Real-Time Computer Animation of Bicyclists and Pedestrians in a Driving Simulator *

Jos B.T.M. Roerdink, Mattijs J.B. van Delden†, Andrea J.S. Hin‡
Department of Mathematics and Computing Science, University of Groningen,
P.O. Box 800, 9700 AV Groningen, The Netherlands
Tel. +31-50-3633931; Fax +31-50-3633800;
Email roe@cs.rug.nl; URL: http://www.cs.rug.nl/~roe
and
Peter C. van Wolffelaar
Traffic Research Centre, University of Groningen,
P.O. Box 69, 9750 AB Haren, The Netherlands
Tel. +31-50-3636764; Fax +31-50-363784; Email wolff@trc.rug.nl

Abstract

The Traffic Research Centre (TRC) of the University of Groningen in the Netherlands has developed a driving simulator with 'intelligent' computer-controlled traffic, consisting at the moment only of saloon cars. The range of possible applications would be greatly enhanced if other traffic participants could be simulated too. This paper presents a study of the possibilities of simulating bicyclists and pedestrians in general, and in the TRC driving simulator in particular. The new traffic participants must be controlled interactively, and move and behave in a natural way. A prototype system has been designed and implemented. The system simulates a bicyclist that is controlled by the user through a graphical interface. Integration of the prototype system in the TRC driving simulator is the subject of future research.

Keywords: Driving simulator, computer animation, bicyclists and pedestrians, motion generation, graphical modelling

1 Introduction

The Traffic Research Centre of the University of Groningen carries out research on the behavioral and social aspects of traffic and transportation. For this purpose a computer controlled

*Report CS-R9606, Department of Computing Science, University of Groningen, December 1996. Postscript version obtainable at http://www.cs.rug.nl/~roe/

†Present address: Virtual Environments, Systems, and Consultancy, Duinrand 2, 9483 TP Zeegse, The Netherlands

‡Present address: TNO Human Factors Research Institute, P.O. Box 23, 3769 ZG Soesterberg, The Netherlands

driving simulator was developed, which provides a safe and programmable environment for road user studies (van Wolffelaar & van Winsum 1995). The simulator is made up of a car installed on a fixed platform with a curved wide-angle projection screen two meters in front of the car to provide a 165 degrees horizontal and 40 degrees vertical out-of-the-window view on the computer generated traffic scenery. All controls of the car have been modified and are linked to a computer system. The system reads input from the simulator driver, and outputs forces to the steering axle and the accelerator pedal. The screen image is generated by a Silicon Graphics Sky-Writer 340 VGXT graphical computer with two hardware graphic pipelines, and is projected on the curved screen by three large video projectors. Rear view mirrors are simulated by small windows on the main projection screen at approximately the correct positions with respect to the driver. A snapshot of the middle projection screen is shown in Fig. 1.

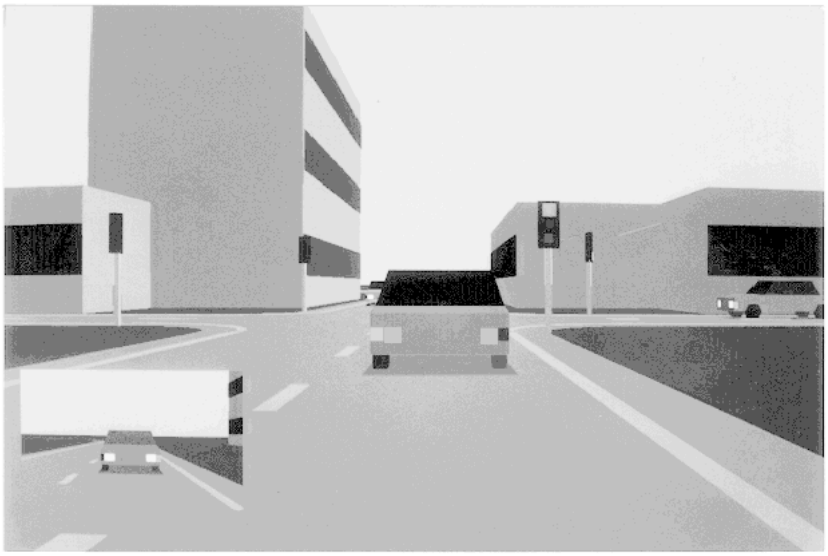


Figure 1: *Screen image of the middle projector of the TRC driving simulator.*

Perhaps the most innovative feature of the TRC simulator is the dynamic traffic environment. Currently this consists of 'intelligent' cars that interact with each other and with the simulator user. A road network can be specified through a programming language or a graphical user interface. The dynamic environment (traffic, traffic lights, animated signs, etc.) can be controlled by means of a scenario, which is a script file that accurately creates desired traffic situations for experiments. In spite of the simplicity of the computer generated environment (see Fig. 1), traffic participation is generally experienced by the simulator driver to be very realistic due to the dynamics of the traffic environment. Currently, an important shortcoming lies in the fact that only saloon cars are simulated. An extension is desirable to trucks, busses, vans, motorcyclists, pedestrians, moped riders, and bicyclists that make up an extremely varied and complex traffic environment. Pedestrians, moped riders and bicyclists are substantially different from the currently implemented cars: they have several parts with internal motion, such as arms and legs.

In this paper, we report on a prototype system for real-time graphical visualization/animation of bicyclists and/or pedestrians, which will be collectively referred to as ViPs, or 'Virtual inter-

active Persons'. For a more extensive discussion of background and technical details the reader is referred to van Delden (1995).

The organization of this paper is as follows. In Section 2 general requirements are formulated and discussed concerning speed, interactivity and naturalness of motion of slow traffic participants in a driving simulator. Section 3 contains the design of the system meeting these requirements. The implementation of a prototype bicyclist real-time animation system is described in Section 4. Finally we give in Section 5 a summary of this work and present conclusions and possibilities for future research.

2 Requirements

2.1 Real-time simulation

The definition of real time given by Hodgins, Wooten, Brogan & O'Brien (1995) states that 'in a real-time system, simulation time is less than wall clock time'. This means that all calculations and actions for a new image that represents the situation one second in the future, must be performed within that second.

The fact that it must be possible to simulate about 10 to 20 pedestrians and bicyclists in real time puts a severe constraint on possible solutions with respect to execution time.

2.2 Interactive simulation

Real time as defined above is not sufficient for visual simulation applications such as a driving simulator. In addition, the visual simulation has to be interactive, meaning that any actions of the simulator user should be immediately reflected in changes of the state of the environment. Actions such as steering, pressing the accelerator pedal and braking, change the state of the computer generated world in such a short interval that the driver gets the impression that the steering wheel is directly controlling the heading of the front wheels, and pedal motion is immediately affecting the speed. The system response delay for a typical simulator session varies between 50 and 90 milliseconds. This means that there can be a delay of about a tenth of a second between turning the wheel and seeing the result on the screen. Generally, delays up to 100 ms are considered acceptable in driving and flight simulators. The addition of bicyclists and pedestrians in the simulation must take as little system time as possible, in order to meet the interactivity requirement.

2.3 Display frames and simulation frames

The TRC driving simulator uses two types of frames: drawing frames and simulation frames that run asynchronous and in parallel. The graphical drawing process generates frames as fast as possible in an endless loop taking actual world position data of the simulated vehicles for each new frame from the simulation process. By extrapolating these positions in time vehicle motions appear very smooth on the screens. For pedestrians and bicyclists, however, extrapolating world positions in time will not be feasible as they are linked to gestural movements. This implies that all calculations necessary for displaying the next frame must be completed before drawing of

that frame starts. An upper bound on the number of simulation frames per second (fps) is given by the display refresh rate, which must be around 60 fps for flicker free display. The actual simulation frame rate of the TRC driving simulator depends on the complexity of the environment, and varies between 10 and 60 fps with an average of 20 to 30 fps.

2.4 Natural motion

When dealing with pedestrians and bicyclists, humans must be modeled while walking, running and cycling. Accurate human models take up thousands of polygons and are not feasible for interactive simulation. The human models to be used in the simulator are allowed to look 'blocky' in order to meet the real-time requirements. However, realistic or natural *motion* will be crucial for making the models appear human. A similar observation holds for the cars in the simulation, which look remarkably simple (see Fig. 1). Apparently, in a dynamic simulator natural movement is more important than accurate shape.

2.5 Range of motions

A desirable set of possible movements for each traffic participant is initially chosen as follows:

pedestrian

- stand still
- start walking
- walk with variable speed - straight ahead - take turns of variable radius
- stop walking

bicyclist

- stand still
- get on bike
- cycle with variable speed - straight ahead - take turns of variable radius
- keep legs still
- get off bike

In addition to natural motion, every ViP needs to have its personal style of walking and cycling, called 'personified motion'.

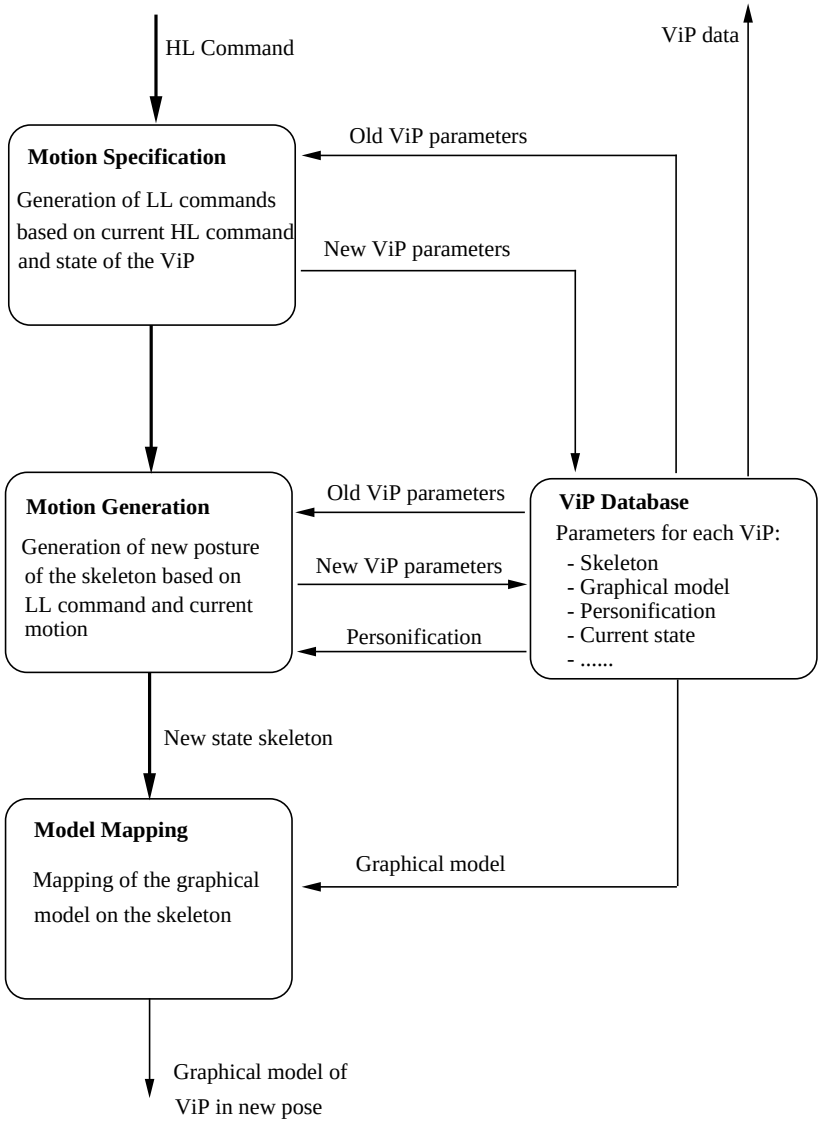


Figure 2: *Conceptual view of the system, showing logical modules with data flow indicated by arrows.*

2.6 Control and interface

The 'intelligent' vehicles in the TRC driving simulator are governed by a model which controls them on the level of behaviour in traffic. The cars do not possess skeletal or intrinsic movements, but slow traffic simulation has to take place at several motion levels. The first level of motion (FLM) consists of the primary motions such as taking a step, taking a turn, or getting off a bike. The behaviour of traffic participants in the virtual world forms the second level of motion (SLM). This level is actually a motion planning system that requires information from the environment database.

This paper only addresses the problem of creating the first level of motion. The existing second level of motion in the TRC driving simulator will be used or adapted to handle slow traffic

participants as well. Only the interface between the first and second level, describing how the simulated persons will be controlled, has been implemented. This will make it possible to add an intelligence module to the system that communicates with the FLM through a well defined interface. Instead of using a SLM module, it is also possible to receive input from a script file processing module, through scenarios, or by direct control via an input device. An important step is therefore the specification of a complete and easy-to-use interface to control the ViPs. The degree of control specifies the extent to which ViPs will be able to obey commands.

3 System design

The task of controlling and moving Virtual interactive Persons (ViPs) can be separated into motion specification and motion generation, as is common usage in most of the systems found in the literature (Morawetz & Calvert 1990, Badler, Phillips & Webber 1993, Girard 1989). Motion generation can be subdivided into internal and external motion generation. A conceptual overview of the system's subdivision into a number of logical parts or modules is given in Fig. 2. The motion specification (MS) and motion generation (MG) modules form the core of the system. Other modules are the ViP database and the graphical mapping of the model.

3.1 Skeleton model

A skeleton-like structure can be used both for motion definition as well as the construction of the graphical model. Motions will be defined on the skeleton, and therefore it must be possible to transform postures of the skeleton to postures of the graphical model (Badler et al. 1993, Badler, Barsky & Zeltzer 1991, Magnenat-Thalmann & Thalmann 1985, Magnenat-Thalmann & Thalmann 1990). A skeleton consists of inflexible segments (the bones) and flexible joints. General joints can be constructed as translational joints, rotational joints, or a combination of both. An internal degree of freedom (DOF) exists for each independent rotation and translation. In Fig. 3 an example skeleton is shown, with the number of rotational degrees of freedom of each joint indicated. The human skeleton as a whole has six *external* degrees of freedom: three translational DOFs and three rotational DOFs of the centre of mass. In the example, the total number of DOFs is 32, which is less than for many of the skeletons mentioned in the literature (Badler et al. 1993, Badler et al. 1991, Magnenat-Thalmann & Thalmann 1990). The problem of rotation over angles that are impossible for joints of real persons can be solved by constraining each DOF. By specifying a minimum and maximum for each angle, the motion can be limited. A natural way to represent the skeleton is by a directed (acyclic) graph struc-

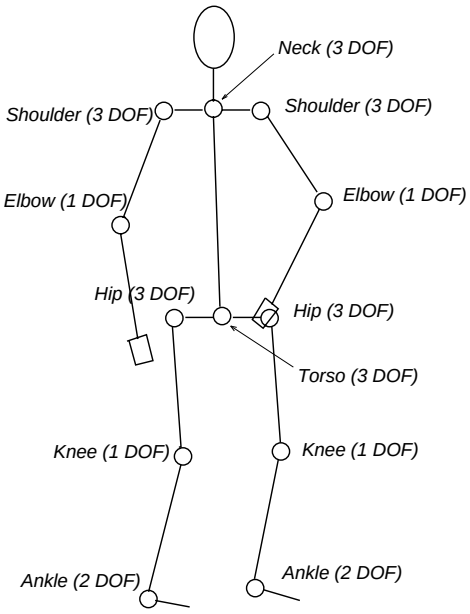


Figure 3: *Example of a simplified human skeleton used for a ViP.*

ture, for example a tree. Each segment is defined in its own coordinate system (CS) defined relative to the CS of its parent. To calculate the position of the child segment of a given joint, the segments are transformed to world coordinates and every child node is assigned a relative transformation with respect to its parent.

3.2 Graphical modelling

The main problem of graphical modelling is to create naturally looking ViPs that can be rendered efficiently. The graphical model constructed from suitable primitives will be wrapped around the skeleton structure. This is called *mapping* the model on the skeleton. Graphical primitives suitable for human modelling can be classified into three groups: wire-frame, volume and surface primitives (Magnenat-Thalmann & Thalmann 1985). Since in the simulator, the ViPs will only be visible from the outside, a surface representation will be appropriate.

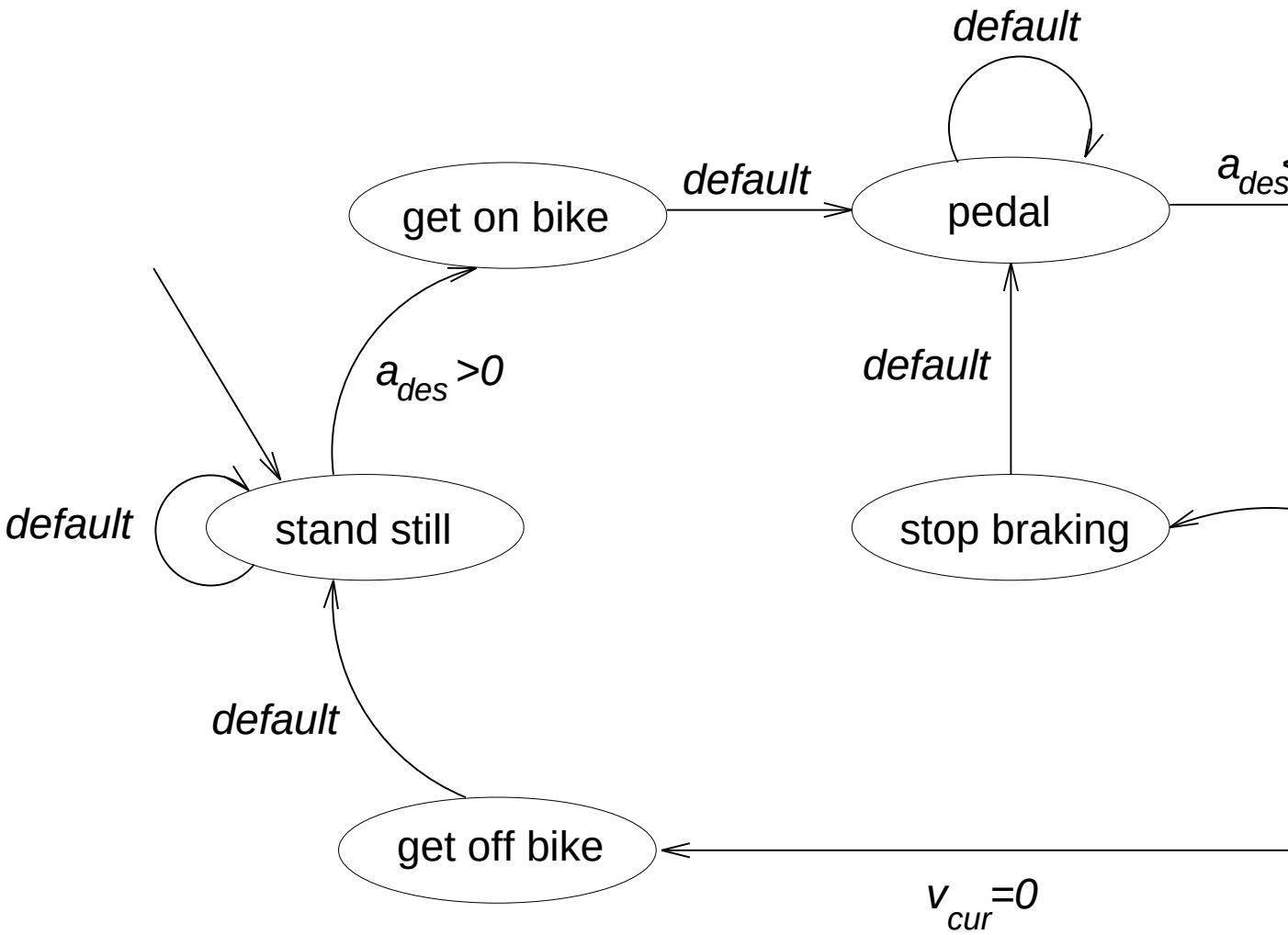


Figure 4: *Finite State Machine for motion specification of the bicyclist.* a_{des} *is the desired acceleration,* v_{cur} *the current velocity.*

3.3 Motion specification

The MS module contains rules describing sequences of motions to be generated by the MG module. It is the interface between the user controlling the ViPs and the database of available motions, translating high level (HL) commands issued by the user to low level (LL) commands of the system. The LL commands are interpreted by the MG module as elementary motion instructions. The ViP database is consulted on the current parameters of each ViP, such as position and orientation in the world, personification parameters, etc. The MS module does not concern itself with low-level details of motion; these are taken care of by the MG module.

3.3.1 Motion specification graph

An obvious logical structure for specifying motion is a graph structure, in this case a finite state machine, cf. Fig. 4. Nodes in the graph are basic motion states of the ViP. The current node indicates the LL command that is sent to the MG module. Transitions from one basic motion to another correspond to traversals to neighbouring states, as indicated by arrows. Transitions have HL command labels or expression labels attached. A transition can only be made if the currently active HL command corresponds to the label, or the attached expression evaluates to true. A default transition is made when no other transitions are valid. The initial state is the one which has an incoming arrow without source state.

3.3.2 Processing high-level commands in time

An important problem is the delay between the moment that the HL command enters the system and the moment it can be obeyed. This implies that the MS and/or MG module must be able to predict how long motions will take. For example, when a moving bicyclist must stop, the ViP has to stop pedalling first and brake until its velocity is almost zero. Therefore, HL commands will have a parameter d, which specifies at what distance from the current location the desired motion has to start. A HL command sent by the SLM module can be obeyed in time if the distance specified in the command is larger than the distance it will take to switch to the specified motion. The FLM cannot guarantee that HL commands will be obeyed in time. The higher levels of motion have to handle the consequences of HL commands that haven't been obeyed in time.

3.4 Motion generation

The MG module determines the new state of the skeleton depending on the current parameters and the LL command to be executed. The new state is returned to the ViP database and also sent to the mapping module. This module maps the graphical model of the ViP onto the skeleton, creating a new posture of the ViP for display. For each frame a LL command is issued, a motion is generated, the ViP database is updated, and the posture of the model is determined. In contrast to LL commands, HL commands are not generated every frame. The SLM has to monitor the behaviour of each ViP to guide the ViP through the traffic environment. In Fig. 2, this is indicated by the arrow extending out of the ViP database module pointing upwards.

3.4.1 Internal motion generation

A number of methods for generating internal motions have been examined to see whether they would meet the requirements of real-time performance, interactivity, naturalness and personification of motion, and the capability of simulating both pedestrians and bicyclists. Among these methods we mention the following:

Keyframe Animation. A key-frame animation system is intended to create moving pictures that generally are not generated real time. Also, natural motion (in contrast to single postures) is very hard to achieve with a key-frame system.

Kinematics. Although inverse kinematics is very suitable to generate a posture for a ViP, it is less successful when used for motion generation. The biggest problem is that kinematics does not take into account physical concepts of mass, force, acceleration and velocity. Inverse kinematics is a computationally expensive algorithm, especially when human figures with a large number of degrees of freedom are simulated.

Dynamics. Dynamic simulation of human figures has recently been examined and implemented by Hodgins et al. (1995). Their system contains algorithms to simulate human skills such as walking/running and cycling. Inverse dynamics results in natural motion, but it is complicated and even slower than inverse kinematics. van Overveld (1991) describes an iterative approach to 3D dynamic simulation for real-time interactive computer animation. Although his algorithm is faster than full dynamic simulation, the reported performance of approximately five frames per second on a SUN-4 is insufficient for simultaneous simulation of a dozen of ViPs. So dynamics violates the speed requirement, but it also needs additional methods to specify motion paths for parts of the body.

Combined Kinematics and Dynamics. With this method, natural motion of limbs is guaranteed by using inverse dynamics while motion paths are specified with a key-frame system. Limbs are positioned on these paths using inverse kinematics. Again, the drawback of this method is the large computational demand. van Overveld & Ko (1994), who use this mixed kinematics/dynamics approach, report a frame rate of about 20 fps on a Silicon Graphics Personal Iris (Indigo) workstation for a flat human figure with 14 degrees of freedom. When simulating a dozen ViPs simultaneously, preferably on a frame rate of 60 Hz, a faster method is desired.

Motion Capture. The two merits of motion capture are speed and naturalness of motion (Maiocchi 1996). Playing back recorded motion is the fastest way of motion ‘generation’. Capturing the movements of real human actors will yield realistic motion provided that the capture process is accurate enough. Captured motions can be transferred to a skeleton structure. Altering the motion to generate personified motion is not possible with motion capture. For each personified motion, the movements of a real actor have to be recorded and stored.

In view of the requirements formulated above, we have decided to use motion capture to create a library of basic internal motions. Using these motions and additional algorithms to create new motions, all required motions can be generated in real time. The MG module will combine basic motions in two ways. First, motions can be executed in a sequence with smooth transitions. Second, a number of basic motions can be interpolated to create a new, different, motion. For example, interpolation is necessary to create walking with a variable step size, or cycling with a variable amount of effort. Also, basic motions, sequenced motions, and blended motions can be played back at different velocities to extend the range of possible motions even further. For efficiency reasons, creation of a new motion by interpolation always uses just two basic motions.

3.4.2 External motion generation

External Motion Generation (EMG) must ensure natural motion of each ViP through the world. A suitable EMG system will allow the ViPs to move through the world as flexibly as possible and in such a way that it appears that the external motions are driven by internal motions. The MS module has to ensure that the internal and external MG modules create 'compatible' motion. External motions can be classified by freedom of movement, i.e., in one, two, or three dimensions. Ideally, each higher degree of motion is constructed out of a lower degree combined with new algorithms. An example of this is walking along a curve being created out of modified straight line walking, with the orientation of the VIP adjusted according to the position on the curve. Although the paths followed by pedestrians and bicyclists are similar, both motion specification and the relation with internal motion differ considerably. We limit ourselves here to a discussion of external motions of bicyclists.

3.4.3 Bicyclist straight-line motion

There are three motion states: starting, moving (accelerating or decelerating) and stopping. A number of parameters are used corresponding to speed and acceleration. Also, depending on the forces slowing the bike down or speeding the bike up, the ViP has to push more or less hard on the pedals. To incorporate this effort we introduce an effort parameter E.

The EMG module uses input parameters like current velocity v_{cur}, desired velocity v_{goal} after a given distance d, and personification parameters. From this, it computes output parameters like possible acceleration, the required effort E and required special case of basic motion, using certain expressions based on physical principles like force, mass and acceleration. (The EMG module doesn't use forces directly, that would be tantamount to using dynamics for motion generation, which is not the case in our approach.)

For *maintaining velocity* captured motions of a person cycling with minimal effort and with maximal effort, respectively, can be interpolated using the parameter E.

Next, consider *changing velocity*. At any time, there is a target velocity v_{goal} (specified by a HL command) that the ViP has to obey. Possible actions are:

1. continue pedalling with unaltered force ($v_{goal} = v_{cur}$)
2. continue pedalling with less force ($v_{goal} < v_{cur}$)
3. continue pedalling with more force ($v_{goal} > v_{cur}$)

4. stop pedalling ($v_{goal} < v_{cur}$)
5. brake ($v_{goal} < v_{cur}$)
6. start (get on bike) ($v_{cur} = 0$ and $v_{goal} > 0$)
7. stop (get off bike) ($v_{cur} \approx 0$ and $v_{goal} = 0$)

A choice from the special cases 1-5 can be made based on the desired acceleration a_{des} of the bike, given by

$$a_{des} = (v_{cur} + v_{goal})(v_{goal} - v_{cur})/(2d).$$

Acceleration alone does not provide a choice out of cases 3, 4 and 5. In order to do that, it is necessary to calculate another parameter: natural deceleration a_{nat} (a negative acceleration). When a_{des} is equal to a_{nat}, the ViP can keep its legs still: case 4. If a_{des} is greater than a_{nat} , the ViP will have to keep pedalling: case 3. If a_{des} is smaller than a_{nat}, the natural deceleration isn't slowing down the bike sufficiently, and the ViP has to brake: case 5.

One of the most important position-critical motions is a special case of braking: *stopping*, as initiated by the HL command 'Velocity $= 0$ after d meters'. Stopping a bike is accomplished by braking, followed by either putting one foot or two feet on the ground or by stepping off the bike, collectively called 'getting off the bike'. The procedure of stopping consists of 'start braking', 'brake', and 'stop braking'. The last phase can mean getting off the bike (when $v_{goal} = 0$) or resume pedalling motion. All three phases will have to be carried out precisely within the distance d given in the HL command.

Starting to cycle is triggered by a high level command specifying a target velocity greater than zero issued to a ViP that is standing still. The actual motion of getting on the bike depends on the current posture of the ViP, which itself depends on the motion used to get off the bike somewhere in the past. For each of the methods of getting off the bike, a corresponding motion must be available to get on again. *Standing still* is a separate state of the ViP. Basic motions are required for standing still and getting on the bike.

3.4.4 Bicyclist curve-based motion

The simplest extension of straight-line motion is moving along a *curve* in the horizontal ground plane. The ideal type of curve will satisfy four important properties:

- resemble motion paths of real humans;
- suitable to describe motions in a traffic environment;
- easy to specify by the SLM module;
- efficient to calculate.

An obvious choice for curves describing movement are Bézier or spline curves. In the case of the traffic environment (consisting of straight or curved roads and intersections) all that is needed are straight line segments and circle arcs. Therefore we will add circular turns to the motion repertoire.

Two HL commands suffice to specify all 2D motions of the bicyclist:

- Speed $= v_{goal}$ (m/s) after d (m)
- Radius $= r$ (m) after d (m)

A left turn has a positive radius, a right turn a negative one, see Fig. 5. Motion along a straight line is specified by setting the radius equal to zero. Each new HL command specifies a new path segment. The midpoint of the circle is calculated at the moment that the ViP switches to the new circle segment. A path segment is called *active* when the ViP is on the segment. A segment ends when another segment becomes active. The position of the ViP on the end of the current segment has to be calculated first. Then the ViP is moved along the new segment covering the remaining distance.

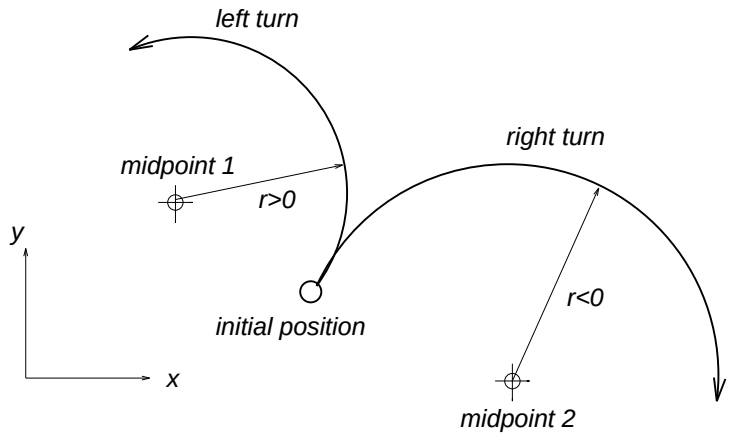


Figure 5: *Left and right turns are specified by a positive or negative radius.*

When riding along circles the bicycle as well as the posture of the ViP changes from upright towards a slanting position. The 'banking angle' α, which is the slant angle with respect to the vertical position, has to be equal to the stable banking angle which can be computed from the physics of centripetal motion. It is given by

$$\alpha = \arctan(\frac{v^2}{r\,g})$$

Velocity v and path radius r refer to the centre of mass, and g is the gravitational constant. Clearly, α increases when the radius of the circle gets smaller. In reality, there are more complicated adjustments of the front and rear wheel of the bicycle, but we have chosen not to model those. When exiting the turn, the bicyclist has to stop banking and therefore is pushed into an upright position.

4 Prototype system

In this section we give an overview of a prototype system which has been implemented, including a simple graphical user interface taking the role of control (second level of motion), discuss

the implementation of the bicyclists as a test-case for the developed method, and give some test results on different platforms for given (graphical) complexity of the bicycle and bicyclist.

In developing the prototype system additional tools were used, such as the PRIMAS motion capture system (van Kasteren 1994) (cf. Section 4.3), *Wavefront Kinemation* human modelling and animation software, *Wavefront Model* 3D object modeller, and IRIS *Performer*, 'a high performance multiprocessing toolkit for real-time 3D graphics' (Rohlf & Helman 1994). *Performer* consists of a low-level library for high performance image drawing and a high-level library that implements, amongst other things, pipelined parallel traversals of scenes defined by hierarchical graphs. It provides functions to create objects, lights, groups of objects, animate objects, handle levels of detail, construct world databases in a tree-like manner, set and control viewports, display views of these viewports into the database and perform collision detection. Additional functions are available for vector and matrix calculations, and multiprocessing control and management.

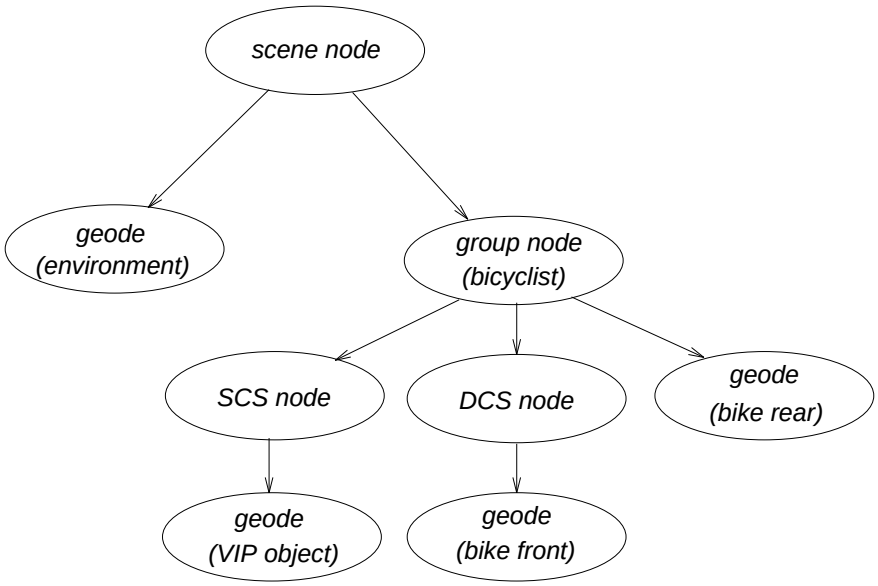


Figure 6: *IRIS* Performer *database structure consisting of an environment and one bicyclist.*

In the current implementation, the graphical mapping module shown in Fig. 2 is absent. Basic motions are not defined in terms of a skeleton, but directly in terms of the graphical model. The captured motions defined on the skeleton are transformed in a preprocessing stage to basic motions defined on the graphical model using the Kinemation software. In this way we bypass the mapping process by providing the polygon structure directly. When this mapping problem is solved efficiently, it will become possible in the future to use basic motions defined on the skeleton. A second missing feature is Level-Of-Detail (LOD) switching. Depending on the size of the object on the screen, a graphical model is chosen out of a set of models with decreasing detail. As each basic motion is currently defined directly on the graphical model, the complete set of basic motions would have to be defined on each model. When motions are defined on the skeleton, the model mapping module would map the model chosen by the LOD algorithm on the skeleton.

4.1 Graphical modelling

Because 3D graphics are handled by IRIS *Performer*, only triangle primitives are used, which are rendered very efficiently by the SGI graphics hardware. A graphical model of a human person has been created with *Wavefront Model*, consisting of 224 vertices and 280 shaded polygons. All polygons are double-sided by default, so a further saving by a factor of 2 is possible. Texture mapping is absent at the moment. A simple bike has also been designed with *Wavefront Model*, consisting of 236 vertices and 178 polygons. To keep the number of polygons as low as possible, the wheels each consist of one double-sided polygon.

Performer functions are applied to create a database with a tree structure consisting of an environment and a bicyclist. The elementary node (the geometry node or *geode*) is the parent of a number of geometry sets or *geosets*: structures that contain the actual geometric primitives of objects. The database tree is shown in Fig. 6.

There are three objects involved: front and rear part of the bike, and the ViP object. An additional transformation with respect to the parent node is used to place the ViP on the 'saddle'. Personification parameters for bike and bicyclist are: (i) Name identifier (ii) Radius of the front and rear wheel; (iii) Pedal-to-rear-wheel angular speed reduction; (iv) Tire friction constant; (v) Mass of the bike; (vi) Distance between front and rear wheel; (vii) Offset to saddle; (viii) Steering axis; (ix) Object files for the front and rear geometry of the bike.

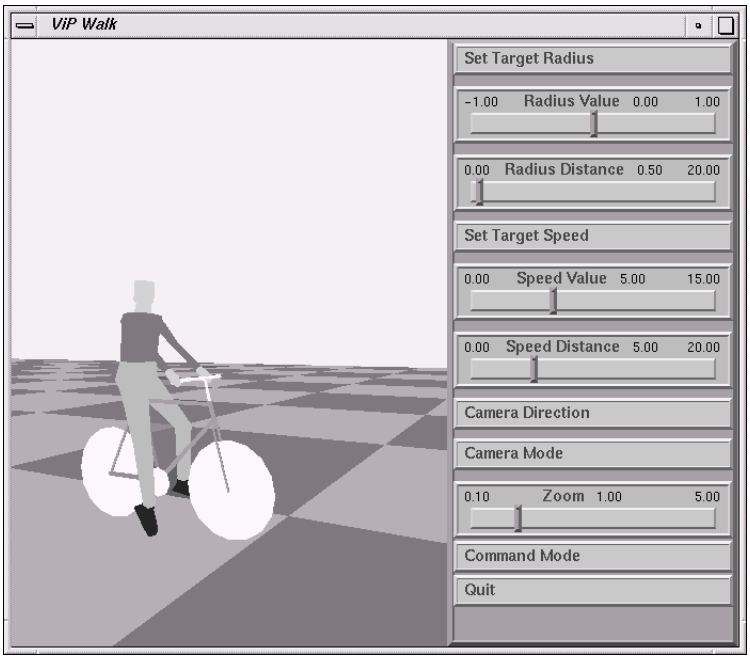


Figure 7: *Screenshot of the prototype system showing a single frame and the user interface.*

4.2 Motion specification

Specification of motion in the prototype system consists of two parts: the user interface and the state machine, as described in Section 3.3. The finite state machine used is shown in Fig. 4. The available HL commands for the bicyclist in the prototype are:

- Speed = v_{goal} (m/s) after d (m)
- Radius = r (m) after d (m)

Personification commands have not yet been included.

The user interface of the prototype system is shown in Fig. 7. There are buttons to control the camera mode (fixed or tracking) and camera position (side view or top view). Sliders control parameter values, such as desired speed or radius, which trigger the appropriate HL commands.

4.3 Motion generation

The basic motions used in the prototype system have been obtained using the PRIMAS motion capture system developed at the Delft University of Technology, The Netherlands (van Kasteren 1994). The system works with reflectors (markers) attached to the body of the test actor. Motions are captured by several cameras up to a frame rate of 200 fps. After recording the motions, special software is used to combine the digital recordings of the cameras resulting in 3D positions of the markers in the recorded frames. All motions have been obtained using only two cameras, so that only one side of the test person could be captured. The other side is obtained as a mirrored version of the captured motion, that has been offset in time. This works fine in the case of cyclic symmetrical motions such as cycling and walking straight on. In other cases (start and stop cycling), it is possible to create the other side of the motion using key-frame animation. Cycling motions have been captured with the bicycle fixed to the ground.

A skeleton consisting of 14 joints was created based on the marker positions of the captured motions and animated using keyframes or by importing captured motion. The graphical model was imported in *Wavefront Kinemation* and attached to the skeleton. External motion generation uses the steps discussed in Section 3.4.2. A number of different motion types as implemented in the prototype system are shown in Fig. 8.

4.4 Performance results

4.4.1 Real-time performance

In the prototype system, the major bottleneck in the simulation loop is the drawing process. The system has been tested on three SGI platforms.

- INDY workstation with a MIPS R4000 processor and software GL implementation. When the window is very small (80 by 60 pixels), the simulation runs at the maximum frame rate of 60 fps. Full-screen simulation (1280 by 1024 pixels) runs at a frame rate of about 2 fps.
- SKYWRITER equipped with four MIPS R3000 processors and a dual VGXT graphics hardware pipeline. Full screen simulation (1280 by 1024 pixels) runs at 30 fps. When the window is reduced to a quarter of the screen (640 by 512 pixels), the frame rate increases to 60 fps. The pixel-fill rate of the VGXT pipeline is the main cause of lower frame rates at high resolution.

- ONYX equipped with four MIPS R4400 processors and Reality Engine^2 graphics hardware. This provides 60 fps full screen animation (1280 by 1024 pixels) running on one processor. The combination of full screen anti-aliasing and 60 Hertz update results in very smooth motion.

4.4.2 Interactive control

The standard control method of specifying a HL command with a distance that describes when the command must be obeyed may be useful for the second level of motion (SLM) module in the driving simulator environment, but it isn't a suitable method for the user controlling ViPs via the graphical user interface of the prototype system. Specifying commands that are directly executed provides the user with a more intuitive control.

4.4.3 Natural motion

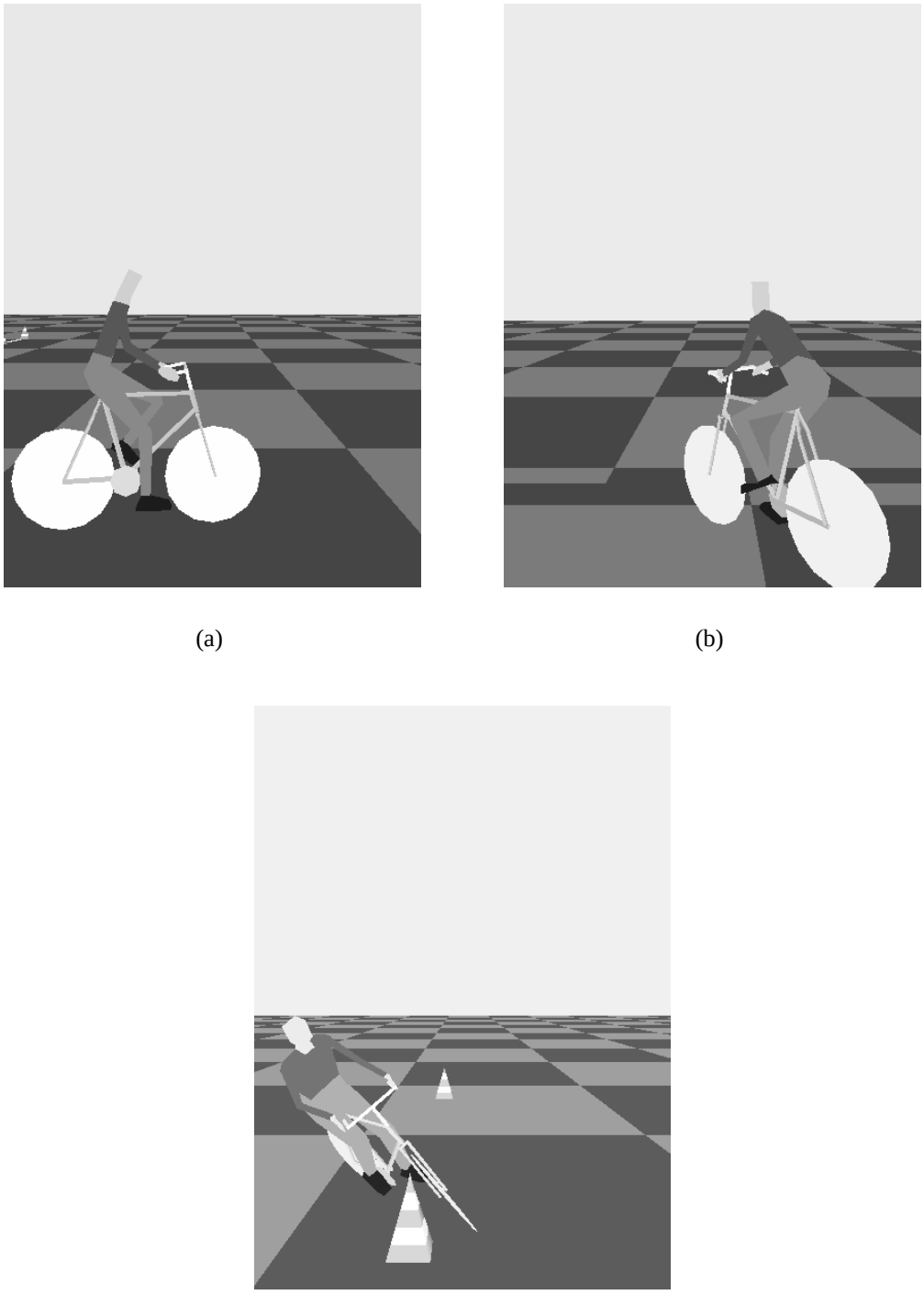
Because motion capture is used, internal motion generation is realistic, although improvements in captured motions and subsequent keyframe animation are still possible. Not all current motions connect seamlessly, which is visually quite disturbing. Linear interpolation of pedalling motion looks convincing, even though the behaviours of the upper body in the two basic motions are quite different. A problem is the transition from start-braking to pedalling with variable effort, because the ViP cannot alter its posture to ensure a smooth connection with the next motion. Steering left and right by rotating the hands and adjusting the elbows looks remarkably well. External motion generation is realistic, even though the path of the ViP consists only of straight and circular segments. However, when the bicyclist is cycling fast, a transition from a left to a right turn or vice versa results in an unnatural rapid change of banking angle.

5 Summary and conclusions

In this paper we have addressed the problem of real-time graphical animation of bicyclists and/or pedestrians as an extension of an existing driving simulator. Requirements are real-time performance, interactivity and naturalness of motion.

A system design has been presented containing a number of modules, of which motion specification (MS) and motion generation (MG) modules are the most important. Control flow is on two levels of motion. The first level of motion consists of the primary motions such as taking a step, taking a turn, or getting off a bike. The behaviour of traffic participants in the virtual world forms the second level of motion (SLM). Other modules are the ViP database and the graphical mapping of the skeleton model of the pedestrians and bicyclists.

From an evaluation of the prototype system which has been implemented, we draw the conclusion that real-time interactively controlled computer animation of pedestrians and bicyclists is feasible on modern simulation systems. The combination of motion capture and interpolation for internal motion generation with simple curve following algorithms for external motions results in a bicyclist that is flexible enough to travel through a flat traffic environment such as provided by a driving simulator. Having a set of pre-generated internal motions is crucial to ensure reasonable performance. Motion capture is the most obvious way to generate natural motion. Altering the motions while the simulation is running is possible even with very simple



(a) (b)

(c)

Figure 8: *Different types of cycling motion. (a): cycling with medium effort; (b): cycling with maximum effort; (c): making a turn.*

algorithms such as linear interpolation. Implementation of the ViPs in the simulator will require a powerful platform like a multiprocessor Onyx with Reality Engine^2 graphics hardware in order to obtain acceptable frame rates.

Many improvements of the prototype system are possible. One is achieving more graphical realism with less polygons by a more careful design of the models. This should make it feasible to simulate a dozen of pedestrians and bicyclists at an acceptable frame rate. Basic motions should be defined on a skeleton structure instead of directly on the graphical model, reducing the required amount of memory and speeding up the interpolation process. An efficient method has to be developed to map a graphical model on the skeleton. Also Level-Of-Detail (LOD) switching is a desired addition to the system. A method is needed to ensure smooth transitions of motions, even when the basic motions themselves do not connect smoothly. A useful feature is to allow individual cycles of periodic basic motions to be interrupted by HL commands. This will enable the ViPs to respond quicker to new HL commands, which is important in emergency situations. It will be necessary to include more physical properties into the system to limit the number of unnatural reactions. Complete 3D motion freedom will be required in general environments with viaducts with access ramps and sloping roads. Finally, the prototype system has to be integrated in the TRC Driving Simulator, with ViPs controlled by a SLM module which guides the ViPs through the traffic environment.

References

Badler, N. I., Barsky, B. A. & Zeltzer, D., eds (1991), *Making Them Move: Mechanics, Control, and Animation of Articulated Figures*, Morgan Kaufmann Publishers inc, San Mateo California.

Badler, N. I., Phillips, C. B. & Webber, B. L. (1993), *Virtual Humans and Simulated Agents*, Oxford University Press, New York, NY.

Girard, M. (1989), The Computer Animation of Legged Animals: Simulation, Design and Control, PhD thesis, Ohio State University.

Hodgins, J. K., Wooten, W. L., Brogan, D. C. & O'Brien, J. F. (1995), Animating human athletics, *in* 'ACM Siggraph Proceedings'.

Magnenat-Thalmann, N. & Thalmann, D. (1985), *Computer Generated Images*, Springer-Verlag, New York–Heidelberg–Berlin.

Magnenat-Thalmann, N. & Thalmann, D. (1990), *Synthetic Actors in Computer Generated Films*, Springer-Verlag, New York–Heidelberg–Berlin.

Maiocchi, R. (1996), 3-D character animation using motion capture, *in* N. M. Thalmann & D. Thalmann, eds, 'Interactive Computer Animation', Prentice-Hall, Englewood Cliffs, NJ, chapter 2, pp. 10–39.

Morawetz, C. L. & Calvert, T. W. (1990), 'Goal-directed human animation of multiple movements', *Proceedings of Graphics Interface* pp. 60–67.

Rohlf, J. & Helman, J. (1994), Iris performer: A high performance multiprocessing toolkit for real-time 3d graphics, *in* 'ACM Siggraph Computer Graphics Proceedings, Annual Conference Series', p. 381.

van Delden, M. J. B. (1995), Real time computer animation of interactively controlled bicyclists and pedestrians, Master's thesis, Dept. of Computing Science, University of Groningen, the Netherlands.

van Kasteren, J. (1994), 'Primas zorgt voor natuurlijker beweging in cartoons', *Delft Integraal (in Dutch)* **4**.

van Overveld, C. W. A. M. (1991), 'An iterative approach to dynamic simulation of 3-d rigid-body motions for interactive computer animation', *The Visual Computer* **7**, 29.

van Overveld, C. W. A. M. & Ko, H. (1994), 'Small steps for mankind: Towards a kinematically driven dynamic simulation of curved path walking', *Journal of Visualization and Computer Animation* **5**, 143.

van Wolffelaar, P. C. & van Winsum, W. (1995), Traffic modelling and driving simulation - an integrated approach, *in* 'Proceeding of the Driving Simulator Conference DSC '95, Sophia Antipolis, France, September 12-13, 1995', Neuf Associes, Teknea.