Local and Global Control in Computer Animation

José Tarcísio Franco de Camargo[1]
Léo Pini Magalhães[2]
Alberto Barbosa Raposo[3]

Departamento de Engenharia de Computação e Automação Industrial (DCA)
Faculdade de Engenharia Elétrica (FEE)
Universidade Estadual de Campinas (UNICAMP)
13081–970 Campinas, SP, Brasil

[1]tarcisio@dca.fee.unicamp.br
[2]leopini@dca.fee.unicamp.br
[3]alberto@dca.fee.unicamp.br

Abstract. In the past few years a lot of effort has been done on modelling computer animation schemes through physical laws. Indeed, the behavior of almost all characters in an animation or in a simulation can be represented by mathematical tools, such as differential calculus. On the other hand, interactions among actors and the animator suggest some other approach. Following this line of inquiry we split the computer animation control problem in two parts: a local control problem and a global control problem. The local control defines the "behavior" of each character, while the global control is responsible for the interactions. In this paper we propose an event oriented scheme to solve the control problem. This has been done using concepts regarding to DEDS **(Discrete Event Dynamic System)** and ESM **(Extended State Machines)**.

1 Introduction

The use of computers in modelling the animation and simulation of many kinds of systems is still a recent research field in computer science. Almost all works done in this line of inquiry have focused on the use of physical laws, such as Newton and Euler dynamics, as the basic modelling tools for every type of system that has to be simulated. This implies the exhaustive use of ordinary or partial differential equations for solving any kind of animation problem. Indeed, without the use of mathematical tools such as calculus it would be hard to model the systems around us. But, on the other hand, the complex nature of many systems will require more complex differential equations, which is a commom fact. In addition, there is no universal solution for any problem. That is, a suitable approach for one type of system may not be the best for another one.

As an alternative for solving computer animation problems (i. e. the problem of controlling the flow of an animation) we suggest to split the model in several parts: local control blocks and a global control block, embedded on a event driven strategy ([Kalra (1992)] also used an event oriented approach, but not considering local and global control blocks). Suppose a system to be simulated with many "actors". We can assign to each actor a local control block that contains the "behavioral rules" of the single actor. That is, the physical laws which control a single actor are embedded in its respective local control block. This suggests a mathematical tool, such as differential calculus, for modelling the respective actor and its behavior. For example, the mass distribution of the actor with respect to some coordinate system is given by its inertia tensor matrix; the motion rules of the actor may be described by Newton-Euler dynamic equations.

Since systems we are talking about are composed by many actors there could be interactions among them and the animator, so, the global control block suggests a logical approach for modelling these interactions. That is, when an event occurs, how must an actor respond ? Which actors are affected by this event ? This way, depending on what is "going on" in the system, the actors can behave in such a way that could be different if something else had occurred.

At next we introduce the initial step of our proposal, i.e., splitting the animation process in two classes of control blocks: the global control block and the local control blocks and the event strategy applied to the control.

2 Modelling tools

Consider, for example, figure 1, that represents an animation, where we have two actors (represented by their respective local control blocks) and a global controller, that is responsible for the interactions.

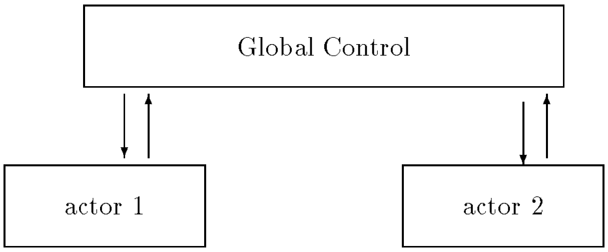


Figure 1: The flow of information among "the actors" of an animation.

According to figure 1 we can see that both actors in this system are connected to the global controller. The connections are made in such a way that each actor has an "outgoing channel" and an "incoming channel" with the global control block. Through these channels the global controller can send and receive any information regarding to the actors. We can also note that this process has feedback loops between the global control block and each actor. In addition, according to the last section, each actor in figure 1 has inside its local control block its behavioral rules and a logical structure that is responsible for the communication with the global controller. On the other hand the global control block contains only a logical structure that is responsible for the coordination of the actors.

Frequently, animators have designed animations through the use of "space-time" constraints. For example, after t_0 move from position p_0 to position p_1. The approch we propose is quite good for this kind of animation and, in addition, it is suitable for modelling animations through the use of "space-time-event" constraints. This means that, when an event occurs in the system, or at any time instant, the global controller may send a proper command to an actor or it can receive any information from an actor about its status or an occurred event. This kind of system is usually called "Discrete Event Dynamic System - DEDS" and, to build a system of this kind, we can use several modelling tools, such as "Extended State Machines - ESM".

ESM features are suitable for modelling "Space-Time-Event" oriented systems and they have been frequently used in our work [Camargo (1993), Camargo (1994), Camargo (1995)]. At next we introduce the basic ESM definitions we use in our research.

2.1 ESM definitions

A basic ESM may be represented by a quintuple **(X, Y, C, L, A)** where, according to [Ostroff (1989)]:

X is a variable whose instace denotes the current state of the machine.

Y is a set of generic data variables that may be used by the ESM.

C is a set of communication channels. A communication channel can be considered as a one direction communication line connecting two ESMs, through which some data can be transmitted.

L is a set of event labels.

A is a set of basic actions.

Each basic action in **A** is given by the quadruple **(cs, guard, operation, ds)**, where:

cs is the current state of the ESM,

ds is the destination state of the ESM,

guard is a boolean-valued expression, and

operation is one of following three possible operations.

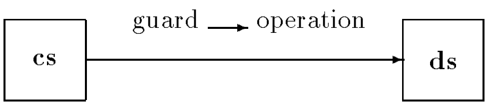
The three possible operations are given by:

1. An **assign operation** given by "α[y:a]" (this means: "a" is assigned to "y" on the occurrence of *alpha*) where "α" is an event label, "y" is a data variable and "a" is an expression. The expression "a" must be of the same type as the data variable "y". The event label may occur by itself in an operation, i. e. [y:a] is optional. Multiple simultaneous assignments in the same operation are denoted "$\alpha[y_1 : a_1, y_2 : a_2]$".

2. A **send operation** is given by "c!m", where "c" is a channel in **C** and "m" is a message. A message is either an expression or an event label.

3. A **receive operation** is given by "c?r", where "c" is a communication channel, and "r" is either a data variable or an event label.

As an example, the graph:



is a pictorial representation of an event. If the **guard** is left out, then it is assumed to be true. An informal interpretation of the above event is: "if the ESM is currently in the state **cs** and if the guard evaluates to true, then the edge is traversed while doing **operation**, after which the ESM will be in state **ds**".

Communicating operations can be used either to command the execution of a specific event or to communicate a message. An informal description of these two types of communication is:

COMMAND TO EXECUTE AN EVENT - The send operation "c!α" in some ESM M_1 is read: "send the command to do "α" over channel "c" (to some other ESM e. g. M_2)". The receive operation "c?α" in M_2 is read: "receive the command to do "α" (e. g. from M_1) over channel "c" and execute the command α ". The send and receive operations, when synchronized, result in the command and execution of the event "α".

COMMUNICATING A MESSAGE - Let "m" be an expression (the value of "m" is the message), and let "r" be a data variable. Then the operation "c!m" in some ESM M_1 is read: "send the message "m" over channel "c" (to some other ESM e. g. M_2)". The corresponding operation "c?r" in M_2 is read: "receive a message (from e. g. M_1) over channel "c" and put the message in "r" ". The send and receive operations togheter result in a distributed assigment of "m" to "r".

Note that, a send operation "c_1!m" matches a receive operation "c_2?r" if $c_1 = c_2$ and, either "m" and "r" are the same event label, or "r" is a data variable and "m" is an expression with the same type as "r". A send action from some ESM "M" cannot have a matching receive action in "M" as there is no internal communication within an ESM ifself (a channel represent a one-to-one connection from "M" to some other ESM).

The above definitions do not complete the entire description of ESMs, but they are sufficient for the purpose of this paper. Further details can be found in [Ostroff (1989)].

3 An Example: modelling a bipedal structure

Consider the bipedal "actor" shown in Figure 2. The figure presents an articulated human-like body that is composed by two identical articulated structures. Both arms and both legs have the same characteristics, i. e., these articulated chains have the same degrees of freedom. When those structures are joined we have the bipedal figure. In order to simulate this articulated character we can consider two control levels, local and global. The local control is responsible for the effective displacement of each leg, that is, a controller of this kind must control the motion of a single leg from one place to another. There are several classical kinematic or dynamic techniques that can be applied to solve this local control problem. The solution we found for the local control problem is much like those presented in [Korein (1982), Craig (1989), Goldenberg (1985), McInnis (1986)], and it can be extended to the solutions presented in [Morgan (1985), Armstrong (1985), Barzel (1988)] for a dynamic model.

On the other hand, it is not enough to control only the motion of each leg individualy, since, for a bipedal structure, there must be coordination between the legs, otherwise we can not move the entire body. For example, both legs are not allowed to be out of the ground at the same time when the actor is walking. In this way, when dealing with the interaction between legs, we are facing a global control problem.

The solution for the this control problem can be found in DEDS formulation. Our global control approach is partially based on the model presented in [Girard (1985)], and the final solution that we propose uses the same tools as those presented in [Ho (1989), Ostroff (1989), Cohen (1989)].

Note that the schem presented in fig 1 represents exactly the control problem that we are dealing.

For this structure, a **gait pattern** describes a sequence of lifting and placing of the feet. The pattern repeats itself as the figure moves: each repetition of the sequence is called the gait cycle. The time (or number of frames) taken to complete a single gait cycle is the period **P** of the cycle.

During each gait cycle period any given leg will spend a percentage of that time on the ground. This fraction is called the duty factor of the leg. In our example, the duty factor may be used to distinguish between the walking and running gaits of bipeds. In addition, we can define the stroke as the distance traveled by the body while one leg is still on the ground.

We can see each one of the legs as a basic ESM where, associated with each leg "i", we have the following states:

S_i The **support state**. We will call the time a leg spends on the ground its **support duration**. Its respective phase is the support state.

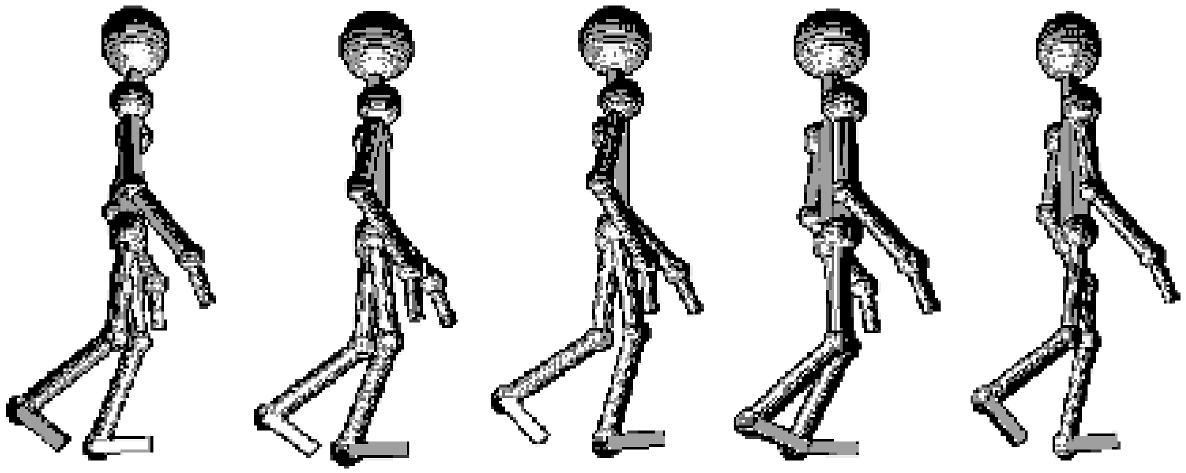


Figure 2: Frames generated for a bipedal structure simulation.

T_i The **transfer state**. We will call the time a leg spends in the air its **transfer duration**. Its respective state is the transfer state.

We still have associated with each leg "i" the following events:

α_i The leg is lifting the ground.

β_i The leg is placing on the ground.

The flow of information between the controller and each leg is done through the communication channels:

m_i Send information from the leg to the controller.

c_l Send information from the controller to the leg.

As a basic control specification we have:
It is not allowed to both legs be in the transfer state at the same time when the body is walking.
For each leg "i", (L_i), we can build the following ESM:

ESM L_i:
$L_i = [(x_i), \emptyset, (m_i, c_i), (\alpha_i, \beta_i), A_i]$
$(x_i) = (S_i, T_i)$

$$A_i = [\ [T_i, \beta_i, \emptyset, S_i]\ ,\ [S_i, c_i?\alpha_i, \emptyset, T_i]\ ,\ [S_i, TRUE, m_i!x_i, S_i]\ ,\ [T_i, TRUE, m_i!x_i, T_i]\ ]$$

The complete system must also consider the torso and the arms of the body. However, for the sake of simplicity, we will only treat the legs in this paper. Now, we will treat the global control problem under a simple point of view. In this case, the actor will move in a straight line indefinitely, and the controller may be represented as follow:

ESM Controller:
Controller =
$[(x_c), (y_l, y_r), (m_l, m_r, c_l, c_r), (\alpha_l, \alpha_r), A_c]$
$(x_c) = (L_1, L_2, L_3, L_4)$

$$A_c = [\ [L_1, m_l?y_l, \emptyset, L_2]\ ,\ [L_2, y_l = T_l, \emptyset, L_1]\ ,\ [L_2, y_l = S_l, c_r!\alpha_r, L_3]\ ,\ [L_3, m_r?y_r, \emptyset, L_4]\ ,\ [L_4, y_r = T_r, \emptyset, L_3]\ ,\ [L_4, y_r = S_r, c_l!\alpha_l, L_1]\ ]$$

A pictorial representation for the set actors - controller is shown in Figure 3.

4 Implementation

We have implemented a system called TOOKIMA (*a TOOl KIt for scripting computer Modeled Animation*) [Hounsell (1992)], that defines a set of tools for the algorithmic description of animations. Some concepts used in TOOKIMA can be found in [Thalmann (1985)]. This system is able to describe the kinematic/dynamic motion of computer modeled objects.

The TOOKIMA system is, indeed, composed by three sub-systems:

Object Modelling Tool: With this sub-system we can model the objects that will take part in the animation process.

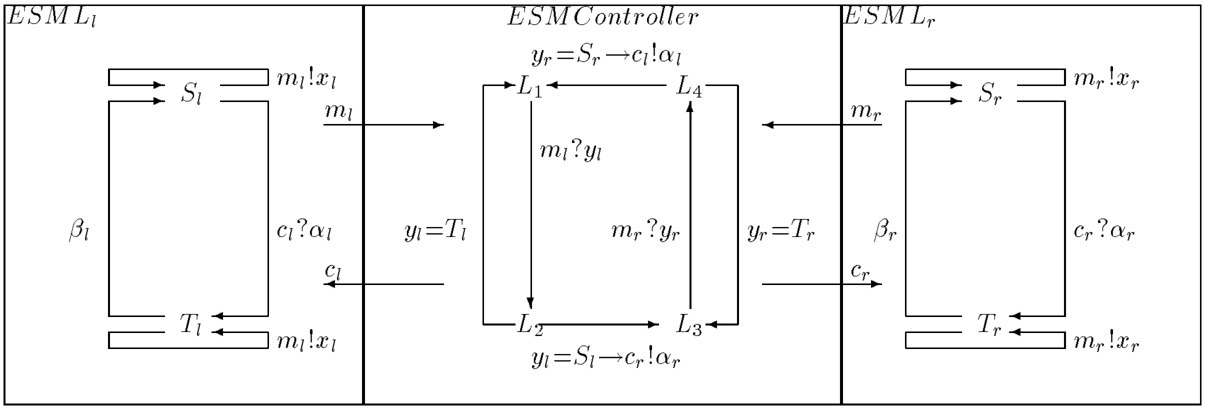


Figure 3: The system actors || controller.

Visualization Tool: A Scanline based algorithm is used for the visualization of the frames generated by the system.

Scripting Tool: This sub-system is responsible for the integration of the two above. In addition, it provides a "language" for temporal synchronization and description of the scenes that take part in the animation. A script of an animation process is described in this pseudo-language that contains all information about the objects, the forniture and the actions that will compose the animation.

According to TOOKIMA's structure, we have built the ESMs of these examples as "sub-routines" in the script of the animation. The script for the frames presented in figure 2 is:

```
/* Definition of ESMs and initial states */
DEF_ESM(CTRL,"L1")
DEF_ESM(LEFT_LEG,"SL")
DEF_ESM(RIGHT_LEG,"SR")

/* Definition of events */
DEF_EVENT(BL)
DEF_EVENT(AL)
DEF_EVENT(BR)
DEF_EVENT(AR)

/* Definition of channels */
DEF_CHANNEL(ML)
DEF_CHANNEL(MR)
DEF_CHANNEL(CL)
DEF_CHANNEL(CR)

TIME_INTERVAL(0.,11.)
SAMPLES(12)

/* External functions */
FUNCTION(move_right_leg)
FUNCTION(move_left_leg)

START_SIMULATION

/* ESM Controller */
BEGIN_ESM(CTRL)
CHANNEL(ML)
CHANNEL(MR)
CHANNEL(CL)
CHANNEL(CR)
ACTION(CTRL,"L1",MATCH(ML,"TL"),NO_TASK,"L1")
ACTION(CTRL,"L1",MATCH(ML,"SL"),CR="AR","L2")
ACTION(CTRL,"L2",MATCH(MR,"TR"),NO_TASK,"L2")
ACTION(CTRL,"L2",MATCH(MR,"SR"),CL="AL","L1")
END_ESM

/* ESM Left Leg */
BEGIN_ESM(LEFT_LEG)
CHANNEL(ML)
CHANNEL(CL)
EVENT(BL)
ACTION(LEFT_LEG,"SL",TRUE,ML=LEFT_LEG,"SL")
ACTION(LEFT_LEG,"SL",MATCH(CL,"AL"),
       move_left_leg(),"TL")
ACTION(LEFT_LEG,"TL",TRUE,ML=LEFT_LEG,"TL")
ACTION(LEFT_LEG,"TL",BL,NO_TASK,"SL")
END_ESM

/* ESM Right Leg */
BEGIN_ESM(RIGHT_LEG)
CHANNEL(CH1R)
CHANNEL(CH2R)
EVENT(BR)
ACTION(RIGHT_LEG,"SR",TRUE,MR=RIGHT_LEG,"SR")
ACTION(RIGHT_LEG,"SR",MATCH(CR,"AR"),
       move_right_leg(),"TR")
```

```
ACTION(RIGHT_LEG,"TR",TRUE,MR=RIGHT_LEG,"TR")
ACTION(RIGHT_LEG,"TR",BR,NO_TASK,"SR")
END_ESM

END_SIMULATION
```

At the top of the simulation file we define the ESMs, the events, and the channels that will take part in the animation. At next, we have the timming definitions of the animation and the definitions of the external functions. These external functions contain the behavioral rules of the motion of each leg. This may be any set of kinematic or dynamic equations. Finally, we have the set of possible actions that an actor may perform.

As we can see in this script, the global control block is composed only by a set of logical expressions used to manage the interactions among the other blocks. On the other hand, each local control block has a set of logical expressions to interact with the controller and, also, a function that will perform the efective displacement of each leg, according to the approach initially proposed.

5 Conclusions

"Space-Time Constraints" paradigm is, today, the most commom modelling tool for computer animation. It is frequently used for modelling many kind of systems around us. However, it is deficient when considering the presence of discrete events in the system, or even considering a large set of possible interactions among the actors in an animation.

On the other hand, any animation process can be thought as a sequence of discrete events where the "actors" in the system are changing their states according to the occurrence of the events. Thinking this way, we can model animations as discrete event dynamic systems (DEDS). A quite commom tool for modelling DEDS is the extended state machine (ESM) that appears to be suitable for applications in computer modeled animation or simulation.

Following this line of inquire, we have considered in this paper an "extended" paradigm for computer modeled animation. "Space-Time-Event Constraints" paradigm appers to cover a lot more than "Space-Time". The walking model for an articulated bipedal structure presented here splits a complex tree structure in several smaller blocks, simplifying the initial problem.

The use of local and global controllers are suitable for the implementation of a wide range of animation models. If we consider the global and local controllers as ESMs, it allows us to split an initial complex problem in several smaller problems.

In addition, the global control structure let us free to think about the local control blocks. That is, no matter which local control technique we adopt, the global control block does not change. This way, when solving the local control problem we can choose among several techniques such as a kinematic or a dynamic model.

6 References

W. W. Armstrong and M. W. Green, *The dynamics of articulated rigid bodies for purposes of animation*, The Visual Computer, **1**,**4**,231-240 (1985)

R. Barzel and A. H. Barr, *A modeling system based on dynamic constraints*, Computer Graphics, **22**,**4**,179-188 (1988)

L. S. Brotman and A. N. Netravali, *Motion interpolation by optimal control*, Computer Graphics, **22**,**4**,309-315 (1988)

J. T. F. de Camargo and L. P. Magalhães, *Técnicas de controle em animação*, Tutorial presented at VI SIBGRAPI, Recife, PE (1993)

J. T. F. de Camargo, L. P. Magalhães and A. B. Raposo, *Modeling motion simulation with DEDS*, Proc. of 13^{th} IFIP World Computer Congress-94, **2**,162-167 (1994)

J. T. F. de Camargo, *Animação modelada por computador: técnicas de controle de movimento*, Ph.D. Thesis, DCA/FEE/UNICAMP (1995)

G. Cohen, P. Moller, J. Quadrat and M. Viot, *Algebraic tools for the performance evaluation of discrete event systems*, Proc. IEEE, **77**,**1**, 39-57 (1989)

J. J. Craig, *Introduction to robotics: mechanics and control*, 2^{nd} ed, Addison-Wesley Publishing Company (1989)

M. Girard and A. A. Maciejewski, *Computational modeling for the computer animation of legged figures*, Computer Graphics, **19**,**3**,263-270 (1985)

A. A. Goldenberg, B. Benhabib and R. G. Fenton, *A complete generalized solution to the inverse kinematics of robots*, IEEE Journal of Robotics and Automation, **RA-1**,**2**,14-20 (1985)

Y. Ho, *Dynamics of discrete event systems*, Proc. IEEE, **77**,**1**, 3-6 (1989)

M. S. Hounsell and L. P. Magalhães, *TOOKIMA - animação cinemática*, Proc. of V SIBGRAPI (1992)

D. Kalra and A. H. Barr, *Modelling with time and events in computer animation*, Proceedings of EUROGRAPHICS-92, **11**,**3**,35-58, (1992)

J. U. Korein and N. I. Badler, *Techniques for generating the goal - directed motion of articulated structures*, IEEE Computer Graphics and Applications, 71-81, November (1982)

B. C. McInnis and C. F. Liu, *Kinematics and dynamics in robotics: a tutorial based upon classi-*

cal concepts of vectorial mechanics, IEEE Journal of Robotics and Automation, **RA-2**,**4**,181-187, (1986)

R. G. Morgan and U. Ozguner, *A descentralizated variable structure control algorithm for robotic manipulators*, IEEE Journal of Robotics and Automation, **RA-1**,**1**,57-65 (1985)

J. S. Ostroff, *Temporal logic for real time systems*, Research Studies Press Ltd., John Wiley & Sons Inc. (1989)

N. M. Thalmann and D. Thalmann, *Computer Animation - Theory and Practice*, Springer-Verlag (1985)

A. Witkin and M. Kass, *Spacetime constraints*, Computer Graphics, **22**,**4**,159-168 (1988)