Initial time : 0

ACCELERATION : ELLIPTIC
Initial angle : 0
Total time : 2
Initial velocity : 7
Maximum acceleration : 8
Time step : 0.1
Initial time : 0 .

**Fig 5.15 : Acceleration - ellipse.**



VELOCITY : STRAIGHT LINE
Initial angle : 0
Total time : 2
Initial velocity : 5
Final velocity : -9
Time step : 0.1
Initial time : 0

VELOCITY : STRAIGHT LINE
Initial angle : 0
Total time : 2
Initial velocity : 5
Final velocity : -9
Time step : 0.1
Initial time : 0

VELOCITY : STRAIGHT LINE
Initial angle : 0
Total time : 2
Initial velocity : 5
Final velocity : -9
Time step : 0.1
Initial time : 0

**Fig 5.16 : Velocity - straight line.**

Although the three points will always define a circle, it is not necessary that they are on the same semi-circle (upper or lower). This condition must be satisfied otherwise the acceleration curve can not be defined properly as shown in Appendix A. In order to ensure that the points are on the same semi-circle, the system detects this problem when it occurs and searches for possible values of $T_1$ (for prescribed $\dot{\theta}_0$, $\dot{\theta}_m$, $\dot{\theta}_1$ and T), such as the three points are on the same semi-circle.

For example the following set of data appears to be erroneous according to our model, (0,1), (1,3) and (0.5,2). The system finds that for $0 < t < 1$ there is only one possibility for $T_1 = 0.4$. Therefore the set of values (0,1), (1,3) and (0.4,2) is correct. None of the three curves has stationary points. Fig 5.17 shows an example of this profile.

### 5.5.3 Circular fairing

Consider that the body is moving on the x-y plane along a path (Fig 5.18) over a time interval $T$. The projected velocity of the body on the x-axis, $V$, is uniform. The body starts from point $A$ at the beginning of time and increases its velocity constantly until it passes through point $B$ at time $T_1$ .It then travels at constant speed until it arrives point $C$ at time $T_2$ and it finally decreases its velocity constantly until it arrives at point $D$ as shown in Fig 5.18. $AB$ is an arc of a circle with radius $r$ and $CD$ is an arc of a circle with radius $R$.

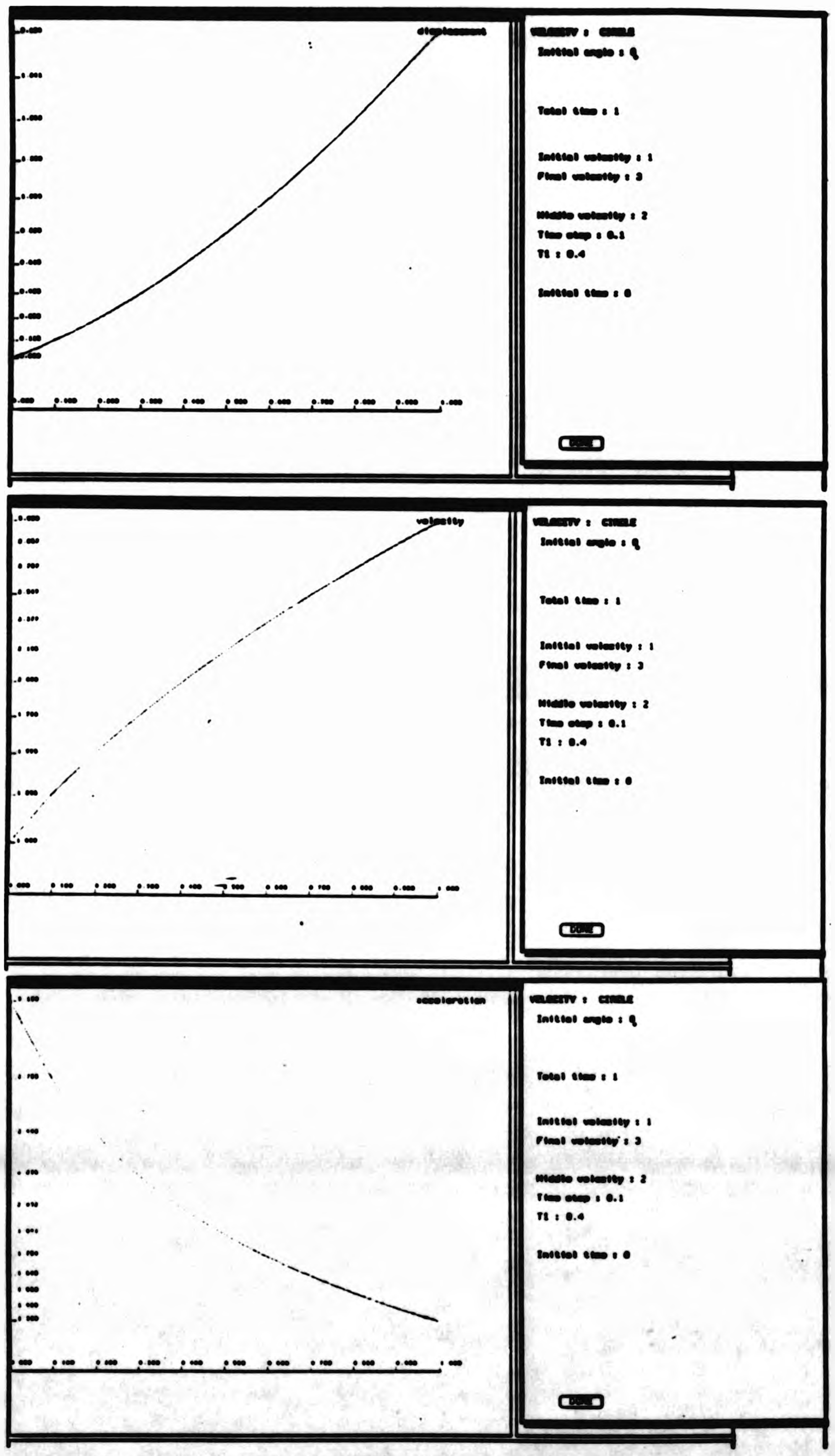
Since the body moves with constant velocity :

$$V = \frac{\pi R}{2T_1} = \frac{\pi R}{2(T - T_1)} = \frac{l}{T_2 - T_1}$$

If t is the time elapsed since the body departed $A$ then the distance travelled in $AB$ is : $r(1 - \cos\theta)$ where

$$\theta = \frac{\pi t}{2T_1} = \frac{Vt}{r}$$

In $BC$ the distance travelled is $V(t - T_1)$



```
VELOCITY : CIRCLE
 Initial angle : 0

 Total time : 1

 Initial velocity : 1
 Final velocity : 3

 Middle velocity : 2
 Time step : 0.1
 T1 : 0.4

 Initial time : 0
```

```
VELOCITY : CIRCLE
 Initial angle : 0

 Total time : 1

 Initial velocity : 1
 Final velocity : 3

 Middle velocity : 2
 Time step : 0.1
 T1 : 0.4

 Initial time : 0
```

```
VELOCITY : CIRCLE
 Initial angle : 0

 Total time : 1

 Initial velocity : 1
 Final velocity : 3

 Middle velocity : 2
 Time step : 0.1
 T1 : 0.4

 Initial time : 0
```

**Fig 5.17 : Velocity - circle.**

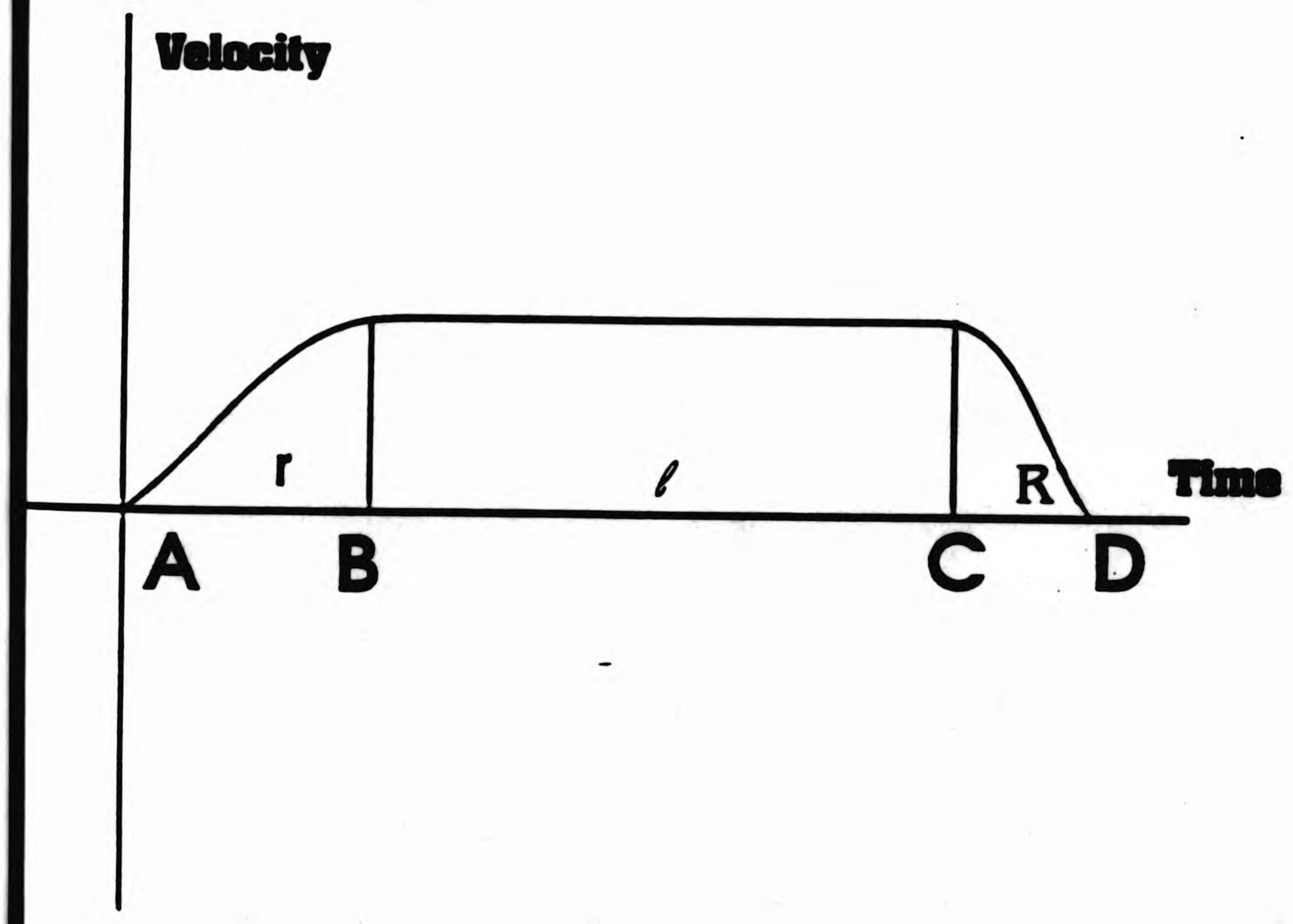


**Fig 5.18 : Circular fairing. The body moves on the x-y plane. The projected velocity on the x-axis is uniform.**

In $CD$ the distance travelled is $R \sin \phi$ where :

$$\phi = \frac{\pi(t - T_2)}{2(T - T_2)} = \frac{V(t - T_2)}{R}$$

Hence :

$$\theta(t) = \begin{cases} r\left(1 - \cos\left(\frac{Vt}{r}\right)\right) & \text{for AB} \\ V(t - T_1) + r & \text{for BC} \\ R \sin\left(\frac{V(t - T_2)}{R}\right) + r + R & \text{for CD} \end{cases}$$

$$\dot{\theta}(t) = \begin{cases} V \sin\left(\frac{Vt}{r}\right) & \text{for AB} \\ V & \text{for BC} \\ V \cos\left(\frac{V(t - T_2)}{R}\right) & \text{for CD} \end{cases}$$

$$\ddot{\theta}(t) = \begin{cases} \frac{V^2}{r} \cos\left(\frac{Vt}{r}\right) & \text{for AB} \\ 0 & \text{for BC} \\ -\frac{V^2}{R} \sin\left(\frac{V(t - T_2)}{R}\right) & \text{for CD} \end{cases}$$

If the times and the velocity are supplied, then the distances can be calculated :

$$r = \frac{2T_1 V}{\pi}$$

$$l = V(T_2 - T_1)$$

$$R = \frac{2V(T - T_2)}{\pi}$$

Alternatively, if the distances and velocity are given, the times are found as :

$$T_1 = \frac{\pi r}{2V}$$

$$T_2 = \frac{l}{V} + \frac{\pi r}{2V}$$

$$T = \frac{\pi(R + r)}{2V} + \frac{l}{V}$$

These two profiles have been implemented into the system and Fig 5.19 shows their shape.

## 5.6 Curve blending

This section describes how the curves described in the previous sections can be blended together to produce profiles that behave differently from the ones above. Given two continuous functions $f_1(t)$ and $f_2(t)$ where $t \in [t_0,\ t_1]$, we can define a function $R(t)$ such that :

$$R(t) = \alpha(t)f_1(t) + (1 - \alpha(t))f_2(t)$$

where $\alpha(t)$ is an arbitrary function with the following properties:

- $\alpha(t)$ is continuous in $[t_0, t_1]$
- $\alpha(t_0) = 1$
- $\alpha(t_1) = 0$

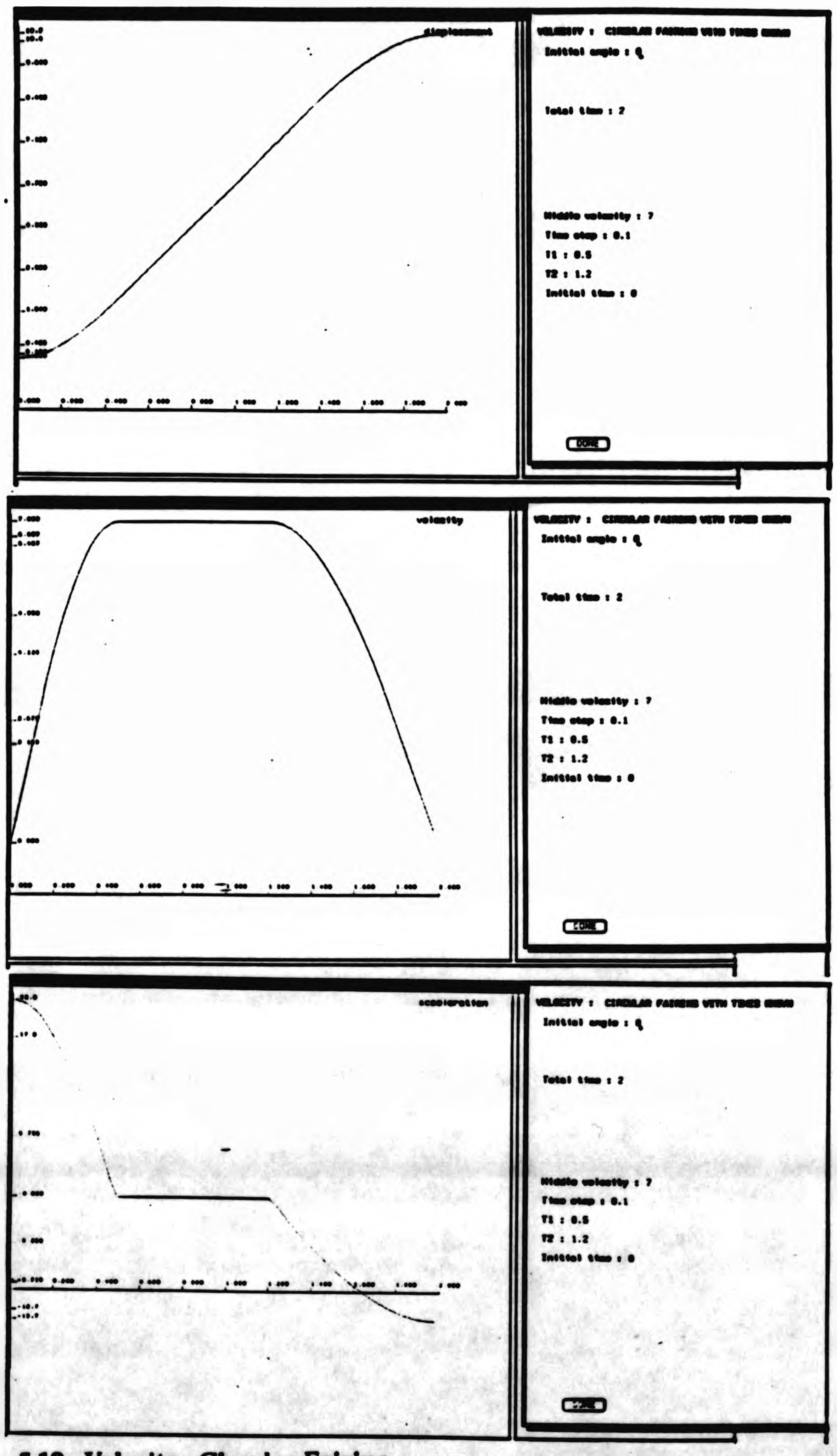
We define function $R(t)$ as being the blended function of $f_1(t)$ and $f_2(t)$. When the three properties of $\alpha(t)$ are satisfied then $R(t_0) = f_1(t_0)$ and $R(t_1) = f_2(t_1)$. For given $f_1(t)$ and $f_2(t)$ the behaviour of $R(t)$ depends solely on $\alpha(t)$. We shall consider four cases :

### Linear case

Let $\alpha(t) = at + b$ then

$$at_0 + b = 1 \quad and \quad at_1 + b = 0 \Rightarrow$$

$$a \;=\; \frac{-1}{t_1 - t_0} \quad and \quad b \;=\; \frac{t_1}{t_1 - t_0}$$



```
VELOCITY :  CIRCULAR FAIRING WITH [illegible]
Initial angle : 0

Total time : 2

Middle velocity : 7
Time step : 0.1
T1 : 0.5
T2 : 1.2
Initial time : 0
```

```
VELOCITY :  CIRCULAR FAIRING WITH [illegible]
Initial angle : 0

Total time : 2

Middle velocity : 7
Time step : 0.1
T1 : 0.5
T2 : 1.2
Initial time : 0
```

```
VELOCITY :  CIRCULAR FAIRING WITH [illegible]
Initial angle : 0

Total time : 2

Middle velocity : 7
Time step : 0.1
T1 : 0.5
T2 : 1.2
Initial time : 0
```

**5.19 : Velocity - Circular Fairing**

**Exponential case**

Consider $\alpha(t) = ae^{t} + be^{2t}$ then

$$0 = ae^{t_1} + be^{2t_1} \quad and \quad 1 = ae^{t_0} + be^{2t_0} \Rightarrow$$

$$b = \frac{1}{e^{2t_0} - e^{t_0+t_1}} \quad and \quad a = \frac{-e^{t_1}}{e^{2t_0} - e^{t_0+t_1}}$$

**Hyperbolic case**

Consider $\alpha(t) = a\cosh(t) + b\sinh(t)$, then :

$$a\cosh(t_0) + b\sinh(t_0) = 1 \quad and \quad a\cosh(t_1) + b\sinh(t_1) = 0 \Rightarrow$$

$$a = \frac{\sinh t_1}{\sinh(t_1 - t_0)} \quad and \quad b = \frac{-\cosh t_1}{\sinh(t_1 - t_0)}$$

**Trigonometric**

Consider $\alpha(t) = a\sin(t) + b\cos(t)$, then

$$a\sin(t_0) + b\cos(t_1) = 1 \quad and \quad a\sin(t_1) + b\cos(t_1) = 0 \Rightarrow$$

$$b = \frac{\sin(t_1)}{\sin(t_1 - t_0)} \quad a = \frac{-\cos(t_1)}{\sin(t_1 - t_0)}$$

The blending function is used when the functions are analytic over the same span of time and it is not used to blend two functions that occupy successive spans of time.

## 5.7 Free hand drawing using Kochanek splines

The previous sections described how profiles which can be produced from analytical functions are constructed individually and how they can be blended with one another.

It is, however, sometimes necessary to use profiles that cannot be expressed by analytic functions. In this case, discrete data is usually available that can be used to define the curve.

The problem addressed here is that given a set of points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ find a smooth curve that passes through these points. The resulting smooth function is called the ***interpolant*** and the portion of the curve between two points $(x_i, y_i)$ and $(x_{i+1}, y_{i+1})$ is called a ***span***. Global and local control of the curve is important to animation. It has been shown that low degree piecewise polynomials are more stable. These splines are used to allow the user to define the value of functions at key-frames where the system calculates values of the function at in-between frames. Kochanek splines were chosen because they are computationally inexpensive and easy to control locally, i.e it is easy to alter individual spans of the interpolating function, Kochanek [84]. We shall consider the parametric form of the interpolating function. Let $P(t)$ be the interpolating function such that :

$$P(t) = (x(t), y(t)) \quad for \quad t \ in \ [0, 1]$$

The curve has to pass through the points $(x_0, y_0), \ldots, (x_n, y_n)$. Let us consider that the number of interpolating points between any two points $(x_i, y_i)$ and $(x_{i+1}, y_{i+1})$ is $N$ and that this is fixed for all portions of the curve. Then the interval between these points is $\Delta$ where $\Delta = \frac{1}{N+1}$ and therefore the parameter t will take the values $t = 0, \Delta, 2\Delta, \ldots, 1$. The interpolant will be a collection of cubic-splines joined together with some continuity constraints.

Each cubic polynomial $at^3+bt^2+ct+d$ can be described by defining the four coefficients a, b, c and d. These coefficients can be found by

- constraining the spline to pass through the starting and finishing points of the interval and
- fixing the tangent vectors at the start and end points.

The tangent vector at $P(x_i, y_i)$ is defined as :

$$D_i \ = \ \left(\frac{dx}{ds}, \frac{dy}{ds}\right)$$

where $s$ is a function of $t$. A cubic polynomial can be expressed as a combination of the four basic functions :

$$s^3,\ s^2,\ s^1,\ s^0$$

but for the purpose of animation the Hermite interpolation basis are chosen in order for the functions to have the following values :

| Value of function or derivative at boundary values | $h_1$ | $h_2$ | $h_3$ | $h_4$ |
|---|---|---|---|---|
| function value at s = 0 | 1 | 0 | 0 | 0 |
| function value at s = 1 | 0 | 1 | 0 | 0 |
| derivative value at s = 0 | 0 | 0 | 1 | 0 |
| derivative value at s = 1 | 0 | 0 | 0 | 1 |

The basis functions are :

$$\begin{aligned} h_1(s) &= 2s^3 - 3s^2 + 1 \\ h_2(s) &= -2s^3 + 3s^2 \\ h_3(s) &= s^3 - 2s^2 + s \\ h_4(s) &= s^3 - s^2 \end{aligned}$$

we can now define the interpolating function as :

$$P(s) \;=\; P_i h_1(s) \;+\; P_{i+1} h_2(s) \;+\; D_i h_3(s) \;+\; D_{i+1} h_4(s)$$

where $P_i = (x_i, y_i)$, $P_{i+1} = (x_{i+1}, y_{i+1})$ and

$$D_i \;=\; \left(\frac{dx_i}{ds}, \frac{dy_i}{ds}\right) \qquad D_{i+1} \;=\; \left(\frac{dx_{i+1}}{ds}, \frac{dy_{i+1}}{ds}\right)$$

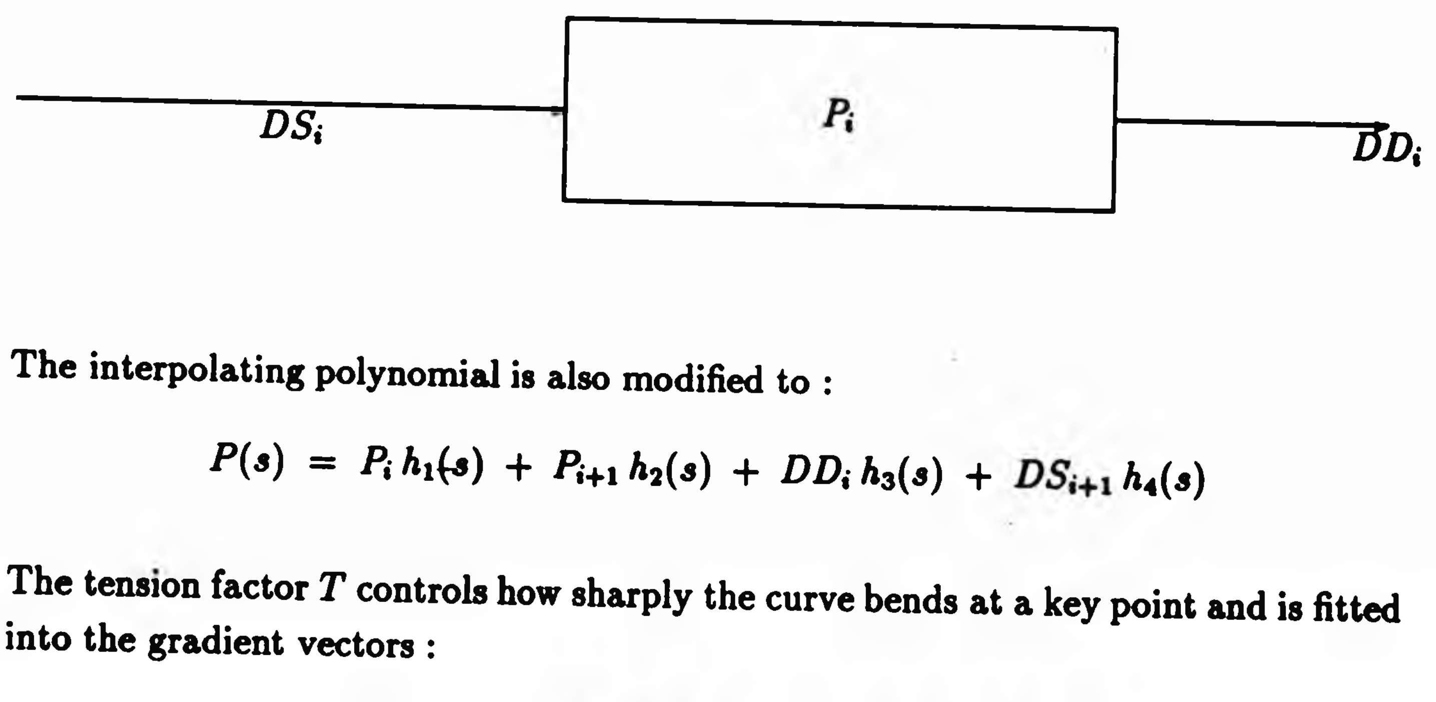
The equation can be re-arranged for computational considerations :

$$P(s) \;=\; (s^3\ s^2\ s\ 1)\begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} P_i \\ P_{i+1} \\ D_i \\ D_{i+1} \end{pmatrix}$$

The calculation of the matrix with the column vector is fixed within the interval so it has to calculated only once. The tangent vectors can be calculated as in the Catmull-Rom spline i.e :

$$D_i = \frac{1}{2}(P_{i+1} - P_{i-1}) = \frac{1}{2}[(P_{i+1} - P_i) + (P_i - P_{i-1})]$$

At the start and end points, $D_0$ and $D_n$ are arbitrarily defined. In order to achieve local control, three coefficients that are associated with the tangent vectors are defined, *bias*, *continuity*, and *tension*. These coefficients can be accommodated by separating the tangent vector into two parts, $DS_i$ and $DD_i$, the source and the destination derivative :



The interpolating polynomial is also modified to :

$$P(s) = P_i\, h_1(s) + P_{i+1}\, h_2(s) + DD_i\, h_3(s) + DS_{i+1}\, h_4(s)$$

The tension factor $T$ controls how sharply the curve bends at a key point and is fitted into the gradient vectors :

$$DS_i = DD_i = \frac{(1-T)}{2}[(P_{i+1} - P_i) + (P_i - P_{i-1})]$$

T ranges between -1 and 1 and its default value is zero in which case the spline is a Catmull-Rom spline. Increasing T above 1 is equivalent to bending the curve at the knot points and it generates loops around them.

The continuity factor C is used where the incoming and outcoming tangent vectors are different. The effect of having different tangent vectors is that the curve can be bent in such a way that it can go 'backwards'. The tangent vectors with tension factor zero are :

$$\begin{aligned} DS_i &= \frac{(1-C)}{2}(P_i - P_{i-1}) + \frac{(1+C)}{2}(P_{i+1} - P_i) \\ DD_i &= \frac{(1+C)}{2}(P_i - P_{i-1}) + \frac{(1-C)}{2}(P_{i+1} - P_i) \end{aligned} \tag{5.1}$$

The range of C is between -1 and 1 with a default value of zero and in this case $DD_i = DS_i$. Increasing the continuity factor to values greater than 1 makes the slope of the curve to look discontinuous at the knot points (as the function is a collection of cubic-curves that are not quite matched at their end points)

Finally the bias parameter B controls the direction of curve as it passes through a point with values between -1 and 1:

$$DS_i = DD_i = \frac{(1+B)}{2}(P_i - P_{i-1}) + \frac{(1-B)}{2}(P_{i+1} - P_i)$$

Combining the three parameters together we have :

$$\begin{aligned} DS_i &= \frac{(1-T)(1-C)(1+B)}{2}(P_i - P_{i-1}) + \frac{(1-T)(1+C)(1-B)}{2}(P_{i+1} - P_i) \\ DD_i &= \frac{(1-T)(1+C)(1+B)}{2}(P_i - P_{i-1}) + \frac{(1-T)(1-C)(1-B)}{2}(P_{i+1} - P_i) \end{aligned}$$

Increasing the bias factor to a large value will make the curve -at the knot points- to twist towards the left or the right as shown in Fig 5.22.

These three parameters can be used to affect the shape of the spline either locally, by changing the local tension, bias and continuity parameters for a span of the interpolant, or globally, by changing the parameters for the entire curve. Although all three parameters are well defined between -1 and 1, the user might want to experiment for values outside this range.The results that one might achieve are shown in Fig 5.20-5.23.

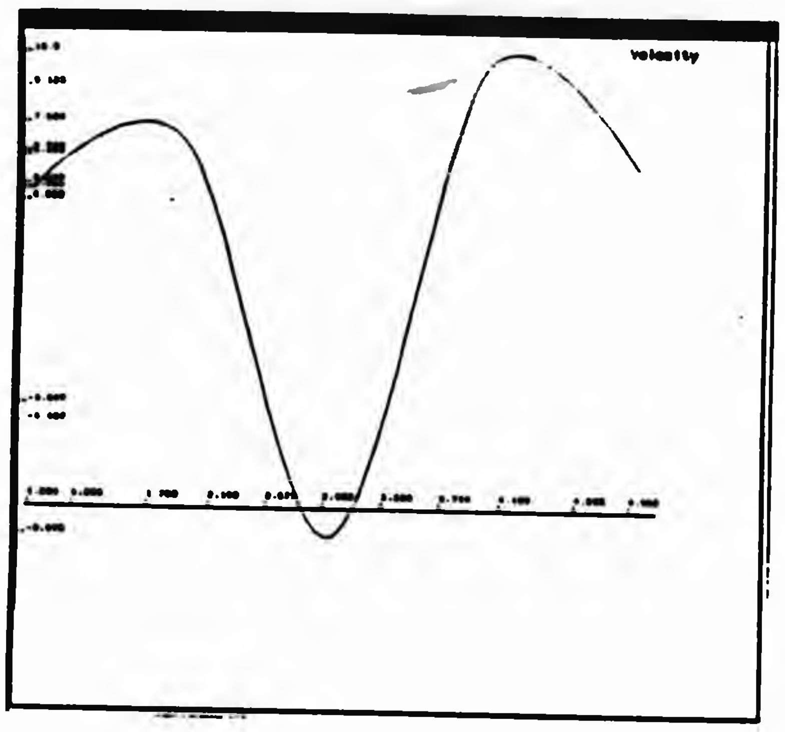


**Fig 5.20 : Konachek spline with all parameters set to zero.**

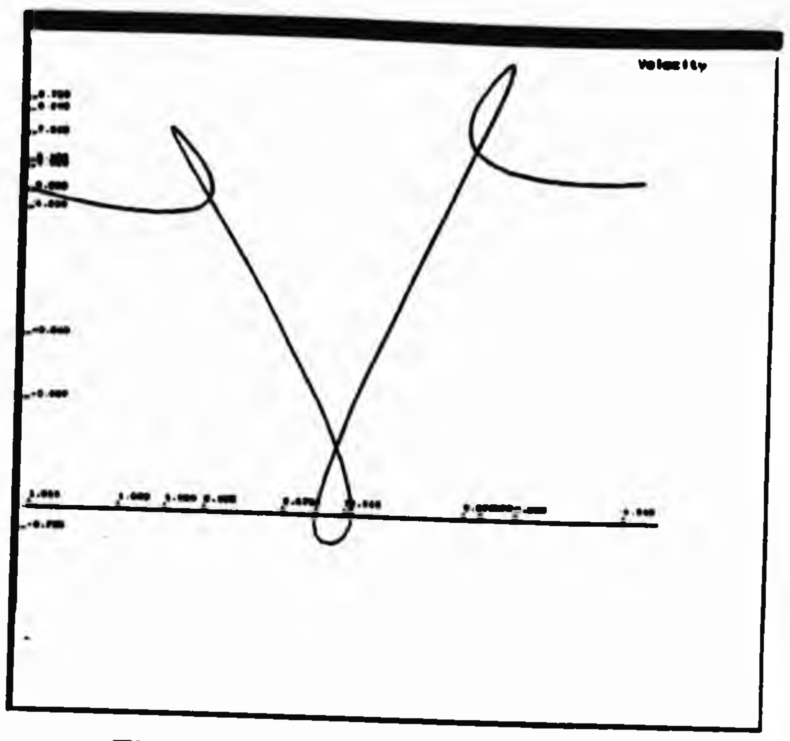


**Fig 5.21 : Same curve as in fig 5.20 but with tension factor 3.**

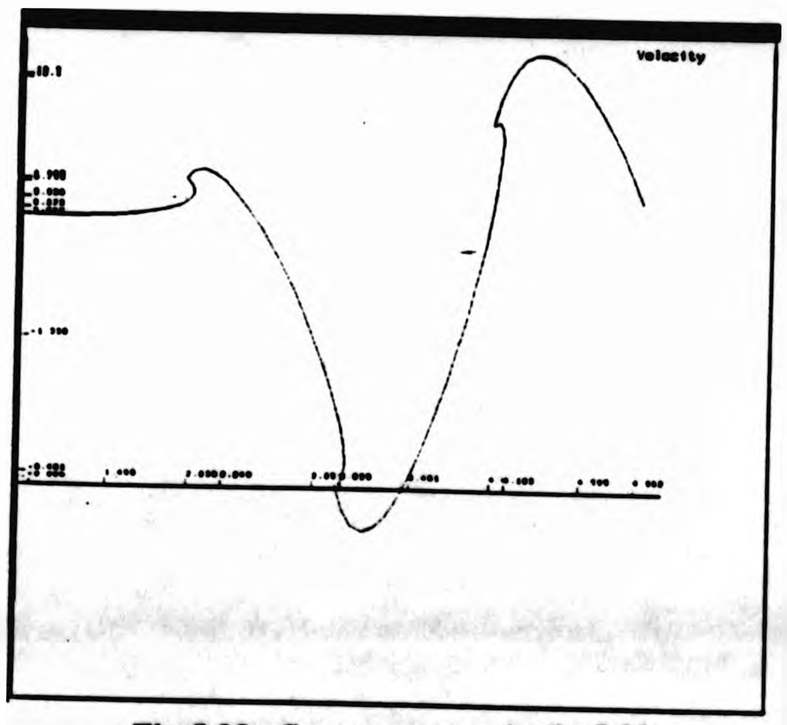


**Fig 5.22 : Same curve as in fig 5.20 but with bias factor 2.**

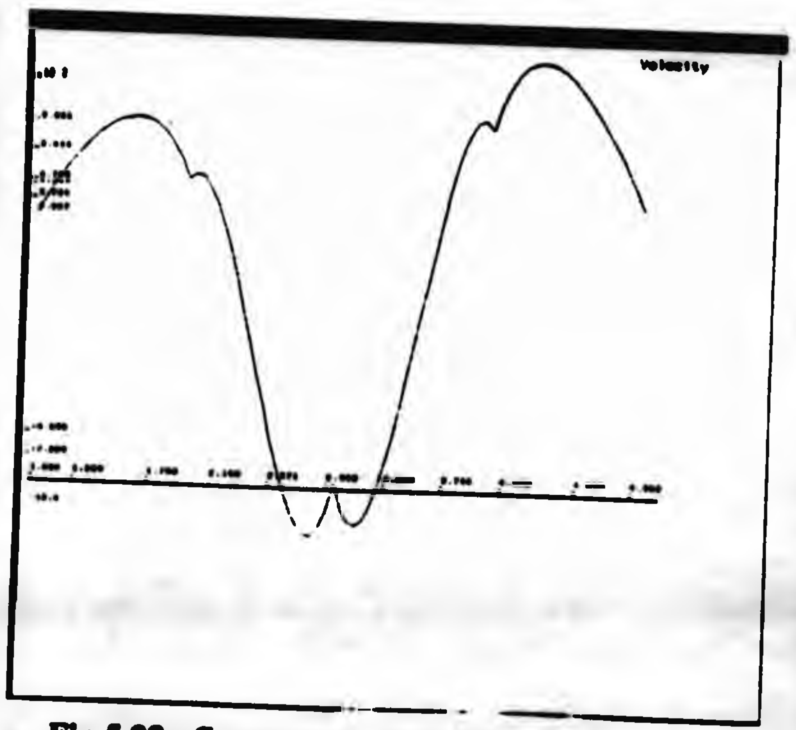


**Fig 5.23 : Same curve as in fig 5.20 but with continuity factor 2.**

Fig 5.20 shows the Kochanek spline for the points (1,5), (2,7), (3,-9), (4,10), (5,6) with tension factor 0 and Fig 5.21 shows the same curve with tension factor 3. Fig 5.22 shows the curve with tension factor 0 and bias factor 2 and Fig 5.23 shows the curve with tension and bias factor 0 but continuity factor set to 2. The changes of factors in the figures were affected globally but they can also be applied to individual spans of the curve.

## 5.8 Conclusions

This chapter described a way of defining the kinematic control profiles for displacement, velocity and acceleration. Table 5.1 illustrates the profiles presented in this chapter. The first column shows the profile type, the second shows the curve type and the third indicates the constraints imposed on that profile.

| Kinematic Profile Type | Curve Type | Constraints | | |
|---|---|---|---|---|
| | | Displacement | Velocity | Acceleration |
| Displacement | Trigonometric | Yes | No | No |
| | Trigonometric | Yes | Yes | No |
| | Hyperbolic | Yes | No | No |
| | Hyperbolic | Yes | Yes | No |
| | Exponential | Yes | No | No |
| | Exponential | Yes | Yes | No |
| | Straight Line | Yes | No | No |
| Acceleration | Constant | Yes | No | No |
| | Trigonometric | Yes | Yes | No |
| | Ellipse | No | Yes | Yes |
| Velocity | Straight Line | No | Yes | No |
| | Trigonometric | No | Yes | No |
| | Ellipse | Yes | Yes | No |

Table 5.1 : Types of constraint that apply to each kinematic profile.

We also presented a method for curve blending. It is used in animation to produce variations of the curves profiles. For example we may want to construct a profile that in the first span of the curve is behaving exponentially and in the second span trigonometrically. This kind of curve can be produced by using the blending technique.

Kochanek splines were used to design free-hand curve drawing. Kochanek splines were chosen for two reasons, firstly, they are computationally inexpensive and secondly, they offer easy local control. This latter feature is essential in designing acceleration and velocity profiles, as it is sometimes necessary to change local spans of the curve rather than the entire curve.

The next chapter discusses the implementation of these techniques together with the user interface. It also examines the design of the first-level motion control in the dynamical system and how dynamic and kinematic modes are combined.

# Chapter 6

# First Level Dynamic Motion Control

## 6.1 Introduction

The motion of humans and other articulated animals is goal directed. This means that in order to accomplish a motion, humans do not think in terms of torques or forces but in terms of the translations and orientations that the various links of the body should take in order to aehieve the desired motion. Most of these calculations are performed subconsciously and with minimal effort and therefore their implementation to an animation system is not a straightforward task.

The task may be simplified if it is divided into a number of layers where the lowest layer deals with individual degrees of freedom and the highest layer is where the body takes decisions upon what are the goals of the motion and how they might be accomplished. This approach is analogous to the onion skin scheme for operating systems. This chapter describes the lowest level layer or the kernel where motions are described by specifying positions of individual degrees of freedom. This layer also deals with configuration specification problems such as defining a new articulated model, order of rotations, and numerical schemes for finding positions for in-between frames.

The major problem in defining motions for articulated bodies is that either the final positions for the various links of the body are not known or even if they are, the method of moving from the starting position to the end position is not known. If the latter is the case, kinematics instead of dynamic analysis can be used to specify the motion and the user must have a number of ways for specifying the behaviour of the motion between starting and end positions. However in the first case, the use of dynamic analysis is apparent so the user can get an initial idea about the profile of the motion. In both cases some input from the user is required, regarding the final positions of the links. In order to take this burden from the user, the immediately higher layer should decide upon these input values.

## 6.2 Dynamic mode

When a degree of freedom is in dynamic mode, its motion is governed by the dynamic analysis torques and accelerations. It has to be stressed that while some degrees of freedom may be in kinematic mode and some in dynamic, nevertheless, dynamic analysis is performed on the entire body i.e degrees of freedom under kinematic control can not be excluded from the dynamic analysis. For a degree of freedom under dynamic control, there are two possible sub-modes of operation :

- Simple Dynamic mode
- Intensive Dynamic mode

In simple dynamic mode, the end position of the degree of freedom should be specified, when it is not specified, it is assumed to be one of the boundary limits of the degree of freedom. The dynamics system then calculates the torque that is needed to accomplish the motion and a torque-time profile is constructed to define the in-between positions of the link over the given time span.

In intensive dynamic mode, again the final angle is specified and the torque is calculated. A numerical integration technique is then used to calculate the next position of the link at the next frame. The new position is fed back into the system and a new torque is calculated. This process is repeated for each frame.

The difference between simple and intensive dynamic modes lies in the frequency of performing dynamic analysis and the way in which displacement profiles are constructed. In simple dynamic mode, dynamic analysis is performed only once and the torque-time profile is constructed from this information. In intensive dynamic mode this process is repeated at each frame and the torque-time profile is constructed from the discrete values that are found from the dynamic analysis.

The use of inverse Lagrangian dynamics requires a knowledge of the final positions, velocities and accelerations and that the body starts its motion from its current position. Hence the torque $\tau$ calculated from the formulation is the torque required to move the body from rest to these given values. Therefore the initial angle will always be the current position of the degree of freedom.

The way that the link is moving during the simulation interval when it operates in simple dynamic mode is fully defined by its torque-time profile. Four profiles are used in order to satisfy different boundary conditions.

## 6.2.1 Simple dynamic mode

### Target - angle oriented profile

Assume that the acceleration found from the dynamic simulation is $\alpha$. This profile is constructed to have constant acceleration $\alpha$ throughout the motion and initial velocity $V_0$ . The profile starts moving the link from its current position and ends when the degree of freedom has achieved its target final angle. The simulation time, therefore is not of any importance as there is no way to know beforehand how long the link will take to complete its motion. The sign of the acceleration has to be the same as the sign of the final target angle otherwise the motion will never be completed. Hence :

$$\begin{aligned} \ddot{\omega} &= \alpha \\ \dot{\omega}_i &= \dot{\omega}_{i-1} + \alpha dt \\ \omega_i &= \omega_{i-1} + \dot{\omega}_i dt \end{aligned} \tag{6.1}$$

## Time oriented profile

In this profile we assume constant acceleration throughout the motion and that motion starts with zero velocity. It starts moving the link from its current position and increases the target angle by $\theta_1$ . The motion lasts as long as the simulation time $T$. In order to achieve the latter condition, the time has to be scaled such that the angular displacement of the link increases by $\theta_1$ exactly when $t = T$, therefore :

$$\begin{aligned}\ddot{\omega}(t) &= \alpha \\ \dot{\omega}(t) &= \alpha t_p \\ \omega(t) &= \frac{\alpha t_p^2}{2}\end{aligned} \tag{6.2}$$

The motion takes place in terms actual time between $0 \leq t \leq T$. In order to ensure the above condition $T_p = \left(\frac{2\theta_1}{\alpha}\right)^{1/2}$ , where $T_p$ is the estimated simulation time, and this means that the time has to scaled according to the following formula :

$$t_p = \frac{t\sqrt{\frac{2\theta_1}{\alpha}}}{T} \tag{6.3}$$

This does not change the behaviour of the profile in any way. It means, however, that there is no continuity in velocity.

## Free - style

This is the simplest profile and is based on direct integration. The acceleration is assumed constant and hence :

$$\begin{aligned}\ddot{\omega}(t) &= \alpha \\ \dot{\omega}_{i+1} &= \alpha dt + \dot{\omega}_i \\ \omega_{i+1} &= \dot{\omega}_i dt + \omega_i\end{aligned} \tag{6.4}$$

where $\dot{\omega}_0$ and $\omega_0$ represent initial velocities and angles. This profile is suited for cases where the link does not have a target angle, but moves freely under the influence of external forces only.

### 6.2.2 Intensive dynamic mode

When a degree of freedom is in intensive mode, the initial and end conditions for the displacement, velocity and acceleration have to be specified, as do the simulation interval and the sample rate. By having this information, the number of frames can be calculated, and dynamic analysis is then performed at each frame. This means that there are as many accelerations as the number of frames and from these accelerations, the velocities and positions of the link at each frame must be calculated. The process can be depicted as follows :

$$\left(0, \dot{\theta}_0, \ddot{\theta}_0\right) \rightarrow \left(\theta_i, \dot{\theta}_i, \ddot{\theta}_i\right) \rightarrow \left(\theta_1, \dot{\theta}_1, \ddot{\theta}_1\right)$$

The 3-tuple $\left(\theta_i, \dot{\theta}_i, \ddot{\theta}_i\right)$ represents the intermediate values for displacement, velocity and acceleration and since the dynamics formulation calculates only the acceleration, a numerical technique must be devised to find velocities and displacements. The simplest numerical scheme can be found as follows :

$$\ddot{\theta} = \frac{d\dot{\theta}}{dt} \Rightarrow \dot{\theta} = \ddot{\theta}\, dt \ , \ \dot{\theta} = \frac{d\theta}{dt} \Rightarrow \theta = \frac{\ddot{\theta} dt^2}{2}$$

This involves one step integration at each frame.

## 6.3 Order of rotations

A link in a kinematic chain of an articulated body can have a maximum of three rotational degrees of freedom except the root that can have a maximum of six degrees of freedom - three rotational and three translational. The experimental model in Chapter 5 fixed the rotations for each link in the order Z-Y-X. This obviously is a special case and it certainly does not represent the order of rotation in a real articulated model. The user of an interactive animation system should be able to define the order of rotation for each link. Two assumptions are made at that stage :

- Any link other than the root has three degrees of freedom whereas the root has six.

- No two rotations in the same link involve the same axis.

This means that rotations of the type X-Z-X are not valid because X appears twice. These two constraints are restrictive but they will be dropped at a later stage. Having these two constraints, the choice of possible orders for rotations is narrowed to six :

| | | |
|---|---|---|
| X-Y-Z | X-Z-Y | Y-Z-X |
| Y-X-Z | Z-X-Y | Z-Y-X |

For each of these rotations, the values of $\alpha_i$ and $\theta_i$ have to specified. As can be seen from Table 4.1 in Chapter 4, there is a pattern for these values when the order of rotation is Z-Y-X. Fortunately this is true for all the other orders as well. Appendix B contains two types of tables, firstly when all degrees of freedom in the body are of the same order, and secondly, when they are not. The way that the second type of Table is read is that if we assume that link i has rotation type A and that link i+1 has rotation type B, then the Tables give the values of $\alpha_i$ and $\theta_i$ for link i+1. In the case that link i+1 is the root (where there is no preceding link), the first type of Table in Appendix B is used. Although the values of $\alpha_i$ and $\theta_i$ can vary according to the direction of the x-axis when the local coordinate systems are constructed, the direction has been taken in all cases such that $\alpha$ and $\theta$ will have the same sign.

## 6.4 Freezing degrees of freedom

Another feature of first level motion control is the ability to freeze, or lock, a degree of freedom to a particular value. The motion begins with the degree of freedom fixed to this value and either throughout the motion not to move the degree of freedom, or when the link tries to move, to apply a torque in the opposite direction in order to keep it in its locked value. The first approach seems easier to employ and it does produce realistic results.

## 6.5 Configuration file

The animation system should not operate with a fixed model only but it should be able to read information about an articulated body and readjust the dynamic and kinematic components, as well the user interface. The configuration file that is used in our animation system contains all the necessary information needed by these components, namely :

- Number of kinematic chains
- List of kinematic chains : the way that the kinematic chains are specified is that the first number represents the number of links in the chains and the remaining numbers identify the links according to the convention that the user devises.
- Positions of the origins and centres of mass : these values represent the initial positions of the origins and centres of mass where the body is currently located. It is wise that these values represent the actual dimensions of the body.
- The points of the tip of the end effectors. This information is used when the hierarchy of the system has to be changed and the tip of an end-effector may become the origin of a link.
- Limiting values for angles : These values are needed in order to avoid internal collisions of links of the body.
- Order of Rotations and shape of links : The order of rotation for each link is specified together with its shape. The shape is used by subsequent parts of the configuration file to generate graphics data for user and graphics interface.
- Inertia tensors and masses : Because inertia tensors are symmetric matrices, only the lower inertia tensor has to specified together with the mass. Although it is true that the mass is equal to $I_{00}$, the mass has to be specified in case the user wishes it to be different from $I_{00}$. This can happen when the user wants to experiment with different inertia tensors.
- Link names : The link names are used by the interface to indicate which link has been currently picked up, in order for the user to specify some parameters

for the link. It is also used by the kinematics interface for defining profiles for the picked link.

- Graphics data for user interface: The animator should be able to see a meaningful representation of the model on the screen. The graphics data for the user interface can be used to depict the articulated model on the screen.
- Graphics data for graphics interface : This data is different from the interface data in the sense that different coordinate systems may be used. In a general animation system, these two sets of data must be separated since the interface may run on one machine and the graphics on another. Another reason for having two different sets of data is that the figure for the user interface will be, in general, a simplified version of the model appearing in the graphics interface (e.g 2D interface model and 3D graphics model).

No information about the values of $\alpha_i$ and $\theta_i$ is needed because this is automatically deduced by the order of rotation and the Tables in Appendix C.

## 6.6 Ground reaction forces

The calculation of external forces is divided into two categories, that of automatic and that of heuristic forces. In the first category is the force of gravity which can be calculated directly from the formulation, once the position of the body is known. Into the latter category fall the ground reaction forces. The calculation of these forces by Wilhelms [85] used a system of springs and dampers. This requires that the forces are given an initial estimate based on the mass of the body and its support profile, and these estimates are then rectified as the dynamic analysis is performed. For example, if, after the initial estimate, the body continues to descend below the ground, an additional force is applied.

The derivation of the accelerations is based on the following assumptions :

- The model has L prototype links

- The model has three translational degrees of freedom at its root and that each link has three rotational degrees of freedom. The reason for this assumption is to simplify the notation of the mathematics involved, but the derivation can be applied to any other hierarchically based model
- The structure can be decomposed into N rationalised kinematic chains
- ${}^{k}P_{j}$ refers to a quantity P on a rationalised chain c and rationalised link j
- Links are numbered as $1, 2, 3 \ldots$ starting from the root towards the end-effectors
- The mass in the rationalised chain is placed at rationalised links 3i $i = 2, 3 \ldots$ and link number 6 for the root (the root is the only link with six degrees of freedom)

The model proposed here is based on an analytic derivation of the linear accelerations which should be applied to the body in order to set the ground reaction forces (normal and tangential) to a predefined value. This value is usually zero for the normal component, and in a static position the tangential forces are usually zero, too. However, when the body is in a dynamic state (moving), the tangential components can be set to a value that reflects the properties of the ground surface, i.e slippery, rough etc. The values of the desired accelerations for the translational degrees of freedom can be found by solving a system of three linear equations (one for each translational degree of freedom). Equation (6.5) gives the formula for finding the actuator force for the third translational degree of freedom. If we want to have zero resultant force then the RHS must be set to zero. However, if the system has a non-zero target linear acceleration ${}_{t}\ddot{q}_{3}$ then the equation can be modified so that the acceleration is equal to that value i.e the RHS of (6.5) must be $M\,{}_{t}\ddot{q}_{3}$ instead of zero. We assume zero resultant acceleration in the rest of this derivation.

From (3.32) we have :

$$F_3 = \sum_{k=1}^{N} tr\left(\frac{{}^{k}\partial W_3}{{}^{k}\partial q_3}\,{}^{k}D_3\right) - \sum_{k=1}^{N} g^T \frac{{}^{k}\partial W_3}{{}^{k}\partial q_3}\,{}^{k}c_3 = 0 \qquad (6.5)$$

Hence :

$$\sum_{k=1}^{N} tr\left(\frac{{}^{k}\partial W_3}{{}^{k}\partial q_3}\,{}^{k}D_3\right) = \sum_{k=1}^{N} g^T \frac{{}^{k}\partial W_3}{{}^{k}\partial q_3}\,{}^{k}c_3 = K_3 \qquad (6.6)$$

The term on the R.H.S represents the gravitational force and so does not depend upon velocities and accelerations. It can also be noticed that the term $\frac{{}^k\partial W_3}{{}^k\partial q_3}$ depends only upon the position of the link so it known. However the term $D$ depends upon the accelerations and velocities of the links.

The term ${}^kT_a^b$ is defined as ${}^kT_a^b = {}^kA_a \dots {}^kA_b$. The term ${}^kD_p$ can be defined from (3.36) as follows

- if $p = 3n$ for $n = 2, 3 \dots$

$$ {}^kD_p = \begin{pmatrix} {}^kA_{p+1}\,{}^kD_{p+1} & \text{if link p is in previous kinematic chain} \\ J_p\,{}^k\bar{W}_p^T + {}^kA_{p+1}\,{}^kD_{p+1} & \text{if link p is not in previous kinematic chain} \end{pmatrix} $$

- if $p = 3n + 1$ or $p = 3n + 2$ for $n = 2, 3 \dots$

$$ {}^kD_p = {}^kA_{p+1}\,{}^kD_{p+1} \tag{6.7} $$

Remember that only the root has six degrees of freedom and thus six rationalised links. The sixth of these links contains mass and since the root is included in all kinematic chains, we have :

$$ {}^kD_6 = \begin{pmatrix} J_6\,{}^k\bar{W}_6^T + {}^kA_7\,{}^kD_7 & \text{for } k = 1 \text{ i.e first kinematic chain} \\ {}^kA_7\,{}^kD_7 & \text{for } k \neq 1 \text{ i.e other kinematic chains} \end{pmatrix} $$

Thus

$$ {}^kD_3 = {}^kA_4\,{}^kD_4 \tag{6.8} $$

In order to evaluate the term ${}^kD_3$ we have to calculate term ${}^kD_4$ which has a general notation ${}^kD_{3i+1}$. The evaluation of this term must not contain the term $D$ in the RHS but only the three translational accelerations so it can be substituted back to (6.8) and eventually to eqn (6.6).

A prototype link (other than the root) is separated into three rationalised links $3i + 1$, $3i + 2$ and $3i + 3$. Consider the calculation of term D for these links i.e for $i = 2, 3, \dots$

- if link $3i + 3$ is in previous kinematic chain

$$ {}^kD_{3i+3} = {}^kA_{3i+4}\,{}^kD_{3i+4} $$

- if link $3i+3$ is not in previous kinematic chains

$$^kD_{3i+3} = J_{3i+3}\,^k\bar{W}^T_{3i+3} + {}^kA_{3i+4}\,^kD_{3i+4}$$

- in both cases the for the other two links

$$\begin{aligned} ^kD_{3i+1} &= {}^kA_{3i+2}\,^kD_{3i+2} \\ ^kD_{3i+2} &= {}^kA_{3i+3}\,^kD_{3i+3} \end{aligned}$$

This leads to the following definition for the term $^kD_{3i+1}$ :

- if $3i+1$ is in previous kinematic chain

$$^kD_{3i+1} = {}^kT^{3i+4}_{3i+2}\,^kD_{3i+4}$$

- if $3i+1$ is not in previous kinematic chain

$$^kD_{3i+1} = {}^kT^{3i+3}_{3i+2}\,J_{3i+3}\,^k\bar{W}^T_{3i+3} + {}^kT^{3i+4}_{3i+2}\,^kD_{3i+4} \tag{6.9}$$

Therefore from (6.9) and (6.8) $^kD_3$ can be evaluated to :

$$^kD_3 = \sum_{\forall b \in \Re_k} \left( ^kT^b_4\, J_b\, ^k\bar{W}^T_b \right) \tag{6.10}$$

where $\Re_k$ is a set that contains all links with mass in kinematic chain $k$.

The next task is to find the general form of $^k\bar{W}_i$ for $i = 3k$ for $k = 2, 3, \ldots, n$. $^k\bar{W}_i$ must be expressed in terms of $^k\bar{W}_3$. In order to do that, we define a new quantity $^kM_i$ by using the definition of $\bar{(W)}_i$ from (3.40):

$$^kM_i = 2\,^k\dot{W}_{i-1}\frac{^k\partial A_i}{^k\partial q_i}\,\dot{q}_i + {}^kW_{i-1}\frac{^k\partial^2 A_i}{^k\partial q_i^2}\,\dot{q}_i^2 + {}^kW_{i-1}\frac{^k\partial A_i}{^k\partial q_i}\,\ddot{q}_i \tag{6.11}$$

and therefore :

$$^k\bar{W}_i = {}^kM_i + \sum_{j=4}^{i-1}\left( ^kM_j\, ^kT^i_{j+1} \right) + {}^k\bar{W}_3 \tag{6.12}$$

Define :

$$^kF^i = {}^kM_i + \sum_{j=4}^{i-1}\left( ^kM_j\, ^kT^i_{j+1} \right) \quad and \quad ^k\Phi^i = {}^kT^i_4$$

Hence (6.10) becomes :

$$ {}^k\ddot{W}_3 = {}^kF^i + {}^k\ddot{W}_3 {}^k\Phi^i \tag{6.13}$$

From (6.10) and (6.13) $\Rightarrow$

$$ {}^kD_3 = \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^kF^b\right)^T \right) + \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^k\Phi^b\right)^T {}^k\ddot{W}_3^T \right) \tag{6.14}$$

Define :

$$\begin{aligned} {}^kP_3 &= 2\,{}^k\dot{W}_2 \frac{{}^k\partial A_3}{{}^k\partial q_3}\,\dot{q}_3 + {}^kW_2 \frac{{}^k\partial^2 A_3}{{}^k\partial q_3^2}\,\dot{q}_3^2 \\ {}^kP_2 &= 2\,{}^k\dot{W}_1 \frac{{}^k\partial A_2}{{}^k\partial q_2}\,\dot{q}_2 + {}^kW_1 \frac{{}^k\partial^2 A_2}{{}^k\partial q_2^2}\dot{q}_2^2 \\ {}^kP_1 &= \frac{{}^k\partial^2 A_1}{{}^k\partial q_1^2}\dot{q}_1^2 \end{aligned}$$

Therefore :

$$\begin{aligned} {}^k\ddot{W}_3 &= {}^k\ddot{W}_2 {}^kA_3 + {}^kW_2 \frac{{}^k\partial A_3}{{}^k\partial q_3}\ddot{q}_3 + {}^kP_3 \\ {}^k\ddot{W}_2 &= {}^k\ddot{W}_1 {}^kA_2 + {}^kW_1 \frac{{}^k\partial A_2}{{}^k\partial q_2}\ddot{q}_2 + {}^kP_2 \\ {}^k\ddot{W}_1 &= \frac{{}^k\partial A_1}{{}^k\partial q_1}\ddot{q}_1 + {}^kP_1 \end{aligned}$$

Hence :

$$ {}^k\ddot{W}_3 = S_1\ddot{q}_1 + S_2\ddot{q}_2 + S_3\ddot{q}_3 + S_4 \tag{6.15}$$

where

$$\begin{aligned} S_1 &= \frac{{}^k\partial A_1}{{}^k\partial q_1} {}^kA_2 {}^kA_3 \\ S_2 &= {}^k\ddot{W}_1 \frac{{}^k\partial A_2}{{}^k\partial q_2} {}^kA_3 \\ S_3 &= {}^k\ddot{W}_2 \frac{{}^k\partial A_3}{{}^k\partial q_3} \\ S_4 &= {}^kP_1 {}^kA_2 {}^kA_3 + {}^kP_2 {}^kA_2 + {}^kP_3 \end{aligned}$$

Let define :

$$ {}^kN = \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^kF^b\right)^T \right) $$

then by using this definition together with (6.15), (6.14) can be written as

$$ {}^kD_3 = {}^kN + \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^k\Phi^b\right)^T [S_1\ddot{q}_1 + S_2\ddot{q}_2 + S_3\ddot{q}_3 + S_4]^T \right) \Rightarrow $$

$$ \begin{aligned} {}^kD_3 = {}^kN &+ \ddot{q}_1 \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^k\Phi^b\right)^T S_1^T \right) + \ddot{q}_2 \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^k\Phi^b\right)^T S_2^T \right) \\ &+ \ddot{q}_3 \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^k\Phi^b\right)^T S_3^T \right) + \sum_{\forall b \in \Re_k} \left( {}^k\Phi^b J_b \left({}^k\Phi^b\right)^T S_4^T \right) \end{aligned} $$

We can use terms ${}^kG_1$ to ${}^kG_4$ to rewrite the above equation in a shorter notation :

$$ {}^kD_3 = \ddot{q}_1 {}^kG_1 + \ddot{q}_2 {}^kG_2 + \ddot{q}_3 {}^kG_3 + {}^kG_4 \tag{6.16} $$

Let

$$ {}^kG_{i+5} = \frac{{}^k\partial W_3}{{}^k\partial q_3} {}^kG_{i+1} \quad i = 0, 1, 2, 3 $$

Then by using (6.16), (6.6) can be re-written as :

$$ \begin{aligned} \left( \ddot{q}_1 \sum_{k=1}^{N} \sum_{i=1}^{4} {}^kG_5 + \ddot{q}_2 \sum_{k=1}^{N} \sum_{i=1}^{4} {}^kG_6 + \ddot{q}_3 \sum_{k=1}^{N} \sum_{i=1}^{4} {}^kG_7 \right)_{ii} &= \\ K_3 &- \left( \sum_{k=1}^{N} \sum_{i=1}^{4} {}^kG_8 \right)_{ii} \end{aligned} \tag{6.17} $$

Similarly the force for the second degree of freedom can be expressed as :

$$ \sum_{k=1}^{N} tr \left( \frac{{}^k\partial W_2}{{}^k\partial q_2} {}^kD_2 \right) = \sum_{k=1}^{N} g^T \frac{{}^k\partial W_2}{{}^k\partial q_2} {}^kc_2 = K_2 \tag{6.18} $$

Since ${}^kD_2 = {}^kA_3 {}^kD_3$, the coefficients of the second equation can be found from the coefficients of the first equation by multiplying them by ${}^kA_3$ :

$$ \begin{aligned} {}^kG_{i+8} &= {}^kA_3 {}^kG_{i+4} \quad i = 1, 2, 3, 4 \\ {}^kG_{i+12} &= \frac{{}^k\partial W_2}{{}^k\partial q_2} {}^kG_{i+8} \quad i = 1, 2, 3, 4 \end{aligned} $$

Which implies that :

$$\left(\ddot{q}_1 \sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{13} + \ddot{q}_2 \sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{14} + \ddot{q}_3 \sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{15}\right)_{ii} = K_2 - \left(\sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{16}\right)_{ii} \quad (6.19)$$

Finally for $F_1$, the first degree of freedom :

$$\sum_{k=1}^{N} tr\left(\frac{{}^k\partial W_1}{{}^k\partial q_1}{}^kD_1\right) = \sum_{k=1}^{N} g^T \frac{{}^k\partial W_1}{{}^k\partial q_1}{}^k c_1 = K_1 \quad (6.20)$$

Since ${}^kD_1 = {}^kA_2\,{}^kD_2$ the coefficients of the third equation can be found from the coefficients of the second by multiplying by ${}^kA_2$, i.e :

$$\begin{aligned} {}^kG_{i+16} &= {}^kA_2\,{}^kG_{i+8} \quad i = 1,2,3,4 \\ {}^kG_{i+20} &= \frac{{}^k\partial W_1}{{}^k\partial q_1}{}^kG_{i+16} \quad i = 1,2,3,4 \end{aligned}$$

This leads to :

$$\left(\ddot{q}_1 \sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{21} + \ddot{q}_2 \sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{22} + \ddot{q}_3 \sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{23}\right)_{ii} = K_1 - \left(\sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_{24}\right)_{ii} \quad (6.21)$$

From eqns (6.17), (6.19) and (6.21) the values of the linear accelerations can be found by solving the system of linear equations. However if the system has target linear accelerations ${}_t\ddot{q}_1$, ${}_t\ddot{q}_2$, ${}_t\ddot{q}_3$ then the equations can be modified so that the offset accelerations $\ddot{q}_1$, $\ddot{q}_2$, $\ddot{q}_3$ can be found i.e from eqn (6.17) we have :

$$\left(({}_t\ddot{q}_1 + \ddot{q}_1)\sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_5 + ({}_t\ddot{q}_2 + \ddot{q}_2)\sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_6 + ({}_t\ddot{q}_3 + \ddot{q}_3)\sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_7\right)_{ii} = K_3 - \left(\sum_{k=1}^{N}\sum_{i=1}^{4} {}^kG_8\right)_{ii} \quad (6.22)$$

Eqns (6.19) and (6.21) are treated similarly. It was noticed that the system of equations results in a diagonal matrix. The reason of this behaviour is not clear mostly because of the complexity of analyzing how the coefficient matrices are constructed. The diagonal matrix certainly simplifies the solution of the equations but it also means that, in order to set the force that applies to a translational degree of freedom, to a particular value, only the acceleration associated with this degree of freedom has to be changed.

The objective of the above derivation is to calculate the forces that are exerted from the ground to the body. These forces are important as they keep the body standing up. The issue that governs the way that the problem is solved, is that the body should not descend below the horizontal ground level at any time during dynamic analysis and this lead to the decision of finding an analytical solution to the problem. This requires a re-formulation from inverse to direct Lagrangian dynamics. The direct Lagrangian dynamics have to be solved only for the three translational degrees of freedom as the ground reaction forces apply to those degrees of freedom only. Therefore, the method can be described in three stages :

1. Perform dynamic analysis to find the forces for each degree of freedom in the body and subsequently, find the accelerations for the translational degrees of freedom
2. Use these translational accelerations to find the ground reaction forces by using the method above and subsequently the translational accelerations
3. Use the accelerations found for the translational degrees of freedom from the ground reaction forces, to calculate the forces for each degree of freedom of the body

Dynamic analysis has to be performed twice and the calculation of ground reaction forces is performed in-between. After the second dynamic analysis, the translational degrees of freedom have zero values (balanced translational forces). The method can be used in cases where the user wishes to assign particular values to the translational degrees of freedom, e.g control the friction of the body with the ground surface.

## 6.7 Implementation of first level motion control

The entire software was written on a Sun 3/60 Network of workstation using C, Sunwindows and SunCore under Berkeley Unix 4.2 BSD and 4.0.3 SunOS software. The user interfaces were designed using the customized Sunwindows offered by SunOS and run in the Suntool environment. For the graphics interface, the implementation of ACM Core for the Sun workstation was used, namely SunCore. The system contains three distinct software units, the dynamics component, the kinematics components and the configuration file component. Fig 6.1 shows the way the three components are connected together.

### 6.7.1 Implementation of kinematics

The kinematics interface is divided into three windows, a *control window*, an *information window* and a *graphics window* as shown in Fig 6.2. The control window contains all the functions that trigger some action about picking a profile, drawing, help facility etc. All functions are represented by buttons that can be picked with the left button of the mouse. There are three controls that are used to manipulate the handling of the three major function of the kinematics component, i.e picking a profile, blending and free-hand drawing. Upon selection, all of them appear a pop-up menu with the various actions in a hierarchical structure.

The user can use the menu (Fig 6.3) to choose a profile. Once the user has input the appropriate data for a profile, there is a control that initiates the calculations for the spline functions. If anything goes wrong during the calculations due to incorrect or unsuitable data, an appropriate message appears on the information window, otherwise information about the maxima and minima of the displacement, velocity and acceleration curves, is displayed. Three buttons are also used in the control window in order to initiate drawing of any of the three curves, i.e displacement, velocity and acceleration.

The blend button gives a pop-up menu of the curves that can be blended as shown in Fig 6.4.

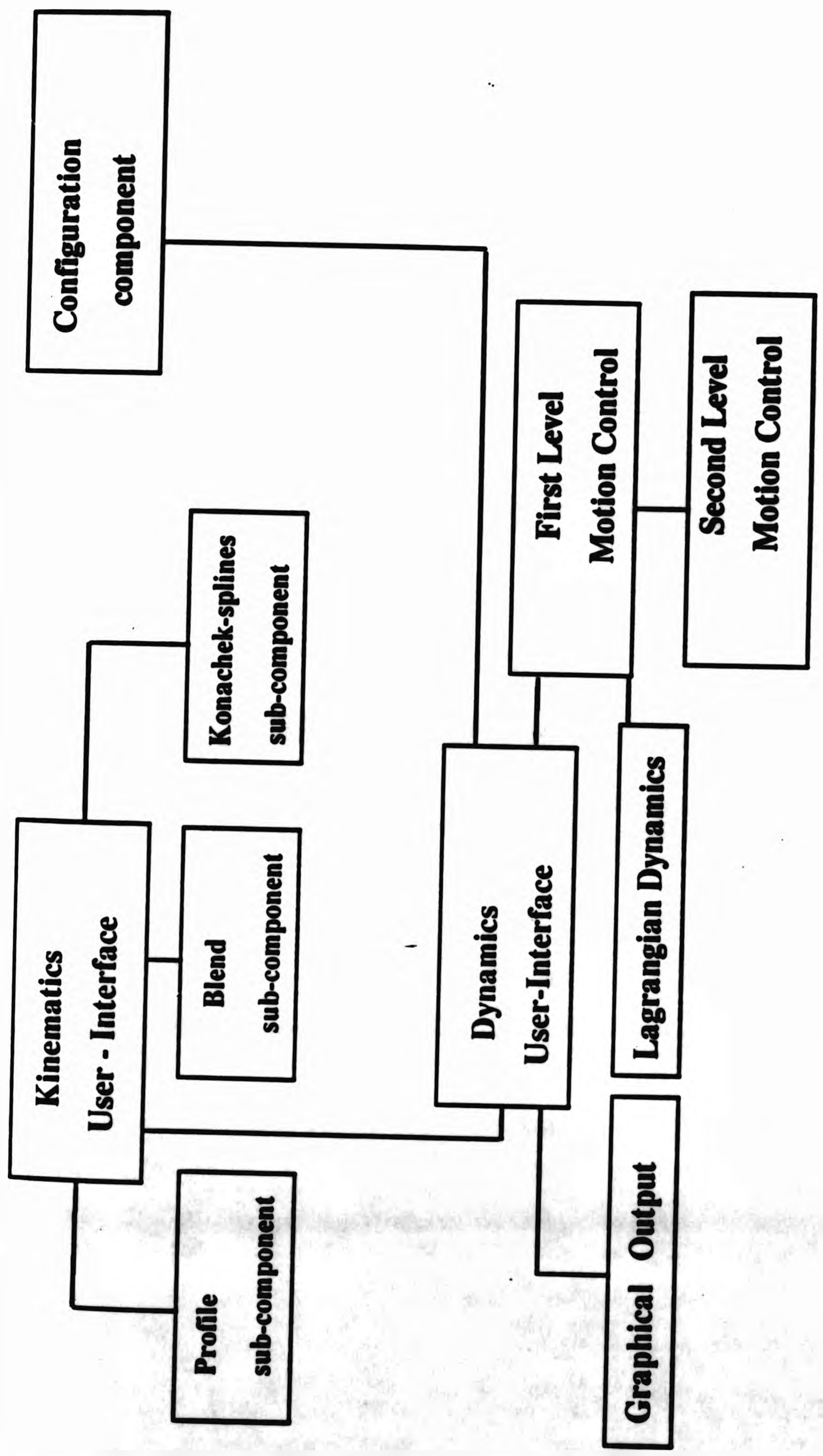


**Fig 6.1 : Schematic representation of the components of the system.**

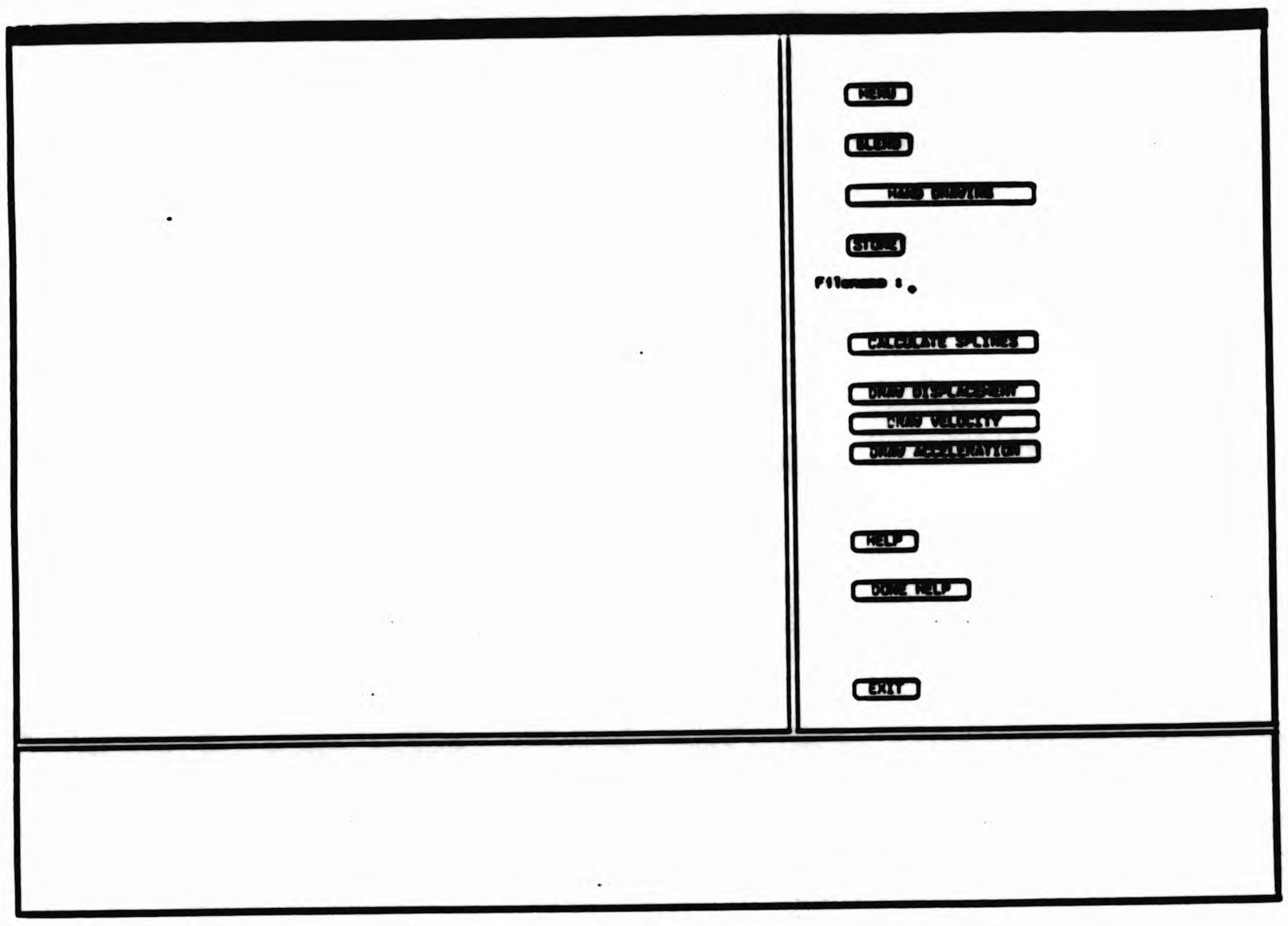


**Fig 6.2 : Kinematics Component User-Interface.**

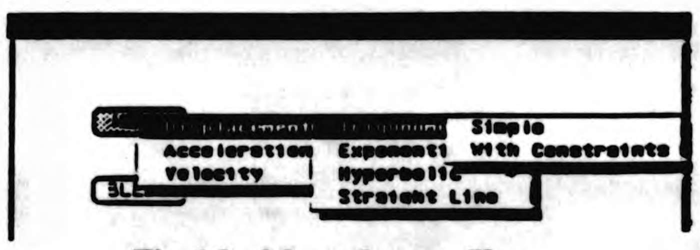


**Fig 6.3 : Menu for profiles**

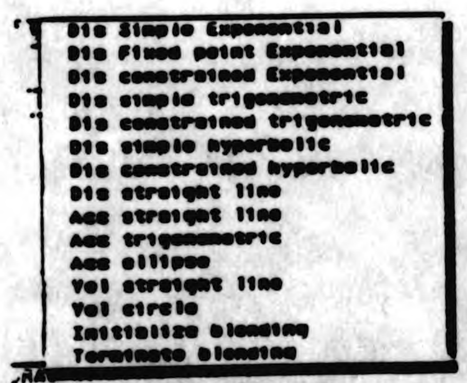


**Fig 6.4 : Blend menu.**

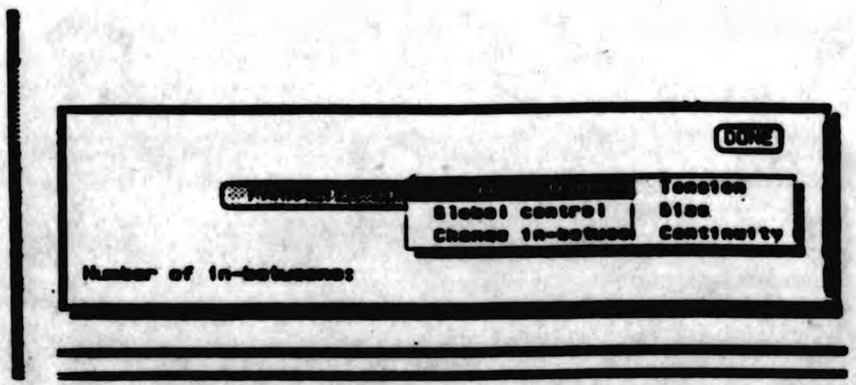


**Fig 6.5 : Hand - Drawing menu.**

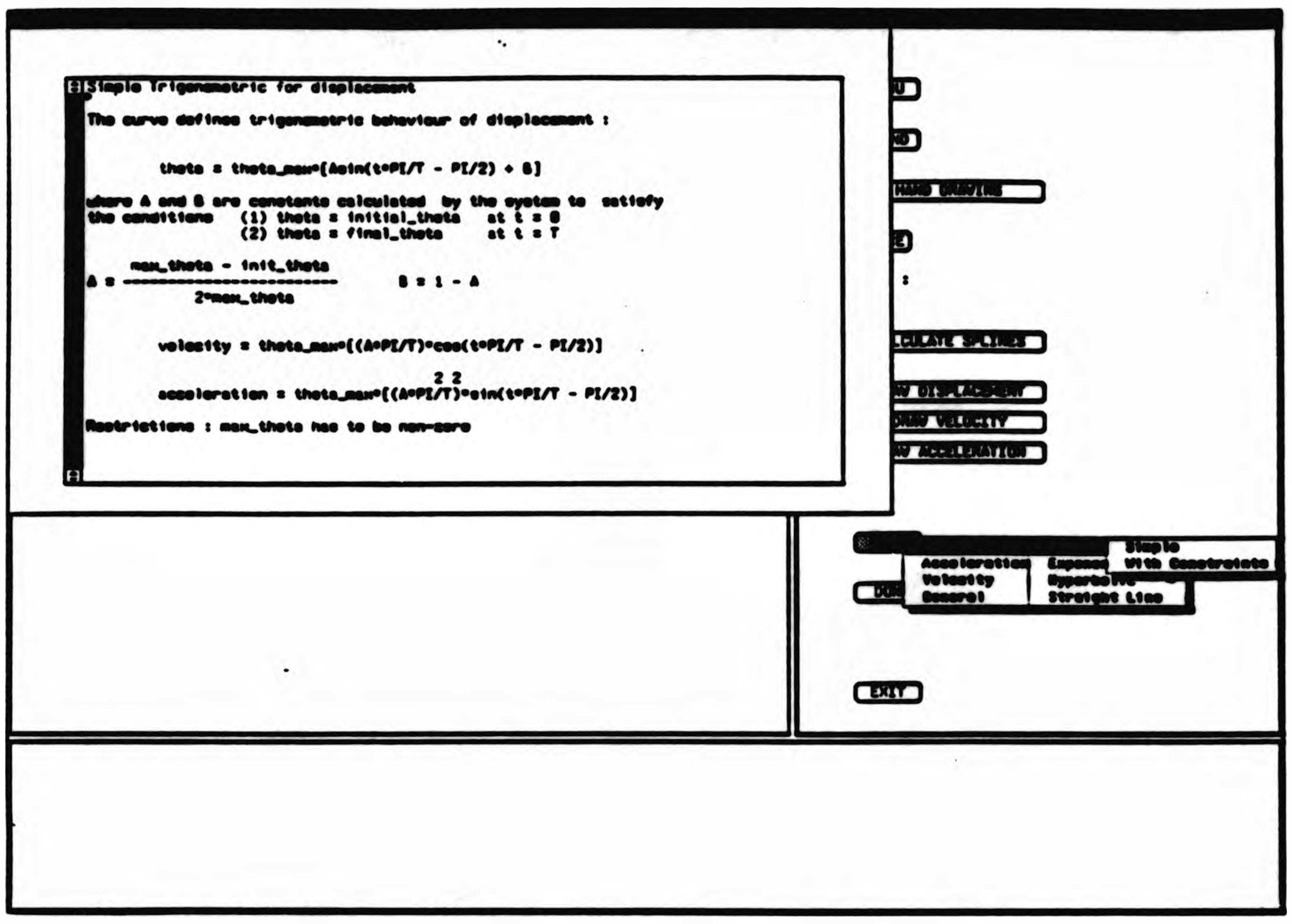
This means blending all three parameters at the same time. Once the curves have been picked, a pop-up window is used to choose the blending function action has to been repeated as many times as a new curve is added to the already blended curves.

The free hand drawing function allows the user to specify the range of the drawing canvas on the screen and to pick up points on this canvas either by using the left button of the mouse or by typing the points manually. A button on the control window is used to picture the resulting Kochanek spline on the canvas and a pop-up window is used to control the bias, tension and continuity parameters of the curve either locally or globally. This menu also controls the number of in-between points in the spline which is a way of making the curve smoother or rougher (Fig 6.5). In the case of local control, the user can choose the appropriate span of the curve by dragging the mouse on the canvas, and the corresponding span is highlighted in inverse video. When the right span has been highlighted, the user can pick it by pressing the right button of the mouse, and any subsequent changes in the parameters refer to this span of the curve. Finally, a help facility can be invoked by using the help button on the control window (Fig 6.6).

A file that contains the link names for which kinematic profiles will be constructed is passed onto the kinematics component. The link names are used by the kinematics program to construct a menu so that the user can pick them up. Two other arguments are passed onto the kinematics component from the command line, the start and final simulation times. The kinematics program allows the user to define different profiles for the same degree of freedom over different spans of time as long as these spans are between the start and finish simulation times. It is not necessary to describe profiles for every degree of freedom, but for those that the user wants to specify the profiles have to be defined over the entire specified time span. This information is then written to the kinematics file so it can be read by the dynamics program.

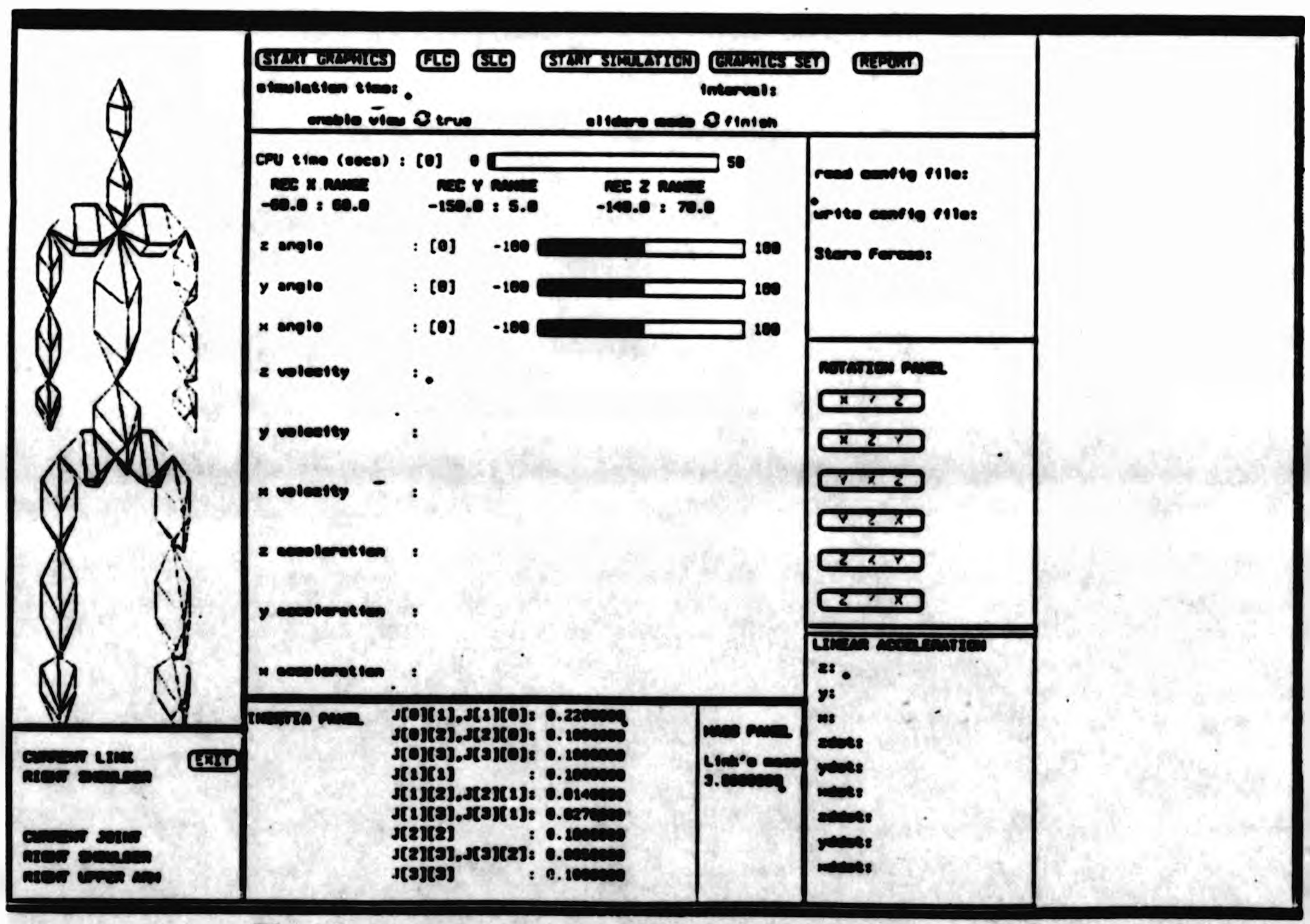
### 6.7.2 Implementation of dynamics

For ease of control the various dynamics control functions have been split into ten windows as shown in Fig 6.7.



```
Simple Trigonometric for displacement

The curve defines trigonometric behaviour of displacement :

        theta = theta_max*[Asin(t*PI/T - PI/2) + B]

where A and B are constants calculated  by the system to  satisfy
the conditions   (1) theta = initial_theta   at t = 0
                 (2) theta = final_theta     at t = T

     max_theta - init_theta
A = -----------------------        B = 1 - A
          2*max_theta

        velocity = theta_max*[(A*PI/T)*cos(t*PI/T - PI/2)]

                                          2 2
        acceleration = theta_max*[(A*PI/T)*sin(t*PI/T - PI/2)]

Restrictions : max_theta has to be non-zero
```

Fig 6.6 : Help facility.



```
simulation time:                        interval:
      enable view ○ true          sliders mode ○ finish

CPU time (secs) : [0]   0               50
 REC X RANGE      REC Y RANGE     REC Z RANGE
 -60.0 : 60.0     -150.0 : 5.0    -140.0 : 70.0

z angle         : [0]   -100            100
y angle         : [0]   -100            100
x angle         : [0]   -100            100
z velocity      :
y velocity      :
x velocity      :
z acceleration  :
y acceleration  :
x acceleration  :

read config file:
write config file:
Store Forces:

ROTATION PANEL

LINEAR ACCELERATION
z:
y:
x:
zdot:
ydot:
xdot:
zddot:
yddot:
xddot:

CURRENT LINK
RIGHT SHOULDER

CURRENT JOINT
RIGHT SHOULDER
RIGHT UPPER ARM

INERTIA PANEL   J[0][1],J[1][0]: 0.2200000
                J[0][2],J[2][0]: 0.1000000
                J[0][3],J[3][0]: 0.1000000
                J[1][1]        : 0.1000000
                J[1][2],J[2][1]: 0.0140000
                J[1][3],J[3][1]: 0.0270000
                J[2][2]        : 0.1000000
                J[2][3],J[3][2]: 0.0000000
                J[3][3]        : 0.1000000

MASS PANEL
Link's mass
3.0000000
```

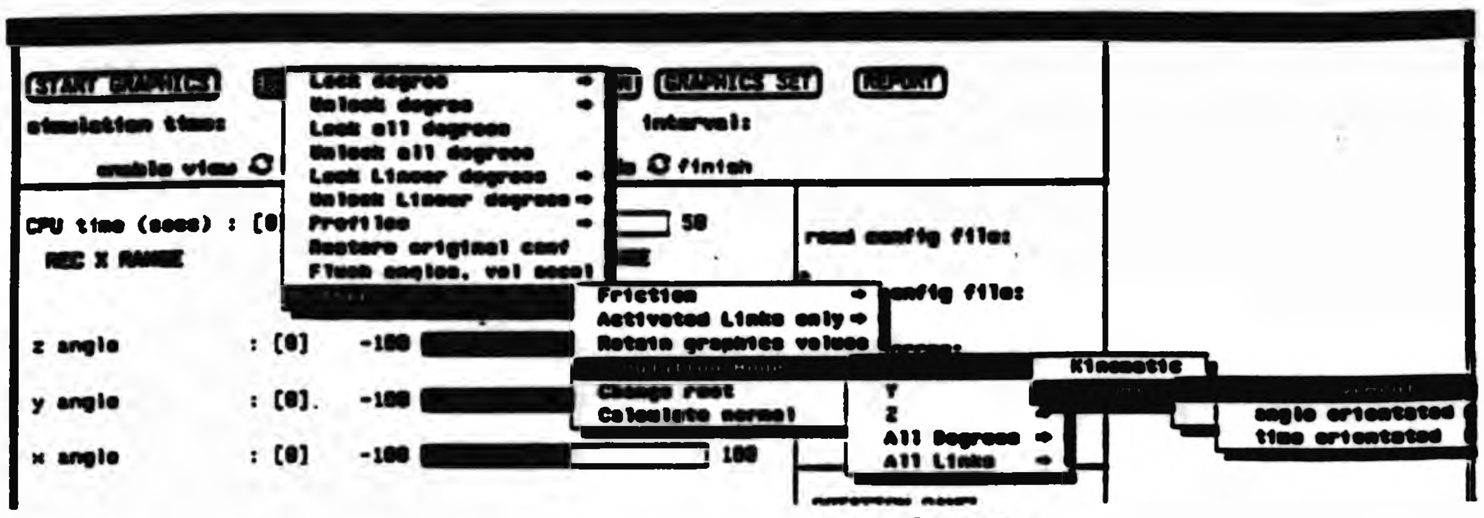
Fig 6.7 : Dynamics Component User-Interface.

The *control window* contains the main functions of the dynamics, such as setting the simulation interval, the sample interval, initiating the dynamics and setting the various profiles for the degrees of freedom. Graphics settings can also be changed using the graphics setting button on the control window. The graphics can either be sent to the screen as the dynamics are calculated or to a file in order to be viewed later. First level motion control functions have been incorporated in a menu ( Fig 6.8) that contains :

- Locking a degree of freedom or link
- Unlocking a degree of freedom or link
- Choosing a profile for a degree of freedom
- Viewing angle and acceleration profiles using smooth cubic spline interpolation
- Switching the calculation of ground reaction on and off

The *figure representation window* is used to help the user to pick up a particular link or joint of the body by using the mouse. Once a link or joint has been picked it is shown on the *link-joint indicator window*. The *input window* is used to input starting and finishing values for displacement, velocity and acceleration for a degree of freedom. The user can adjust whether the condition is initial or final by using a control button on the control window. The *configuration window* is used to read or write a configuration file and is also used to write the accelerations found by the dynamics to a file. The *inertia window* together with the *mass window* is used to show the inertia tensors and masses of links. Both windows can be accessed and their values can be changed on-line. The *linear motion window* is used to input the values for displacement, velocity and acceleration for the three linear degrees of freedom. The *rotation window* is used to pick up one of the six orders of rotation for a link. Finally the *acceleration window* shows the values obtained for accelerations by running the dynamics.

The graphics are displayed in a graphical window which is overlaid by the interface and appears only when the graphics are initiated. A typical output on the graphics window is shown in Fig 6.9.



CPU time (secs) : [0]

z angle : [0] -100

y angle : [0]. -100

x angle : [0] -100

Fig 6.8 : First Level Motion Control menu.

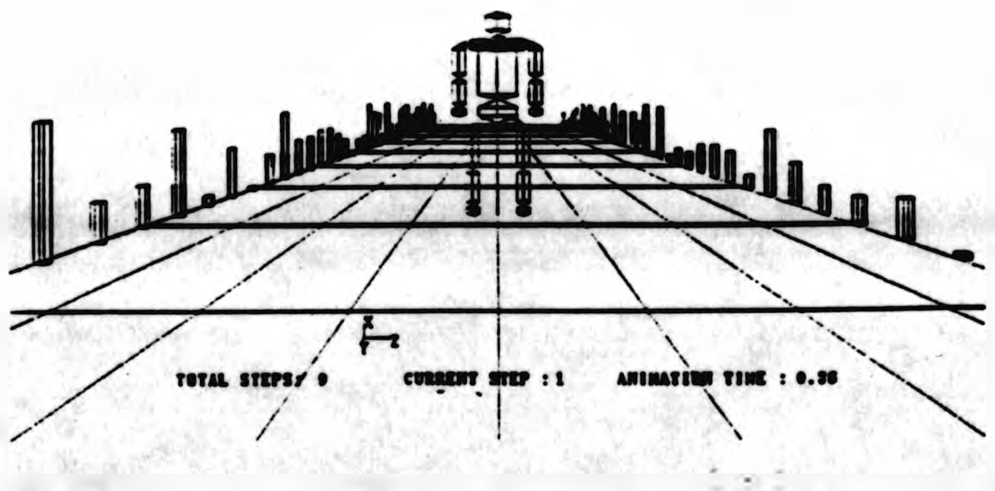


Fig 6.9 : Graphical output.

### 6.7.3 Configuration file component

The construction of the configuration file is not a straightforward matter. For this reason an individual software component was designed to help the user to specify configuration files. The use of this component can be separated into two distinct tasks :

- specification of the hierarchical structure of the body and
- specification of the dynamic quantities associated with this hierarchy.

The user can specify the hierarchy by drawing it on a digital canvas on the screen and by using a suitable menu as shown in Fig 6.10. Once the hierarchy has been specified, the user can proceed to decompose the prototype chain into rationalised kinematic chains by using the menu in Fig 6.11. If the prototype chain is ill-defined, this will be picked up by the system which gives an informative message for correcting the hierarchy.

Upon successful decomposition of the prototype chain, the user can start specifying the dynamics quantities such as order of rotations, inertia tensors etc. Some of the information that appears in the configuration file, does not have to be specified here because it can be deduced by the rest of the information. The dynamics information is specified by using five pop-up windows as shown in Fig 6.12.

The system can also read configuration files and display all the information to the user who can change the values of any dynamic variable but not the hierarchy itself.

The advantage of using this software component instead of editing the configuration file is twofold :

- It simplifies the task of specification as the system guides the user in a comprehensive way and
- Ill-defined hierarchies are rejected by the system and a file produced by this system will never fail to be read by the dynamics component.

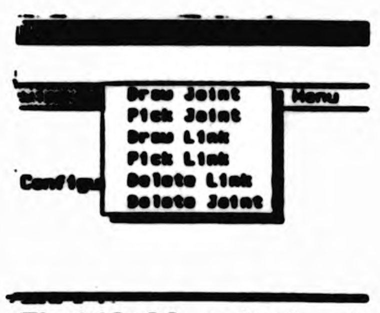


Fig 6.10 : Menu for drawing hierarchical structures.

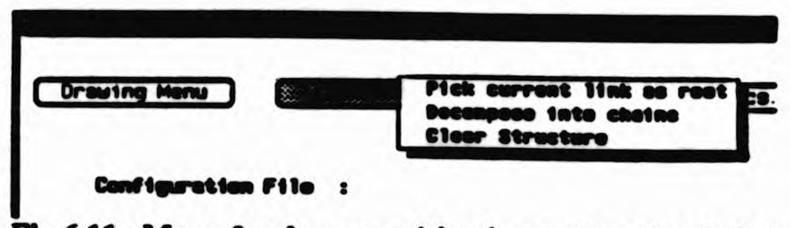
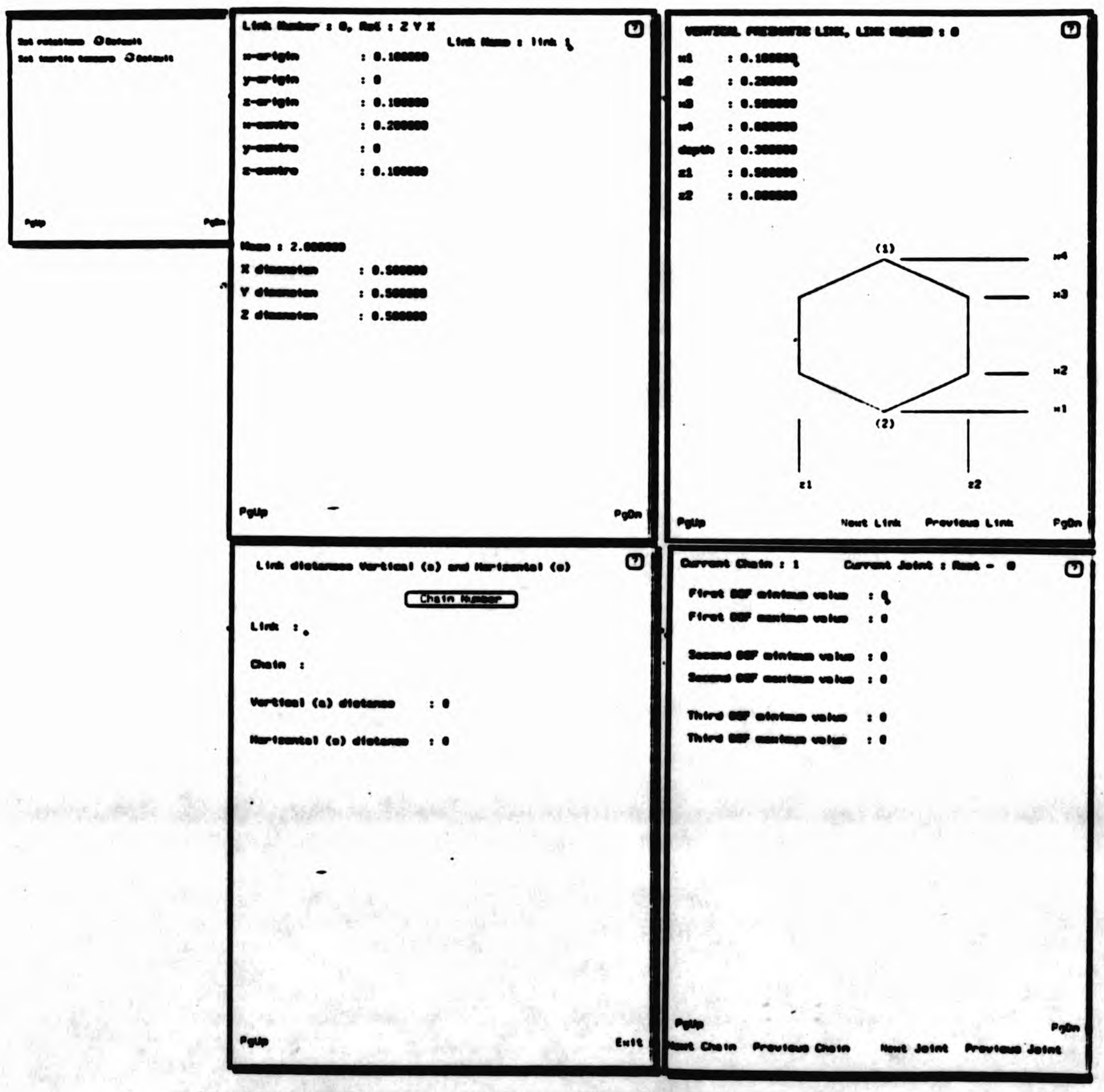


Fig 6.11 : Menu for decomposition into rationalised chains.



```
Set rotations  ○Default
Set inertia tensors  ○Default

PgUp                PgDn
```

```
Link Number : 0, Rot : Z Y X
                         Link Name : link 1
x-origin      : 0.100000
y-origin      : 0
z-origin      : 0.100000
x-centre      : 0.200000
y-centre      : 0
z-centre      : 0.100000

Mass : 2.000000
X dimension   : 0.500000
Y dimension   : 0.500000
Z dimension   : 0.500000

PgUp                                PgDn
```

```
VERTICAL PRISMATIC LINK, LINK NUMBER : 0
x1     : 0.100000
x2     : 0.200000
x3     : 0.500000
x4     : 0.600000
depth  : 0.300000
z1     : 0.500000
z2     : 0.600000

PgUp          Next Link    Previous Link     PgDn
```

```
Link distances Vertical (a) and Horizontal (e)
                  Chain Number
Link :

Chain :

Vertical (a) distance      : 0

Horizontal (e) distance    : 0

PgUp                                Exit
```

```
Current Chain : 1        Current Joint : Root -  0
  First DOF minimum value    : 0
  First DOF maximum value    : 0

  Second DOF minimum value   : 0
  Second DOF maximum value   : 0

  Third DOF minimum value    : 0
  Third DOF maximum value    : 0

PgUp                                          PgDn
Next Chain   Previous Chain   Next Joint   Previous Joint
```

Fig 6.12 : The five pop-up windows for inputting dynamical information into the configuration component.

## 6.8 Conclusions

This chapter described the lowest level for dynamic control. This level contains the functions necessary for controlling the motion of individual degrees of freedom.

Once the dynamic analysis has been performed, each degree of freedom can be assigned a torque-time profile which describes the behaviour of the degree of freedom during the simulation interval. A degree of freedom can be in simple or intensive dynamic mode.

When a link is in simple dynamic mode, three such profiles are offered which can be applied for achieving different results in the style of motion. The free-style profile is the simplest and leaves the degree of freedom to move under the action of the forces. It has no control upon the direction of motion or its duration. The angle-profile has unknown duration as it stops only when it gets to its target angle. The time-profile is the only one that guarantees that the motion will last exactly as long as the the simulation time.

In intensive dynamic mode, dynamic analysis is performed at each frame and the final target angles that are achieved are supplied to the system as the new initial angles. Even if only some of the degrees of freedom are in intensive mode, dynamic analysis has to be performed on the entire body.

It was also illustrated that for the six different types of rotation, there are certain patterns for the construction of angles $\alpha$ and $\theta$. This simplifies enormously the specification of local coordinate systems.

Bodies with different hierarchies can be simulated by the system with the help of a configuration file that contains the necessary information for specifying the body.

A direct method for the calculation of ground reaction forces based on a reformulation of Lagrangian dynamics was also presented. The reformulation of the method resulted in a direct dynamics solution for the translational degrees of freedom only. Since the primary aim of this work is maximum realism in the motion, the inclusion of ground reaction forces is of paramount performance. It is the first time that such a direct approach to the problem has been taken, and it proved successful. The results from the implementation of the method showed that the method is stable and correct to five or six decimal places which is enough for our purposes. The inclusion of the ground reaction forces is essential to the study and implementation of bipedal gaits which are presented in the next chapter.

Finally, this chapter described the implementation of the three components of the system and their interfacing. Their implementation is centred about a single design decision which is to make the system totally interactive, error-free and expandable as much as possible.

The motion functions presented here form the foundation for building the second layer of motion control which deals with the analysis and implementation of human locomotion.

# Chapter 7

# Second Level Dynamic Control : Walking

## 7.1 Introduction

The commonest types of motion of humans and articulated bodies in general, are those of walking and running. These types of motion must be incorporated into the animation system in order to be able to offer a greater variety of motion styles. As they are performed very often, the objective is to analyze the mechanics of locomotion and identify the parameters of the problem. It is also important to minimise the number of parameters that the user has to specify. In order to accomplish this objective, the interrelationships between the various parameters of the problem have to be established. This has to be done, though, in such a way that the resulting motion is general and that the fine details of locomotion are preserved.

Work in locomotion has been done by Boulic [90] and Brudelin [89] in animation, and McMahon [84] in biomechanics, while Vubobratovic [90] offers a large selection of research papers in the locomotion of anthropomorphic robots. Additional work in robotics for the purpose of constructing biped robots has been done by Song [87] and Zheng [88]. Early work in animation was performed by Girard [87], who produced some good sequences of human locomotion and Zeltzer [82] who also produced moving human skeletons using the synergic control approach, discussed in Chapter 2.

In robotics, the problem of locomotion is centred around the issue of balance and efficiency. Issues that are important in computer animation, such as style and realism, are not addressed. On the other hand, most researchers in animation approach the human locomotion by concentrating mostly on the motion of the legs, ignoring the problem of co-ordination of the rest of the body, such as the torso and the arms, with the legs. One solution is to use dynamics to control the motion of the legs and use prescribed kinematics to define the movement of the rest of the body. The approach that is to be followed in this work is to control the entire motion dynamically without involving any kinematic specification.

The process of locomotion can be considered to consist of simple motion control tasks as described in the first layer of motion control. Therefore, locomotion constitutes the second layer of the motion control system. The other strand of functionality of this layer is to decide upon motion specification of articulated body locomotion. This is closely related to the way that parameters are defined and implemented internally in the animation system. For instance, one way of specifying a walking sequence is to provide some points on the path that the body has to follow and to use inverse kinematics to find the target angles of the links. On the other hand, the motion can be specified as a sequence of simple commands such as 'perform 10 steps', or 'walk 30 meters'. In both cases, some parameters of the locomotion have to be specified.

Another issue that is important for locomotion is the use of interactive versus scripted systems. The way the software system has been designed is totally interactive and so far, for simple first level motion control functions, this method has proved adequate. For the more complicated commands for generating bipedal locomotion, it is not certain that an interactive system will be efficient and easy to use, and so this question has also to be answered.

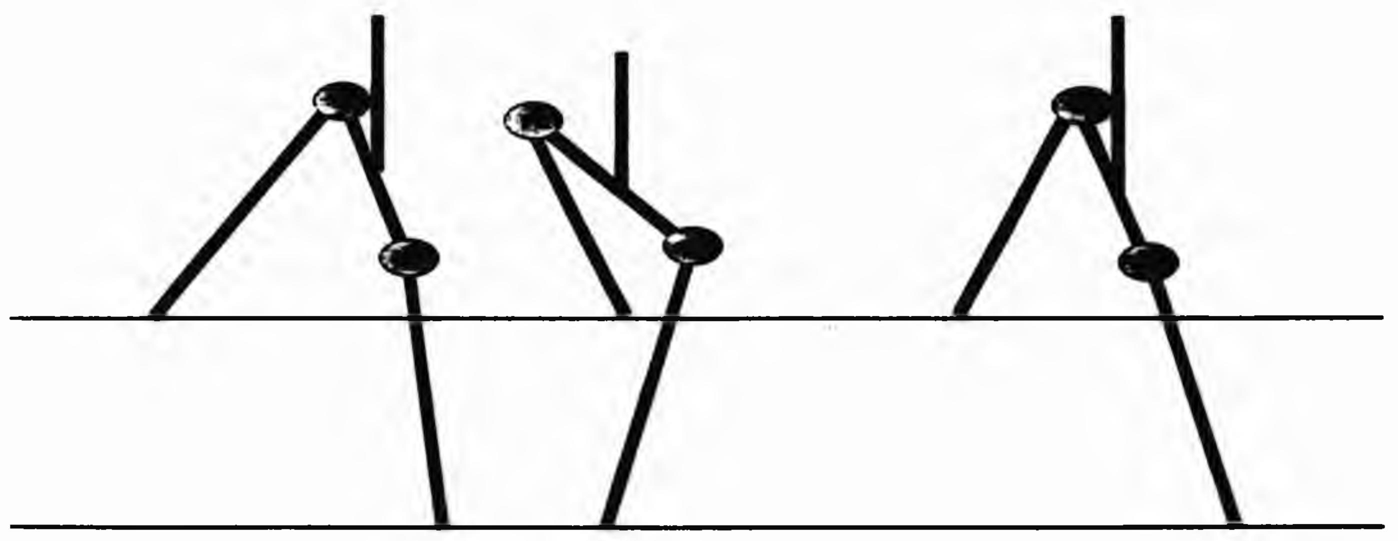
Bipeds can move by using three different gaits : hopping, walking and running. In hopping the two lower limbs of the body are in phase, whereas in walking and running they move in alternate phases. In walking, a lot of time is spent with both lower limbs on contact with the ground, whereas in running, most of the time both limbs are off the ground. The gait determinants describe the movements of the individual limbs that are performed in order to make the body walk. Humans do not usually use hopping as the means for locomotion, and therefore walking and running are the two most important gaits.

When both legs are in contact with the ground, the body is in ***double support state*** and when one leg is off the ground, the body is in ***single support state***. In single support state, the leg off the ground is called the ***swing leg*** and the leg in contact with the ground is called the ***stance leg***.

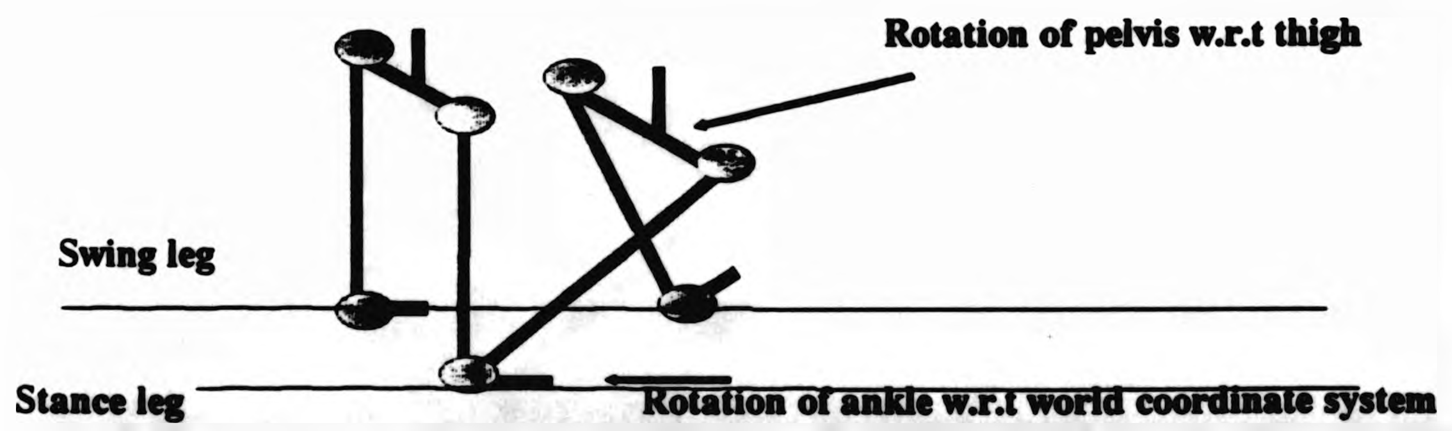
## 7.2 Gaits determinants for bipedal walking

Although there is not a unique way of describing the gait determinants for walking, McMahon [84] defines them as a sequence of movements that depend upon a single degree of freedom in one of the joints. This approach particularly suits dynamic analysis since it deals with individual degrees of freedom and the target angles can be found by establishing the relationships between these determinants. However in the process of dynamic analysis, for reasons that will be explained later, a gait determinant may involve more that a degree of freedom. McMahon uses six gait determinants for walking, but only three of them are used here since the aim of the animation system is to produce a fairly generalized motion while keeping the number of parameters minimal. The six gait determinants are described briefly here :

- Compass gait : This determinant refers to the motion of the pelvis through a series of horizontal arcs. The radius of the motion is determined by the length of the legs ( Fig 7.1).
- Pelvic tilt : Pelvic tilt involves the lowering of the pelvis from the side of the swing leg as shown in Fig 7.2. This necessitates the introduction of knee flexion of the swing leg, as otherwise the swing leg would hit the ground as it starts moving.
- Stance leg knee flexion : This is used by the body to flatten the arcs of the compass gait (Fig 7.3).
- Pelvic rotation : This allows the pelvis to rotate around a vertical axis. During walking the angle of rotation is around $\pm3$ degrees, but it increases for running and trotting.



**(a) Compass gait determinant.**



Rotation of pelvis w.r.t thigh
Swing leg
Stance leg
Rotation of ankle w.r.t world coordinate system

**(b) Compass gait rotations**

**Fig 7.1 : Compass gait determinant and degrees of freedom involved in the motion.**

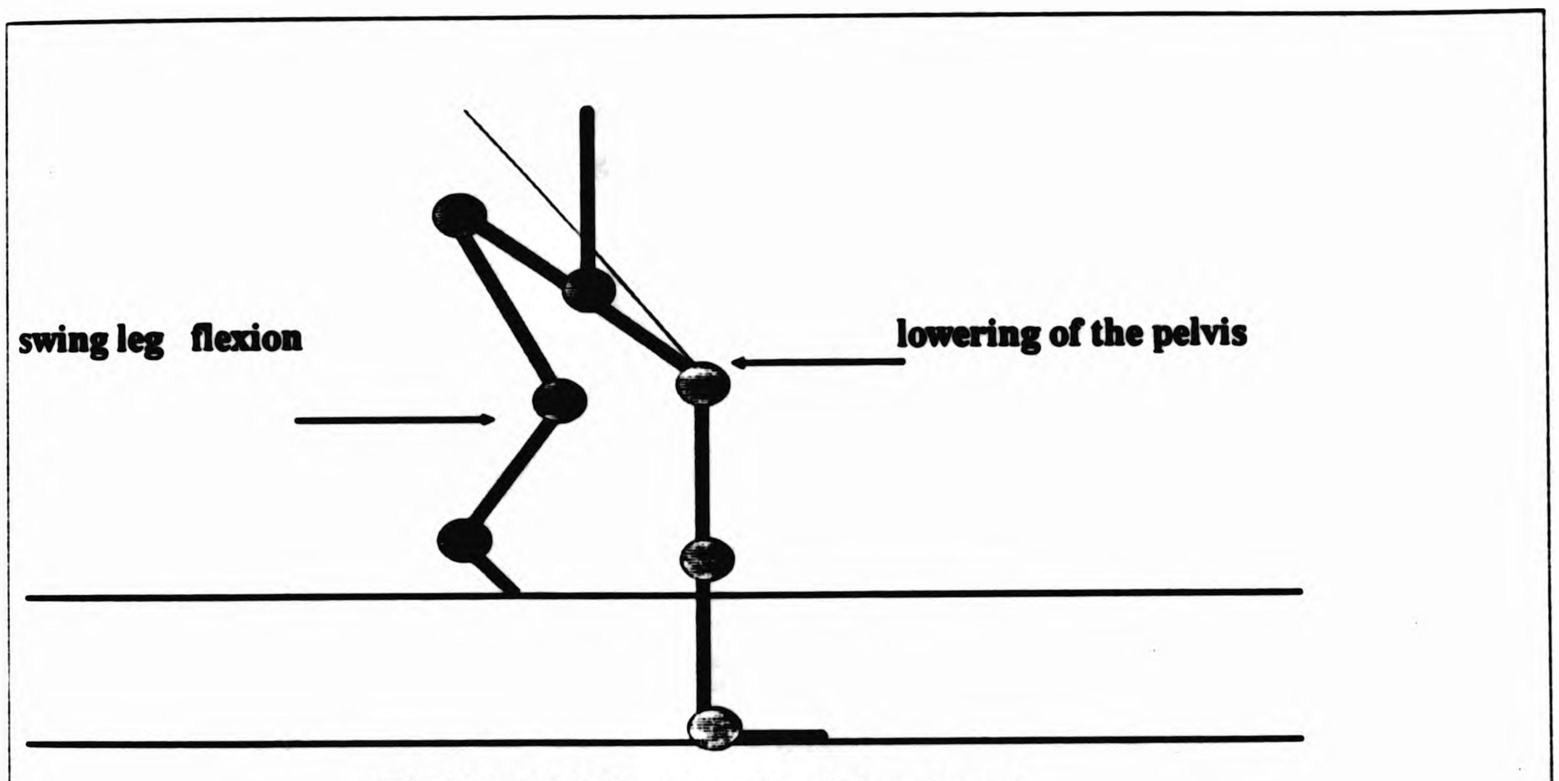


**Fig 7.2 : Pelvic tilt gait determinant.**

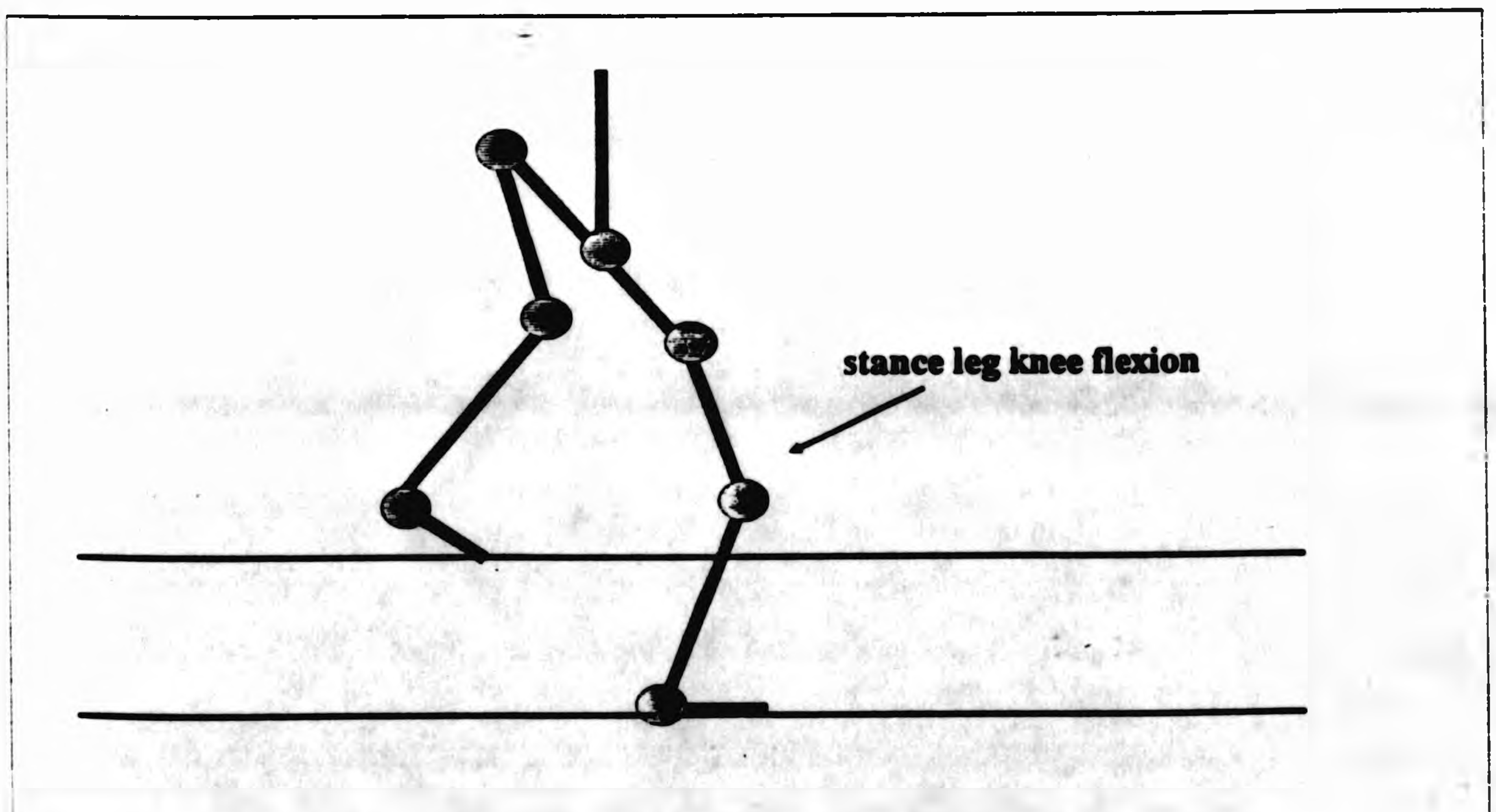


**Fig 7.3 : Stance leg knee flexion gait determinant.**

- Plantar flexion of the swing ankle : This determinant makes the plantar surface of the swing foot(sole) flexes just before the foot lifts off. This is done in order to smooth the transition from the double support state to the swing phase.
- Lateral displacement of the pelvis : This determinant is a lateral movement of the body where the body bearing is transferred from one leg to the other.

From these six gait determinants only the *compass gait*, *pelvic tilt* and *stance leg flexion* are used by the system. The idea is to start with a quite general model that is not very expensive computationally and build up on it by adding more gait determinants. In this way the importance of each gait determinant can be assessed and computational adjustments from one model to another will be easier.

## 7.3 Adjusting the gait determinants for the purpose of the model

In the proposed model by McMahon each gait determinant involves the movements of a single degree of freedom. The degree of freedom that moves during each gait determinant depends directly upon the decision of which link is the root of the motion. In the subsequent discussion the term limb means a single link of the body such as, thigh, foot etc.

*i. Compass Gait* : The degree of freedom involved in compass gait is that of the trunk with respect to the world coordinate system as shown in Fig 7.1(a). This is true if the trunk is considered as the root of the body. However, because of the way rotations are performed in the dynamic model - additive effect of rotations from the root to the end effectors - it was proved that the only way to perform realistic rotations is to use the feet of one of the legs as the root of the model. This leg has to be the stance leg since it remains in contact with the ground during the dynamic analysis. This approach is used in the other gait determinants as well. As a result of this change two degrees of freedom are involved as shown in Fig 7.1(b) :

- A rotation of the root (foot of stance leg) w.r.t the world coordinate system which makes the body move towards the direction of motion

- A rotation of the pelvis w.r.t the upper limb (thigh) of the stance leg which extends the swing leg in the direction of motion.

*ii. Pelvic Tilt* : Again two degrees of freedom are involved here as shown in Fig 7.2 :

- A rotation of the pelvis w.r.t the upper limb of the stance leg. The rotation is performed in the vertical direction and the effect it produces is to lower that part of pelvis attached to the swing leg.
- A rotation of the upper limb of the swing leg w.r.t its lower limb. This is due to the vertical rotation of the pelvis and is necessary as the vertical height of the leg is now shorter.

*iii. Stance leg knee flexion* : It involves only one degree of freedom i.e a rotation of the lower limb of the stance leg w.r.t the thigh of the stance leg (Fig 7.3).

## 7.4 Analysis of bipedal walking

In order to analyze walking, three terms have to be defined. A cycle is the set of movements that the body performs in order to cover a stride, as shown in Fig 7.4. A cycle will always relate to the swing leg, and since walking occurs in alternate states, a typical sequence of events is : left leg cycle, then right leg cycle and so forth (Fig 7.4). It is obvious that after two successive cycles the body will start repeating the same sequence of movements. Two successive cycles form a *walking period*. Each cycle is divided into phases. A *phase* represents a portion of time within a cycle. The number of phases is decided from the number of times that new target angles have to be achieved by some or all limbs involved in the motion. In other words, there are as many phases as the number of times that dynamic analysis is performed within a cycle. When using the three gait determinants model, two phases are sufficient for each cycle. Fig 7.5 shows how walking progresses if the body starts moving with the left leg.

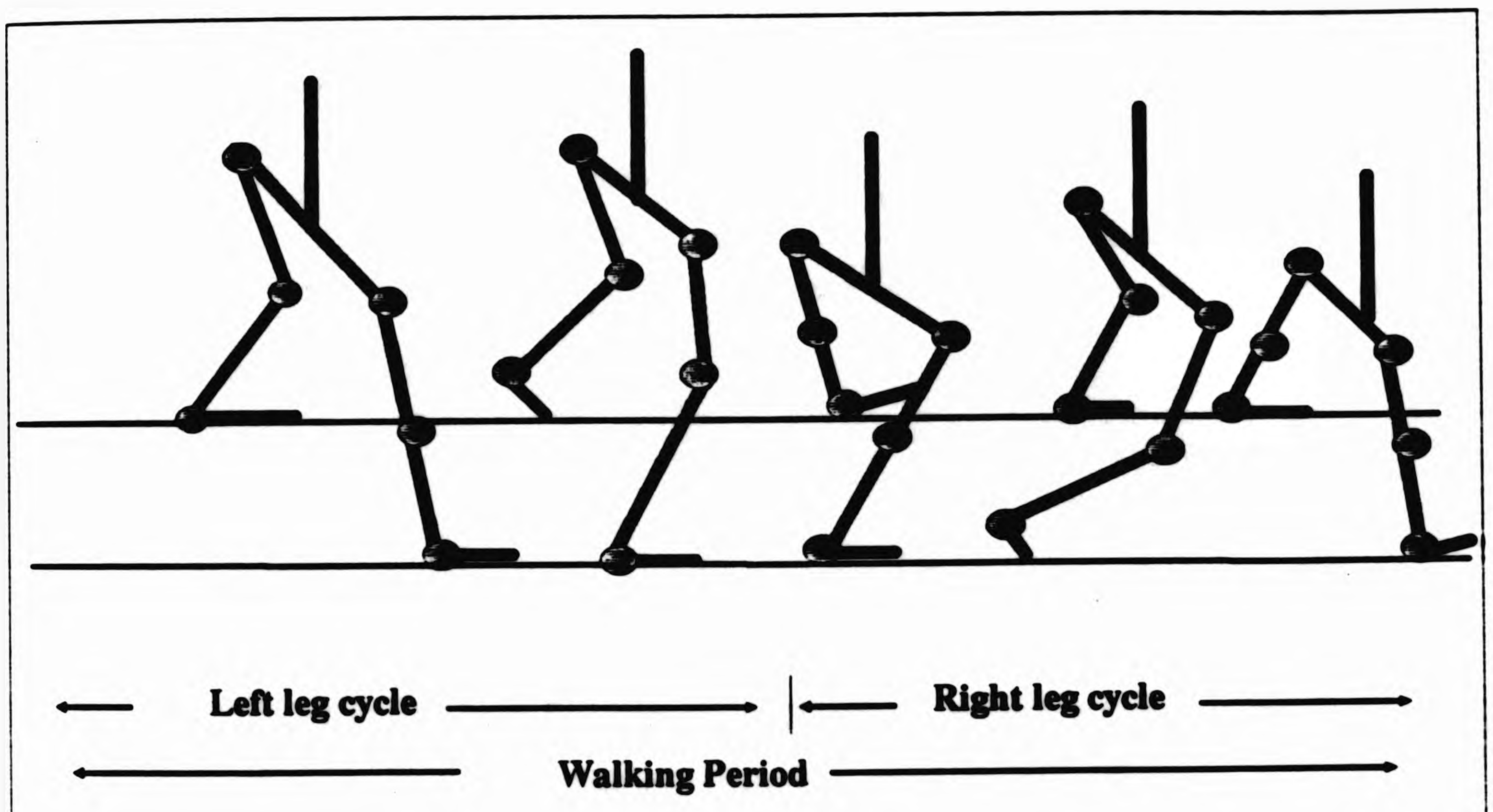


**Fig 7.4 : Progression of walking in cycles and periods.**

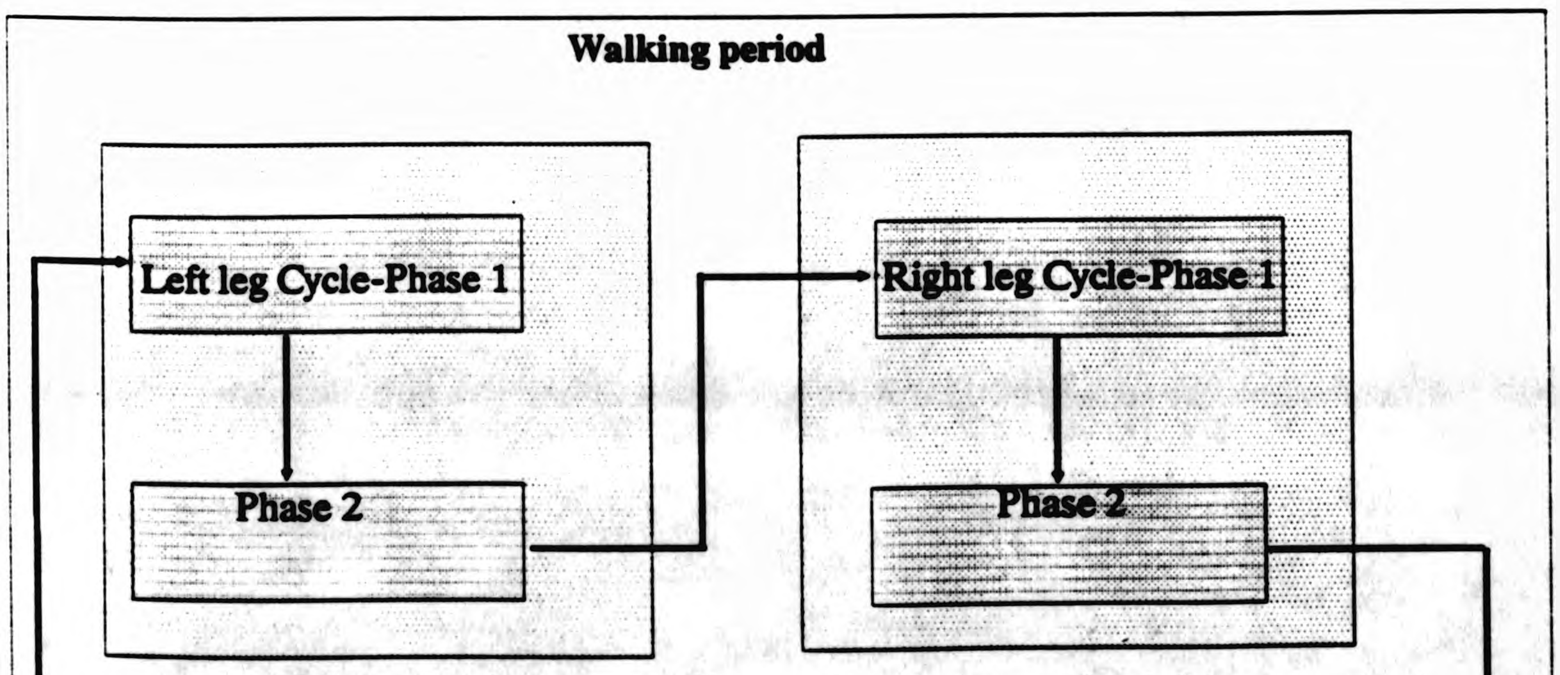


**Fig 7.5 : Walking periods, cycles and phases**

The two phases within a cycle can be described in terms of the gait determinants as follows :

- Phase 1 : brings the swing leg to liftoff position
    1. Positive pelvic tilt
    2. Stance leg flexion
- Phase 2 : moves the body in the direction of motion.
    1. Negative pelvic tilt
    2. Compass gait

Positive pelvic tilt occurs when the swing leg lowers at the pelvis, whereas negative pelvic tilt refers to the opposite movement i.e elevation of the swing leg at the pelvis.

The body is considered to start and finish its motion at rest, in which state both legs are in the same vertical plane. Therefore for simplification reasons the cycles are classified into three categories : *starting cycle*, *intermediate cycle* and *finishing cycle*, as shown in Fig 7.6. All these cycles have two phases and the next task is to find the target angles for these phases in each cycle.

## 7.4.1 Starting cycle

*Phase 1* : Fig 7.7 shows how the body stands at the end of phase 1. Two assumptions are made to keep the kinematic calculations simple :

- For the stance leg, the ankle joint and the hip joint are in the same vertical line.
- For the swing leg, the ankle joint remains at a right angle throughout this phase.

There are five angles involved in this phase. $\delta$ and $\chi$ represent the pelvic tilt and the stance leg flexion respectively, while $\theta_1$, $\theta_2$ and $\theta_3$ (Fig 7.7) refer to the liftoff angles of the swing leg and occur as a result of the two gait determinants. If $L_1$, $L_2$ and $L_3$

are the lengths of the thigh, lower leg and foot respectively and $L$ is the width of the pelvis, then some relationships between the angles can be established.

The original height of the stance leg was $L_1 + L_2$ which, after the knee flexion, becomes M, expressed as :

$$M = L_1 \cos\xi + L_2 \cos\chi$$

where $L_1 \sin\xi = L_2 sin\chi$. If the height of the swing leg is $D$, allowing for the descent owing to pelvic tilt :

$$D = M - L\sin\delta = \left(L_1^2 - L_2^2 \sin^2\chi\right)^{\frac{1}{2}} + L_2\cos\chi - L\sin\delta \qquad (7.1)$$

From the second assumption above we have $\theta_2 = \theta_3$ and hence :

$$A_1 + A_2 + A_3 = D \qquad (7.2)$$

where $A_1 = L_3 \sin\theta_3$, $A_2 = L_2 \cos\theta_2$ and $A_3 = L_3 \cos\theta_1$ (see Fig 7.7). Hence :

$$\cos\theta_1 = \frac{-L_2\cos\theta_3 - L_3\sin\theta_3 + D}{L_1} \qquad (7.3)$$

where $D$ is given by eqn (7.1). Therefore, if the swing leg liftoff angle $\theta_3$, the pelvic tilt angle $\delta$ and the stance leg flexion angle $\chi$ are given, $\theta_1$ can be calculated.

*Phase 2* : In Fig 7.8 the state of the body after phase 2 is shown. During this phase a negative pelvic tilt rotation $\delta$ takes place and the legs stretch such that both knees are stiff. $\beta$ is the angle that the stance leg makes with the horizontal at the end of phase 2, whereas $\alpha$ is the angle that the swing leg makes with the vertical plane. The vertical heights of the two legs are equal and this leads to the following relationship :

$$\cos\alpha = \sin\beta \qquad (7.4)$$

Table 7.1 shows the rotations of the degrees of freedom that are involved during the starting cycle. One direction is assigned arbitrarily to be positive and the opposite to be negative. The angles indicated reflect the rotations which are passed through the hierarchy by successive rotations at the joints, starting at the root which is fixed at the ankle of the stance leg.

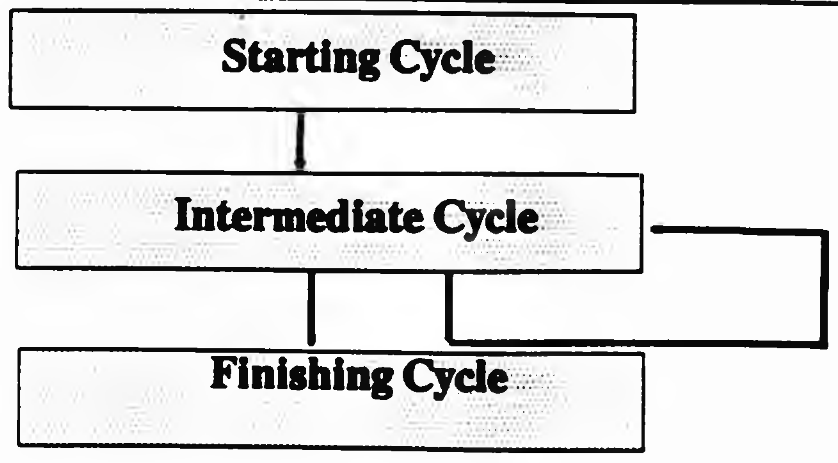


**Fig 7.6 : Schematic representation of different types of cycles.**

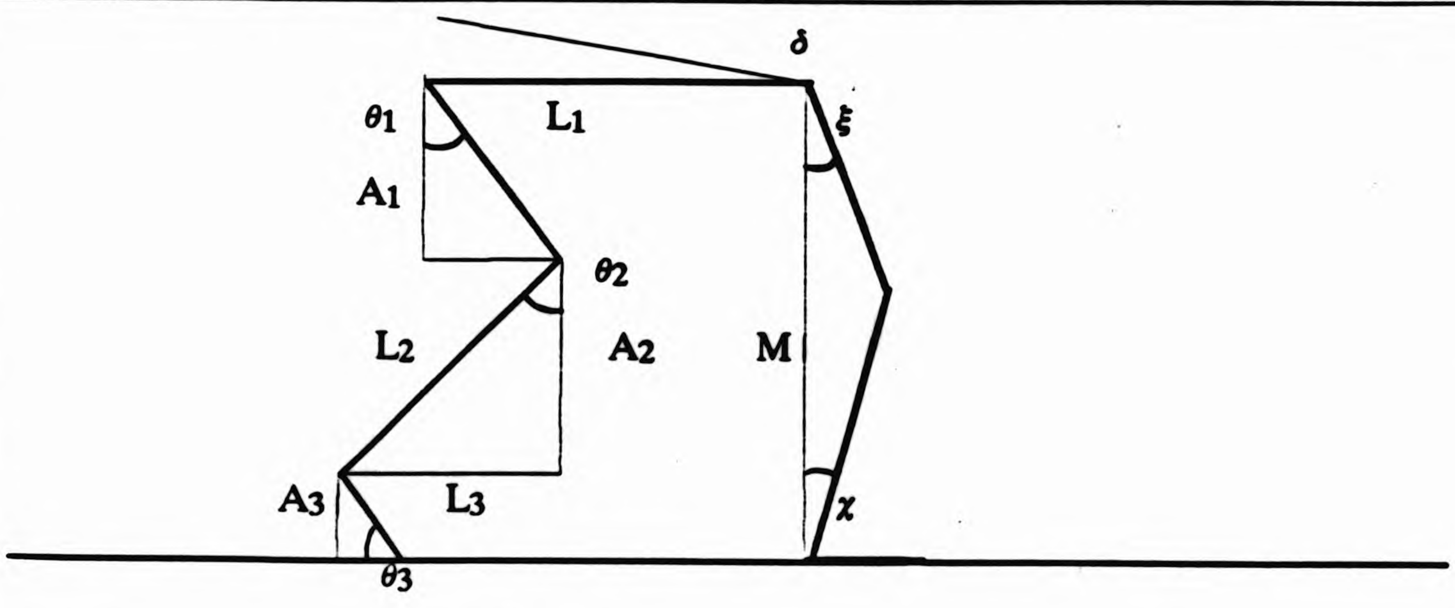


**Fig 7.7 : Position of the body after phase 1 in starting cycle.**

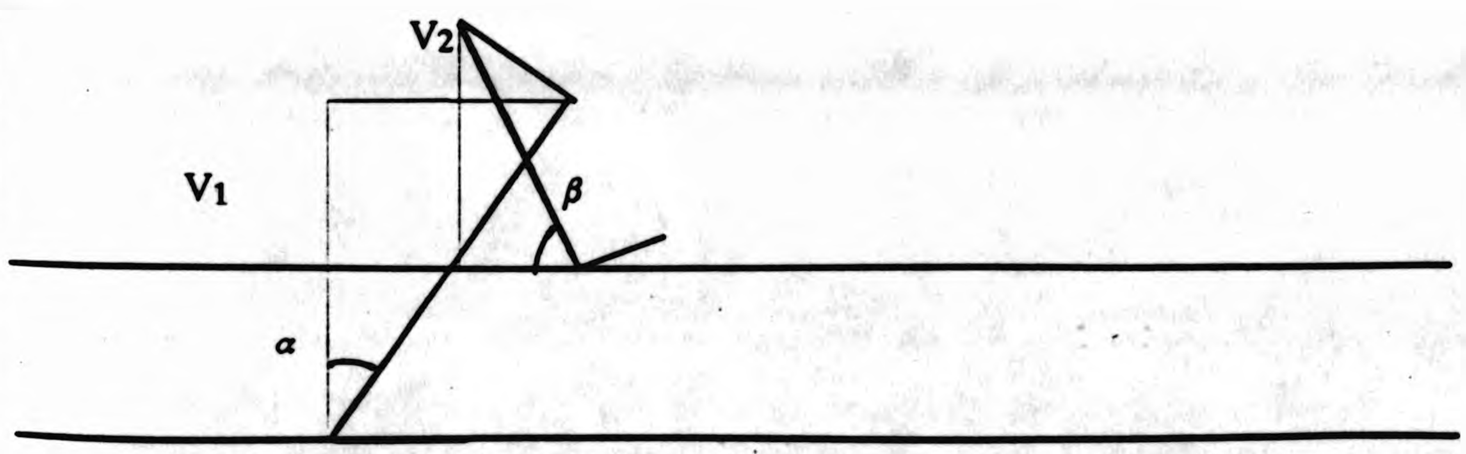


**Fig 7.8 : Position of the body after phase 2 of the starting cycle.**

| Starting | Pelvic | Stance Leg | | Swing Leg | |
|---|---|---|---|---|---|
| Cycle | Tilt | Ankle | Knee | Hip | Knee |
| Phase 1 | $\delta$ | $\chi$ | $-\chi-\xi$ | $-\theta_1$ | $\theta_1+\theta_3$ |
| Phase 2 | $-\delta$ | $\alpha-\chi$ | $\chi+\xi$ | $-90+\beta+\theta_1$ | $-\theta_1-\theta_3$ |

Table 7.1 : Target angles for starting cycle

## 7.4.2 Intermediate cycle

Phase 1 : After the starting cycle the motion is repeated in walking periods. The first phase of the intermediate cycle involves liftoff of the swing leg, stance knee flexion and positive pelvic tilt. In Fig 7.9 the state of the body after performing the first phase of the intermediate cycle is shown. The two assumptions that held for phase 1 of the starting cycle, are again adopted. Two new angles associated with the swing leg are introduced. $\omega$ is the angle that the thigh of the swing leg makes with the vertical and $\phi$ is the angle that the lower swing leg makes with the thigh. By using the vertical height in Fig 7.9, in a similar way to that used in Fig 7.7 for the starting cycle we obtain :

$$\cos\omega = \frac{-L_2\cos\phi - L_3\sin\phi + D}{L_1} \tag{7.5}$$

The similarity of (7.5) and (7.3) is obvious. If $\theta_3$ is equal to $\phi$ then $\omega$ is equal to $\theta_1$ then eqns (7.3) and (7.5) become identical. For generality however, $\theta_3$ and $\phi$ are assumed different.

Phase 2 : During phase 2, a negative pelvic tilt rotation takes place and both legs stretch so that there is no knee flexion (Fig 7.10). The direction of the pelvis is considered always to be perpendicular to the direction of motion so with this condition each leg covers half a stride. There are two relationships regarding the horizontal distance that the body covers :

$$S_1 = (L_1 + L_2)\sin\alpha \quad \textit{and} \quad S_2 = (L_1 + L_2)\cos\beta$$

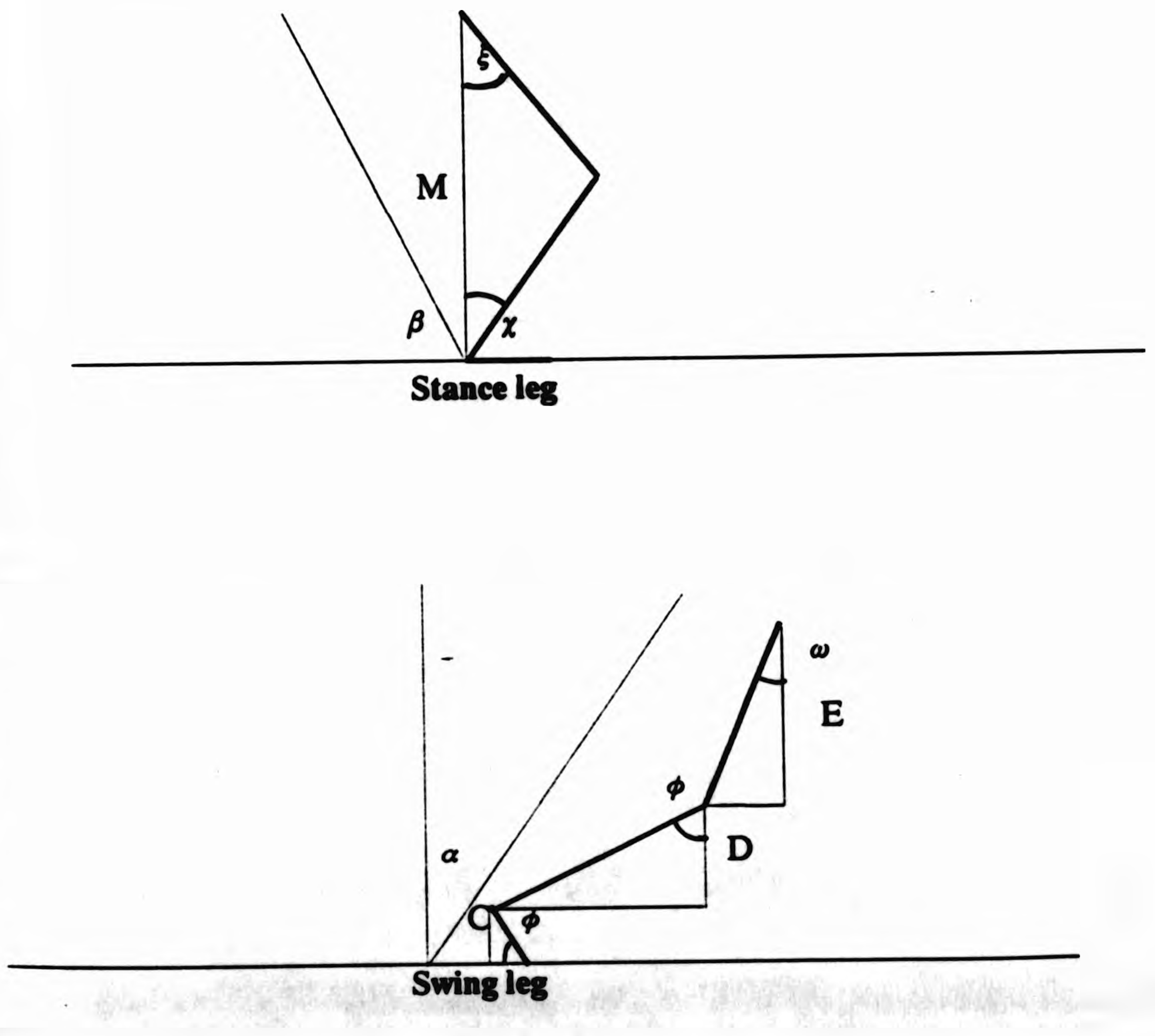


**Fig 7.9 : Position of the body after phase 1 of the intermediate cycle.**

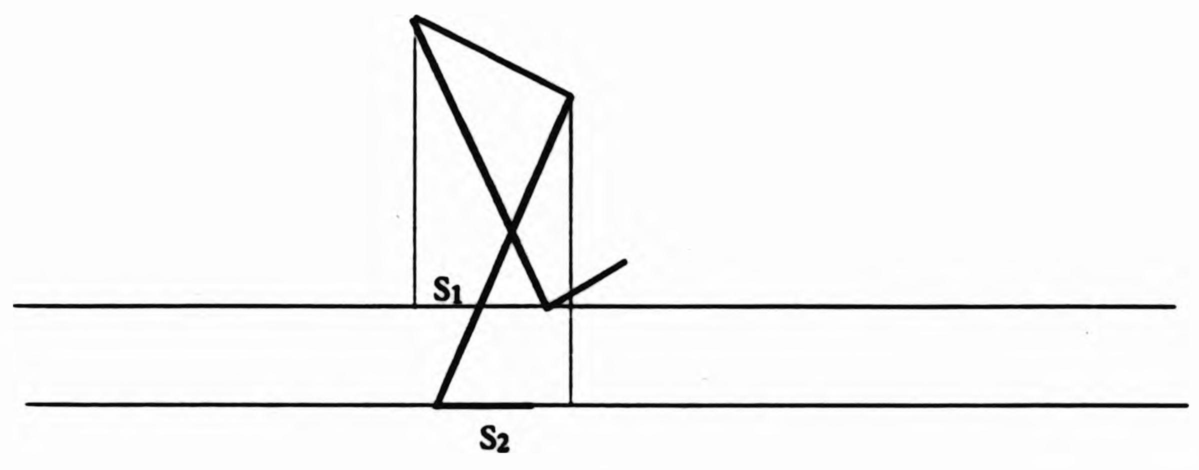


**Fig 7.10 : Position of the body after phase 2 of the intermediate cycle.**

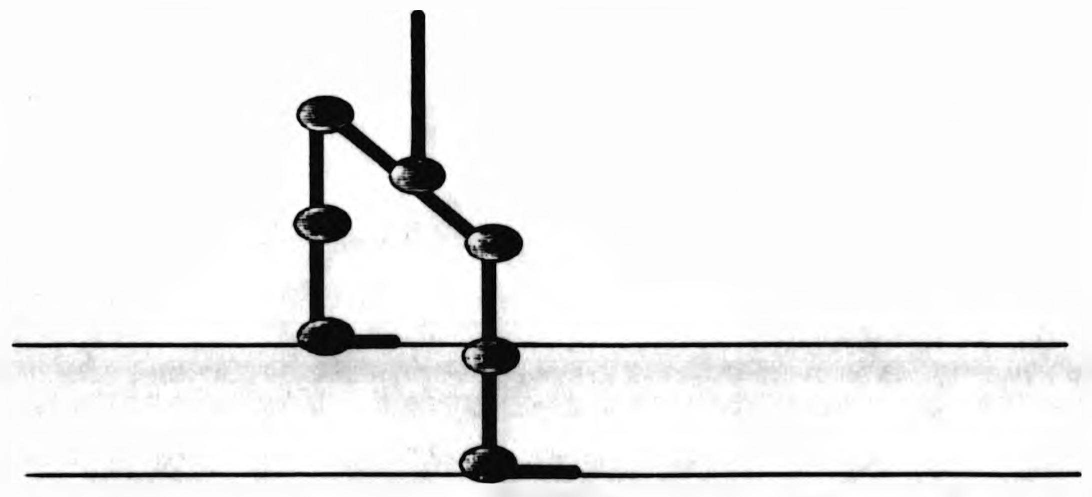


**Fig 7.11 : Position of the body after phase 2 of finishing cycle. The body comes to its rest position.**

The horizontal distance that is covered by the body during a cycle (stride) is denoted by $S$ where $S = S_1 + S_2$, hence :

$$\cos\beta = \sin\alpha = \frac{S}{2(L_1 + L_2)} \tag{7.6}$$

Table 7.2 shows the rotations of the joints during the intermediate cycle.

| Intermediate | Pelvic Tilt | Stance Leg | | Swing Leg | | |
|---|---|---|---|---|---|---|
| Cycle | Tilt | Ankle | Knee | Hip | Knee | Ankle |
| Phase 1 | $\delta$ | $90 - \beta + \chi$ | $-\chi - \xi$ | $\omega - \alpha$ | $\phi - \omega$ | $\alpha$ |
| Phase 2 | $-\delta$ | $\alpha - \chi$ | $\chi + \xi$ | $-90 + \beta - \omega$ | $-\phi + \omega$ | 0 |

Table 7.2 : Target angles for intermediate cycle

### 7.4.3 Finishing cycle

During the finishing cycle no new relationships are derived. Phase 1 of the finishing cycle is identical to phase 1 of the intermediate cycle. Phase 2 of the finishing cycle must bring the body to the rest position as shown in Fig 7.11. Table 7.3 shows the target angles for the finishing cycle.

| Finishing | Pelvic | Stance Leg | | Swing Leg | | |
|---|---|---|---|---|---|---|
| Cycle | Tilt | Ankle | Knee | Hip | Knee | Ankle |
| Phase 1 | $\delta$ | $90 - \beta$ | 0 | $\omega - \alpha$ | $\phi - \omega$ | $\alpha$ |
| Phase 2 | $-\delta$ | 0 | 0 | $-\omega$ | $-\phi + \omega$ | 0 |

Table 7.3 : Target angles for finishing cycle

### 7.4.4 Calculating the unknown parameters from the gait determinants

From the above analysis, the unknown parameters of the problem can be found by using relationships between the gait determinants. Given :

- $\theta_3$, the swing leg liftoff angle in the starting cycle
- $S$, the stride of the walking
- $\delta$, the pelvic tilt angle
- $\chi$, the stance knee flexion
- $\phi$, the swing leg liftoff angle in the intermediate cycle

the angle $\theta_1$ can be found using (7.3), $\omega$ can be found using (7.5) and angles $\alpha$ and $\beta$ can be found using (7.6). In all calculations, the lengths of the limbs are assumed to be known.

### 7.4.5 Eliminating parameters by using motion constraints

The number of input parameters can be decreased if the constraints of the motion are taken into account. There are four constraints that arise from the way the motion is prescribed :

• *Constraint (i)* : $\alpha - \chi > 0$. This constraint arises from phase 2 of the starting cycle. The physical meaning of this constraint is that the stance leg inclines forward (with the ankle fixed on the ground) as the swing leg moves forward to hit the ground. With $\alpha, \chi < 90$, this constraint implies that $\sin\alpha > \sin\chi$. From (7.6) :

$$\chi < \sin^{-1}\left(\frac{S}{2(L_1 + L_2)}\right) \tag{7.7}$$

Therefore there is a maximum bound for $\chi$ given by (7.7). On the other hand (7.7) can be used to calculate $\chi$ given the stride $S$.

• *Constraint (ii)* : $90 - \beta - \theta > 0$. The second constraint also arises from phase 2 of the starting cycle. It means that the thigh of the swing leg rotates forward as the leg moves from liftoff to its next impact with the ground. Therefore :

$$\sin(90 - \beta) > \sin\theta_1 \Rightarrow \cos\beta > \sin\theta_1 \Rightarrow \cos\theta_1 > \left(1 - \cos^2\beta\right)^{\frac{1}{2}}$$

and from (7.3) $L_1 \cos\theta_1 = -L_2\cos\theta_3 - L_3\sin\theta_3 + D$. These two expressions imply that :

$$L_2\cos\theta_3 + L_3\sin\theta_3 < D - L_1\left(1-\cos^2\beta\right)^{\frac{1}{2}}$$

and by letting $Q = D - L_1(1-\cos^2\beta)^{1/2}$ we have : $L_2\cos\theta_3 + L_3\sin\theta_3 < Q$ which can expanded to be

$$\begin{aligned} L_2^2\left(1-\sin^2\theta_3\right) &< (Q - L_3\sin\theta_3)^2 \Rightarrow \\ \left(L_2^2+L_3^2\right)\sin^2\theta_3 &- 2QL_3\sin\theta_3 + \left(Q^2 - L_2^2\right) > 0 \end{aligned} \qquad (7.8)$$

From (7.3) we also have that :

$$-L_2\cos\theta_3 - L_3\sin\theta_3 + D < L_1$$

By solving the quadratic inequality (7.8) as an equality we have two roots :

$$\theta_3 = \sin^{-1}\left(\frac{QL_3 \pm L_2(L_2^2+L_3^2-Q^2)^{\frac{1}{2}}}{(L_2^2+L_3^2)}\right) \qquad (7.9)$$

From (7.9) we have $L_2\cos\theta_3 + L_3\sin\theta_3 > Q_1$ where $Q_1 = D - L_1$ which can be expressed as :

$$(L_2^2+L_3^2)\sin^2\theta_3 - 2Q_1L_3\sin\theta_3 + (Q_1^2 - L_2^2) < 0$$

which if it solved as an equality gives another two roots for $\theta_3$ :

$$\theta_3 = \sin^{-1}\left(\frac{Q_1L_3 \pm L_2(L_2^2+L_3^2-Q_1^2)^{\frac{1}{2}}}{(L_2^2+L_3^2)}\right) \qquad (7.10)$$

The four solutions for $\theta$, two from (7.9) and two from (7.10), regard inequalities and not equalities. Assume that (7.9) gives roots ${}^1\theta_3$ and ${}^2\theta_3$ whereas (7.10) gives roots ${}^3\theta_3$ and ${}^4\theta_3$ with ${}^1\theta_3 \leq {}^2\theta_3$ and ${}^3\theta_3 \leq {}^4\theta_3$. Fig 7.12 and 7.13 show the solution of eqns (7.9) and (7.10) respectively.

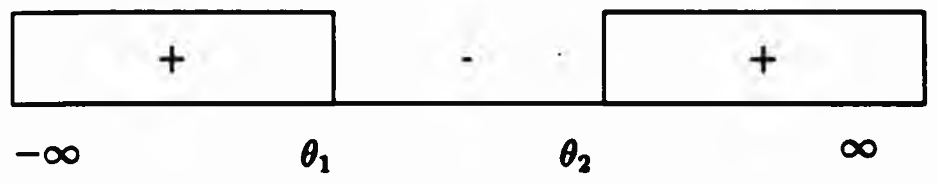


Fig 7.12 : Solution of eqn 7.9

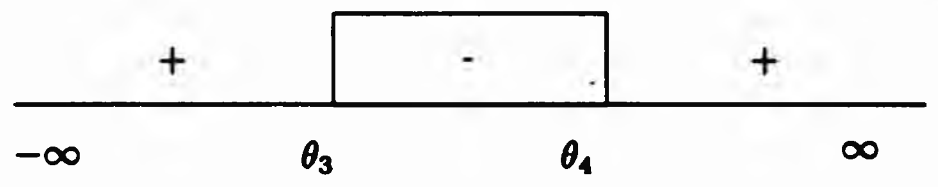


Fig 7.13 : Solution of eqn 7.10

| $^1\theta_3$ | $^2\theta_3$ | $^3\theta_3$ | $^4\theta_3$ | Conditions | Solutions |
|---|---|---|---|---|---|
| | | | | $^1\theta_3 > {}^3\theta_3$ | $\theta_3 = ({}^1\theta_3 + {}^3\theta_3)/2$ |
| | | | | $^3\theta_3 = {}^1\theta_3$ | $\theta_3 = {}^3\theta_3$ |
| +ve | +ve | +ve | +ve | $^1\theta_3 < {}^3\theta_3 < {}^4\theta_3 < {}^2\theta_3$ | No solution |
| | | | | $^1\theta_3 < {}^3\theta_3$ & $^2\theta_3 = {}^4\theta_3$ | $\theta_3 = {}^2\theta_3$ |
| | | | | $^1\theta_3 < {}^3\theta_3 < {}^2\theta_3 < {}^4\theta_3$ | $\theta_3 = ({}^4\theta_3 + {}^2\theta_3)/2$ |
| | | | | $^1\theta_3 \geq {}^4\theta_3$ | $\theta_3 = {}^4\theta_3/2$ |
| +ve | +ve | -ve | +ve | $^4\theta_3 > {}^1\theta_3$ & $^4\theta_3 < {}^2\theta_3$ | $\theta_3 = {}^1\theta_3/2$ |
| | | | | $^4\theta_3 \geq {}^2\theta_3$ | $\theta_3 = {}^1\theta_3/2$ |
| | | | | $^3\theta_3 < {}^2\theta_3$ & $^4\theta_3 < {}^2\theta_3$ | No solution |
| | | | | $^3\theta_3 < {}^2\theta_3$ & $^4\theta_3 = {}^2\theta_3$ | $\theta_3 = {}^2\theta_3$ |
| -ve | +ve | +ve | +ve | $^3\theta_3 < {}^2\theta_3$ & $^4\theta_3 > {}^2\theta_3$ | $\theta_3 = ({}^4\theta_3 + {}^2\theta_3)/2$ |
| | | | | $^3\theta_3 = {}^2\theta_3$ | $\theta_3 = {}^2\theta_3$ |
| | | | | $^3\theta_3 = {}^2\theta_3$ | $\theta_3 = ({}^4\theta_3 + {}^3\theta_3)/2$ |
| | | | | $^4\theta_3 < {}^2\theta_3$ | No solution |
| -ve | +ve | -ve | +ve | $^2\theta_3 = {}^4\theta_3$ | $\theta_3 = {}^2\theta_3$ |
| | | | | $^4\theta_3 > {}^2\theta_3$ | $\theta_3 = ({}^4\theta_3 + {}^2\theta_3)/2$ |

Table 7.4 : Solution of inequalities for $\theta_3$

Both inequalities have to be satisfied simultaneously and the solution must be positive although, this may not be possible sometimes. Table 7.4 shows the solutions for each possible combination. In most cases there is a multiplicity of solutions so an average values is taken as a solution. In cases where the solution can be chosen from two ranges, the average of the boundary values of the range closer to zero is taken as a solution.

Constraint (iii) : $\omega - \alpha > 0$. This constraint arises from phase 1 of the intermediate cycle and means that the thigh of the swing leg swings backwards before liftoff. Therefore from (7.6) :

$$\sin\omega > \sin\alpha \;\Rightarrow\; \omega > \sin^{-1}\left(\frac{S}{2(L_1 + L_2)}\right) \tag{7.11}$$

This can be considered to be a lower bound for $\omega$.

Constraint (iv) : $\phi - \omega > 0$. This constraint also arises from phase 1 of the intermediate cycle and means that the knee joint of the swing leg bends further during liftoff. Hence : $\cos\phi < \cos\omega$ and from (7.5) :

$$\begin{aligned} \cos\omega &= \frac{-L_2\cos\phi - L_3\sin\phi + D}{L_1} > \cos\phi \;\Rightarrow \\ -L_2\cos\phi - L_3\sin\phi + D &> L_1\cos\phi \end{aligned}$$

which can be expanded to :

$$\sin^2\phi[L_3^2 + (L_1 + L_2)^2] \;-\; 2DL_3\sin\phi \;+\; [D^2 - (L_1 + L_2)^2] \;>\; 0 \tag{7.12}$$

By solving (7.12) as an equality, it yields two roots :

$$\phi \;=\; \sin^{-1}\left(\frac{DL_3 \pm (L_1 + L_2)[L_3^2 + (L_1 + L_2)^2 - D^2]^{1/2}}{L_3^2 + (L_1 + L_2)^2}\right) \tag{7.13}$$

From (7.5) we also have :

$$-L_2\cos\phi - L_3\sin\phi + D \;<\; L_1 \tag{7.14}$$

with a similar treatment to that of (7.9), it yields two solutions :

$$\phi \;=\; \sin^{-1}\left(\frac{Q_1L_3 \pm L_2[L_2^2 + L_3^2 - Q_1^2]^{1/2}}{(L_2^2 + L_3^2)}\right) \tag{7.15}$$

Eqns (7.13) and (7.15) give four values for $\phi$ and the two inequalities have to be treated in the same way as in Constraint (ii). This will result to a table similar to table 7.4. In order to avoid repetition, we assume that (7.13) yields solutions ${}^1\phi$ and ${}^2\phi$ whereas (7.15) gives ${}^3\phi$ and ${}^4\phi$. Table 7.4 can, then, be read for $\phi$ instead of $\theta_3$.

## 7.5 Time durations

A phase represents a proportion of time within a cycle. There are experimental results that relate the walking speed with the time that the body takes to perform a cycle. Brudelin [89] refers to some expressions that relate these various quantities. *Step frequency* ($sf$ steps/min) is the quantity that gives the number of steps that the body performs per minute. Experimental results show that the *maximum step frequency* ($sf_{max}$) for walking is :

$$sf_{max} = 182 \textit{ steps per min} \tag{7.16}$$

There is also a normalized relationship between the step frequency and the stride :

$$\frac{stride}{sf \times body\_height} = 0.004 \tag{7.17}$$

The *body_height* is the quantity that normalize the relationship. If the forward velocity of the body is $V_f$, then $V_f = sf \times Stride$, and from (7.17) $\Rightarrow$

$$\frac{V_f}{0.004 \times body\_height} = sf^2 \tag{7.18}$$

The body can have both feet on the ground, i.e double support period or only one foot in contact with the ground, i.e swing period. Two times are defined, $t_{double\_support}$ and $t_{swing}$. The time for a cycle is

$$t_{cycle} = \frac{1}{sf} \tag{7 19}$$

Experimental data shows that :

$$t_{double\_support} = \frac{2(-0.16 \times sf + 29.08) \times t_{cycle}}{100} \tag{7.20}$$

Hence given the step frequency, $t_{cycle}$ and $t_{double\_support}$ can be calculated from (7.19) and (7.20) and $t_{swing}$ can be found as :

$$t_{swing} = t_{cycle} - t_{double\_support} \tag{7.21}$$

which leads to the result that given $sf$, all the times can be calculated. From these results the time duration of each phase can be calculated.

| Cycle | Simulation time for Phase 1 | Simulation time for Phase 2 |
|---|---|---|
| Starting | $\frac{1}{2}$ double support time | swing time |
| Intermediate | double support time | swing time |
| Finishing | double support time | $\frac{1}{2}$ swing time |

Table 7.5 : Simulation times for the three cycles

Table 7.5 shows the times of the different walking phases. In the starting cycle, the duration of phase 1 can be assumed to be equal to the double support time although this would not be accurate, but it would give uniformity to the duration of the starting and intermediate cycles. The same can be done with phase 2 of the finishing cycle i.e assign the time to be equal to half the swing time.

## 7.6 Arm co-ordination

Section 7.4 described how to establish the target angles for the lower extremes of the body in each phase and cycle. This section investigates the motion of the arms during walking. It seems more difficult to find a way of describing the motion of the arms, mostly because their motion is less stereotyped. However, one can observe that the arms move in opposite phases with respect to the corresponding legs. This fact, coupled with the assumption that the target angles of the arms are related to the target angles of the legs, can be used to establish a correlation between the two sets of angles. There are many ways of finding a relationship between them, but since they move in opposite phases, it seems natural to relate the target angles of the left arm with the target angles of the right leg, and the target angles of the right arm with the target angles of the left leg. Figures 7.14, 7.15 and 7.16 show the state of the arms in each cycle and phase.

According to this arrangement, there are nine angles involved, labelled $\alpha_1$ to $\alpha_9$. Table 7.6 gives the rotations during each phase. The angles indicated reflect the rotations which are passed through the hierarchy by successive rotations at the joint, starting at the root which is fixed at the stance leg.

| Cycle | Arm | Shoulder | Elbow |
|---|---|---|---|
| Starting Cycle - Phase 1 | Stance | $-\alpha_1$ | $-\alpha_2$ |
| | Swing | $-\alpha_3$ | 0 |
| Starting Cycle - Phase 2 | Stance | $-\alpha_4 + \alpha_1$ | 0 |
| | Swing | $-\alpha_5 - \alpha_3$ | 0 |
| Intermediate Cycle - Phase 1 | Stance | $-\alpha_5 + \alpha_6$ | $-\alpha_7$ |
| | Swing | $-\alpha_4 - \alpha_8$ | $\alpha_2 - \alpha_9$ |
| Intermediate Cycle - Phase 2 | Stance | $-\alpha_6 - \alpha_4$ | $-\alpha_2 + \alpha_7$ |
| | Swing | $\alpha_8 + \alpha_5$ | $\alpha_9$ |
| Finishing Cycle - Phase 1 | Stance | $-\alpha_5 + \alpha_6$ | $-\alpha_7$ |
| | Swing | $\alpha_4 - \alpha_8$ | $\alpha_2 - \alpha_9$ |
| Finishing Cycle - Phase 2 | Stance | $-\alpha_6$ | $\alpha_7$ |
| | Swing | $\alpha_8$ | $\alpha_9$ |

Table 7.6 : Target angles for the arms

The target angles for the arms can be arbitrarily assigned by the user in order to generate a particular motion. However, this task can be extremely difficult as the angles that generate realistic motion are unknown. Hence, a relationship between leg and arm angles would be more natural. According to the assumption that the left arm angles should be related to the right leg angles, $\alpha_1$, $\alpha_2$ and $\alpha_3$ will relate to $\theta_1$, $\theta_2$ and $\chi$. This approach can be extended to the other angles. In this way, target angles can be expressed as ratios of the leg angles, and this results in the following assignments :

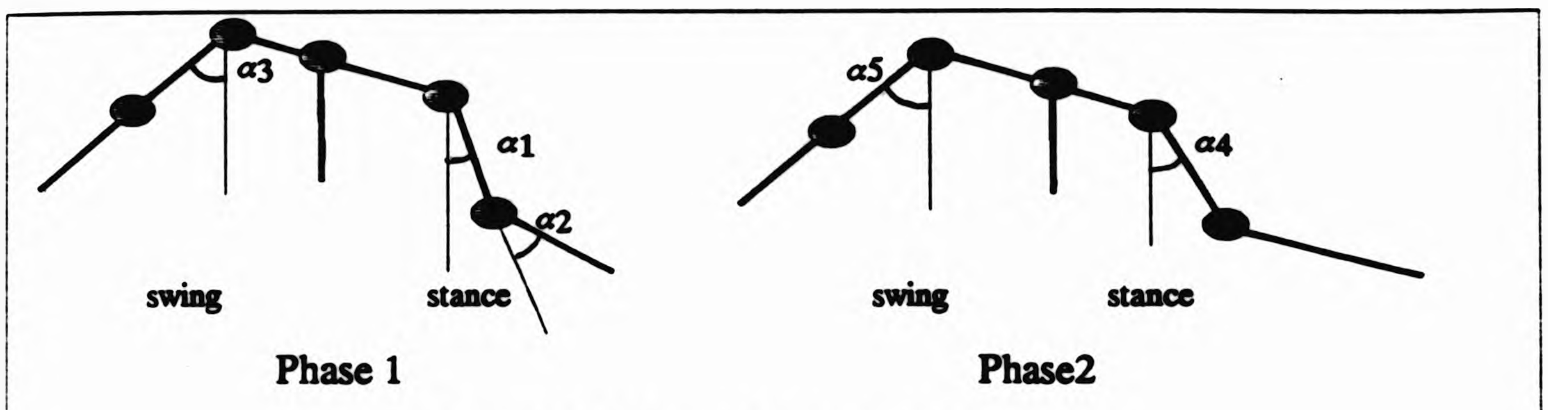


**Fig 7.14 : Position of arms for the starting cycle.**

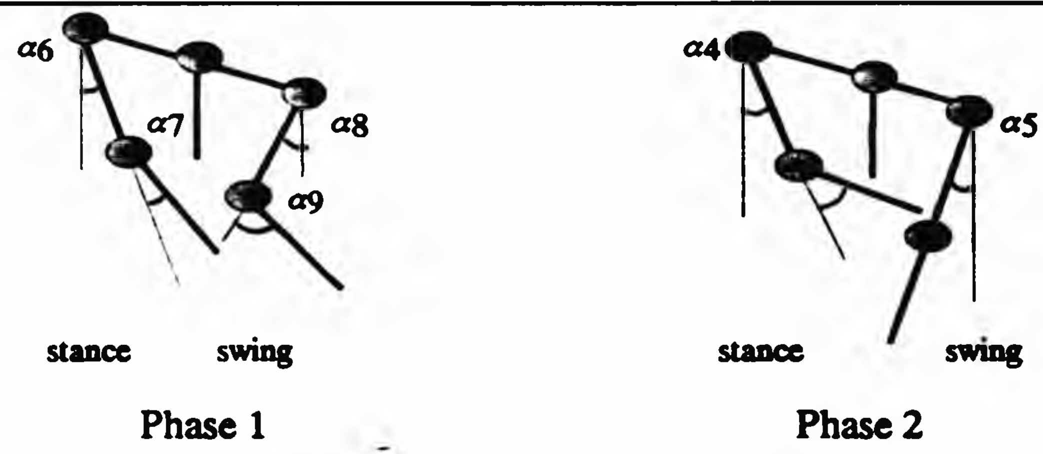


**Fig 7.15 : Position of arms for the intermediate cycle.**

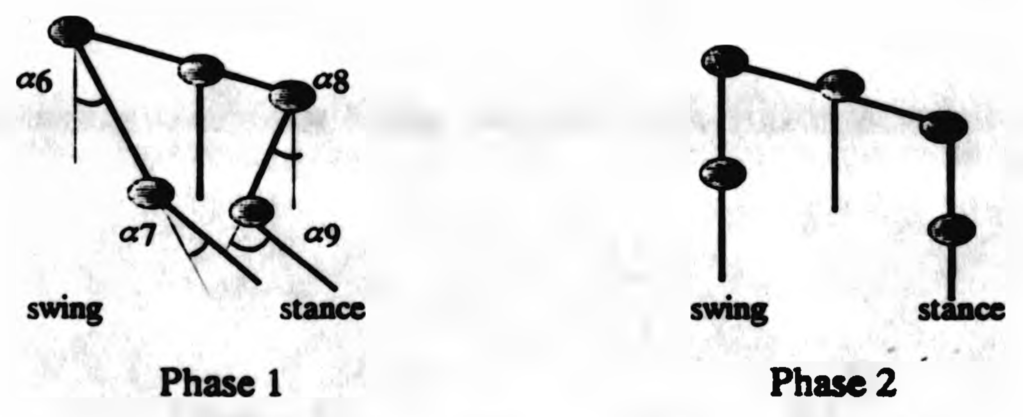


**Fig 7.16 : Position of arms for the finishing cycle.**

$$\begin{array}{llll} \alpha_1 = k_1\theta_1 & \alpha_2 = k_1\theta_3 & \alpha_3 = k_1\chi & \\ \alpha_4 = \alpha_1 + k_2\beta & \alpha_5 = \alpha_3 + k_2\alpha & & \\ \alpha_6 = \alpha_5 - k_3\omega & \alpha_7 = k_3\phi & \alpha_8 = \alpha_5 - k_3\beta & \alpha_9 = \alpha_2 - k_3\chi \end{array} \tag{7.22}$$

The three coefficients $k_1$, $k_2$ and $k_3$ are user-prescribed and allow a great variety of motions. Limiting them to lie between 0 and 1 is not a firm constraint, although setting them to be greater than one might result in unnatural motion.

## 7.7 Investigation of walking parameters

The motion generated by the model presented in this Chapter depends upon a number of parameters which in turn depend upon each other. For a given set of parameters, the user does not know if the kinematics equations of motion can produce a solution, or, when they do, what kind of motion the user should expect. This is due to the solution of the inequalities that give the values of $\theta_3$ and $\phi$. In addition to the uncertaincy of the inequalities, certain constraints must be satisfied in order to have natural motion as explained in previous sections.

In this section, an attempt is made to correlate the walking parameters in some way. This investigation looks into the solution of $\theta_3$, $\omega$ and finally into solution of the entire system of the kinematic constraints to see which parameters produce realistic motion and which do not.

o $\theta_3$ : The value of $\theta_3$ as found from (7.9) and (7.10) depends upon the solution of the two quadratic inequalities. The solution of these inequalities depends upon three angles $\chi$, $\delta$ and $\beta$. The first two angles are supplied directly by the user and $\beta$ can be calculated from $sf$. One way of visualising the behaviour of $\theta_3$ is to draw the values of $\theta_3$ against those of $\chi$, $\beta$ and $sf$. This would result in a $4D$ function which will be difficult to visualise in $3D$ space. Alternatively we can visualise slices of this hyper-function. This can be done by drawing $\theta_3$ against $\chi$ and $\delta$, for discrete values of $sf$.

A discrete set of $sf$ was chosen such that it is large enough to investigate the behaviour of $\theta_3$ i.e $sf = (20, 40, 60, 80, 100, 120, 140, 160, 180)$. For a prescribed $sf$ value, $\theta_3$ is sketched against a large set of $\chi$ and $\delta$ values that we believe covers the whole spectrum of walking i.e $1^0 \leq \delta \leq 10^0$ in steps of $0.5^0$ and $1^0 \leq \chi \leq 60^0$ in steps of $1^0$.

The functions have three interesting characteristics :

- For a prescribed $\delta$, the value of $\theta_3$ increases as the value of $\chi$ increases until it gets to a maximum value.
- The solution of the inequalities are of two types, either $(-ve, +ve, -ve, +ve)$, i.e type 5, or $(+ve, +ve - ve, +ve)$, i.e type 2. Because these two types produce values for $\theta_3$ that have considerable difference in magnitude, they appear in the curve as plateaus.
- For some values the inequalities are unsolvable. In these cases the function is assigned arbitrarily to -0.2 radians. As a result these values form a plateau which in any case will be lowest w.r.t the horizon.

Another interesting result that concerns the overall behaviour of $\theta_3$ is that its maximum value obtained for a prescribed $sf$ is monotonically decreasing as $sf$ increases. Table 7.7 gives the maximum values of $\theta_3$ and the values of $\chi$ at which they occur for prescribed $sf$ and $\delta$. An interesting aspect of the function is that after it reaches its maximum value (for prescribed $\delta$ and $sf$) the system becomes unsolvable. As the $sf$ increases the system becomes unsolvable (for prescribed $\delta$) for small as well as large $\chi$ values. Table 7.7 also gives the values of $\chi$ for unsolvability for prescribed $\delta$ and $sf$. These values show the overall behaviour of the function. A final point on the behaviour of $\theta_3$ is that as $sf$ increases, the type 2 solutions of the inequalities increase. Fig 7.17 shows the 3D functions for representative values of $sf$ i.e $sf = (40, 100)$.

∘ $\phi$ : The behavior of $\phi$ is similarly investigated. Like $\theta_3$, $\phi$ depends upon $\chi$, $\delta$ and $sf$. The ranges of $sf$, $\chi$ and $\delta$ are as before. The general characteristics of $\phi$ are :

- For a prescribed $sf$, the function has a saddle point.
- For a prescribed $sf$ and $\delta$, $\omega$ increases to a maximum value respectively with $\chi$ and then decreases.
- All solutions are of type 5 i.e $(-ve, +ve, -ve, +ve)$.

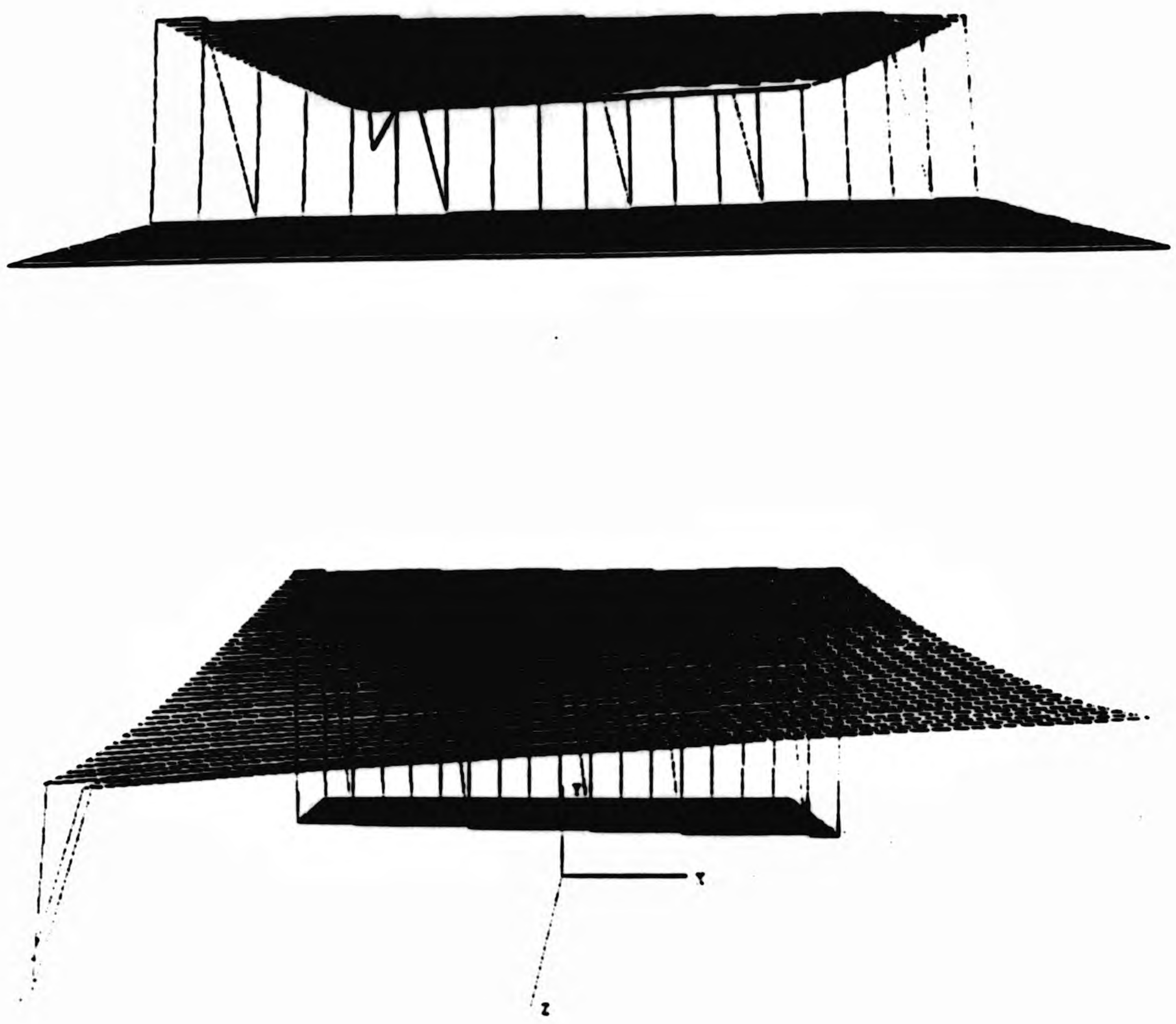
- **The maximum value of $\omega$ remains the same irrespective of the values of $sf$ and $\delta$ and this value is $\omega = 74^0$.**

| Step frequency | Function maximum value | | | Range of unsolvability | |
|---|---|---|---|---|---|
| | $\delta$ | $\chi$ | $\theta_3$ | $\delta$ | $\chi$ |
| 20 | $1^0$ | $47^0$ | $90^0$ | $1^0$ | $> 47^0$ |
| | $10^0$ | $42^0$ | $90^0$ | $10^0$ | $> 42^0$ |
| 40 | $1^0$ | $47^0$ | $89^0$ | $1^0$ | $> 47^0$ |
| | $10^0$ | $42^0$ | $89^0$ | $10^0$ | $> 42^0$ |
| 60 | $1^0$ | $47^0$ | $88^0$ | $1^0$ | $> 47^0$ |
| | $10^0$ | $42^0$ | $88^0$ | $10^0$ | $> 42^0$ |
| 80 | $1^0$ | $48^0$ | $88^0$ | $1^0$ | $> 48^0$ |
| | $10^0$ | $43^0$ | $88^0$ | $10^0$ | $> 43^0$ |
| 100 | $1^0$ | $49^0$ | $87^0$ | $1^0$ | $> 49^0$ & $< 10^0$ |
| | $10^0$ | $44^0$ | $87^0$ | $10^0$ | $> 44^0$ & $< 15^0$ |
| 120 | $1^0$ | $49^0$ | $85^0$ | $1^0$ | $< 16^0$ & $> 49^0$ |
| | $10^0$ | $45^0$ | $85^0$ | $10^0$ | $< 20^0$ & $> 45^0$ |
| 140 | $1^0$ | $51^0$ | $84^0$ | $1^0$ | $< 21^0$ & $> 51^0$ |
| | $10^0$ | $46^0$ | $84^0$ | $10^0$ | $< 6^0$ & $> 46^0$ |
| 160 | $1^0$ | $52^0$ | $81^0$ | $1^0$ | $< 27^0$ & $> 52^0$ |
| | $10^0$ | $47^0$ | $81^0$ | $10^0$ | $< 17^0$ & $> 47^0$ |
| 180 | $1^0$ | $54^0$ | $79^0$ | $1^0$ | $< 32^0$ & $> 54^0$ |
| | $10^0$ | $49^0$ | $79^0$ | $10^0$ | $< 25^0$ & $> 49^0$ |

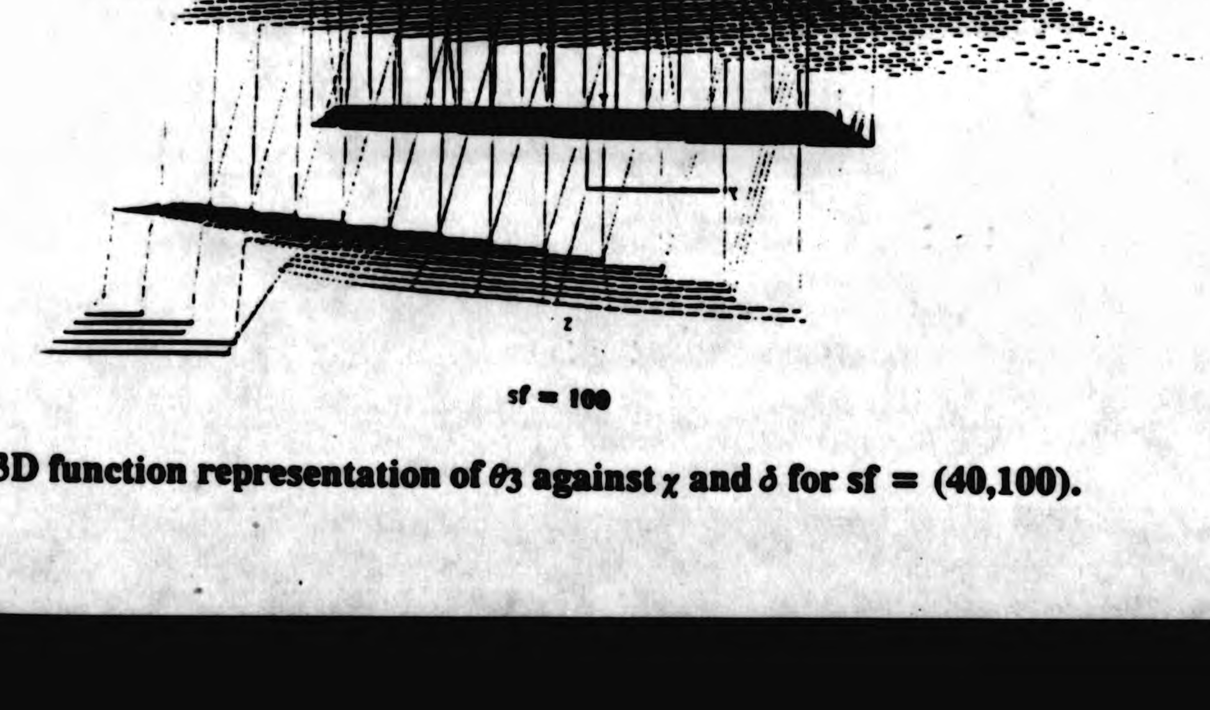
Table 7.7 : Analysis of $\theta_3$

**The system starts to become unsolvable for values of $\delta > 2^0$ and $\chi > 55^0$ (for a prescribed $sf$). These $\chi$ values remain fixed over the whole spectrum of step frequency values. Because for all $sf$ the function has almost the same shape, Fig 7.18 shows the function for $sf = 20$.**

**○ *Solvability of the parameters* : Now that the behaviour of the two 'unpredictable' parameters of the model have been examined, we can investigate the overall behaviour of the model.**

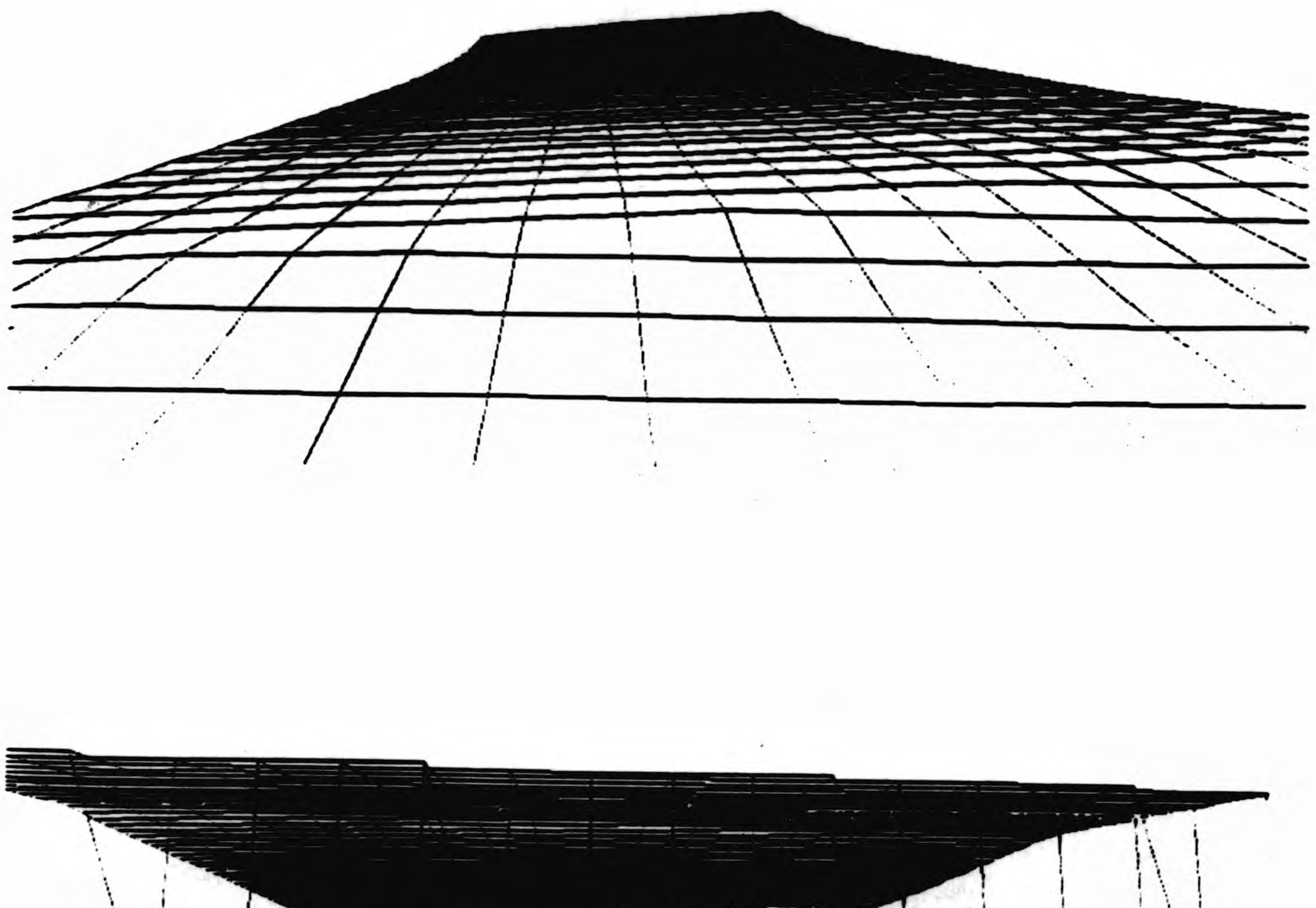


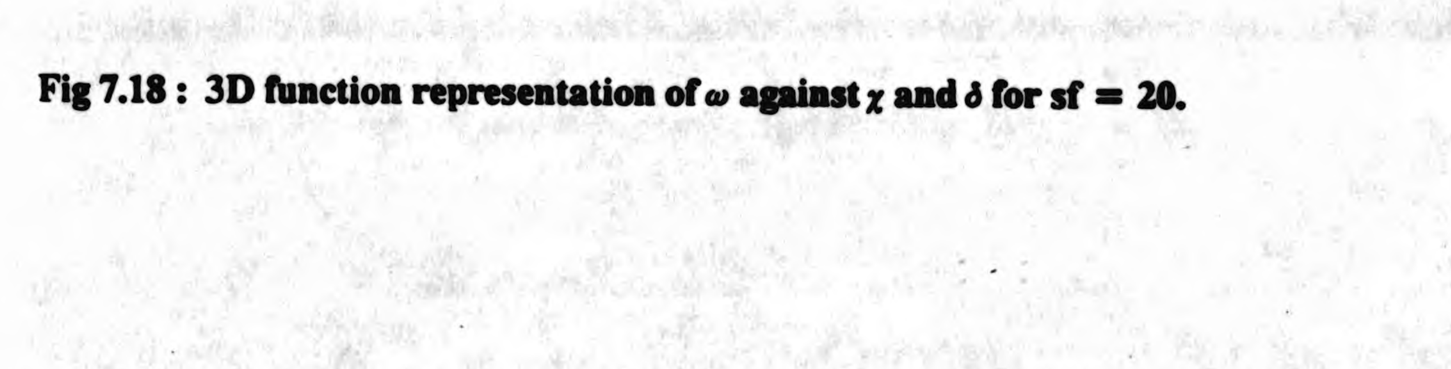
sf = 40



sf = 100

**Fig 7.17 : 3D function representation of $\theta_3$ against $\chi$ and $\delta$ for sf = (40,100).**





**sf = 20**

**Fig 7.18 : 3D function representation of $\omega$ against $\chi$ and $\delta$ for sf = 20.**

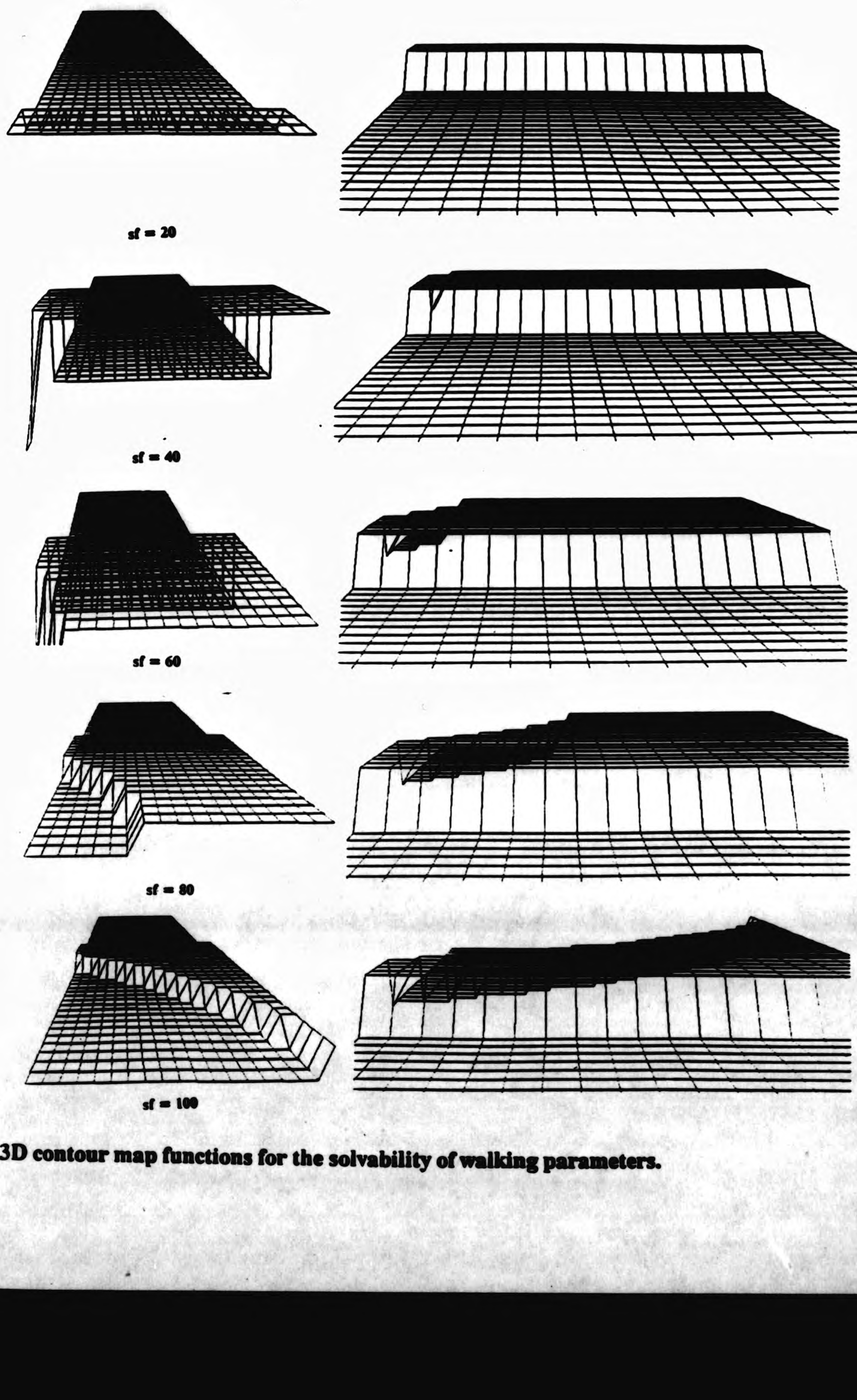


**Fig 7.19 : 3D contour map functions for the solvability of walking parameters.**

The main question to answer is for what set of parameters the model produces realistic motion. In order to have realistic motion the four constraints of the motion have to be satisfied.

A satisfactory way for depicting the behaviour of the model is to draw a sort of 3D contour map. The way that the map is constructed is that given $sf$, $\theta_3$ and $\chi$, the other parameters are calculated and if the parameters satisfy the constraints then the contour function is given value 1 for this set of parameters or 0 otherwise. The contour function is constructed for the same values of $sf$, $\chi$ and $\delta$ as it was done for $\phi$ and $\theta_3$.

The analysis of the contour maps show that the number of possible set of parameters decreases as the step frequency increases. For a prescribed step frequency there is a maximum $\chi$ value for which the system becomes unsolvable and this value increases as the step frequency decreases. Table 7.8 gives a detailed account of the values for which the system produces realistic motion.

| Step frequency | Solvable |
|---|---|
| 20 | $\chi < 4.96^0$ for all $\delta$ |
| 40 | $\chi < 9.95^0$ except for $\delta = 1^0$ where solution exists for $3.0^0 \leq \chi \leq 9.95^0$ |
| 60 | for $\delta = 1^0$ $9^0 \leq \chi \leq 15^0$ and after $\delta = 4^0$ (incl) solution exists for $\chi \leq 15^0$ |
| 80 | for $13^0 \leq \chi \leq 20.2^0$ and after $\delta = 8^0$ (incl) $\chi \leq 20.2^0$ |
| 100 | for $\delta = 1$ $21^0 \leq \chi \leq 25.6^0$ only for $\delta = 10^0$ $\chi \leq 25.6^0$ |
| 120 | for $\delta = 1$ $30^0 \leq \chi \leq 31.24^0$ and for $\delta = 10^0$ $22^0 \leq \chi \leq 31.2^0$ |
| 140 | solutions start at $\delta = 4^0$ $for$ $37^0 \leq \chi \leq 37.2^0$ for $\delta = 10^0$ for $34^0 \leq \chi \leq 37.2^0$ |
| 160 | solutions only for $\delta = 10^0$ and $43^0 \leq \chi \leq 43.75$ |
| 180 | solutions exist for all $\delta$ and $\chi$ around $51^0$ |

Table 7.8 : Solvability of the model

Fig 7.19 show the contour functions for step frequencies $sf = (20, 40, 60, 80, 100)$. In general, as the step frequency increases, two things happen :

- The number of possible solutions decrease.
- For a prescribed $\delta$, the range of value of $\chi$ for which the parameters have a solution becomes smaller and the boundary values of this range increase with the step frequency.

These results are somewhat expected because large walking speeds require larger stance knee flexion angles.

## 7.8 Motion specification

The way that motion is specified by the system is entirely interactive within the window environment. This has proved adequate and easy to use for simple motions. There is, however, a penalty incurred by such a design decision and this is that once the motion has started, the user can not interact with the system until it finishes. On the other hand only one command at the time can be issued to the system. We believe that more sophisticated motion control commands should be described in the form of a script and at the same time the user should be able to interact with the system while the body is in motion. This is of paramount importance in cases where the user does not have a clear idea about the resulting motion for a given set of parameters. Since dynamic analysis requires high computational times, the user should be able to interact and rectify motion on-line.

So, ideally, a mixture of a scripted and interactive system should be used. The motion could be described by a script file which initiates the motion and, as the system is displaying the motion, the user should be able to interact and change the script. This might mean the design and implementation of a parser for a motion specification language. This language should contain a small set of commands, initially, and the animator should be able to build new commands by using this small set.

In a UNIX environment this parser can be implemented in a less economise way than in other operating systems since a lexical analyser (LEX) and a parser generator (YACC) are present.

## 7.9 Conclusions

The analysis of walking parameters for the legs resulted in nine parameters, eight angles and the step frequency. By imposing the motion constraints, the number of parameters is restricted to three, the pelvic tilt angle, the stance knee flexion angle and the step frequency.

The analysis of of arm co-ordination resulted in relations between the angles of the arms and the angles of the legs. In these, the arm angles were expressed as ratios of the leg angles by introducing three coefficients $k_1$, $k_2$ and $k_3$. Hence walking (legs and arms), can be described by using six parameters.

At the beginning of each phase, the target angles are calculated and supplied into the dynamics component so that dynamic analysis can be performed and find the forces. The calculation of the normal forces can be included to the dynamic analysis. This adds to the realism of the motion but contributes to the computational complexity of the method.

For simplicity in the previous chapters, the root of the hierarchy was considered to be the torso. During the analysis of walking it emerged that the most suitable link for the root is the foot of the stance leg. This has the important consequence that the root changes from one cycle to another. The software has be therefore been adapted to cope with changing the root during consecutive dynamic analysis. The system is capable of changing the root of the body to any link, including the associated internal rearrangement of the hierarchy.

The most important result is the ease of describing walking. The step frequency and the two angles (pelvic tilt and stance leg flexion) are easily comprehended by the user of the system. The effects on arm behaviour which result from the particular choice of the three parameters can be determined from the values of the leg angles. Because

**the movement of the arms are less of a stereotype that those of the legs, the three arm coefficients can be used for finely tuning the motion such that it acquires a more personal style.**

**Experience has shown that simple motion sequences can be described via an interactive system, but for more complicated control sequences, the system should support scripted motion specification together with a level of interactability.**

# Chapter 8

# An Improved Walking Model

## 8.1 Introduction

The previous chapter introduced a model of walking that took into consideration three out of the six gait determinants. In this chapter, an improved model is described that takes into account five out of the six gait determinants involved in walking. This should make the model more realistic, although the mathematical complexity of the kinematic calculations is increased together with the number of parameters.

The five gait determinants in the model include, stance leg knee flexion, pelvic tilt and compass gait which are explained in the previous chapter. The two new gait determinants are :

- Pelvic rotation : this allows the pelvis to perform a rotary motion about the vertical axis. This axis in reality passes through the centre of mass of the pelvis. The rotation angle is quite small for normal walking speeds (around $\pm 3^0$) although it can be increased at higher speeds, McMahon [84]. This motion enables the legs to have a longer stride and thus achieve higher walking speeds.
- Plantar flexion of the swing ankle : During the transition of between double support phase (both feet in contact with the ground) and swing phase (stance leg only in contact with the ground), the heel of the swing leg moves down just

before toe-off. This gait determinant is important in establishing the initial velocity of the swing leg.

The distinction between stance and swing leg is not as clear as in the previous model. They are instances in the motion where either leg can be considered as being the stance leg. The plantar flexion determinant introduces the necessity of using toes into the model. Therefore, the model has two more links since each leg is considered to have one toe only. Although this makes the model more realistic it keeps the mathematics much simpler than using five toes for each leg.

The introduction of the two extra gait determinants increases the number of phases for each cycle. The number of cycles remains the same i.e *starting*, *intermediate*, and *finishing cycle*. However the starting cycle has two phases, the intermediate four phases, and the finishing three phases.

An additional introductory point about this model is that the root of the motion changes within a cycle. This is because of the introduction of toes. The root in this model will be considered to be a joint associated with a link rather than a link alone. The link in question will always be the foot of either the stance or swing leg, and the joint will either be the ankle or the toe of this leg. This will become obvious in the subsequent discussion.

This chapter offers a feasibility study of the possible extensions that can be made to the model presented in the previous chapter. It investigates the possibility of including toes and pelvic rotation to the model. Since, to our knowledge, this is the first time that such an attempt has been made in computer animation, it can be used as a pilot study for further research to the subject. Due to time constraints, the realisation of this model within the animation system was not possible. The major outcome of such an investigation is whether or not it is feasible to formulate bipedal walking, given the five gait determinants that are involved in the model. Two objectives are set during this investigation as to the way the motion is described :

- The construction of the model corresponds as closely as possible to the results produced from research in physiology and biomechanics, Thorstensson [84 & 85], McMahon [84] and Nilsson [85], and from an extended set of photographic sequences of human locomotion, Muybridge [55/1].

- As the complexity of the model increases considerably in comparison to the model presented in the last chapter, some assumptions about the behaviour of the body are made in order to keep the mathematical expressions simple. In some cases, these assumptions are justified by research in physiology, and in other cases we make these assumptions in order to keep the complexity of the model minimal. In the subsequent discussion, whenever such an assumption is made, its physical meaning will be given together with whether or not it is substantiated by physical evidence.

The chapter starts with the description of each cycle and its phases, together with the derivation of the mathematical relationships at each phase. It then proceeds with the description of the motion constraints and the elimination of parameters with the help of these constraints. The movements of the arms is then considered and linked to the motion of the legs. This model also discusses the movements of the trunk in an attempt to consider the interaction of the links of the entire body. Because most of the phases of the starting and finishing cycles are identical to phases of the intermediate cycle, we start with the intermediate cycle.

## 8.2 Intermediate cycle

The intermediate cycle contains four phases. The number of phases is determined by the complexity of the model and they are sufficient to describe realistic walking. As a result of using four phases instead of two, the behaviour of the pelvic tilt becomes more complicated and its amplitude with respect to time is given in Fig 8.1. The data for these gait determinants has been found from studies in physiology by many researchers McMahon [84] and Nilsson [85]. The top of the figure shows a typical human performing an intermediate cycle with the left leg being the swing leg, so phase 1 starts with right heel contact. If we assume a maximum amplitude $\delta$ for the pelvic tilt, then the curve gives the proportion of $\delta$ at the end of each phase. For example, at the end of phase 1 the pelvis has descended by $\delta$ at the end of the second phase the pelvis has descended an absolute value of approximately $0.4 \times \delta$ which means that between phases 1 and 2, the relative rotation of the pelvic tilt is $0.6 \times \delta$ and this is how this curve should be read for the other phases as well. It is interesting to see that pelvic tilt has diminished almost completely at phase 4.

The curve at the bottom of Fig 8.1 gives the behaviour of the pelvic rotation during the phases. Again the values are given as rations of the maximum pelvic rotation $\tau$. Notice that during phases 1 and 2 the curve stays in the positive plane whereas during phases 3 and 4 it is in the negative plane and it is also symmetrical. This means that during phases 1 and 2, the pelvis is rotated towards the left and during phases 3 and 4 towards the right. The time axis in the figure gives the percentage of $t_{cycle}$ elapsed after the completion of a phase. For example, phase 1 is completed after 25% of $t_{cycle}$, phase 2 after 50% of $t_{cycle}$ and so on. The rest of this section describes the four phases in detail.

*i. Phase 1* : Fig 8.2 shows the position of the body after completion of phase 1. The heel of the swing leg of the previous phase makes contact with the ground and the swing leg of this phase starts its liftoff. There are two assumptions :

- For the stance leg, the ankle joint is at a right angle when it makes contact with the ground. In reality the angle of the stance ankle is not exactly $90^0$ when it makes contact with the ground but it is close to this value, Muybridge [55/1].
- For the swing leg, the ankle joint remains at a right angle throughout the phase. This condition reflects reality as can be seen in biomechanical studies, McMahon [84].

There are four angles involved in the motion. The pelvic tilt is $\delta$ and the pelvic rotation is $\tau$. $\alpha$ and $\beta$ represent the angles that the stance and swing legs make with the vertical respectively (Fig 8.2). The pelvis descends by $\delta$ towards the side of the swing leg and the sole of the swing leg flexes just before liftoff which is occurring in the next phase. The pelvis rotates towards the side of the stance leg by $\frac{2\tau}{5}$.

Let $L_1$ and $L_2$ be the lengths of the thigh and lower leg respectively, $L_3$, $L_4$ be the lengths of the foot and toe respectively, and $L$ be the width of the pelvis, then the relationships between the angles can be derived.

The vertical heights of the stance and swing legs $V_{st}$ and $V_{sw}$ are :

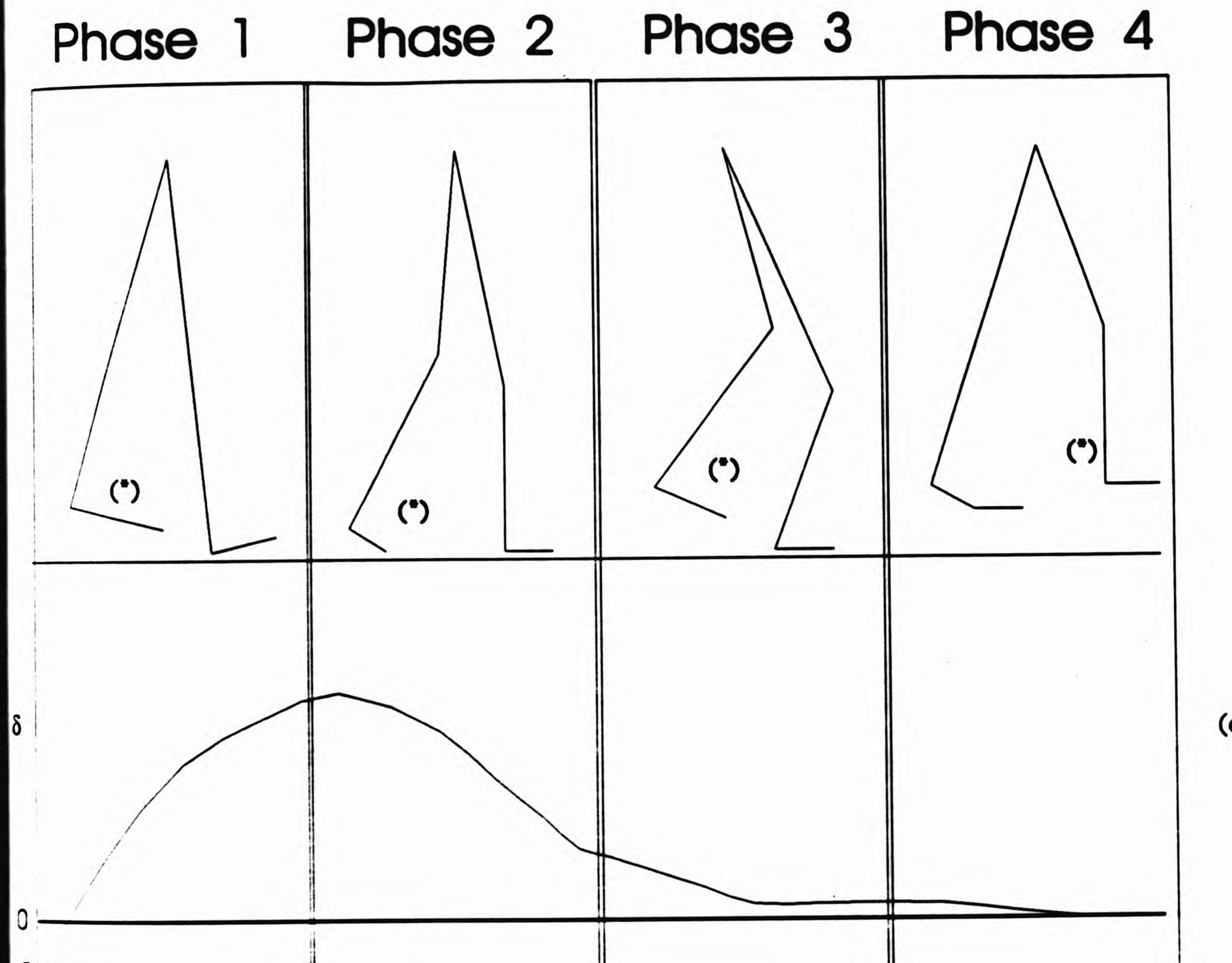


Fig 8.1 : (a) shows the behaviour of pelvic tilt and (b) shows the pelvic rotation. During phases 1 and 2 pelvic rotations are performed towards the side of the stance leg and then towards the side of the swing leg. (*) indicates swing leg.

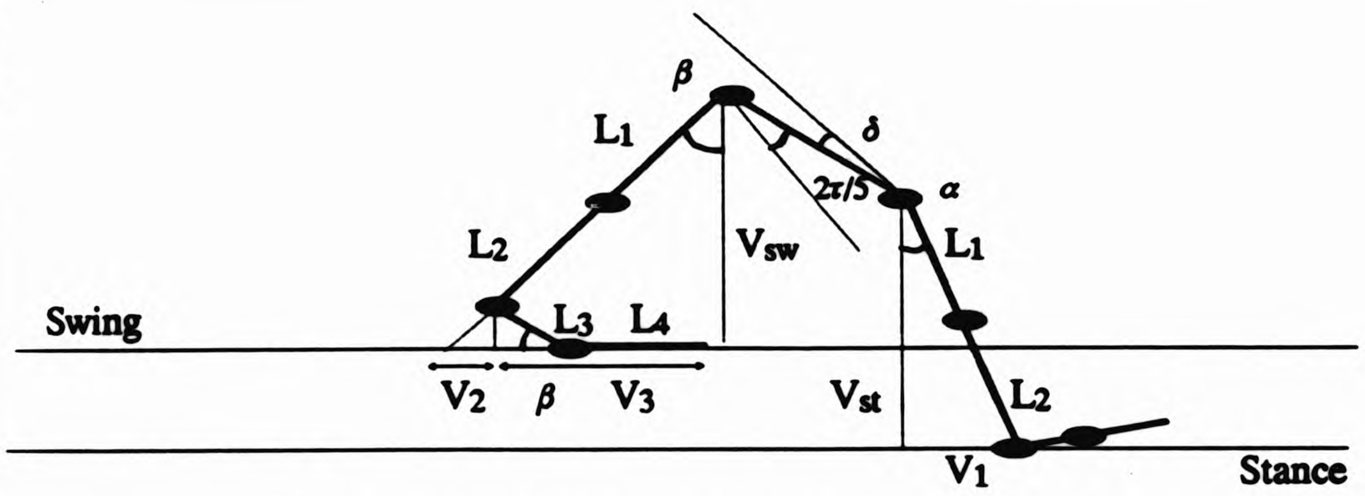


**Fig 8.2 : Position of the body after phase 1 of intermediate cycle.**

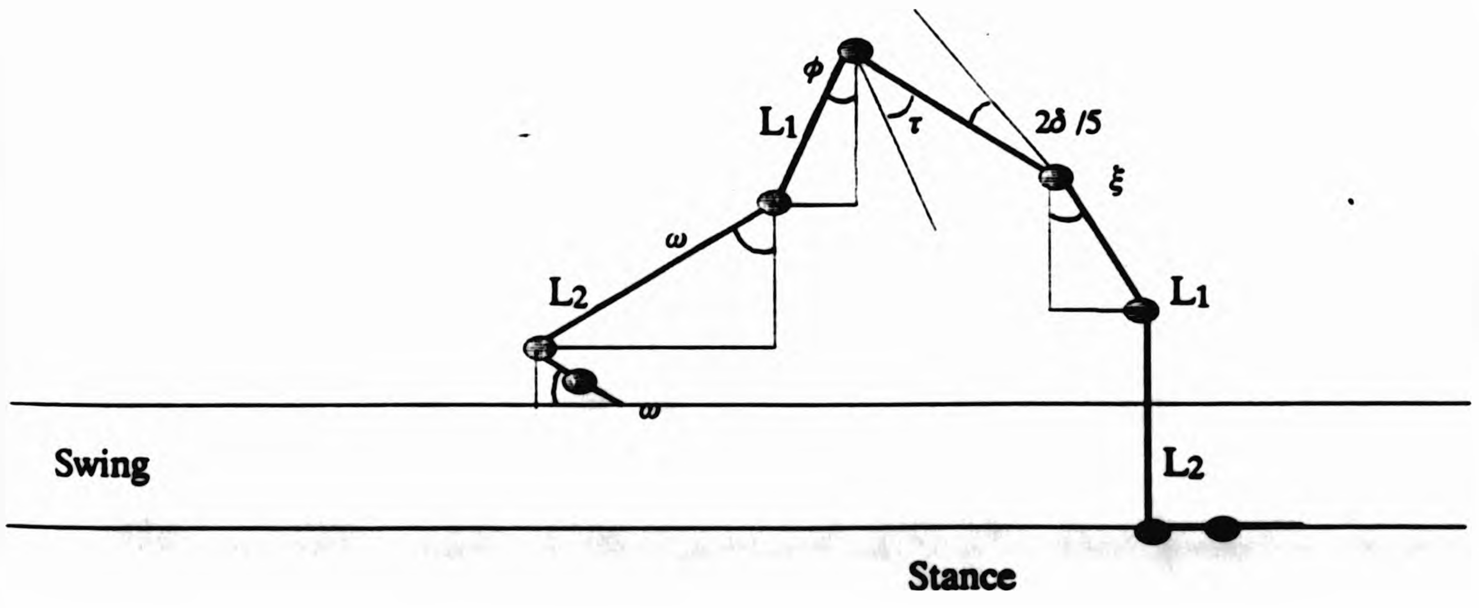


**Fig 8.3 : Position of the body after phase 2 of intermediate cycle.**

$$V_{st} = (L_1 + L_2)\cos\alpha \tag{8.1}$$
$$V_{sw} = (L_1 + L_2)\cos\beta + L_3\sin\beta \tag{8.2}$$

**Allowing for the descent owing to pelvic tilt :**

$$V_{sw} + L\sin\delta = V_{st} \tag{8.3}$$
$$(L_1 + L_2)\cos\beta + L_3\sin\beta + L\sin\delta = (L_1 + L_2)\cos\alpha \tag{8.4}$$

**Assuming that the stride is $S$, then from Fig 8.2 can be seen that :**

$$S = V_1 + V_2 + V_3 + L\sin\left(\frac{2\tau}{5}\right)$$

**Therefore :**

$$V_1 = (L_1 + L_2)\sin\alpha$$
$$V_2 = L_3\tan\beta\sin\beta$$
$$V_3 = (L_1 + L_2)\sin\beta$$

**Hence :**

$$S = (L_1 + L_2)\sin\alpha + L_3\tan\beta\sin\beta + (L_1 + L_2)\sin\beta + L\sin\left(\frac{2\tau}{5}\right) \tag{8.5}$$

**The root is the foot of the swing leg and the joint associated with the root is the toe. Notice that the pelvic rotation does not contribute to the vertical height of either legs. As a matter of fact the pelvic rotation is not involved in the vertical heights in any phase.**

***ii. Phase 2* : Fig 8.3 shows the position of the body after completion of phase 2. There are two assumptions for this phase :**

- **For the stance leg, the lower leg is vertical. This seems to be the case in actual walking as found from footage in Muybridge [55/1].**
- **For the swing leg, the ankle remains at a right angle throughout the motion. This condition is certainly true at the end of phase 2, McMahon[84] and Nilsson[85], and since the ankle of the swing leg starts its motion in phase 2 at right angles, it is reasonable to assume that it maintains this position throughout this phase.**

At the end of this phase, the swing leg is ready for liftoff and the stance leg has begun to flex at the knee. Angles $\omega$ and $\phi$ represent the liftoff angles of the swing leg and $\xi$ is the angle that the thigh of the stance leg makes with the vertical. The root is the foot of the stance leg and the joint associated with the root is the ankle. The body performs a negative pelvic tilt rotation $\frac{3\delta}{5}$ i.e it elevates towards the side of the swing leg and therefore the net value of the pelvic tilt is $\frac{2\delta}{5}$.

The vertical heights of the stance and swing legs $V_{st}$ and $V_{sw}$ are :

$$V_{st} = L_2 + L_1 \cos\xi \tag{8.6}$$

$$V_{sw} = L_1 \cos\phi + L_2 \cos\omega + (L_3 + L_4) \sin\omega \tag{8.7}$$

Allowing for the elevation owing to pelvic tilt :

$$L_1 \cos\phi + L_2 \cos\omega + (L_3 + L_4) \sin\omega + L \cos\left(\frac{2\delta}{5}\right) = L_2 + L_1 \cos\xi \tag{8.8}$$

*ii. Phase 3* : Fig 8.4 shows the position of the body at the end of phase 3. There are three assumptions during this phase :

- For the stance leg, the ankle and hip joints are vertically aligned. Physical evidence shows that at normal walking speeds the angles that the ankle and hip joints make with the vertical are small and, furthermore, they combine in effect to make the supposition accurate (Muybridge [55/1], McMahon [84] and Nilsson [85]).
- For the swing leg, the ankle joint remains at a right angle throughout the motion. This condition is also substantiated by researchers in physiology (Inman [81] and McMahon [84]).
- For the swing leg, the knee joint remains stiff i.e does not rotate at all. This condition is made in order to keep the mathematical complexity of the model minimal. Physical evidence suggest that there is a slight rotation at the knee during this phase, Muybridge [55/1].

At the end of this phase the swing leg is in the air swinging forward and the stance leg has completed its flexion at the knee. We assume that the lowest extremity of the swing leg (the tip of the toe) is at a vertical height $d$ from the ground.

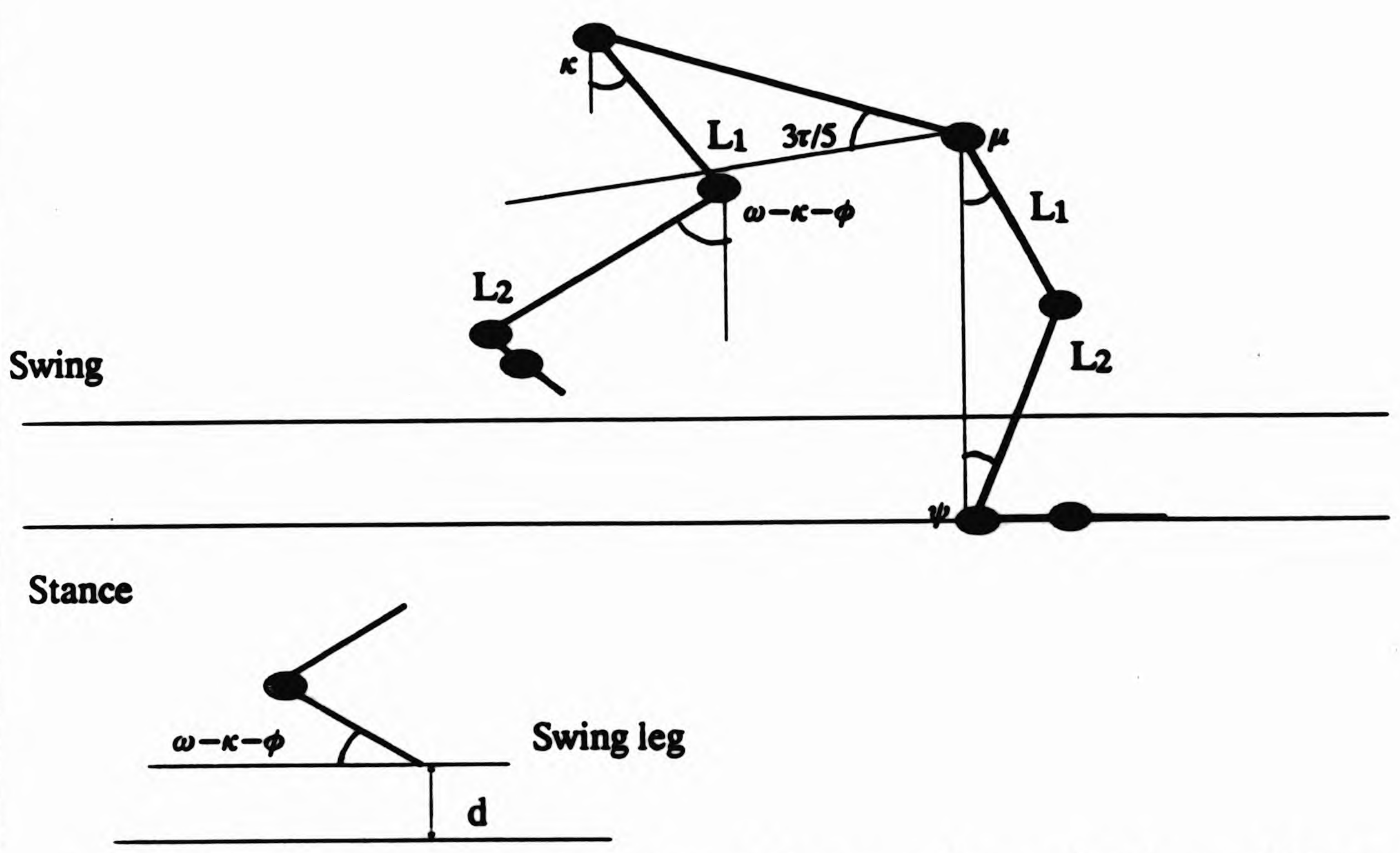


**Fig 8.4 : Position of the body after phase 3 of intermediate cycle. The small figure shows the vertical distance of the tip of the toe from the ground.**

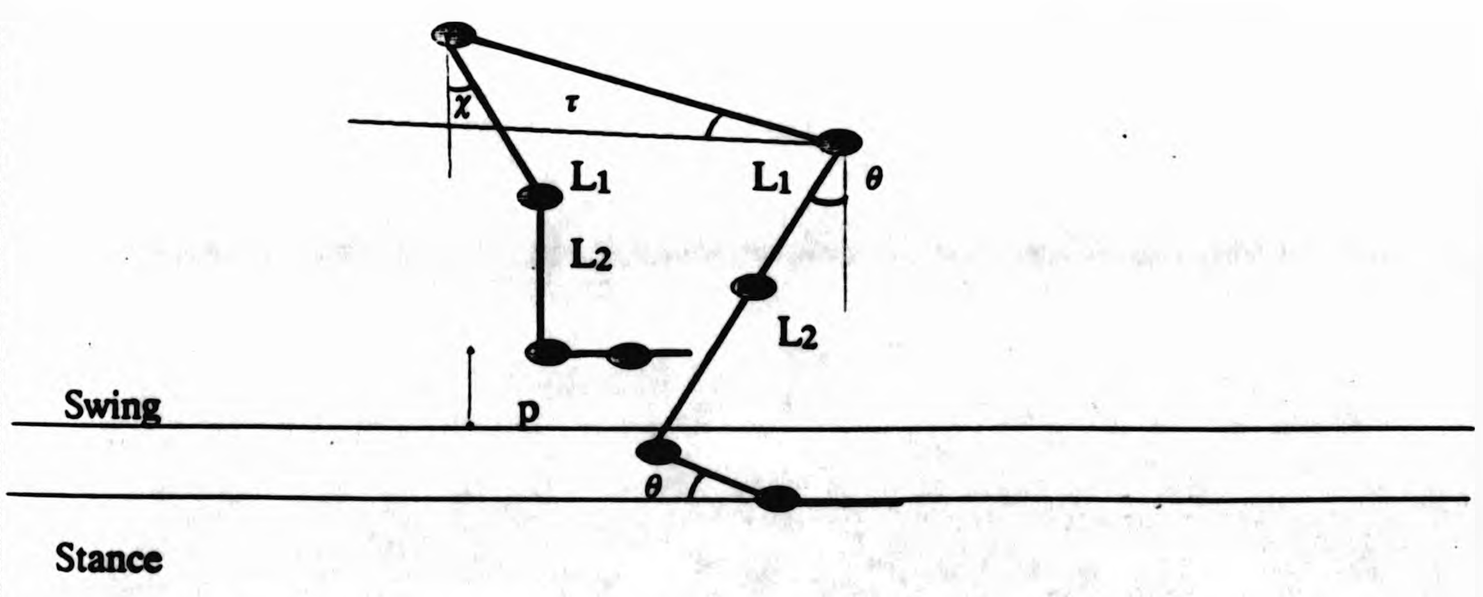


**Fig 8.5 : Position of the body after phase 4 of intermediate cycle.**

The pelvis performs a negative tilt by $\frac{2\delta}{5}$ towards the side of the swing leg and it, therefore completes its cycle. Although there is an elevation of the pelvis during the next phase (phase 4), as shown in the pelvic tilt curve in Fig 8.1, it is negligible and can be therefore absorbed by this phase. The pelvis rotates towards the side of the swing leg by $\frac{3\tau}{5}$. Three new angles are introduced here, $\kappa$ and $\mu$ are the angles that the thighs of the swing and stance legs make with the vertical respectively, and $\psi$ is the angle that the lower stance leg makes with the vertical (Fig 8.4).

The vertical heights of the stance and swing legs $V_{st}$ and $V_{sw}$ are :

$$\begin{aligned} V_{sw} &= L_1 \cos\kappa + L_2 \cos(\omega - \kappa - \phi) + (L_3 + L_4)\sin(\omega - \kappa - \phi) + d \quad (8.9) \\ V_{st} &= \left(L_1^2 - L_2^2 \sin^2\psi\right)^{1/2} + L_2 \cos\psi \quad (8.10) \end{aligned}$$

$\omega - \kappa - \phi$ is the angle that the lower swing leg makes with the vertical. The swing leg knee does not rotate during this phase and therefore it preserves the angle it has at phase 2 (which was $\omega$) but since the thigh of the swing leg rotates by $\phi + \kappa$ this affects the angle that the knee makes with the vertical. Therefore :

$$\begin{aligned} L_1 \cos\kappa &+ L_2 \cos(\omega - \kappa - \phi) + (L_3 + L_4)\sin(\omega - \kappa - \phi) + d = \\ \left(L_1^2\right. &- \left.L_2^2 \sin^2\psi\right)^{1/2} + L_2 \cos\psi \quad (8.11) \end{aligned}$$

The root is the foot of the stance leg and the joint associated with the root is the ankle.

*ii. Phase 4* : Fig 8.5 shows the position of the body after phase 4. There are two assumptions during this phase :

- For the stance leg, the ankle remains at a right angle throughout the motion. Physical evidence show that this angle is slightly smaller than $90^0$, but this condition is introduced in order to minimise the mathematical complexity.
- For the swing leg, at the end of this phase, the lower leg is vertical and, since the ankle remains at a right angle, this results in the sole of the swing foot being parallel with the ground. This is substantiated by evidence from Muybridge [55/1] and McMahon [84].

At the end of this phase the swing leg moves even more forward and the stance leg gets ready to become the swing leg of the next phase by starting to lift its ankle w.r.t the toe. There is no pelvic tilt during this phase. $\theta$ and $\chi$ are the angles that the swing and stance legs make with the vertical respectively. $p$ is the vertical distance that the tip of the toe has from the ground. The root is the foot of the stance leg and the joint associated with the root is the toe.

The vertical heights of the stance and swing legs $V_{st}$ and $V_{sw}$ are :

$$\begin{aligned} V_{sw} &= L_1 \cos\chi + L_2 + p && (8.12) \\ V_{st} &= (L_1 + L_2)\cos\theta + L_3 \sin\theta && (8.13) \end{aligned}$$

The two heights are equal since there is no pelvic tilt :

$$L_1 \cos\chi + L_2 + p = (L_1 + L_2)\cos\theta + L_3 \sin\theta \qquad (8.14)$$

Table 8.1 shows the rotations of the degrees of freedom that are involved during the intermediate cycle. One direction is assigned arbitrarily to be positive then the opposite direction will be negative. The angles indicated, reflect the rotations which are passed through the hierarchy by successive rotations at the joint, starting at the root joint (indicated in the first column of the table). Remember that the root link is the foot of the stance leg except for phase 1. The values of the pelvic rotation must be read carefully. During phases 1 and 2 the pelvis rotates towards the side of the stance leg i.e the hip joint of the swing leg rotates forward (positive direction). During phases 3 and 4 the pelvis rotates towards the side of the swing leg i.e the hip joint of the stance leg rotates forward (negative direction).

| | Pelvis | | Stance | | | | Swing | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | T | R | H | K | A | T | H | K | A | T |
| T | $\delta$ | $2\tau/5$ | $\alpha-\chi$ | $\chi$ | 0 | 0 | 0 | 0 | 0 | $\theta-\beta$ |
| A | $-3\delta/5$ | $3\tau/5$ | 0 | $\xi$ | $-\alpha$ | 0 | $\beta-\phi$ | $\phi-\omega$ | 0 | $-\beta$ |
| A | $-2\delta/5$ | $-3\tau/5$ | 0 | $\mu-\xi+\psi$ | $-\psi$ | 0 | $\kappa+\phi$ | 0 | 0 | 0 |
| T | 0 | $-2\tau/5$ | 0 | $-\mu-\psi$ | $\psi$ | $-\theta$ | $\chi-\kappa$ | $\omega-\chi-\phi$ | 0 | 0 |

Table 8.1 : Target angles for intermediate cycle. T and R stand for Tilt and Rotation respectively. H, K, A and T mean Hip, Knee, Ankle and Toe joints respectively. First row gives the rotations for phase 1, second row for phase 2 and so on.

## 8.3 Starting cycle

The starting cycle has two phases :

- Starting cycle phase 1 is identical to phase 3 of intermediate cycle.
- Starting cycle phase 2 is identical to phase 4 of intermediate cycle.

The starting cycle is designed as such in order to avoid the introduction of more angles to the model. It keeps the model quite general but it does not add more parameters. As a consequence of the above arrangement, the time that the body takes to complete the starting cycle is half the time that it takes for the intermediate cycle. Table 8.2 gives the target angles for this cycle.

| | Pelvis | | Stance | | | | Swing | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | T | R | H | K | A | T | H | K | A | T |
| A | $-2\delta/5$ | $-3\tau/5$ | 0 | $\psi+\mu$ | $-\psi$ | 0 | $\kappa$ | $\phi-\omega$ | 0 | 0 |
| T | 0 | $-2\tau/5$ | 0 | $-\mu-\psi$ | $\psi$ | $-\theta$ | $\chi-\kappa$ | $\omega-\chi-\phi$ | 0 | 0 |

Table 8.2 : Target angles for starting cycle. The same conventions as in Table 8.1 are followed here.

## 8.4 Finishing cycle

The finishing cycle has three phases :

- Finishing cycle phase 1 is identical to phase 1 of intermediate cycle.
- Finishing cycle phase 2 is identical to phase 2 of intermediate cycle.
- Finishing cycle phase 3 brings the body to rest and adds no new parameters.

The time duration of the finishing cycle is 75% of the time for the intermediate cycle. Table 8.3 gives the target angles for this cycle.

| | Pelvis | | Stance | | | | Swing | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | T | R | H | K | A | T | H | K | A | T |
| T | $\delta$ | $2\tau/5$ | $\alpha-\chi$ | $\chi$ | 0 | 0 | 0 | 0 | 0 | $\theta-\beta$ |
| A | $-3\delta/5$ | $3\tau/5$ | 0 | $\xi$ | $-\alpha$ | 0 | $\beta-\phi$ | $\phi-\omega$ | 0 | $-\beta$ |
| A | $-2\delta/5$ | $-\tau$ | 0 | $-\xi$ | 0 | 0 | $\phi$ | $\omega-\phi$ | 0 | 0 |

Table 8.3 : Target angles for finishing cycle. The same conventions as in Table·8.1 are followed here.

## 8.5 Motion constraints

The target angles have to obey some motion constraints in order for the generated motion to be realistic. There are six motion constraints for the described model which are explained here together with the relationships derived from them, and their physical meaning.

• *Constraint (i)* : $\alpha - \chi > 0$. This constraint is derived from phase 1 of the intermediate cycle and ensures that the swing leg stretches forward just before touchdown. Therefore, $\cos\alpha < \cos\chi$. From (8.4) and (8.14) :

$$\begin{aligned} L_1(L_1 &+ L_2)\cos\beta + L_1L_3\sin\beta + L_1L\sin\delta < \\ (L_1 &+ L_2)^2\cos\theta + (L_1+L_2)L_3\sin\theta - (L_1+L_2)(L_2+p) \end{aligned} \tag{8.15}$$

Let

$$\begin{aligned} A &= L_1(L_1+L_2) \\ B &= L_3L_1 \\ C &= (L_1+L_2)^2\cos\theta + (L_1+L_2)L_3\sin\theta - (L_1+L_2)(L_2+p) - LL_1\sin\delta \end{aligned}$$

then this results in :

$$A\cos\beta \ + \ B\sin\beta \ < \ C$$

Expressing this inequality in powers of $\sin\beta$ :

$$\left(A^2+B^2\right)\sin^2\beta - 2BC\sin\beta + \left(C^2-A^2\right) > 0 \tag{8.16}$$

Inequality (8.16) can be solved as a quadratic equation to find two roots $\beta_1$ and $\beta_2$ where $\beta_1 < \beta_2$. Also from (8.4) :

$$(L_1 + L_2)\cos\beta + L_3 \sin\beta + L\sin\delta < (L_1 + L_2)$$

Let $D = (L_1 + L_2)$, $E = L_3$ and $F = (L_1 + L_2) - L\sin\delta$ then the equation can be written as :

$$\left(D^2 + E^2\right)\sin^2\beta - 2EF\sin\beta + \left(F^2 - D^2\right) > 0 \qquad (8.17)$$

Solving (8.17) as a quadratic equation gives two solutions for $\beta$, $\beta_3$ and $\beta_4$ where $\beta_3 < \beta_4$. The simultaneous solution of (8.16) and (8.17) gives a value for $\beta$. This value can be found from table 8.4.

| $\beta_1$ | $\beta_2$ | $\beta_3$ | $\beta_4$ | Condition | Solution |
|---|---|---|---|---|---|
| +ve | +ve | +ve | +ve | $\beta_3 \le \beta_1$ | $\beta = \beta_3/2$ |
| | | | | $\beta_1 < \beta_3 < \beta_2$ | $\beta = \beta_1/2$ |
| | | | | $\beta_3 = \beta_2$ | $\beta = \beta_2$ |
| | | | | $\beta_3 > \beta_2$ | $\beta = \beta_1/2$ |
| -ve | +ve | +ve | +ve | $\beta_2 < \beta_3$ | $\beta = \beta_2 + (\beta_3 - \beta_2)/2$ |
| | | | | $\beta_2 = \beta_3$ | $\beta = \beta_3$ |
| | | | | $\beta_3 < \beta_2 \le \beta_4$ | $\beta = \beta_4$ |
| | | | | $\beta_2 > \beta_4$ | $\beta = \beta_2$ |
| +ve | +ve | -ve | +ve | $\beta_4 < \beta_1$ | $\beta = \beta_4 + (\beta_1 - \beta_4)/2$ |
| | | | | $\beta_4 = \beta_1$ | $\beta = \beta_1$ |
| | | | | $\beta_1 < \beta_1 \le \beta_2$ | $\beta = \beta_2$ |
| | | | | $\beta_4 > \beta_2$ | $\beta = \beta_4$ |
| -ve | +ve | +ve | +ve | $\beta_4 \le \beta_2$ | $\beta = \beta_2$ |
| | | | | $\beta_4 > \beta_2$ | $\beta = \beta_4$ |

Table 8.4 : Solution for $\beta$.

• *Constraint (ii)* : $\theta - \beta > 0$. This constraint is derived from phase 2 of the intermediate cycle. Its physical meaning is that the sole of the swing leg flexes just before liftoff. This determines the initial velocity of the swing leg. There are no expressions derived from this constraint.

• ***Constraint (iii)*** : $\xi - \alpha > 0$. This constraint is derived from phase 2 of the intermediate cycle. This means that the thigh of the stance leg moves forward so that the knee will start flexing. Hence $\cos\xi < \cos\alpha$. From (8.4) and (8.8) :

$$\begin{aligned} L_1(L_1+L_2)\cos\beta &+ L_1L_3\sin\beta + L_1L\sin\delta > \\ (L_1+L_2)(L_1\cos\phi &+ L_2\cos\omega + (L_3+L_4)\sin\omega + L\cos\left(\frac{2\delta}{5}\right) - L_2) \end{aligned} \quad (8.18)$$

Let :

$$\begin{aligned} A &= L_2(L_1+L_2) \\ B &= (L_1+L_2)(L_3+L_4) \\ C &= L_1(L_1+L_2)\cos\beta + L_1L_3\sin\beta + LL_1\sin\delta - L_1(L_1+L_2)\cos\phi \\ &- L(L_1+L2)\cos\left(\frac{2\delta}{5}\right) + L_2(L_1+L_2) \end{aligned}$$

Then this results to :

$$A\cos\omega \;+\; B\sin\omega \;<\; C$$

Expressing this inequality in powers of $\sin\omega$ :

$$\left(A^2+B^2\right)\sin^2\omega - 2BC\sin\omega + \left(C^2-A^2\right) > 0 \quad (8.19)$$

This inequality can be solved as a quadratic equation to find two roots $\omega_1$ and $\omega_2$ where $\omega_1 < \omega_2$. Also from (8.8) :

$$L_1\cos\phi + L_2\sin\omega + (L_3+L_4)\sin\omega + L\cos\left(\frac{2\delta}{5}\right) - L_2 < L_1$$

Let $D = (L_3+L_4)$, $E = L_2$ and $F = L_1 + L_2 - L\cos(2\delta/5) - L_1\cos\phi$ then the equation can be written as :

$$\left(D^2+E^2\right)\sin^2\omega - 2EF\sin\omega + \left(F^2-D^2\right) > 0 \quad (8.20)$$

Solving (8.21) as a quadratic equation gives two solutions $\omega_3$ and $\omega_4$ where $\omega_3 < \omega_4$. The simultaneous solution of (8.19) and (8.20) gives a solution for $\omega$. This solution can be found again from table 8.4 by reading $\omega$ instead for $\beta$.

• ***Constraint (iv)*** : $\phi - \beta > 0$. It is derived from phase 2 of the intermediate cycle. This constraint means that the swing stretches backwards before liftoff.

• *Constraint (v)* : $\omega - \phi > 0$. It is also derived from phase 2 of the intermediate cycle and it means that the lower swing leg bends backwards more than the thigh before liftoff.

• *Constraint (vi)* : $\chi - \kappa > 0$. It is derived from phase 4 of the intermediate cycle. Its significance is that the swing leg moves forward before touchdown. Hence $\cos\chi < \cos\kappa$ and from (8.11) and (8.14) $\Rightarrow$

$$\begin{aligned}(L_1 &+ L_2)\cos\theta + L_3\sin\theta - L_2 - p < (L_1^2 - L_2^2\sin^2\psi)^{1/2} \\ &+ L_2\cos\psi - (L_3 + L_4)\sin\epsilon - d - L_2\cos\epsilon \qquad (8.21)\end{aligned}$$

where $\epsilon = \omega - \kappa - \phi$. Let

$$M = (L_1 + L_2)\cos\theta + L_3\sin\theta - L_2 - p + (L_3 + L_4)\sin\epsilon + d + L_2\cos\epsilon$$

Then a minimum bound for $\psi$ is found to be :

$$\psi > \cos^{-1}\frac{M^2 - L_1^2 + L_2^2}{2ML_2} \qquad (8.22)$$

Also from (8.11) we can deduce a maximum bound for $\psi$ :

$$\psi < \cos^{-1}\frac{K^2 - L_1^2 + L_2^2}{2KL_2} \qquad (8.23)$$

where

$$K = L_1 + (L_3 + L_4)\sin\epsilon + d + L_2\cos\epsilon$$

The two inequalities (8.22) and (8.23) can be used to calculate $\psi$. It can be seen that this set of inequalities require a different type of solution from the one that was used for inequalities in this and the previous chapter. Table 8.5 gives the values for the system of inequalities.

| sign of value of $\psi_{min}$ | sign of value of $\psi_{max}$ | Solution for $\psi$ |
|---|---|---|
| +ve | +ve | $\psi = \psi_{min} + (\psi_{max} - \psi_{min})/2$ |
| -ve | +ve | $\psi = \psi_{max}/2$ |
| +ve | -ve | No solution |
| -ve | -ve | No solution |

Table 8.5 : Solution table for $\psi$.

## 8.6 Calculation of parameters

The model has fourteen parameters :

- the stride $S$
- the pelvic tilt angle $\delta$
- the pelvic rotation angle $\tau$
- angles $\alpha$, $\beta$ from phase 1
- angles $\phi$, $\omega$ and $\xi$ from phase 2
- angles $\kappa$, $\psi$ and distance $d$ from phase 3
- angles $\chi$, $\theta$ and distance $p$

We can now examine which relationships to use in order to calculated these quantities:

*i.* Given $\theta$, $\delta$ and $p$, then (8.15) and (8.17) can be used to calculate $\beta$.

*ii.* Given $\beta$, $\phi$ and $\delta$, then (8.19) and (8.20) can be used to calculate $\omega$.

*iii.* Given $\beta$ and $\delta$, use (8.4) to calculate $\alpha$ :

$$\alpha = \cos^{-1}\left(\frac{(L_1 + L_2)\cos\beta + L_3 \sin\beta + L\sin\delta}{(L_1 + L_2)}\right) \tag{8.24}$$

*iv.* Given $\alpha$ and $\beta$, use (8.5) to calculate $\tau$ :

$$\tau = \frac{5}{2}\sin^{-1}\left(\frac{S - (L_1 + L_2)\sin\alpha - L_3\tan\beta\sin\beta - (L_1 + L_2)\sin\beta}{L}\right) \tag{8.25}$$

*v.* Given $\phi$, $\omega$ and $\delta$, (8.8) can be used to calculate $\xi$ :

$$\xi = \cos^{-1}\left(\frac{L_1\cos\phi + L_2\cos\omega + (L_3 + L_4)\sin\omega + L\cos\left(\frac{2\delta}{5}\right) - L_2}{L_1}\right) \tag{8.26}$$

*vi.* Given $\theta$ and $p$, (8.14) can be used to calculate $\chi$ :

$$\chi = \cos^{-1}\left(\frac{(L_1 + L_2)\cos\theta + L_3\sin\theta - L_2 - p}{L_1}\right) \tag{8.27}$$

*vii.* Given $\omega$, $\kappa$, $\phi$, $d$ and $\theta$, then (8.22) and (8.23) can be used to calculate $\psi$.

Therefore we arrive to the conclusion that the user must supply the following parameters :

$$\theta,\ \delta,\ \phi,\ d,\ p,\ \kappa \text{ and } S$$

and the system calculates :

$$\alpha,\ \beta,\ \omega,\ \kappa,\ \psi,\ \xi,\ \chi \text{ and } \tau$$

Therefore, the user has to supply seven parameters in total i.e three distances and four angles.

## 8.7 Arm co-ordination

The co-ordination of the arms will be treated as in the previous model :

- The target angles of the left arm will be related to the target angles of the right leg.
- The target angles of the right arm will be related to the target angles of the left leg.

These two assumptions are substantiated by the photographic sequences in Muybridge, but one can see from his plates that although the overall behaviour of the arms is the same as the one described here, the are variations from one person to another (for the same walking speed) which suggest that the motion of the arms depend upon the personal style of walking of the individual. Therefore a reasonable approach is to adopt a parametrised model similar to the one used in the previous chapter. Four parameters are needed, $k_1$, $k_2$, $k_3$ and $k_4$, one for each phase of the intermediate cycle. This arrangement results to having 12 angles for the arms labelled $\alpha_1$ to $\alpha_{12}$. These angles are shown in Fig 8.6, 8.7, 8.8 and 8.9. The arm angles can be found from the leg angles by connecting them with the four coefficients :

$$\begin{array}{lll} \alpha_1 = k_1\alpha & \alpha_2 = k_1\beta & \alpha_3 = k_1\beta \\ \alpha_4 = k_2\xi & \alpha_5 = k_2\phi & \alpha_6 = k_2\omega \\ \alpha_7 = k_3\kappa & \alpha_8 = k_3\mu & \alpha_9 = k_3\psi \\ \alpha_{10} = k_4\chi & \alpha_{11} = k_4\chi & \alpha_{12} = k_4\theta \\ 0 \leq & k_1,\, k_2,\, k_3,\, k_4 & \leq 1 \end{array} \tag{8.28}$$

The rotations of the arms during each phase and each cycle are shown in Table 8.6. One direction is assigned positive and thus the other direction is negative.

The motion of arms is bound to constraints that restrict their values :

- Phase 1
  1. $\alpha_{12} - \alpha_1 > 0$
  2. $\alpha_2 - \alpha_{10} > 0$
  3. $\alpha_3 - \alpha_{11} > 0$
- Phase 2
  1. $\alpha_{11} - \alpha_4 > 0$
  2. $\alpha_2 - \alpha_5 > 0$
  3. $\alpha_3 - \alpha_6 > 0$
- Phase 3
  1. $\alpha_4 - \alpha_7 > 0$
  2. $\alpha_5 - \alpha_8 > 0$
  3. $\alpha_6 - \alpha_9 > 0$
- Phase 4 : No constraints.

The way that the values of the target angles are evaluated is such that most of these constraints are satisfied, but not all of them, and therefore not every set of leg angles will generate realistic motion for the arms.

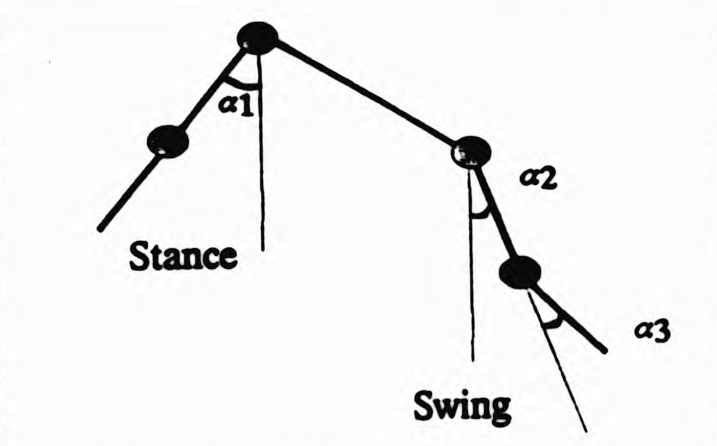


**Fig 8.6 : Target angles for phase 1 of intermediate cycle.**

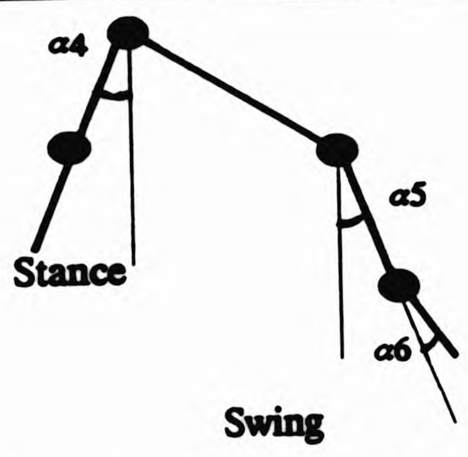


**Fig 8.7 : Target angles for phase 2 of intermediate cycle.**

**Fig 8.8 : Target angles for phase 3 of intermediate cycle.**

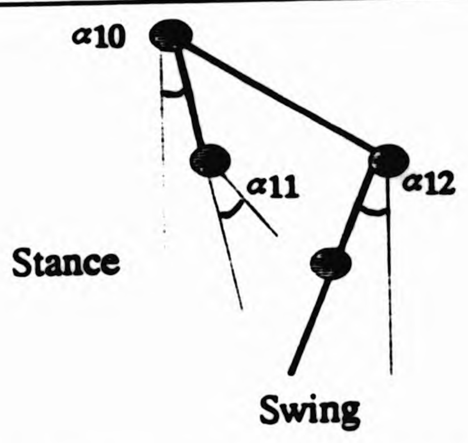


**Fig 8.9 : Target angles for phase 4 of intermediate cycle.**

| Cycle | Phase | Arm | Shoulder | Elbow |
|---|---|---|---|---|
| Intermediate cycle | Phase 1 | Stance | $\alpha_{12} - \alpha_1$ | 0 |
| | | Swing | $\alpha_2 - \alpha_{10}$ | $\alpha_3 - \alpha_{11}$ |
| | Phase 2 | Stance | $\alpha_{11} - \alpha_4$ | 0 |
| | | Swing | $-\alpha_2 + \alpha_5$ | $-\alpha_3 + \alpha_6$ |
| | Phase 3 | Stance | $\alpha_4 - \alpha_7$ | 0 |
| | | Swing | $\alpha_5 - \alpha_8$ | $-\alpha_6 + \alpha_9$ |
| | Phase 4 | Stance | $\alpha_7 + \alpha_{10}$ | $\alpha_{11}$ |
| | | Swing | $-\alpha_8 - \alpha_{12}$ | $-\alpha_9$ |
| Starting cycle | Phase 1 | Stance | $\alpha_7$ | 0 |
| | | Swing | $\alpha_8$ | $\alpha_9$ |
| | Phase 2 | Stance | $\alpha_7 + \alpha_{10}$ | $\alpha_{11}$ |
| | | Swing | $-\alpha_8 - \alpha_{12}$ | $-\alpha_9$ |
| Finishing cycle | Phase 1 | Stance | $-\alpha_1$ | 0 |
| | | Swing | $\alpha_2$ | $\alpha_3$ |
| | Phase 2 | Stance | $\alpha_{11} - \alpha_4$ | 0 |
| | | Swing | $-\alpha_2 + \alpha_5$ | $-\alpha_3 + \alpha_6$ |
| | Phase 3 | Stance | $\alpha_4$ | 0 |
| | | Swing | $-\alpha_5$ | $-\alpha_6$ |

Table 8.6 : Target angles for the arms.

## 8.8 Trunk movements

One issue that was not addressed by the model presented in the previous chapter was the motion of the trunk i.e the movement of the upper and lower torso. In that model the motion of the trunk was determined by the rotations that propagated from the root of the motion to the kinematic chains and thus the torso and lower body. Few researchers in physiology have investigated the motion of the trunk under locomotion for different walking speeds, Thorstensson [84 & 85] and Inman [81].

The motion of the trunk can be described by three factors :

1. Vertical movement which refers to an elevation and descendance of the torso

during walking, labelled $Vd$.

2. Lateral movement which refers to
   (a) a z-rotation of the torso w.r.t the lower body which makes the torso to lean sideways, labelled $\gamma$.
   (b) a z-rotation of the torso about the pelvis which causes the torso and lower body to move sideways, labelled $\nu$.
3. Sagittal movements which refer to an x-rotation of the pelvis and cause the torso to bend forward w.r.t the rest of the body and it is labelled $\zeta$.

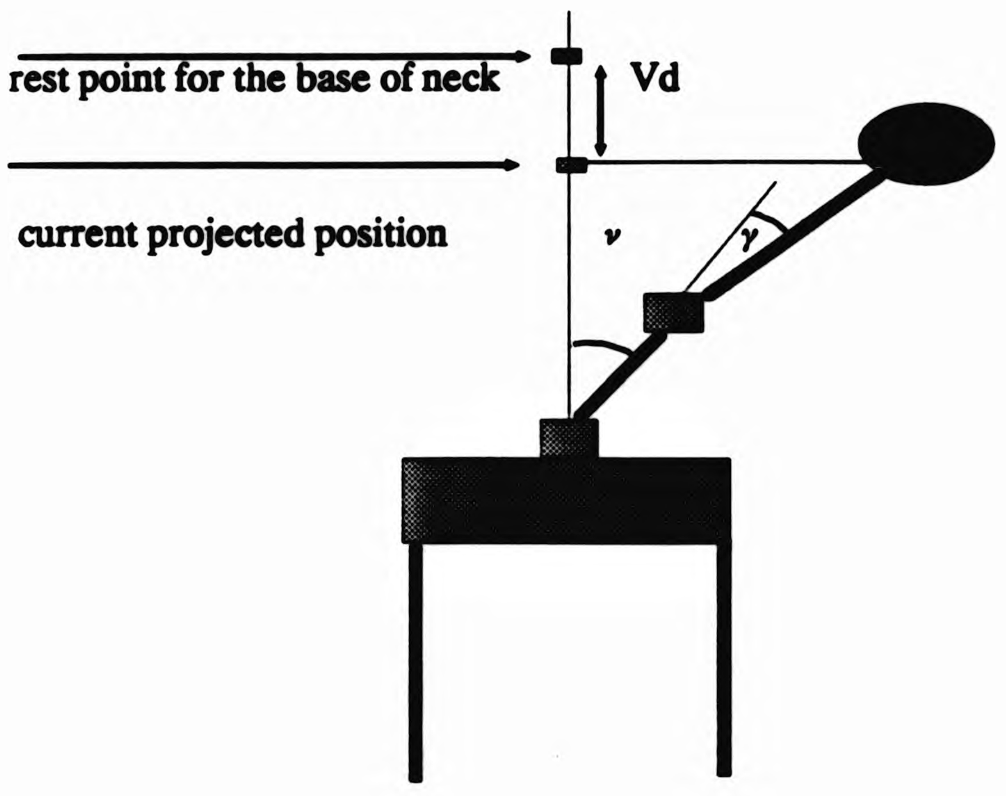
Fig 8.10(a) and (b) shows how these angles are defined. Fig 8.10(a) shows a rear view of the body and Fig 8.10(b) shows a side view of the body.

### 8.8.1 Vertical displacement

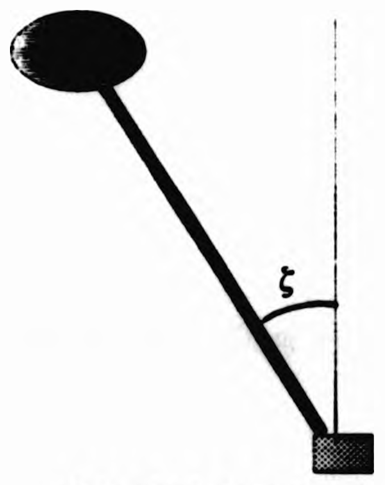
During a walking cycle, the trunk makes a full oscillation, that is, it reaches a maximum point (highest amplitude) and then it returns to its middle position and continues until it reaches its minimum value (lowest amplitude) which is achieved in the middle of the double support phase (see Fig 8.11), Thorstensson [84 & 85]. This can be interpreted in our model as the end of phase 1 (since the body is in double support phase in phases 1 and 2). The highest point of the oscillation occurs in the middle of the single support phase i.e end of phase 3. The motion can then be described as follows :

- End of phase 1, reach lowest descendance i.e $-Vd$.
- End of phase 2, return to middle position.
- End of phase 3, reach highest elevation i.e $Vd$.
- End of phase 4, return to middle position.

Experimental results in Thorstensson shown that the displacement of the neck from its middle position varies together with the forward velocity and this relationship is quadratic. The experiments were carried out by attaching an LED sensor to the base of the neck and measuring vertical displacements from its originally recorded value.



**(a) Rear view**



**(b) Side view**

**Fig 8.10 : Rear and side view of the human figure. (a) illustates the lateral trunk movements and (b) shows the sagittal rotation.**

From his experimental data we managed to produce a normalized relationship for the vertical displacement $Vd$ (see Fig 8.11):

$$Vd = -0.00012 \times V_f^2 + 0.068 \times V_f - 2.8 \tag{8.29}$$

$V_f$ is measured in cm per second and can be found from the normalised formula used in chapter 7, given the $sf$ :

$$\frac{V_f}{0.004 \times body\_height} = sf^2 \tag{8.30}$$

## 8.8.2 Lateral rotations

Both lateral angular displacements, $\gamma$ and $\nu$ have oscillatory behaviour and perform half an oscillation per cycle, i.e they both have period equal to two walking cycles or a walking period Nilsson [84] and Thorstensson [84 & 85]. The motion is somewhat complicated and can be described as follows (assuming the left leg to be the stance leg):

-At the end of phase 1, $\nu$ is slightly inclined to the right and $\gamma$ to the left but the entire trunk is on the left side of the middle position.
-At the end of phase 2, $\nu$ is inclined to the left where it obtains its maximum value and $\gamma$ is inclined to the left.
-At the end of phase 3, $\gamma$ is continuing to incline to the left where it obtains its maximum value at peak $\nu$ value.
-At the end of phase 4, $\nu$ starts to incline to the left at peak $\gamma$ inclination.

Experimental results from physiology has shown that $\gamma$ is almost invariant with walking speed and has amplitude of approximately $7^0 - 9^0$. Therefore, without loss of generality, $\gamma$ can be considered to have an average value $8^0$.

$\nu$ increases with velocity and its values ranges between $2^0$ and $5^0$. Its behaviour is quadratic and is given by the following formula (as calculated from experimental data found in Thorstensson):

$$\nu = -0.00005 \times V_f^2 + 0.035 \times V_f - 1.0 \tag{8.31}$$

The values of $\nu$ are given in degrees and $V_f$ in cm per second.

### 8.8.3 Sagittal rotation

The sagittal rotation results in the trunk leaning forward and backward during walking. The movement is oscillatory and completes one oscillation per walking cycle. At the end of phase 1, the rotation obtains its maximum amplitude and then during, phases 2 and 3, the trunk moves backwards until it reaches its minimum amplitude. During phase 4 it starts moving forward again. Hence, the motion can be described as :

- During phase 1, obtains maximum value $\zeta$.
- During phase 2, starts moving backwards.
- During phase 3, obtains minimum value $-\zeta$.
- During phase 4, starts moving forwards.

Research in physiology shows that $\zeta$ is invariant w.r.t velocity and its value ranges between $1.5^0 - 6^0$. The average value of $\zeta$ is about $3^0$.

### 8.8.4 Overall view of trunk movements

The motion of the trunk depends upon many overlapping factors, and any attempt of exact mathematical modelling is over ambitious. However, the above sections tried to give a model of the trunk movements as close to reality as possible while at the same time, keeping the mathematical complexity minimal. We discussed the behaviour of the four parameters for different walking speeds. The analysis showed that two of the parameters $\zeta$ and $\gamma$ are independent of walking speeds and can be fixed to average values that have been found from experimental data. The other two parameters $Vd$ and $\nu$ have shown positive correlation with walking speed.

$Vd$ represents the disposition of the base of the neck from its middle position during walking. This is the result of two factors :
- the pelvic tilt $\delta$ and
- lateral and sagittal rotations $\nu$, $\gamma$ and $\zeta$.

This observation can be used to relate $\nu$, $\gamma$, $\zeta$, $\delta$ and $Vd$, although, this relationship could be quite complicated since it has to consider rotations on different planes (sagit-

**tal and lateral). The existence of such a relationship has the importance consequence that there is a possibility for $\delta$ to be calculated given the step frequency only, since $Vd$ and $\nu$ can be found by (8.30) and (8.31) respectively, whereas $\zeta$ and $\gamma$ can be considered to have average values of about $3^0$ and $8^0$ respectively. Table 8.7 gives the target angles for the trunk movements. The values are obtained by experimental data and represent the real model as close as possible.**

| Cycle | Phase | Lateral rotations | | Sagittal rotation |
|---|---|---|---|---|
| | | Pelvis-Trunk | Torso-Lower Body | |
| Intermediate cycle | Phase 1 | $4\nu/5$ | $6\gamma/5$ | $+\zeta$ |
| | Phase 2 | $-6\nu/5$ | $2\gamma/5$ | $-\zeta$ |
| | Phase 3 | 0 | $4\gamma/5$ | $-\zeta$ |
| | Phase 4 | $2\nu/5$ | 0 | $+\zeta$ |
| Starting cycle | Phase 1 | $-\nu$ | $\gamma$ | $-\zeta$ |
| | Phase 2 | $2\nu/5$ | 0 | $+\zeta$ |
| Finishing cycle | Phase 1 | $4\nu/5$ | $6\gamma/5$ | $+\zeta$ |
| | Phase 2 | $-6\nu/5$ | $2\gamma/5$ | $-\zeta$ |
| | Phase 3 | $\nu$ | $-\gamma$ | 0 |

**Table 8.7 : Target angles for trunk movements.**

**Overall, the investigation of trunk movements not only did not introduce new parameters to the model since these can all be calculated from the normalized formulae introduced earlier, but it also helps us to calculate the pelvic tilt angle $\delta$. This is mostly due to the available experimental data about the movements of the trunk during human locomotion.**

## 8.9 Conclusions

**This chapter presented an extended model for human walking which was based upon improvements of the model presented in Chapter 7. This model considers five out of the six gait determinants involved in human locomotion : *compass gait*, *stance leg knee flexion*, *pelvic tilt*, *pelvic rotation* and *plantal flexion of the swing ankle*. As a**

consequence of these extension, the human model used must include toes and therefore it was assumed that each feet had a single toe, contributing thus to the introduction of two more links to the model. This model is considered as a feasibility study of a bipedal walking model with toes since its implementation has not been included within the current animation system due to limited time resources. The model serves as a platform for further research to the subject as it is the first time in computer animation that a toe model, for bipedal locomotion, is investigated.

The concept of the root of the motion also had to change as now it is not only a link but it is assigned a joint as well. This has two consequences :

- the root changes within a walking cycle and
- the root is always the foot of the swing or the stance leg and the joint associated with it, is either the toe or the ankle of that foot.

Notice that the model presented in Chapter 7 had the root fixed to the foot of the stance leg and the root remained the same within a cycle.

The number of cycles remained the same with respect to the original model but the number of phases increased in order to accommodate the new gait determinants, and so the number of phases are two, four and three for the starting, intermediate and finishing cycles respectively. As it turned out the root has to be assigned to the foot of the stance leg except in phase 1 of the intermediate and finishing cycles where it is assigned to the foot of the swing leg. This is because during phase 1 both legs are in contact with the ground and any of the legs can be the root but by assigning the root to the swing leg simplifies the accumulated rotations as it would much more difficult to start rotations from the stance leg as shown in Fig 8.2.

The analysis of the model and its motion constraints indicated that the user has to specify seven parameters for controlling the motion of the legs whereas the total number of leg parameters are fourteen. The investigation of the arms were carried out in a similar way to that of the original model, linking left leg angles with right arm angles and right leg angles with left arm angles. This resulted in the use of four parameters $k_1$ to $k_4$ for determining arm angles.

A new element of this model is the examination of trunk movements during walking.

Although there is considerable research in physiology about this subject, the format of results are not directly applicable to computer animation and thus it was necessary to introduce adaptations that made the data meaningful for the computer model. It turned out that the movement of the trunk is important to walking and that its behaviour is somewhat complicated. The investigation showed that four parameters can be used to specify trunk movements for locomotion, but further analysis resulted in relating these parameters to walking speed thus allowing them to be directly evaluated from the forward velocity of walking. Therefore the inclusion of trunk movements to the model did not necessitate the introduction of more user-prescribed parameters. On the contrary, using the experimental data we were able to express the pelvic tilt in terms of the walking speed and thus reducing the number of leg parameters from seven to six.

The presented model exhibits coherence between the three set of parameters (legs, arms, trunk). It shows that toes can be included to the model and its most important aspect is the low number of user-prescribed parameters (ten in total, six for the leg and four for the arms). The next task is to implement this model and investigate how it behaves for the purposes of human computer animation. This constitutes a target for further research as the model has not implemented. Some of the assumptions that were made in the course of the investigation of this model were imposed purely from the necessity to minimize the number of parameters. Although these assumptions are not far from reality, we do not know how they affect bipedal walking. Therefore an implementation of the model should investigate those issues.

The model can be further improved by including the sixth gait determinant i.e *lateral displacement of the pelvis*. Other issues that can be investigated within the subject of walking include, the adaptation of step length for rough terrain locomotion, Hodgins [91] and adaptations of gaits during variable speed walking, Grillner [79]. Another issue is that of body orientation, i.e turning the body, or walking along arcs.

The investigation in this chapter indicates that the model outlined is viable within the context of computer animation. Data is coming at an increasing rate from biomechanics, physiology and robotics which can be helpful to researchers in the field. The subject is a fruitful area of research and there are great possibilities for extending this approach to consider other forms of locomotion.

# Chapter 9

# Epilogue

## 9.1 Solaris : An interactive environment for dynamic analysis

The ideas presented in this thesis were applied into the construction to an interactive environment for utilizing dynamic analysis (Solaris). The software system is designed around two concepts :

- It is constructed of layers of motion control functions.
- It is entirely interactive.

These two design decisions lead to a windowing system within which the user can specify motions. The system consists of two motion layers of increasing complexity. The first layer contains routines for controlling individual degrees of freedom and the second is used to specify bipedal walking.

The system can operate on any articulated body as long as it is specified as a hierarchy stored in a file called the *configuration file*. If no file is specified, the system uses a human model with 63 degrees of freedom which was the figure used most during this research. However, if the user wants to specify a different model, a subcomponent of the system, called the *configuration specification component* can be used to generate the configuration file. The interface of this component is also window-based and the user can specify the hierarchical structure of the model as well as the physical quantities needed for the dynamic simulation of the model.

Once the model has been read by Solaris, the user can perform one of the following tasks :

- Make modifications to the dynamic quantities(order of rotation, inertia tensors, masses) as they are all accessible from the interface.
- Initiate dynamic analysis
- Specify boundary conditions to the problem i.e initial and/or final velocities and accelerations.
- Initiate the human walking function.
- View graphical output

The user can also assign attributes to the motion such as inclusion of ground reaction forces, assigning time-torque profiles and freezing degrees of freedom. Walking operates under two submodes, automatic and manual. In automatic walking mode, the motion constraints are used to calculate the parameters whereas, in manual mode the user specifies all parameters explicitly. When dynamic analysis is initiated, control is turned to the dynamics subcomponent which calculates the forces. The user can view the results by starting the graphics option. The graphics operate in three submodes :

- Viewing them on the screen (in 25 or 30 frames per second).
- Storing the frames into bitfiles.
- Storing the frames into vector-files.

These two types of files can also be played back onto the screen by using utilities of the system.

The kinematic component of the system (Nostromo) is used to specify kinematic profiles to degrees of freedom of the model, whenever this is desirable. It communicates with Solaris by a file that contains the kinematics information.

## 9.2 The problematics of dynamic analysis

The application of dynamics to human bodies presents a number of problems that do not occur in conventional methods such as kinematics. These problems are inherent in the method due to the use of physical laws. These problems are listed below together with some proposed solutions :

*i.Inertia tensors* : The implementation and testing of inverse Lagrangian dynamics leads to the conclusion that inertia tensors and masses are of paramount importance to the form of motion produced. The values of the masses of links were deduced from data in anthropometrics, but not the inertia tensors. The problem is that there is no recorded data that will help to find the inertia tensors of the human body. The approach used in the thesis was to find the average dimensions of each link of the body and calculate the tensors by assuming that the links have uniform mass distribution and box-like shape. This is rather simplistic but given the lack of concrete data, it seems a reasonable approach. The original calculations lead to small values of the inertia tensors which in certain instances of the simulation proved inadequate as they resulted in large acceleration values. After experimentation, we found that when the tensors are multiplied by a scalar value of between 5 and 8, the system behaves more realistically. Although this may look like a distortion of the physical properties of the model, it is not, because the original model for calculating the tensors did not represent reality i.e links have non-uniform mass distribution and irregular shape. Another explanation for this behaviour may be that the scaling of the inertia tensors represents a change in the physical units of the model i.e from centimetres to metres. The values of the masses were more or less correct. This explains why the value used for $I_{00}$ may be different from the value of the mass of the body, although in reality they should be the same. Finally, early attempts for calculating the inertia tensors used data from Armstrong's system, but these proved unsatisfactory within our working model.

*ii.Determination of initial velocities and accelerations* : One of the major problems in dynamic analysis is a lack of knowledge of the initial conditions of the problem. The animator would know the final position that the body has to achieve at the end of the motion but may not have detailed information about the individual initial velocities and accelerations of each link unless the body starts from rest. As a general rule we found that greater initial velocities and accelerations result in finding greater forces during the process of dynamic analysis. It was found that initial accelerations can in general be set to zero, and faster transitions between frames can be achieved by increasing the values of initial velocities only. However there is no way of determining how fast the body will reach its end position with respect to its initial velocity condition but motion profiles can be used to control the speed of the body in in-between frames i.e determining the curves of velocity and acceleration during the motion.

*iii.Ground reaction forces and other external forces* : The inclusion of external forces within the model requires a knowledge of their values. The most important external forces proved to be the ground reaction forces that provide balance for the body. Previous methods used an amount of guesswork in order to predict what these forces will be. The proposed model here used an analytical solution to the problem by re-formulating the Lagrangian dynamics from inverse to direct formulation. The direct formulation solves only for the three translational degrees of freedom attached to the root of the motion. The solution proved accurate and the results quite realistic, although the computational time increased by a factor of 3. Extensions of this idea can be applied to the inclusion of other external forces such as wind pressure and weight bearing. Although in the the course of calculating ground reaction forces we solve only for translational degrees of freedom, the method can be re-arranged to find the torques applied to other degrees of freedom as well.

## 9.3 Advantages of dynamic analysis

The problems presented above are not unique to dynamic analysis, but their manifestation is. For example, the use of kinematics assumes that the animator knows how these factors affect the motion and tries to allocate matural-looking positions to degrees of freedom either by direct assignment or use of techniques such as rotoscoping. The use of dynamic analysis moves these problems from the animator to the implementer of the system, and this means that all these problems should be solved during design and implementation and not during usage of the system. In this way, dynamic analysis offers an attractive solution to the animator.

Dynamic analysis also embodies the capability of building more 'intelligent' motion control systems that are no longer passive but can interact with the environment. The model can be tuned to react to changes to its environment and physical properties.

The use of research findings in physiology and biology can be used in conjunction with dynamics to automate common human activities such as walking. This approach offers a high level of automation i.e from the specification of few parameters, realistic motion sequences can be produced. The animator can specify walking by using a handful of parameters that are meaningful and well-understood variants of human locomotion. This technique can be extended to other types of motion such as running, grasping objects and other more specialised physical activities.

## 9.4 Conclusions

The modelling of motion of articulated bodies and humans in particular is a complicated task for a number of reasons. We have a limited knowledge of the mechanics of human motion, and the physical properties lead to complex equations of motion. Our limited knowledge of modelling the motion of humans and other articulated bodies is a product of research in physiology and medicine. Robotics also presents a large source of information about the motion equations of articulated bodies.

The use of dynamic analysis is one approach out of the plethora of methods in motion control. It exploits concepts from robotics and physics to model the motion of articulated bodies as well other rigid or flexible objects. The computational cost of dynamics is greater in comparison to other methods in motion control but it offers two major advantages :

- More realistic motion
- Possibility of automating the motion control

Both improvements are of great importance to the animation of articulated bodies. Few animation sequences have been produced involving articulated bodies and humans in particular because the process is very laborious and long. Because humans, mammals, birds and other objects such as robots, are articulated bodies, their animation is important in the advertising and entertainment industry.

The method used in this thesis is based on inverse Lagrangian dynamics, that is, the user supplies the accelerations, velocities and positions of the body and a solution is obtained for the forces. To the author's knowledge, this is the first time that such an approach is used for motion control of articulated bodies as previous attempts to model articulated bodies used direct dynamics i.e the user provides the forces and solves for accelerations, velocities and positions. The method produces realistic motion and sufficiently general to accommodate any articulated body that can be expressed in a hierarchical structure. The method can also be re-formulated to calculate the effect of ground reaction forces as well as other external forces. The application of dynamics to bipedal locomotion showed positive results and the model can be extended to more sophisticated locomotion patterns as well as to other types of locomotion.

## 9.5 Future extensions and trends

Dynamic analysis on its own, does not present a complete solution to the task of motion control of articulated bodies. It can, however, be used together with findings from other disciplines to generate realistic motion. In this thesis we applied dynamic analysis to bipedal walking and showed that the resulting motion was not only realistic but easy to generate as well.

Bipedal locomotion still offers some interesting points for research such as, the study of variable-speed walking. Another issue is locomotion on rough terrains where the body has to adjust its walking style according to the texture and shape of the terrain. The next level of complexity is obstacle detection and avoidance. This problems may necessitate the use of other principles such as robotics and expert systems but at the lower level of motion should be calculated using dynamics. It is therefore our belief that an animation system for articulated bodies should be built in layers of increasing complexity.

Other areas for investigation include the study of other types of motion such as run-

**ning, grasping objects, jumping, etc. The major problem with any type of motion is converting the high level motion specification to a low level specification i.e accelerations, velocities and positions, that can be supplied to dynamics for calculating the forces. The subject remains a fruitful area for research.**

# Bibliography

| | |
|---|---|
| [Alexander 83] | R.M Alexander : *Animal Mechanics.* Blackwell scientific publications, 1983, pp 1-36. |
| [Armstrong 87/1] | W.W Armstrong, M.Green, R.Lake : *Near real time control of human figure models.* IEEE Computer Graphics and Applications, June 1987, pp 52-61. |
| [Armstrong 85] | W.W Armstrong, M.W Green : *The dynamics of articulated rigid bodies for purposes of animation.* The Visual Computer, 1985, 1, pp 231-240. |
| [Armstrong 87/2] | W.W Armstrong, T.A Marsland, M.Olafsson, J.Schaeffer : *Solving equations of motion on a virtual tree machine.* SIAM Journal of Scientific and Statistical Computing, Vol 8, No 1, January 1987 pp s59- s72. |
| [Badler & Smoliar 87] | N.Badler, S.W.Smoliar : *Digital representation of human movement.* Computing Surveys, Vol 11, No 1, March 1987, pp 19 - 38. |
| [Badler 83] | N.Badler : *Motion graphics description and control.* Proceedings of the ACM SIGGRAPH/SIGART, Interdisciplinary workshop on Motion : Representation and Perception, Toronto, Ontario 1983 pp 295-302. |
| [Badler 87] | N.Badler, K.H.Manoochehri, G.Walters : *Articulated figure positioning by multiple constraints.* IEEE Computer Graphics and Applications, June 1987, pp 28-37. |

| | |
|---|---|
| [Badler 91] | N.I.Badler, B.L.Webber, J.Kalita, J.Esakov : *Animation from instructions.* Make Them Move, Kaufmann, Los Altos, California 1991, pp 51-93. |
| [Baraff 89] | D.Baraff : *Analytical methods for dynamic simulation of non-penetrating rigid bodies.* Computer Graphics, Volume 23, No 3, July 1989, pp 223-231. |
| [Barzel & Barr 88] | R.Barzel, A.H.Barr : *A modelling system based on dynamic constraints.* ACM Computer Graphics, Vol 22, No 4, pp 179-188, 1988. |
| [Bendal 69] | J.R Bendal : *Muscles molecules and movement.* Heinemann Education Books, 1969, pp 112-136. |
| [Brotman 88] | L.S.Brotman, A.N.Netravali : *Motion interpolation by optimal control.* IEEE Computer Graphics and Applications, Aug 1988, Vol 22, No 4, pp 309-315. |
| [Bruderlin 89] | A.Bruderlin, T.W.Calvert : *Goal-directed dynamic animation for human walking.* Computer Graphics, Vol 23, No 3, pp 233-241, 1989. |
| [Calvert 91] | T.Calvert : *Composition of realistic animation sequences for multiple human figures.* Make Them Move, 1991 Kaufmann, Los Altos, California pp 35-50. |
| [Cameron 85] | S.Cameron : *A study of the clash detection problem in robotics.* Proceedings, IEEE Conference in Robotics and Automation 1985 pp 448-493. |
| [Canny 87] | J.F.Canny : *The complexity of robot motion planning.* PhD Dissertation, MIT, 1987. |
| [Cotsaftis 88] | M.Cotsaftis, C.Vibet : *Lagrange formalism decoupling method and control law generation.* International Journal of Robotics and Automation, Vol 3, No 2, 1988, pp 86-89. |
| [Croney 71] | J.Croney : *Anthropometrics for designers.* B.T.Batsford Ltd, London 1971. |

[Croney 80] J.Croney : *Anthropometry for designers.* The Anchor Press Ltd, Essex, England, 1980.

[Dagg 77] A.I.Dagg : *Running, walking and jumping, the science of locomotion.* Wykeham Publications, 1977, pp 29 -47.

[Faugeras 86] O.D.Faugeras, M.Hebert : *The representation, recognition and location of 3D objects.* The International Journal of Robotic Research, 1986, Vol 5, No 3, pp 27-71.

[Featherstone 83] R.Featherstone : *The calculation of robot dynamics using articulated body inertias.* The International Journal of Robotics Research Vol 2, No 1, 1983, pp 13-30.

[Ginsberg 83] C.M.Ginsberg, D.Maxwell : *Graphical marionette.* Proceedings of the ACM SIGGRAPH/SIGART, Interdisciplinary workshop on Motion : Representation and Perception, Toronto, Ontario 1983 pp 303-310.

[Girard 87] M.Girard : *Interactive design of 3D computer animated legged animal motion.* IEEE Computer Graphics and Applications, June 1987, pp 39-51.

[Girard 91] M.Girard : *Constrained optimization of articulated animal movements in computer animation.* Make Them Move, Kaufmann, Los Altos, California 1991, pp 209-222.

[Glazzard 87] N.Glazzard : *Particle systems : methods, implementation and experiences.* Computer Graphics 87, On-Line London, 1987.

[Grillner 79] S.Grillner, J.Harbertsma, J.Nilsson, A.Thorstensson : The adaptation to speed in human locomotion. Brain Research, 165(1979), pp 177-182.

[Devanit & Hartenberg 55] J.Devanit, R.Hartenberg : *A kinematic notation for lower-pair mechanisms based on matrices.* Journal of Applied Mechanics, 1955, pp215-221.

[Hashimoto 89] K.Hashimoto, H.Kimura : *A new parallel algorithm for inverse dynamics.* The International Journal of Robotics Research, 1989, Vol 8 No 1, pp 63-76.

[Hahn 88] J.K.Hahn : *Realistic animation of rigid bodies.* IEEE Computer Graphics and Applications, Aug 1988, Vol 22, No 4, pp 299-308.

[Hodgins 91] J.K.Hodgins, M.H.Raibert : Adjusting step length for rough terrain locomotion. IEEE Transactions on Robotics and Automation, Vol 7, No 3, 1991.

[Hollerbach 80] J.Hollerbach : *A recursive lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity.* IEEE Transactions on Systems, Man, and Cybernetics, Vol SMC-10, No 11, November 1980, pp 730-736.

[Hollerbach 89] J.M.Hollerbach : *Kinematics and dynamics for control.* Robotics Science, Edited by M.Brady, The MIT Press, 1989.

[Inman 81] V.T.Inman, H.T.Ralston, F.Todd : Human walking. Williams and Wilkins. Baltimore 1981.

[John 89] N.W.John, P.J.Willis : *The controller animation system.* 7th Annual Eurographics(UK) Conference, Manchester, March 1989, pp 85-95.

[Kant 86] K.Kant, S.W.Zucker : *Towards efficient trajectory planning : the path-velocity decomposition.* The International Journal of Robotics Research, 1986, Vol 5, No 3, pp 72-89.

[Kochanek 84] D.H.U.Kochanek : *Interpolating splines with local tension, continuity, and bias control.* SIGGRAPH 84, Conference Proceedings, 1984, Minneapolis, Minnesota, pp 33-41.

[Lansdown 85] J.Lansdown : *Object and movement description techniques for animation, an informal review.* Fundamental Algorithms for Computer Graphics, Editor R.A.Earnshaw 1985, NATO ASI series, VOl F17, pp 1029-1036.

[Li 88] Chang-Jin Li : *A fast computational method of Lagrangian dynamics for robot manipulators.* International Journal of Robotics and Automation, Vol 3, No 1, 1988, pp 14-20.

[Luh 79] J.V.S Luh, M.W Walker, R.P Paul : *Newton-Euler formulation of manipulator dynamics for computer control.* 2nd IFAC/IFIP Symposium, Information Control Problems in Manufacturing Technology, Stuttgart, 1979, pp 165-172.

[Mason 86] T.M.Mason : *Mechanics and planning of manipulator pushing Operations.* The International Journal of Robotics Research, Vol 5, No 3, 1986, pp 53-71.

[McGovern 87] T.McGovern : *The use of live-action footage as a tool for the animator.* Tutorial Notes In Character Animation, SIGGRAPH, 1987.

[McMahon 84] T.A.McMahon : *Mechanics of Locomotion.* The International Journal of Robotics Research, Vol 3, No 2, 1984, pp 4-28.

[McMahon 87] T.A.McMahon : *Muscles, reflexes and locomotion.* Princeton University Press, 1987.

[Miller 88] G.S.P.Miller : *The motion dynamics of snakes and worms.* ACM, Computer Graphics, Vol 22, No 4, pp 169-177, 1988.

[Mohamed 88/1] A.S Mohamed, W.W Armstrong : *Measuring learning progress in intelligent autonomous robots.* Third Annual

Rocky Mountain Conference on Artificial Intelligence, Denver, Colorado, 1988, pp 280-295.

[Mohamed 88/2] A.S Mohamed, W.W Armstrong : *A hybrid numerical knowledge based locomotion control system for a multi-legged manipulator robot.* Eighth International Workshop in Expert Systems and their Applications, Paris, 1988, pp 511-529.

[Mohamed 89] A.S Mohamed, W.W Armstrong : *Towards a computational theory for motion understanding : The expert animators model.* Fourth Conference on Artificial Intelligence for Space Applications, Alabama, 1989, pp 289-302.

[Moore & Wilhelms 88] M.Moore, J Wilhelms : *Collision detection and response for computer animation.* IEEE Computer Graphics and Applications, Aug 1988, Vol 22, No 4, pp 289-297.

[Muybridge 55/1] E.Muybridge : *The human figure in motion.* Dover 1955.

[Muybridge 55/2] E.Muybridge : *Animals in motion.* Dover 1955.

[Nilsson 85] J.Nilsson, A.Thorstensson, J.Halbertsma : Changes in leg movements and muscle activity with speeds of locomotion and mode of progression in humans. Acta Physiol Scand 1985, pp 457-475.

[Olafsson & Marsland 85] M.Olafsson, T.A.Marsland : *A UNIX based virtual tree machine.* Proceedings of the Canadian Information Processing Society (TIPS) Congress, Montreal 1985, pp 176-181.

[Raibert 78] M.H Raibert, B.K.P.Horn : *Manipulator control using the configuration space method.* The Industrial Robot, June 1978, pp 69-73.

[Perez 79] T.Lozano-Perez, M.Wesley : *An algorithm for planning collision-free paths among polyhendra obstacles.* Communications of the ACM, Vol 22, No 10, pp 560-570, 1979.

[Perez 84] T.L.Perez, M.T.Mason, R.H.Taylor : *Automatic synthesis of fine motion strategies for robots.* The International Journal of Robotics Research, Vol 3, No 1, 1984, pp 3-24.

[Price 71] W.L.Price : *Graphs and networks.* Butterworths, Operational Research Series.

[Reynolds 87] C.W.Reynolds : *Flocks, herds and schools : A distributed behavioral model.* ACM, Computer Graphics, Vol 21, No 4, 1987.

[Schwartz & Shariz 87] J.Schwartz, M.Shariz : *Planning, geometry and complexity of robot motion.* Edited by J.Schwartz, M.Shariz and J.Hopcroft, Ablex publishing corp, New Jersey, 1987, Chapter 5, pp 154-186.

[Silver 81] W.Silver : *On the equivalence of Lagrangian and Newton-Euler dynamics.* Proc. 1981 Joint Automatic Conference, Green Valley, Arizona, American Automatic Control Council, Paper no TA-2A.

[Song 87] S.M.Song, J.K.Waldron : *An analytical approach for gait study and its applications to wave gaits.* The International Journal of Robotics Research, 1987, Vol 6, No 2 pp 60-71.

[Steketee & Badler 85] S.Steketee, N.Badler : *Parametric keyframe interpolation incorporating kinetic adjustment and phrasing control.* ACM SIGGRAPH, San Francisco, 1985, Vol 19, No 3, pp 255-262.

[Thalmann 89] N.M.Thalmann, D.Thalmann : *The problematics of human prototyping and animation.* 7th Annual Eurographics(UK) Conference, Manchester, 1989

| | |
|---|---|
| [Thalmann 87] | N.M.Thalmann, D.Thalmann : *The direction of the synthetic authors in the film : Rendezvous a Montreal.* IEEE Computer Graphics and Applications, December 1987, pp 9-19. |
| [Tost & Pueyo 88] | D.Tost, X.Pueyo : *Human body animation - a survey.* The Visual Computer, 1988, 5, pp 254-264. |
| [Thorstensson 84] | A.Thorstensson, J.Nilsson, H.Carlson : Trunk movements in human locomotion. Acta Physiol Scand 1984, 121, pp 9-22. |
| [Thorstensson 85] | A.Thorstensson, L.Oddsson, H.Carlson : Motor control of voluntary trunk movements in standing. Acta Physiol Scand 1985, pp 309-321. |
| [Stepanenko 88] | Y.Stepanenko : *Dynamic robot models.* International Journal of Robotics and Automation, Vol 3, No 1, 1988 pp 1-8. |
| [Walker 82] | M.W.Walker, P.E.Orin : *Efficient dynamic computer simulation of robotic mechanisms.* Journal of Dynamic Systems Measurement and Control, September 1982, Vol 104, pp 205-211. |
| [Wilhelms & Barsky 85] | J.Wilhelms, B.Barsky : *Using dynamic analysis to animate articulated bodies such as humans and robots.* Graphics Interface 85, Canadian Information Processing Society, Toronto, May 1985, pp 97-104. |
| [Wilhelms 86] | J.Wilhelms : *Virya - A motion control editor for kinematic and dynamic animation.* Graphics Interface 86, Morgan Kaufmann, Los Altos, California, May 1986, pp 141-146. |
| [Wilhelms 87/1] | J.Wilhelms : *Towards automatic motion control.* IEEE Computer Graphics and Applications, April 1987, pp 11-22. |

| | |
|---|---|
| [Wilhelms 87/2] | J.Wilhelms : *Using dynamic analysis for realistic animation of articulated bodies.* IEEE Computer Graphics and Applications, June 1987, pp 12-27. |
| [Witkin & Kass 88] | A.Witkin, M.Kass : *Spacetime constraints.* ACM Computer Graphics, Vol 22, No 4, pp 159-168, 1988. |
| [Wyvill 89] | B.Wyvill : *Navigating the animation jungle.* Editor R.A.Earnshaw 1989. |
| [Vubobratovic & Stokic 77] | M.Vubobratovic, D.Stokic : *New control concepts of anthropomorphic manipulators.* Mechanism and Machine Theory, 1977, Vol 12, pp 515-530 |
| [Vubobratovic 90] | M.Vubobratovic, B.Borovac, D.Stokic : *Legged locomotion.* Springer-Verlag Berlin 1990. |
| [Uicker 69] | J.J.Uicker : *On the dynamic analysis of spatial linkages using 4x4 matrices.* Ph.D Dissertation, Northwestern University, Aug 1965. |
| [Zeltzer 82/1] | D.Zeltzer : *Motor control techniques for figure animation.* IEEE Computer Graphics and Application, November 1982, pp 53-59. |
| [Zeltzer 82/2] | D.Zeltzer : em Knowledge based animation. Motion : Representation and Perception (1982) Editor N.I.Badler, Elsevier Science Publishing Co Inc, pp 318-323. |
| [Zeltzer 91] | D.Zeltzer : *Task-level graphical simulation : abstraction, representation, and control.* Make them Move, 1991, Kaufmann, Los Altos, California pp 3-33. |
| [Zheng 88] | Y.F.Zheng, F.R.Sias Jr : *Design and motion control of practical biped robots.* International Journal of Robotics and Automation, Vol 3, No 2, 1988. |

# Appendix A

# Derivation of Kinematics Profiles

This Appendix derives the solution for the constants for the displacement, velocity and acceleration profiles used for a degree of freedom under kinematic control. General notation :

- $T$ for simulation interval
- $\theta_0$, $\theta_1$ for starting and finishing angle respectively
- $\dot{\theta}_0$, $\dot{\theta}_1$ for starting and finishing velocity respectively
- $\ddot{\theta}_0$, $\ddot{\theta}_1$ for starting and finishing acceleration
- t the time variable
- A, B, C, D and s are constants

## A.1 Curves for displacement

### A.1.1 Trigonometric

#### Without velocity constraints

We consider $\sin(x)$. Then $0 \le t \le T \Rightarrow -\frac{\pi}{2} \le \frac{t\pi}{T} - \frac{\pi}{2} \le \frac{\pi}{2}$. Let :

$$\theta(t) = \theta_1 \left( A \sin\left(\frac{t\pi}{T} - \frac{\pi}{2}\right) + B \right)$$

The velocity is given by :

$$\dot{\theta}(t) = \theta_1 \left( \frac{A\pi}{T} \cos \left( \frac{t\pi}{T} - \frac{\pi}{2} \right) \right)$$

and acceleration by :

$$\ddot{\theta}(t) = -\theta_1 \left( \frac{A\pi^2}{T^2} \sin \left( \frac{t\pi}{T} - \frac{\pi}{2} \right) \right)$$

Solving for constraints : $\theta(0) = \theta_0$ and $\theta(T) = \theta_1 \Rightarrow$

$$A = \frac{\theta_1 - \theta_0}{2\theta_1} \quad and \quad B = \frac{\theta_1 + \theta_0}{2\theta_1}$$

**With velocity constraints**

Consider the displacement to be given by :

$$\theta(t) = \theta_1 \left( A \sin \left( \frac{t\pi}{T} - \frac{B\pi}{2} \right) + Ct + D \right)$$

Then the velocity is given by :

$$\dot{\theta}(t) = \theta_1 \left( \frac{A\pi}{T} \cos \left( \frac{t\pi}{T} - \frac{B\pi}{2} \right) + C \right)$$

and the acceleration by :

$$\ddot{\theta}(t) = -\theta_1 \left( \frac{A\pi^2}{T^2} \sin \left( \frac{t\pi}{T} - \frac{B\pi}{2} \right) \right)$$

Solving for constraints : $\theta(0) = \theta_0$, $\theta(T) = \theta_1$, $\dot{\theta}(0) = \dot{\theta}_0$ and $\dot{\theta}(T) = \dot{\theta}_1$, the constants are calculated as :

$$B = \frac{2}{\pi} \tan^{-1} \left( \frac{k_1 \pi}{2Tk_2} \right) \quad and \quad A = \frac{Tk_2}{\pi \cos \left( \frac{B\pi}{2} \right)}$$

and

$$C = \frac{\dot{\theta}_0 - \dot{\theta}_1}{2\theta_1} \quad and \quad D = \frac{\theta_0}{\theta_1} + A \sin \frac{B\pi}{2}$$

where $k_1$ and $k_2$ are given by :

$$k_1 = 1 - \frac{\theta_0}{\theta_1} - \frac{T(\dot{\theta}_0 - \dot{\theta}_1)}{2\theta_1} \quad and \quad k_2 = \frac{\dot{\theta}_0 - \dot{\theta}_1}{2\theta_1}$$

## A.1.2 Exponential

**Simple exponential without velocity constraints**

The displacement is given by :

$$\theta(t) = A\left(\frac{t}{T}\right)^n \theta_1 + \theta_0$$

Then the velocity is :

$$\dot{\theta}(t) = An\left(\frac{t}{T}\right)^{n-1} \theta_1$$

and the acceleration :

$$\ddot{\theta}(t) = An(n-1)\left(\frac{t}{T}\right)^{n-2} \theta_1$$

Constraint $\theta(0) = \theta_0$ is satisfied by the displacement profile and constraint $\theta(T) = \theta_1$ gives A to be :

$$A = \frac{\theta_1 - \theta_0}{\theta_1}$$

**Exponential without velocity constraints**

Let :

$$\theta(t) = \theta_1\left(\frac{A\left(t - \frac{T}{s}\right)^n + \left(\frac{T}{s}\right)^n}{B\left(\frac{T}{s}\right)^n}\right)$$

The velocity is, then, given by :

$$\dot{\theta}(t) = \theta_1\left(\frac{An\left(t - \frac{T}{s}\right)^{n-1}}{B\left(\frac{T}{s}\right)^n}\right)$$

and the acceleration :

$$\ddot{\theta}(t) = \theta_1\left(\frac{An(n-1)\left(t - \frac{T}{s}\right)^{n-2}}{B\left(\frac{T}{s}\right)^n}\right)$$

From the two constraints $\theta(0) = \theta_0$ and $\theta(T) = \theta_1$, the constants are calculated to be

$$A = \frac{\theta_1 - \theta_0}{(s-1)^n \theta_0 + \theta_1} \quad and \quad B = A(s-1)^n + 1$$

**Exponential with velocity constraints**

Let the displacement curve be :

$$\theta(t) = \theta_1\left(A\left(t-\frac{T}{s}\right)^n + B\left(t-\frac{T}{s}\right)^{n-1} + C\left(t-\frac{T}{s}\right)^{n-2} + D\right)$$

Then the velocity is :

$$\dot{\theta}(t) = \theta_1\left(An\left(t-\frac{T}{s}\right)^{n-1} + B(n-1)\left(t-\frac{T}{s}\right)^{n-2} + C(n-2)\left(t-\frac{T}{s}\right)^{n-3}\right)$$

and the acceleration :

$$\begin{aligned}\ddot{\theta}(t) &= \theta_1\left(An(n-1)\left(t-\frac{T}{s}\right)^{n-2} + B(n-1)(n-2)\left(t-\frac{T}{s}\right)^{n-3}\right)\\ &+ \theta_1\left(C(n-2)(n-3)\left(t-\frac{T}{s}\right)^{n-4}\right)\end{aligned}$$

The two displacement constraints $\theta(0) = \theta_0$, $\theta(T) = \theta_1$ and the two velocity constraints $\dot{\theta}(0) = \dot{\theta}_0$ and $\dot{\theta}(T) = \dot{\theta}_1$ results to system of four linear equations.The solution of this system of these linear equations will give the values of constants A, B, C and D. The system can be solved analytically or by using any numerical analysis technique for linear systems such as Gauss elimination. The solution is omitted here because of its length.

### A.1.3 Hyperbolic

**Without velocity constraints**

Let the displacement be :

$$\theta(t) = \theta_1\left(A\sinh\left(\frac{t}{T}\right) + B\right)$$

Then the velocity is given by :

$$\dot{\theta}(t) = \theta_1\left(\frac{A}{T}\cosh\left(\frac{t}{T}\right)\right)$$

and the acceleration :

$$\ddot{\theta}(t) = \theta_1\left(\frac{A}{T^2}\sinh\left(\frac{t}{T}\right)\right)$$

From the constraints $\theta(0) = t_0$ and $\theta(T) = \theta_1$, the two constants are found to be :

$$A = \frac{1 - \frac{\theta_0}{\theta_1}}{\sinh(1)} \quad \textit{and} \quad B = \frac{\theta_0}{\theta_1}$$

**With velocity constraints**

Let the displacement be :

$$\theta(t) = \theta_1 \left( A \sinh\left(\frac{t}{T}\right) + Bt \sinh(t) + C \sinh(t) + D \right)$$

Thus, the velocity is :

$$\dot{\theta}(t) = \theta_1 \left( \frac{A}{T} \cosh\left(\frac{t}{T}\right) + B \sinh(t) + Bt \cosh(t) + C \cosh(t) + D \right)$$

and the acceleration :

$$\ddot{\theta}(t) = \theta_1 \left( \frac{A}{T^2} \sinh\left(\frac{t}{T}\right) + 2B \cosh(t) + Bt \sinh(t) + C \sinh(t) + D \right)$$

The two displacement constraints $\theta(0) = \theta_0$, $\theta(T) = \theta_1$ and the two velocity constraints $\dot{\theta}(0) = \dot{\theta}_0$ and $\dot{\theta}(T) = \dot{\theta}_1$ result to asystem of four linear equations which can be solved to find A, B, C and D. The method is similar as in the case of the exponential profile with velocity constraints and the solution can be obtained with the same techniques.

### A.1.4 Straight line

Let the displacement profile be :

$$\theta(t) = \theta_1 \left( \frac{At}{T} + B \right)$$

The velocity and acceleration are :

$$\dot{\theta}(t) = \frac{\theta_1 A}{T} \qquad \ddot{\theta}(t) = 0$$

From the two displacement constraints $\theta(0) = \theta_0$ and $\theta(T) = \theta_1$, the constants are found to be :

$$A = \frac{\theta_1 - \theta_0}{\theta_1} \qquad B = \frac{\theta_0}{\theta_1}$$

## A.2 Curves for acceleration

### A.2.1 Constant

Let the acceleration be constant : $\ddot{\theta}(t) = \ddot{\theta}_1$, then by taking into consideration the constraints $\theta_0 = \theta_0$ and $\dot{\theta}(0) = \dot{\theta}_0$ the velocity and acceleration are given by:

$$\dot{\theta}(t) = \ddot{\theta}_1 t + \dot{\theta}_0 \quad , \quad \theta(t) = \frac{\ddot{\theta}_1 t^2}{2} + \dot{\theta}_0 t + \theta_0$$

### A.2.2 Trigonometric

Let the acceleration be :

$$\ddot{\theta}(t) = \ddot{\theta}_1 \sin\left(\frac{\pi t}{T}\right)$$

Then the velocity is :

$$\dot{\theta}(t) = -\ddot{\theta}_1 \left(\frac{T}{\pi}\right) \cos\left(\frac{\pi t}{T}\right) + A$$

and the displacement is :

$$\theta(t) = -\ddot{\theta}_1 \left(\frac{T^2}{\pi^2}\right) \sin\left(\frac{\pi t}{T}\right) + At + B$$

where, from the constraints $\dot{\theta}(0) = \dot{\theta}_0$ and $\theta(0) = \theta_0$, the constants are :

$$A = \dot{\theta}_0 + \frac{\ddot{\theta}_1 T}{\pi} \quad and \quad B = \theta_0$$

### A.2.3 Ellipse

The equation for acceleration for the ellipse profile is :

$$\frac{\ddot{\theta}^2(t)}{\alpha^2} + \frac{(t-T)^2}{\beta^2} = 1$$

From the two acceleration constraints $\ddot{\theta}(0) = \ddot{\theta}_0$ and $\ddot{\theta}\left(\frac{T}{2}\right) = \ddot{\theta}_m$ , $\alpha$ and $\beta$ can be found so the acceleration constraint becomes :

$$\frac{3\ddot{\theta}^2(t)}{4\ddot{\theta}_m^2} + \frac{(t-T)^2}{T^2} = 1$$

Hence :

$$\ddot{\theta}(t) = \frac{2\ddot{\theta}_m}{\sqrt{3}T}\left(T^2 - (t-T)^2\right)^{\frac{1}{2}}$$

By integrating the acceleration and considering the constraint $\dot{\theta}(0) = \dot{\theta}_0$, the velocity curve is found to be :

$$\dot{\theta}(t) = \frac{\ddot{\theta}_m T}{\sqrt{3}}\left(x - \frac{1}{2}\sin 2x\right) + \dot{\theta}_0 + \frac{\ddot{\theta}_1 T\pi}{2\sqrt{3}}$$

where $x = \sin^{-1}\left(\frac{(t-T)}{T}\right)$. The displacement is found by integrating the velocity and by using $\theta(0) = \theta_0$ :

$$\begin{aligned}\theta(t) &= \frac{\ddot{\theta}_m T^2}{\sqrt{3}}\left(x\sin x + \cos x + \frac{1}{3}\cos^3 x\right) + \dot{\theta}_0 t + \frac{\ddot{\theta}_m T\pi t}{2\sqrt{3}} + \\ &+ \theta_0 + \frac{4\ddot{\theta}_m T^2}{3\sqrt{3}}\end{aligned}$$

## A.3 Curves for velocity

### A.3.1 Straight line

Let the velocity be $\dot{\theta}(t) = \dot{\theta}_1\left(\frac{\alpha t}{T} + \beta\right)$. The two velocity constraints $\dot{\theta}(0) = \dot{\theta}_0$ and $\dot{\theta}(T) = \dot{\theta}_1$ are used to find $\alpha$ and $\beta$ :

$$\alpha = \frac{(\dot{\theta}_1 - \dot{\theta}_0)}{\dot{\theta}_1} \qquad \beta = \frac{\dot{\theta}_0}{\dot{\theta}_1}$$

The displacement is found by integration to be

$$\theta(t) = \dot{\theta}_1\left(\frac{\alpha t^2}{2T} + \beta t\right) + C$$

Using the displacement constraint $\theta(0) = \theta_0$, we find $C = \theta_0$. The acceleration is finally found to be :

$$\ddot{\theta}(t) = \frac{\dot{\theta}_1 - \dot{\theta}_0}{T}$$

## A.3.2 Circle

The equation of a circle is given by :

$$\left(\dot{\theta}(t)-\alpha\right)^2+(t-\beta)^2=\gamma^2$$

Consider that the velocity has to pass through three points : $(0,\dot{\theta}_0)$, $(T_1,\dot{\theta}_m)$ and $(T,\dot{\theta}_1)$. These three constraints form a set of three equations which define the constants to be :

$$\alpha = \frac{\dot{\theta}_m-\dot{\theta}_0-k_1^2+(T_1-k_1)^2}{2\left(\dot{\theta}_m-\dot{\theta}_0-k_1k_2-(T_1-k_1)\,k_2\right)}$$

$$\beta = k_1-\alpha k_2 \quad \textit{and} \quad \gamma^2 = \beta^2+(\dot{\theta}_0-\alpha)^2$$

where the two constants $k_1$ and $k_2$ are :

$$k_1 = \frac{(T^2-T_1^2)+(\dot{\theta}_1^2-\dot{\theta}_m^2)}{2(T-T_1)} \qquad k_2 = \frac{\dot{\theta}_1-\dot{\theta}_m}{T-T_1}$$

By integration the displacement is found to be :

$$\theta(t) = \alpha t\pm\frac{1}{2}\gamma^2\left(\frac{1}{2}\sin 2x+x\right)+C$$

where from the constraint $\theta(0)=\theta_0$ :

$$x = \sin^{-1}\left(\frac{-\beta}{\gamma}\right) \quad \textit{and} \quad C = \theta_0\mp\frac{\gamma^2}{2}\left(\frac{1}{2}\sin 2x+x\right)$$

Finally the acceleration is given by :

$$\ddot{\theta}(t) = \pm\frac{1}{2}\left(\gamma^2-(t-\beta)^2\right)^{-\frac{1}{2}}$$

The sign is chosen so that the three points are on the same semicircle otherwise the acceleration will switch from one branch of the curve to another causing instability.

# Appendix B

# Transformation Tables

This Appendix shows the transformation default values for $\alpha$ (angle between $z_{i-1}$ and $z_i$ axes, measured in the right hand sense about $x_i$) and $\theta$(angle between $x_{i-1}$ and $x_i$ axes, measured in the right hand sense about $z_{i-1}$). The values of $\alpha$ are constant throughout the motion whereas the given values of $\theta$ show their offset values. The appendix has two parts. The first part contains the values of $\alpha$ and $\theta$ when all joints in the rationalised kinematic chain have the same rotation type. The second part of the appendix gives the values of $\alpha$ and $\theta$ in the case where the joints in the kinematic chain have a mixture of rotation types.

## B.1 Homogeneous rotation types

•*i. Rotation type :* $X$-$Y$-$Z$ For this type of rotation all values of $\alpha$ and $\theta$ are +90.

•*ii. Rotation type* $X$-$Z$-$Y$

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 90 |
| 2 | 90 | 0 |
| 3 | -90 | -90 |
| 4 | -90 | -90 |
| 5 | -90 | -90 |
| 6 | -90 | -90 |

All the following degrees of freedom have values $\alpha = -90$ and $\theta = -90$.

•*iii. Rotation type A: Y-X-Z*

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | 0 |
| 2 | -90 | -90 |
| 3 | -90 | -90 |
| 4 | -90 | -90 |
| 5 | -90 | -90 |
| 6 | -90 | -90 |

All the following degrees of freedom have values $\alpha = -90$ and $\theta = -90$.

•*iv. Rotation type Y-Z-X*

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | 0 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |
| 4 | 90 | 90 |
| 5 | 90 | 90 |
| 6 | 90 | 90 |

All the following degrees of freedom have $\alpha = 90$ and $\theta = 90$.

•*v. Rotation type Z-X-Y*

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 90 | 90 |
| 3 | 90 | 90 |
| 4 | 90 | 90 |
| 5 | 90 | 90 |
| 6 | 90 | 90 |

All the following degrees of freedom have $\alpha = 90$ and $\theta = 90$.

•*vi. Rotation type Z-Y-X*

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | -90 | 0 |
| 3 | 90 | 90 |
| 4 | -90 | 90 |
| 5 | -90 | -90 |
| 6 | -90 | -90 |

All following degrees of freedom are $\alpha = -90$ and $\theta = -90$.

## B.2 Heterogeneous rotation types

The previous tables are valid only if all the degrees of freedom of the body are of the same rotation type. However, there will be cases where the body will contain a mixture of rotation types. the following tables are constructed to give guidance about the values of $\alpha$ and $\theta$ when there are mixed type rotations. In the following tables, it is assumed that there are two links of three degrees of freedom each where the first link has rotation type A and the second link rotation type B and the values of $\alpha$ and $\theta$ are given for the second link (rotation type B). The tables are sufficient to define the values of $\alpha$ and $\theta$ for any type of mixed rotations :

•*i. Rotation type A: X-Y-Z*

a. To Rotation type X-Z-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 90 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

b. To Rotation type Y-X-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | 0 |
| 2 | -90 | -90 |
| 3 | -90 | -90 |

c. To Rotation type Y-Z-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | 0 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

d. To Rotation type Z-X-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 90 | 90 |
| 3 | 90 | 90 |

e. To rotation type Z-Y-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

• ***ii. Rotation type A: X-Z-Y***

a. To Rotation type X-Y-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | -90 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

b. To Rotation type Y-X-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | -90 | -90 |
| 3 | -90 | -90 |

c. To Rotation type Y-Z-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

d. To Rotation type Z-X-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 0 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

e. To Rotation type Z-Y-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 0 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

●*iii. Rotation type A: Y-X-Z*

a. To Rotation type X-Y-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 0 |
| 2 | 90 | 90 |
| 3 | 90 | 90 |

b. To Rotation type X-Z-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 0 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

c. To Rotation type Y-Z-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | -90 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

d. To Rotation type Z-X-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | -90 |
| 2 | 90 | 90 |
| 3 | 90 | 90 |

e. To Rotation type Z-Y-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | -90 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

***•iv . Rotation type A: Y-Z-X***

a. To rotation type X-Y-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 90 | 90 |
| 3 | -90 | 90 |

b. To Rotation type X-Z-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 0 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

c. To Rotation type Y-X-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 90 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

d. To Rotation type Z-X-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 0 |
| 2 | 90 | 90 |
| 3 | 90 | 90 |

e. To Rotation type Z-Y-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | 0 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

***•v. Rotation type A: Z-X-Y***

a. To Rotation type X-Y-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | 0 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

b. To Rotation type X-Z-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 0 |
| 2 | -90 | -90 |
| 3 | -90 | -90 |

c. To Rotation type Y-X-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 90 |
| 2 | 90 | -90 |
| 3 | 90 | -90 |

d. To Rotation type Y-Z-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 90 |
| 2 | 90 | 0 |
| 3 | 90 | -90 |

e. To Rotation type Z-Y-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 90 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

•*vi. Rotation type A: Z-Y-X*

a. To Rotation type X-Y-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 90 |
| 2 | 90 | 90 |
| 3 | 90 | 90 |

b. To Rotation type X-Z-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 0 | 90 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

c. To Rotation type Y-X-Z

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 180 |
| 2 | -90 | 0 |
| 3 | -90 | -90 |

d. To Rotation type Y-Z-X

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | 90 | 180 |
| 2 | 90 | 90 |
| 3 | 90 | 90 |

e. To Rotation type Z-X-Y

| Rationalised Link Number | $\alpha_i$ default value | $\theta_i$ default value |
|---|---|---|
| 1 | -90 | 90 |
| 2 | 90 | 0 |
| 3 | 90 | 90 |

# Appendix C

# Combined Transformation Matrices

This appendix lists the six transformation matrices. Since we are using column vectors the rotation matrices are :

$$z - rotation \ : \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$y - rotation \ : \begin{pmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{pmatrix}$$

$$x - rotation \ : \begin{pmatrix} 1 & 0 & 1 \\ 0 & \cos\chi & -\sin\chi \\ 0 & \sin\chi & \cos\chi \end{pmatrix}$$

Hence :

*Order of rotation : z, y, x* :

$$\begin{pmatrix} \cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi \\ \cos\chi\sin\theta + \sin\chi\sin\phi\cos\theta & \cos\chi\cos\theta - \sin\chi\sin\phi\sin\theta & \sin\chi\cos\phi \\ \sin\chi\sin\theta - \sin\phi\cos\theta\cos\chi & \sin\chi\cos\theta + \cos\chi\sin\phi\sin\theta & \cos\chi\cos\phi \end{pmatrix}$$

*Order of rotation : z, x, y :*

$$\begin{pmatrix} \cos\phi\cos\theta + \sin\theta\sin\phi\sin\chi & -\cos\phi\sin\theta + \sin\phi\sin\chi\cos\theta & \sin\phi\cos\chi \\ \cos\chi\sin\theta & \cos\chi\cos\theta & -\sin\chi \\ -\sin\phi\cos\theta + \cos\phi\sin\chi\sin\theta & \sin\phi\sin\theta + \cos\phi\sin\chi\cos\theta & \cos\phi\cos\chi \end{pmatrix}$$

*Order of rotation : y, x, z :*

$$\begin{pmatrix} \cos\theta\cos\phi - \sin\theta\sin\chi\sin\phi & -\sin\theta\cos\chi & \cos\theta\sin\phi + \sin\theta\sin\chi\cos\phi \\ \sin\theta\cos\phi & \cos\theta\cos\chi & \sin\theta\sin\phi - \cos\theta\sin\chi\cos\phi \\ -\sin\phi\cos\chi & \sin\chi & \cos\chi\cos\phi \end{pmatrix}$$

*Order of rotation : y, z, x :*

$$\begin{pmatrix} \cos\theta\cos\phi & -\sin\theta & \cos\theta\sin\phi \\ \cos\chi\sin\theta\cos\phi + \sin\chi\sin\phi & \cos\chi\cos\theta & \cos\chi\sin\theta\sin\phi - \sin\chi\cos\phi \\ \sin\chi\sin\theta\cos\theta - \sin\phi\cos\chi & \sin\chi\cos\theta & \sin\chi\sin\theta\sin\phi + \cos\chi\cos\phi \end{pmatrix}$$

*Order of rotation : x, y, z :*

$$\begin{pmatrix} \cos\theta\cos\phi & \cos\theta\sin\phi\sin\chi - \sin\theta\cos\chi & \cos\theta\sin\phi\cos\chi + \sin\theta\sin\chi \\ \sin\theta\cos\phi & \sin\theta\sin\phi\sin\chi + \cos\theta\cos\chi & \cos\theta\sin\phi\cos\chi + \sin\theta\sin\chi \\ -\sin\phi & \cos\phi\sin\chi & \cos\phi\cos\chi \end{pmatrix}$$

*Order of rotation : x, z, y :*

$$\begin{pmatrix} \cos\phi\cos\theta & -\cos\phi\sin\theta\cos\chi + \sin\phi\sin\chi & \cos\phi\sin\theta\sin\chi + \sin\phi\cos\chi \\ \sin\theta & \cos\theta\cos\chi & -\sin\chi\cos\theta \\ -\sin\phi\cos\theta & \sin\phi\sin\theta\cos\chi + \cos\theta\sin\chi & \cos\phi\cos\theta - \sin\phi\sin\theta\sin\chi \end{pmatrix}$$

DX

171569