1988

# Model Generation and Modification for Dynamic Systems from Geometric Data

Christoph M. Hoffmann
*Purdue University*, cmh@cs.purdue.edu

John E. Hopcroft

Report Number:
88-767

# MODEL GENERATION AND MODIFICATION FOR DYNAMIC SYSTEMS FROM GEOMETRIC DATA

Christoph M. Hoffmann
John E. Hopcroft

CSD-TR-767
April 1988

# MODEL GENEATION AND MODIFICATION FOR DYNAMIC SYSTEMS FROM GEOMETRIC DATA

Christoph M. Hoffmann*
John E. Hopcroft†

Computer Sciences Department
Purdue University

Technical Report CSD-TR-767
CAPO Report CER-87-23
April, 1988

Abstract

We are experimenting with a prototype implementation of a simulation system for rigid body motion where the objects to be simulated are specified by a geometric description and a few symbolic data, such as object density, the type of hinge between certain bodies, and evnironmental factors. In our approach, the system automatically generates the appropriate mathematical model describing the current system dynamics, and revises the model so as to account for unanticipated changes in the system's topology. For example, when simulating impact behavior, the system will not know in advance what objects might collide at which points, thus invalidating the usual approach of defining constraint forces between point pairs that are negligible except when the points are in close proximity. We describe the overall organization of the system, and give a general method to effect this self-modification. Focusing on characteristic cases such as gaining or losing contact and low-velocity collision, we discuss our experience with the system, its flexibilities, and its limitations.

* Research supported in part by NSF grants CCR-8619817 and ONR contract N0014-86-K-0465, and a grant from the AT&T Foundation.

† Computer Science Department, Cornell University, Ithaca, NY 14853, USA. Supported in part by NSF Grant DMC 86-17355, and ONR Contract N00014-86-K-0281.

# Model Generation and Modification for Dynamic Systems from Geometric Data

Christoph M. Hoffmann*
John E. Hopcroft†

**Abstract**

We are experimenting with a prototype implementation of a simulation system for rigid body motion where the objects to be simulated are specified by a geometric description and a few symbolic data, such as object density, the type of hinge between certain bodies, and environmental factors. In our approach, the system automatically generates the appropriate mathematical model describing the current system dynamics, and revises the model so as to account for unanticipated changes in the system's topology. For example, when simulating impact behavior, the system will not know in advance what objects might collide at which points, thus invalidating the usual approach of defining constraint forces between point pairs that are negligible except when the points are in close proximity. We describe the overall organization of the system, and give a general method to effect this self-modification. Focusing on characteristic cases such as gaining or losing contact and low-velocity collision, we discuss our experience with the system, its flexibilities, and its limitations.

## 1 Introduction

Traditional approaches to simulation and analysis of mechanical systems proceed as follows:

*Computer Science Department, Purdue University, West Lafayette, IN 47907, USA. Supported in part by NSF Grant CCR 86-19817, ONR Contract N00014-86-K-0465, and a grant from the ATT Foundation

†Computer Science Department, Cornell University, Ithaca, NY 14853, USA. Supported in part by NSF Grant DMC 86-17355, and ONR Contract N00014-86-K-0281

1. Formulate a set of algebraic and differential equations that describes the behavior of the system, for example, as an initial value problem in time. This set of equations is the *mathematical model* of the mechanical system.

2. Discretize the mathematical model (in space and/or time) and integrate it using some numerical scheme appropriate to the characteristics of the equation system.

3. If possible, estimate the error incurred during the previous step comparing the computed approximate solution with the exact solution of the mathematical model.

4. Validate the results by experimentation. This step is only sometimes included.

The geometric characteristics of the system are usually abstracted out. For example, in discrete mechanical problems, extended bodies are replaced by points with identical mass and inertia properties.

There are several reasons why geometry is not an integral part of the process. For one, the mathematical model is a precise object that can be analyzed and studied using mathematics. Moreover, truly accounting for complex geometric shapes sharply increases the computational load and cannot be done with satisfactory speed unless parallel hardware or very fast single processor machines are used. Until recently, the cost of such hardware has fully justified the traditional desire to simplify or abolish altogether the geometry of the mechanical system. Moreover, many mechanisms possess a sufficiently simple structure, so that the kinematics of its parts can be understood beforehand. Thus, all potential motion of these parts can be known in principle, and often can be accounted for in the formulation of the mathematical model. Many excellent simulation and animation programs for mechanical systems are based on this principle, e.g., [1,3]. Gilmore develops a similar approach in [4], but limits himself to a two-dimensional world.

It is a simple fact that many mechanical systems are not naturally described in time by a single mathematical model. Rather, as time evolves, so must the set of equations describing the system's behavior. For example, consider a block resting on a table. As the block is pushed towards the edge of the table, there is a kinematic hinge between the table and the block with one rotational and two translational degrees of freedom. As the block begins to move over the edge of the table, the nature of this hinge changes, and a new degree of freedom must be described corresponding the possible rotation of the block about the edge of the table. Finally, as the block falls, no kinematic relationship exists between the table top and the block. Thus, the evolution of this mechanical system is best described by three different mathematical models, used in sequence, each describing the respective topological nature of the system, as it exists at that time.

A major device to accommodate predictable intermittent geometric contact is to formulate a fictitious contact force between the two points known to come into contact. This force is postulated to depend on the distance between the two points. At larger distances, the force is negligible, but it rises sharply when the two points are in close proximity of each other. Thus, the contact force is a substitute for the impulse that would be transmitted if the two points collided. Evidently, this approach requires a priori knowledge of the locus of two points. Without accurate and complete knowledge of possible contact, the model is incapable of approximating real behavior.

From a computer science perspective, a periodic reformulation of the mathematical model is in principle straightforward. Technically, it requires a modicum of symbolic processing, capable of formulating and manipulating equations efficiently. However, the correct formulation of the relevant equations interfaces with the geometry of the system's current state, and it becomes necessary to automate the equation formulation based on *geometric* data.

In this paper, we review Project Newton, an ongoing development project for simulating mechanical systems based on geometric models and their spatial interrelationship. The approach chosen is to derive automatically the relevant mathematical models from the geometric data, and to reformulate these models throughout the simulation, whenever appropriate. The work stresses flexibility and generality, rather than efficiency that might be possible by exploiting special circumstances prevailing at certain times, such as tree-structured component connectivity.

## 2 User Input and Translation

The Newton system accepts as input a description of the mechanical world that is to be simulated. This description is structured in several ways: Objects are either *primitive*, i.e., rigid bodies of arbitrary shape, or they are *composite*, consisting of variously hinged primitive objects. In each case, the object definition is understood to be *generic*, that is, a single description of an object can be instantiated several times, denoting different objects with the same mechanical characteristics. Final object instantiation happens in the *world* definition where objects are also placed into initial position and orientation, with initial velocities. Moreover, the world definition includes environmental declarations such as the presence or absence of gravity.

The description of individual objects is structured in sections, each describing a physical category understood by the simulation system. Presently, the following sections are understood:

1. Abstract properties: This section describes summary aspects of the object including density and color.

2. Geometric properties: This section describes the shape of the body as well as its *features*. (A feature is a point, line or plane used to place objects in relationship to each other.)

3. Control properties: This section interfaces to external control programs simulating, e.g., actuating devices by supplying external forces at certain times based on current values of the state variables.

4. Dynamic properties: This section describes the motion equations of relevance to the object. The equations are system derived, and the section specifies only whether they should be generated. For example, if the object is assumed to be perfectly controlled, i.e., all accelerations are supplied over time by external control programs, then no motion equations are produced.

5. Interference properties: This section specifies whether to test for collision and impact. If so, the relevant models are generated automatically.

For a composite object, we must describe its constituent parts and how they are related to each other. The parts may be primitive objects or they may be composite objects in turn. Parts are named and may be hinged together with a variety of standard hinges. Possible hinges are springs, dampers, ball and socket hinges, pin hinges, and temporary point/surface contact hinges. When hinging two parts, they are placed in relation to each other, the hinge points or lines are identified, and the hinge type is given. Note that parts can be hinged in an arbitrary topology.

A key aspect of this structured description is that it allows us to include additional physical categories eventually. Although the system does not know about other categories at this time, we envision interfacing the simulation of motion with elasticity, deformation and stress model formulation and evaluation.

The user's description of the world is compiled into two logical levels of data structures. The highest level system routines, implementing the flow of control in the simulation, understand the upper level organization of the data. They use this knowledge to extract lower level data structures and pass those to system components that understand them and can use them for specific computations. For example, objects and hinges are represented at the upper level. When the high level simulation routines determine that some computation involving the geometry of an object must be done, they extract the relevant lower level data structures from the object and pass them on to the geometry processing subsystem.

The structure of the lower level is hidden by access functions that jointly comprise the interface to the major system components and their functionalities. Interfaces exist to the dynamic modeler, to the geometric modeler, to the display subsystem, etc. With this two-tiered structure we are in the position to replace individual system components without having to rewrite the software extensively. For example, we are presently replacing the geometric modeling subsystem. To do so, we only rewrite the interface functions. These functions must understand the conventions by which the respective information is obtained from the new modeler, and include, for example, queries such as volume, volumetric inertia, and whether and how two objects interfere.

## 3 Dynamics Model Creation, Editing, and Evaluation

One of the central functionalities of the software is the creation, evaluation, and modification of the mathematical models describing the system dynamics. These models are created from vectorial equation schemata that are instantiated and edited as needed. The model construction goes through the following stages:

1. An equation is created by instantiating a schema. This involves setting up appropriate data structures and state variables.
2. The equation may be modified according to the presence or absence of newly imposed constraints that require, e.g., constraint forces.
3. The equation is compiled into a set of routines that effectively evaluate the equation during the simulation.

An individual equation consists of named quantities and operations on them. A named quantity is any variable or constant that can be evaluated, such as mass, position, angular acceleration, hinge constraint force, etc. Each quantity has instances that belong to individual objects or object components. For instance, every primitive object has a mass, every hinge of a fixed type gives rise to a constraint force, and so on. As example, consider the creation of the linear acceleration equation for an object linked at a single hinge and being under the influence of gravity. The final equation will have the form

$$m\ddot{r} - X - mg = 0,$$

where $m$ is the mass, $\ddot{r}$ the linear acceleration, $X$ the constraint force exerted at the hinge, and $g$ is the gravitational constant. Note that the other object with which this one is hinged would contain a term $+X$ in its linear acceleration equation.

The equation schema is created in stages:

1. When the primitive object is instantiated as part of some composite object, the equation $m\ddot{r} = 0$ is created. At this time, the quantities $m$, $r$, $\dot{r}$, $\ddot{r}$, $q$, $\omega$, and $\dot{\omega}$ are created for the object. Here, $q$ are the Euler parameters.

2. When the object is linked by a hinge, the constraint force term $-X$ is added, i.e., $X$ is created, and the equation is modified. Also added is a constraint equation summarizing the kinematic constraints due to the hinge.

3. When processing the world declaration, finally, the term $-mg$ is added.

Other hinges on the object would similarly result in further equation modifications. Note that 'hinge' is understood in the most general sense and includes springs, dampers, and contacts with other objects.

Named quantities are grouped into *classes* where each quantity in the class is evaluated or assigned by essentially the same procedure. For example, evaluation of $m$ entails locating the object whose mass is referred to, querying its density and volume, and multiplying the results of these queries. For each class, a unique dictionary entry is created and to it is linked a list of all its members along with a reference to the owning object.

Dynamic equations are compiled for efficiency of evaluation. The compilation strategy is predicated on the fact that the final system of dynamic equations is integrated by repeatedly formulating a linear system $Ax+b=0$ and solving it. To this end, each term in the vector equation is compiled separately. When examining a term, such as $m\bar{r}$, the instance of each named quantity in it must be found. From it, and from the class dictionary, it is determined whether the quantity instance will be known or unknown at the time of integration. In forward dynamics, $\bar{r}$ will be unknown and ultimately will be determined from the external forces, whereas in backwards dynamics $\bar{r}$ is a known quantity that ultimately determines unknown external forces. As long as the final system remains determined and linear, some objects may have known accelerations whereas others may have unknown accelerations.

A vectorial term consisting of known named quantities will evaluate to a vector of known quantities. The scalar components of this term must be added to the corresponding components of $b$ in the final system $Ax+b=0$. Similarly, assume that the term contains unknown quantities, e.g., the components of $\bar{r}$. Then the coefficients of these components must be added in the matrix $A$ in the proper position. Thus, the compilation of a vectorial equation requires

1. assigning to the scalar components of the equation row indices in the final system $Ax+b=0$, and

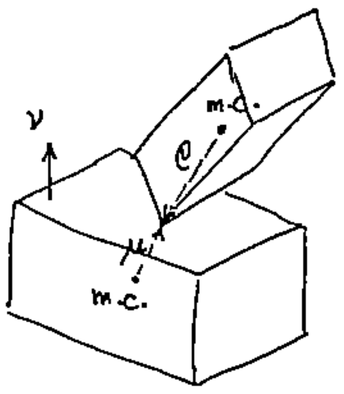


Figure 1: Point/Surface Collision

2. assigning column indices to the unknown occurrences of named quantities.

Thereafter, the terms are compiled into functions that appropriately modify the system $Ax + b = 0$ at the respective positions. For efficiency, constant terms such as $mg$ are recognized in the compilation process, the necessary evaluation is performed beforehand, and the value so determined stored in an instance variable belonging to the object. See also [2].

## 4 Interaction of Geometry and Dynamics during Collisions

The reformulation of a dynamic model during simulation depends on the geometric configuration of the system of objects simulated, and requires interaction between the geometric and dynamic models. As first example, consider two colliding objects. We assume that all collisions take place at velocities that are sufficiently small so that no large deformations ensue. In that case, for point/surface collisions, the impulse exchanged at the point of collision depends on the geometric quantities shown in Figure 1. Summarizing this geometric data as the vectors $\rho$, $\mu$, and $\nu$, we have outlined in [5] how to formulate the collision equations.

The determination of the geometric quantities is a subtle problem. Since the simulation constructs events at discrete points in time, the general sit-

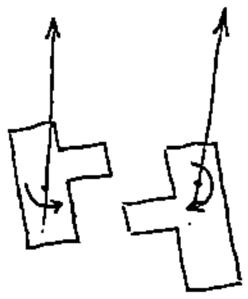


Figure 2: Nonconvex Bodies that Might Interfere

uation is that at time $t$ the objects are apart, and at time $t + \Delta t$ they interpenetrate. The motion of the two objects in between these two points in time is approximately a screw, and even for two cuboids we have no closed form solutions that fix the instant of first contact, on which the geometric quantities depend. If $\Delta t$ is large, moreover, we may not notice a collision. That is, a block may be above a table top at time $t$, and below it at time $t + \Delta t$, without an interference at either point in time. While this specific situation can be remedied fairly easily, using adaptive time stepping, the general problem is not solved without substantial geometric computation. Consider Figure 2, in which two nonconvex bodies and their trajectories are drawn schematically. Depending on the precise relationship of the angular motion of the bodies, they may or may not be colliding. The swept volume in four-dimensional space-time must be computed for the two objects and intersected for a correct treatment of possible interference.

In the current implementation we rely on time steps that are sufficiently small so that no unexpected interference in between steps is likely. When a collision occurs, the moment of first contact is determined by repeatedly halving the time interval until the colliding features have been identified, to an acceptable resolution. This resolution is specified by the user.

## 5 Contact Hinge Evolution

Two objects that are in contact at a point are connected by a special hinge which sustains compression but not tension. While the contact persists, and

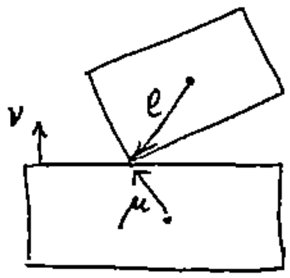


Figure 3: Contact Hinge

with the geometric quantities shown in Figure 3, the equation constraining the relative motion is as follows.

$$(\dot{r_1} + \omega_1 \times \rho) \cdot \nu = (\dot{r_2} + \omega_2 \times \mu) \cdot \nu$$

This contact equation is differentially valid. For surface/surface contact, as in the case of rolling, $\rho$, $\mu$ and $\nu$ change over time and must be updated.

The hinge gives rise to a constraint force $X$ acting at the point of contact. By convention, $X$ acts positively on object 1 and negatively on object 2. Since the hinge cannot sustain tension, $X$ must satisfy the inequality

$$X \cdot \nu \geq 0.$$

In particular, in the absence of friction, $X$ is collinear with $\nu$. Rather than incorporating this inequality into the mathematical model, we satisfy it procedurally as follows: The hinge equations are formulated and added to the dynamic models as if the hinge could sustain tension as well as compression. When the constraint force no longer satisfies the inequality, the hinge is removed. Similarly, the hinge is added subsequent to an inelastic impact of the two objects.

We now consider the case of a block $B$ moving on a table top $T$. The argument developed makes use only of the convexity of $B$ and of $T$. The block and the table top are in contact in a common plane $P$. The surface of $B$ intersects $P$ in a set of points, lines and faces $J_B$. Likewise, the table top intersects $P$ in the set $J_T$. The two sets intersect each other in the set $J_{B \cap T}$. We must analyze this set $J_{B \cap T}$ so as to determine the kinematic nature of the hinge. Note that $J_{B \cap T}$ is a convex polygon.

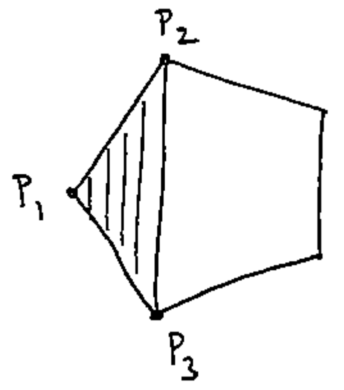


Figure 4: Supporting Triangle, three Degrees of Freedom

Assuming no friction, if there are three noncollinear points $P_i$ in $J_{B\cap T}$ such that the contact equation above is satisfied at each point and the corresponding constraint forces are compressive or zero, then the hinge has only three degrees of freedom. Since $J_{B\cap T}$ is convex, the required points may be chosen to be vertices of it. We call the triangle spanned by the three points a *support triangle*.

We identify the vertices $P_i$ of a support triangle initially as follows: Pick three vertices of $J_{B\cap T}$. If the constraint forces are all compressive or zero, we are done. Otherwise, we pick a new vertex of $J_{B\cap T}$ dropping a vertex that has a tensile constraint force. If we cannot find three noncollinear points at which the constraint forces are compressive, we may have a degenerate support triangle, as discussed next.

During the simulation, the constraint forces change at the vertices of the chosen support triangle. If one of them becomes negative, say at $P_3$, then we must find a new vertex $Q$ such that $P_1$, $P_2$ and $Q$ span a new support triangle. Such a triangle cannot be found when the segment $(P_1, P_2)$ is an edge of $J_{B\cap T}$, because $J_{B\cap T}$ is convex. Then the kinematic hinge between the two objects gains one additional degree of freedom, corresponding to a rotation about the line through $P_1$ and $P_2$. It is also possible that the constraint force at two support triangle vertices becomes tensile, say at $P_2$ and $P_3$. Again, a new support triangle may not exist, and then the hinge has gained two new degrees of freedom, by possible pivoting about $P_1$. The situations are graphically summarized in Figures 4 through 6.

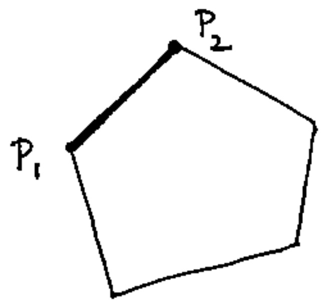


Figure 5: Supporting Triangle Contracts to a Line, four Degrees of Freedom

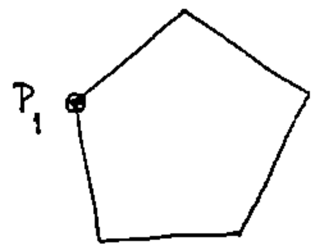


Figure 6: Supporting Triangle contracts to a Point, five Degrees of Freedom

## 6 Experiments

We have thoroughly experimented with collisions between multiple bodies linked in various ways. Some examples are shown in the figures below, and include simulation of breaking pool balls, collision with a linearly hinged chain with links of equal masses, and collision with a sheet of masses hinged by springs. The major limitation in these experiments stems from the possibility of underdetermined systems of equations. Thus, when hinging the masses in the sheet with ball and socket joints, a singular system has to be integrated. This cannot be done presently, and we are exploring ways to circumvent this difficulty. Similarly, simultaneous collisions of a body at more than four points of impact leads also to underdetermined equations.

Acknowledgements

Jim Cremer and Bill Bouma implemented this second version of the simulator system.

## 7 References

1. M. Chace (1985), "Modeling of Dynamic Mechanical Systems" CAD/CAM Robotics and Automation Institute and Intl. Conf., Tucson, Feb. 1985.

2. J. Cremer (1988), Ph. D. Dissertation, Cornell University, in preparation.

3. P. Cundall, R. Hart (1985), "Development of Generalized 2-D and 3-D Distinct Element Programs for Modeling Jointed Rock", Misc. Paper SL-85-1, US Army Corps of Engineers, Waterways Experiment Station, Vicksburg, Miss., Jan. 1985.

4. B. Gilmore (1986), "The Simulation of Mechanical Systems With a Changing Topology," Ph.D. Dissertation, Mechan. Engr., Purdue University, August 1986.

5. C. Hoffmann, J. Hopcroft (1987), "Simulation of Physical Systems from Geometric Models," *IEEE J Robotics and Aut. RA-3*, June 1987, 194 – 206.

6. J. Latombe, C. Laugier, J. Lefebvre, E. Mazer, J. Miribel (1984), "The LM Robot Programming System", *Robotics Research*, 2nd Intl. Symp., Kyoto, 1984, MIT Press, 1985, 377–391.

7. T. Lozano-Perez, M. Mason, R. Taylor (1984), "Automatic Synthesis of Fine-Motion Strategies for Robots", *Intl. J. of Robotics Research* 3:1, 3–24.

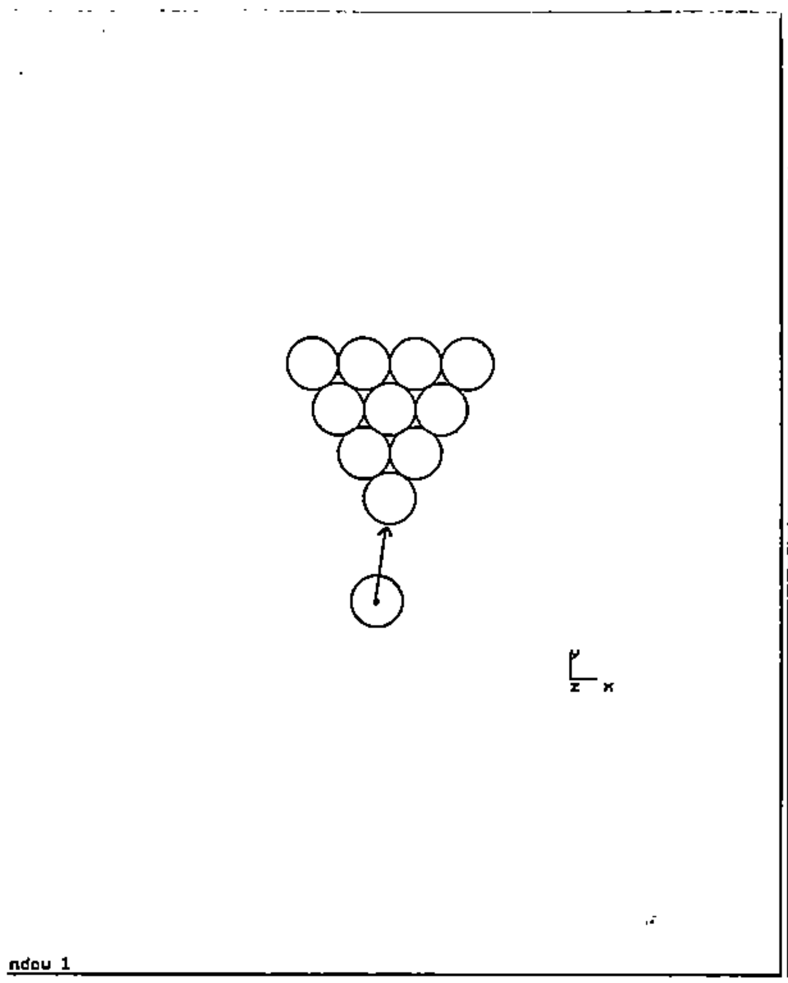
8. M. Mason (1984), "Mechanics of Pushing", *Robotics Research*, 2nd Intl. Symp., Kyoto, 1984, MIT Press, 1985, 421–428.

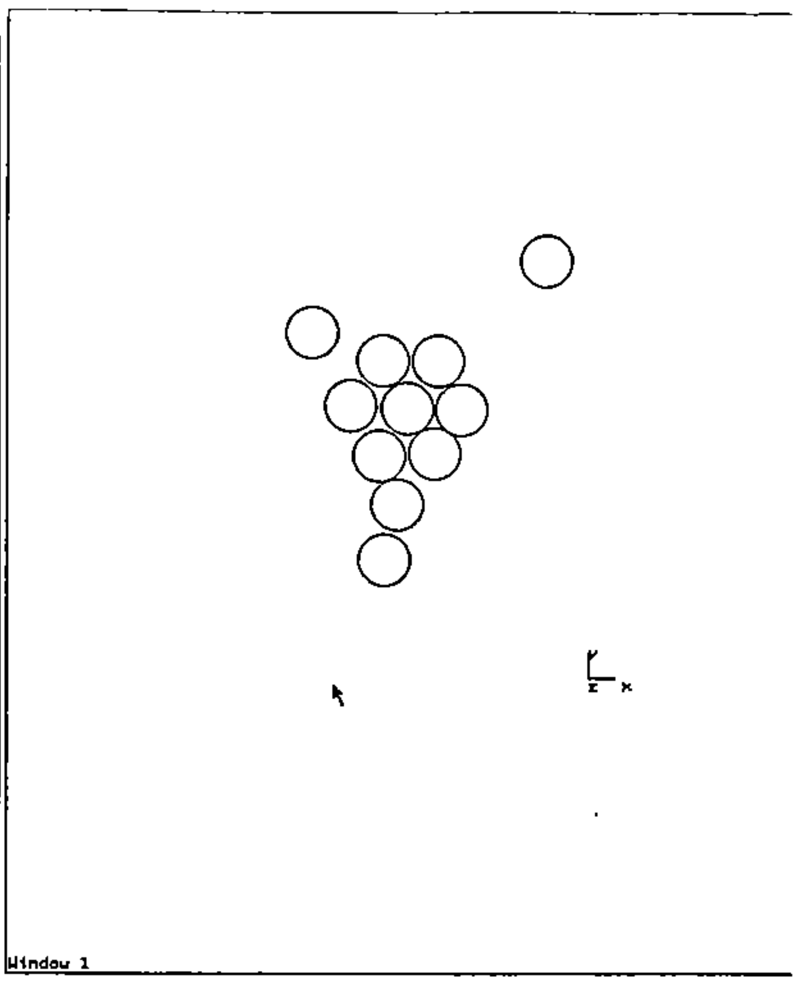
9. M. Peshkin, A. Sanderson (1985), "The Motion of a Pushed, Sliding Object; Part 1: Sliding Friction", Robotics Inst., Carnegie-Mellow Univ., TR 85-18

10. M. Peshkin, A. Sanderson (1986), "The Motion of a Pushed, Sliding Object; Part 2: Contact Friction", Robotics Inst., Carnegie-Mellon Univ., TR 86-7

11. M. Takano (1984), "Development of Simulation System of Robot Motion and its Role in Task Planning and Design Systems", *Robotics Research*, 2$^{nd}$ Intl. Symp., Kyoto, 1984, MIT Press, 1985, 223–230.

12. J. Wittenburg (1977), *"Dynamics of Systems of Rigid Bodies"*, B. G. Teubner, Stuttgart, W. Germany, 1977.





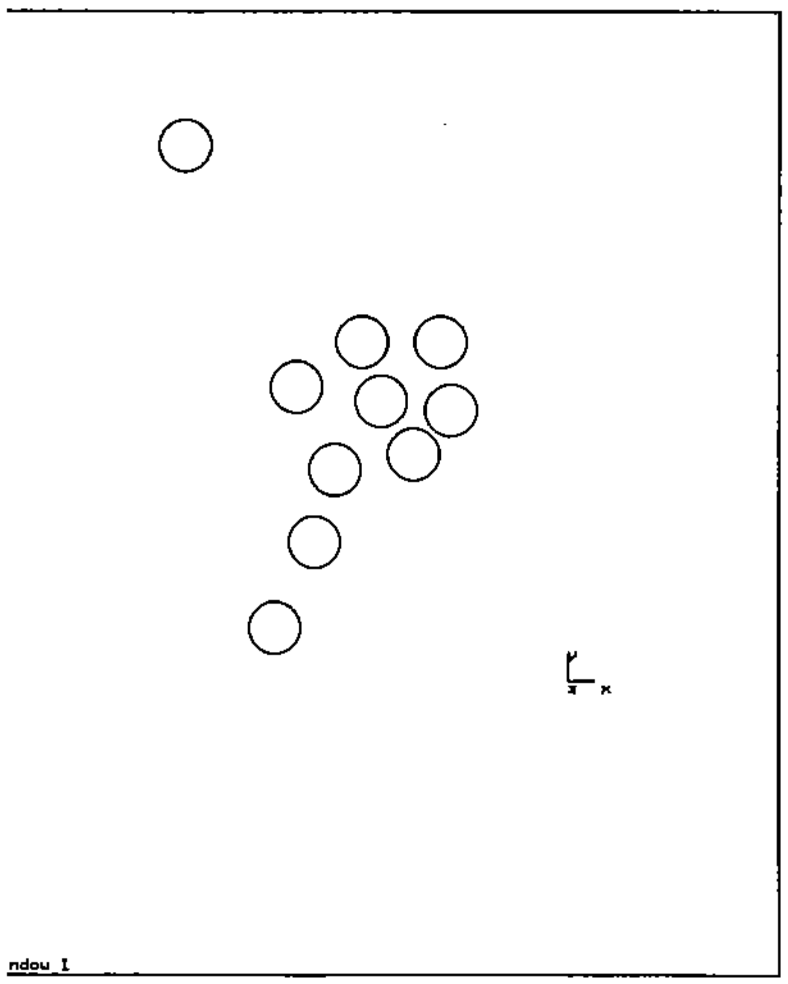
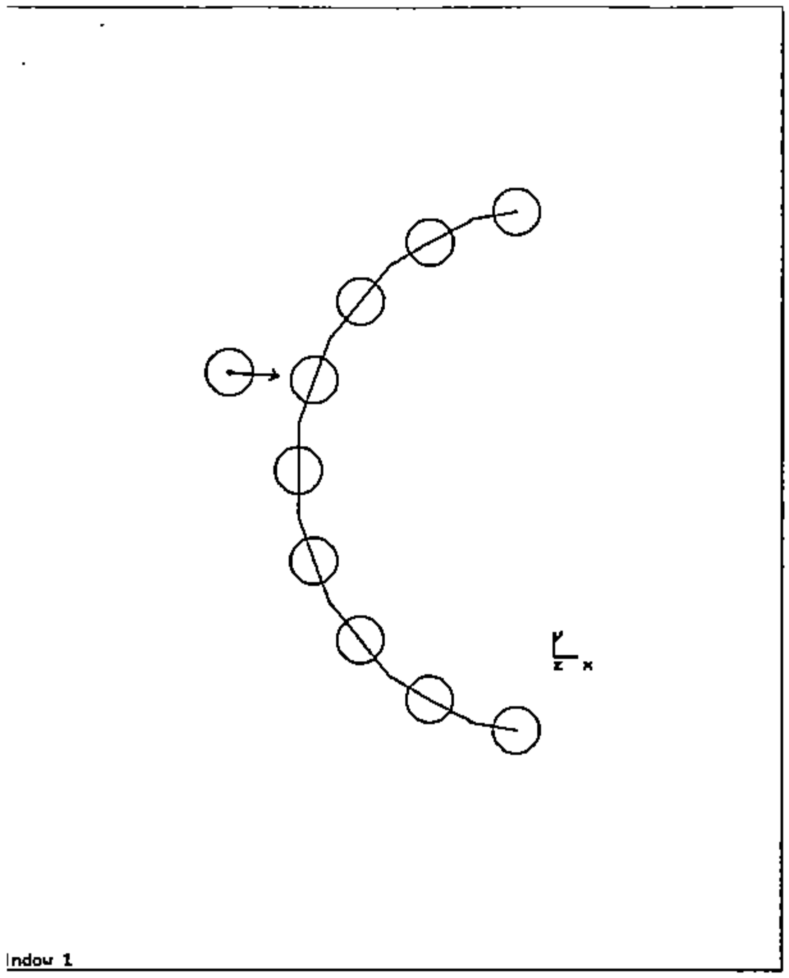
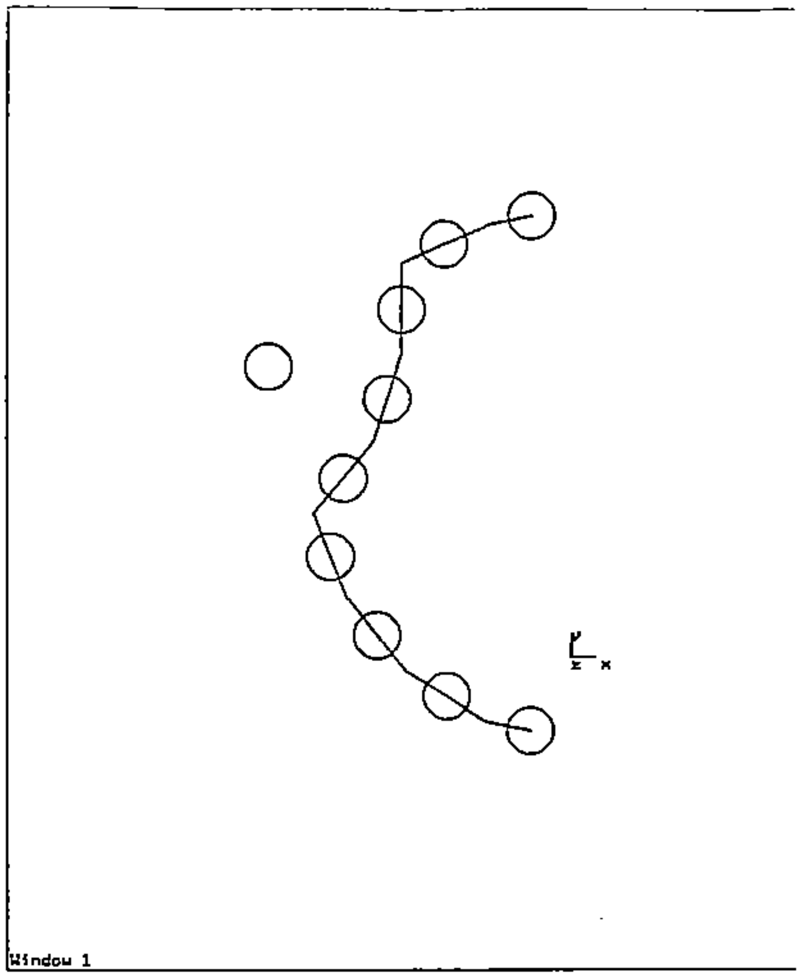


Plate 1: Pool Break

A ball colliding with ten other balls.





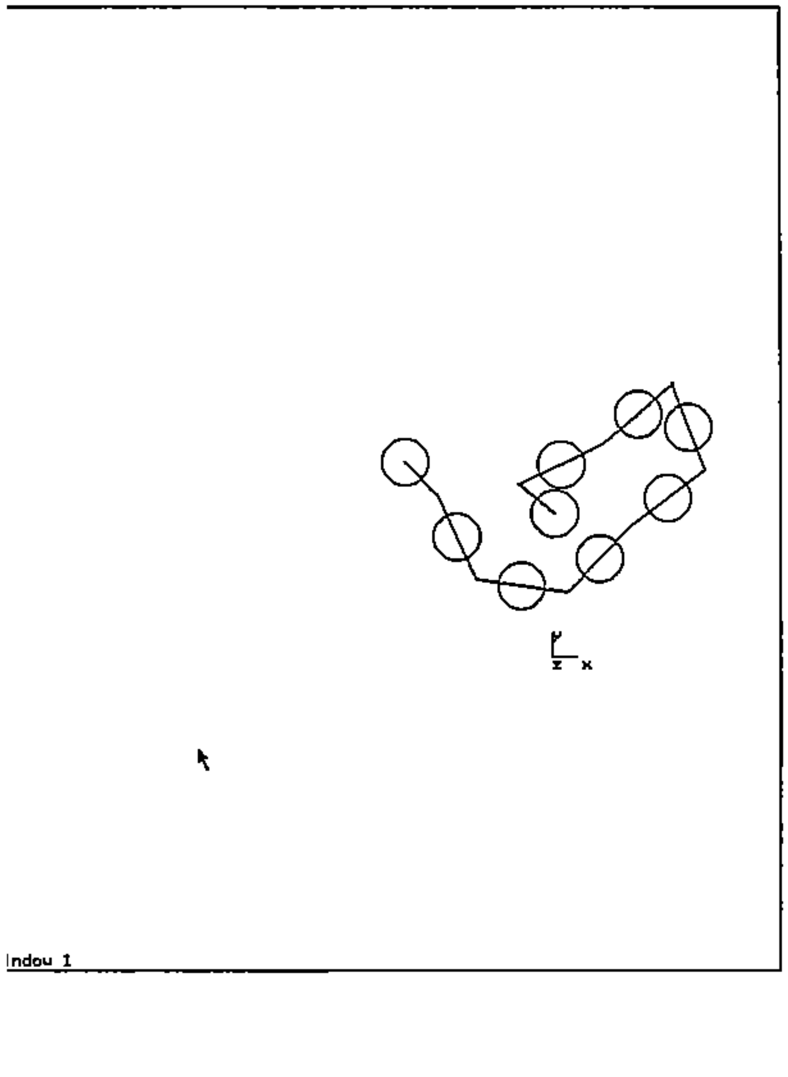
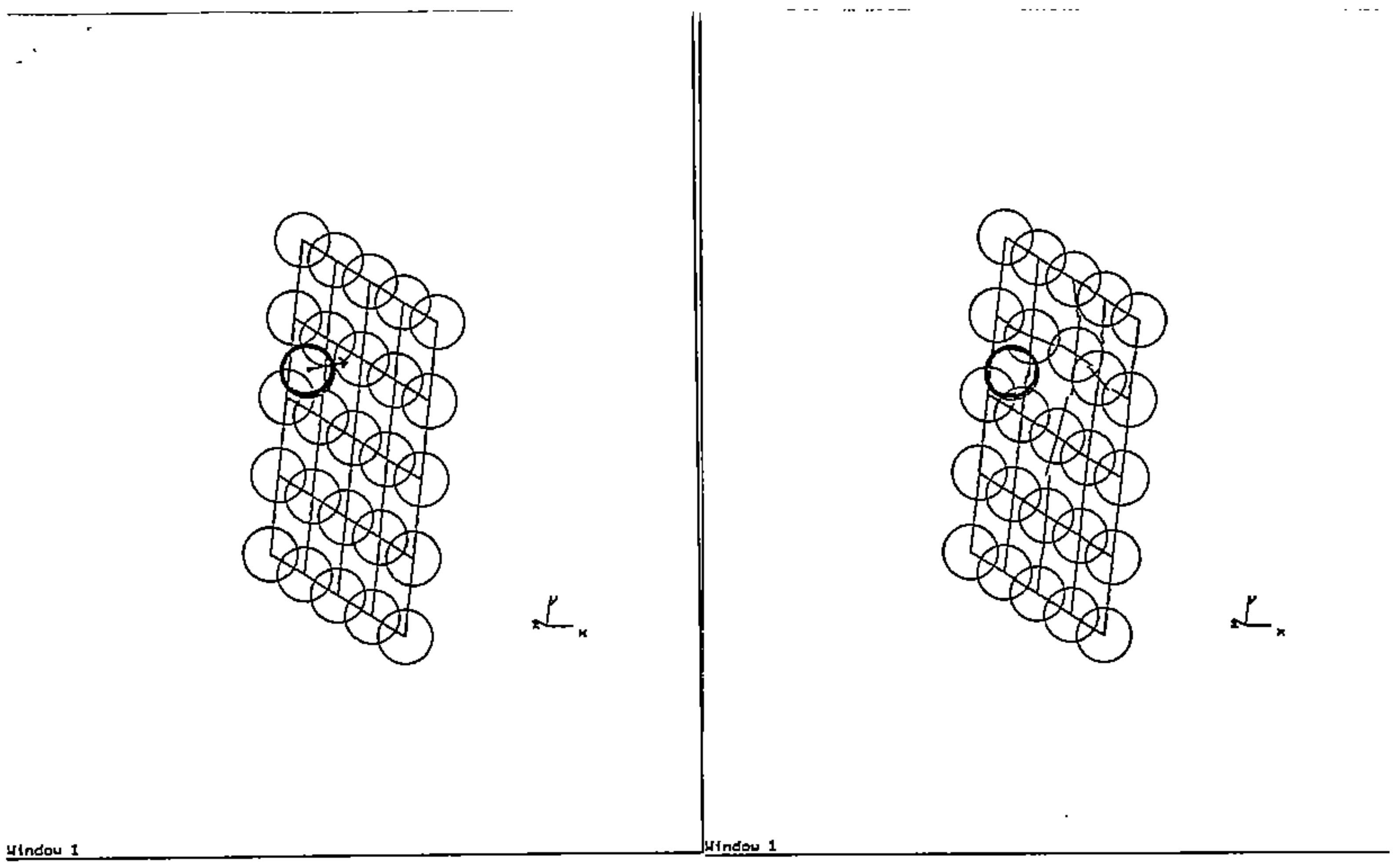


Plate 2: Chain Collision

A ball colliding with a chain of balls linked with ball and socket hinges.



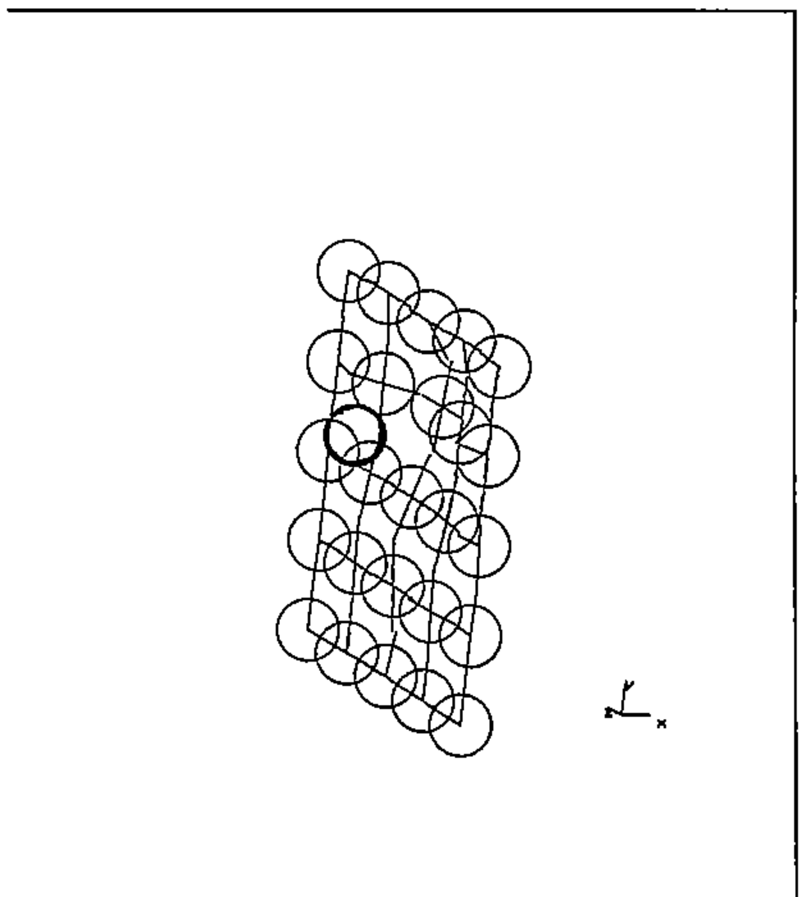


Plate 3: Sheet Collision

A ball colliding with a sheet of balls linked by springs.