Schematic Modeling for Simulation of Physical Systems

Xue Dong Yang*

TR 90-1079
January 1990

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*Work on this paper is supported by a grant from Sandia National Laboratories #75-6562.

Schematic Modeling
for Simulation of Physical Systems

Xue Dong Yang [1]
Cornell University
Computer Science Department
Ithaca, NY 14853

ABSTRACT

This paper proposes a method for classifying and implementing schematic representations in multiple domains within a simulation system. Our initial work is restricted to the modeling of rigid components for kinematic and dynamic simulations. We hope to demonstrate that the use of abstract schematic representations will significantly reduce the effort needed to create models of physical systems, which will in turn greatly simplify the effective modeling and simulation of large, complicated physical systems.

[1] Work on this paper is supported by a grant from Sandia National Laboratories #75-6562.

1 Introduction

Computers have been used extensively for the simulation and analysis of physical systems. Early simulation systems concentrated on the analysis of particular problems. They often used sophisticated, special purpose models of a particular class of physical systems. The capabilities of simulation programs have grown in proportion to the growth in available computing power. The basic techniques used have changed as well. Surveys of mechanical engineering simulation systems can be found in [16], [20]. The ADAMS [6] and DADS [16] systems are examples of large state-of-the-art systems from the mechanical engineering domain. These systems have very sophisticated methods to provide efficient formulations of dynamics, intricate numerical techniques, object flexibility and elasticity, etc. However, they have largely ignored geometric issues. *Newton*, a dynamic simulation system developed at Cornell and Purdue Universities [14], [8], provides many general-purpose analysis capabilities by integrating geometric design and dynamic analysis.

More powerful systems will allow us to build and test with "electronic prototypes", sophisticated object models incorporating information about object behavior from a variety of domains [15]. Electronic prototyping consists of constructing models of physical objects in a computer to support activities such as computer-aided design, engineering analysis, design verification, and automated manufacturing. *Newton* represents our initial effort towards this goai. This system automatically formulates and integrates object motion equations, uses geometric models to detect and resolve impacts, uses a general-purpose event-handling mechanism to support incremental modifications of object-behavior descriptions as object relationships change (due, for instance, to temporary contacts being made or broken), and provides facilities for specification of control algorithms.

However, from our experience, conventional geometric modelers are not practical for simulating large assemblies. For example, the time needed to model an automobile engine to a sufficient level of detail for verifying such properties as combustion would be prohibitive. Thus, we are exploring the use of schematic editors. Previous simulation systems have limited their use of schematic representations to the study of particular domains. We believe that a system which offers multi-level representations of physical objects will have a significant impact in the design and analysis of complex systems.

Such a multi-representational system will allow designs and analyses to begin at very abstract levels. For example, in the case of an automobile engine, one could represent a cylinder by an an axis, a piston head by a dot point, and a crankshaft by an axis. Certain points on the crankshaft would be constrained to be a fixed distance from the center of the axis. At this level, one could

carry out a kinematic simulation of the engine. By adding mass and inertia, one could carry out a dynamic simulation, and so on. If one needed to perform a lower-level simulation, the design could be expanded into a geometric level. Again, if one wanted to study combustion, one could add a finite element mesh in the cylinder. At each level, the appropriate analysis could be performed with the minimum specification of information. Many analyses could be carried out at very early design stages.

A multi-level system could also be used as an adaptive representation. We would choose the simplest model necessary for the desired analysis of each component of a physical system. Thus, different components in a system would be modeled at different levels of representation. For example in the case of a robot hand, finger linkages could be modeled effectively at the schematic level, while geometric models could be used to study surface contact manipulation during a grasping operation. This scheme is analogous to the concept of multi-resolution simulations of VLSI design in which different modules of a large circuit are modeled at different levels of abstraction (see, for example, [13]).

In order to explore the above ideas, we have developed and implemented a small set of schematic primitives on *Newton*. In this article, we are mainly concerned with object representation for dynamic simulation, analysis and control.

2 Schematic Modeling

2.1 Review of Previous Work

Schematic representations are not new. They have been used as fundamental tools in the study of engineering mechanics in college, and in engineering analyses of mechanical system designs for centuries. Figure 1 shows a diagram of a typical mechanical mechanism found in mechanical engineering books. Each rigid component in the system is usually treated as a point mass with associated moments of inertia. The dimensional parameters and geometric shapes of components in the diagram are used primarily for defining kinematic inter-relationships between the components, and sometimes used for deriving dynamic properties of rigid bodies such as center of mass, moment of inertia, etc. Often a designer may directly specify estimated or desired dynamic properties for each component. In such a case the geometric shapes in the diagram are used solely for graphic appearances of the mechanism and play almost no role in its dynamic analysis. Though this kind of representation is very simple, it can satisfy many requirements in engineering analysis problems and has been used extensively.

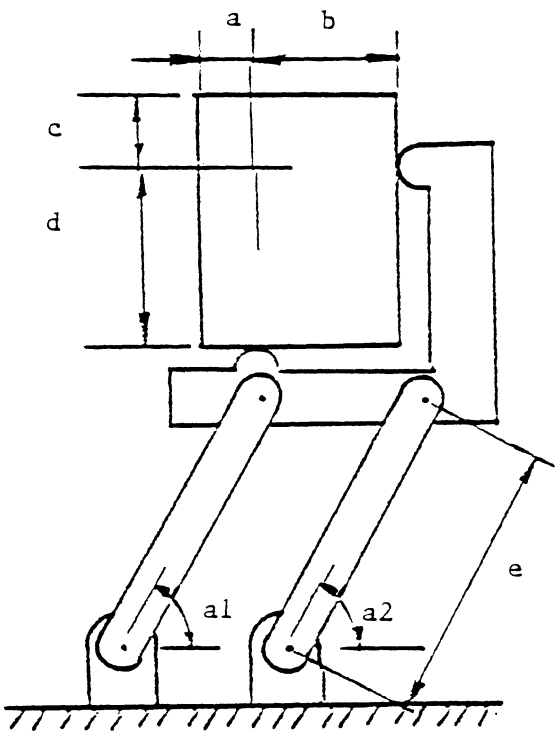


Figure 1: Schematic diagram of an example mechanical mechanism

The ADAMS [6] and DADS [16] systems are examples of large state-of-the-art systems using schematic representations. These systems typically represent objects as point masses with associated inertias and coordinate systems in their internal representations. Shapes of rigid components of the mechanisms may be described by wire frame models at the user interface level, but this is done mostly for graphic displays in simulation or engineering drafting. These systems include sophisticated automatic formulation and integration of motion equations and the use of efficient numerical analysis techniques. Object collision handling is supported in these systems, but in a limited way, for example, all possible collisions must be explicitly foreseen by the experimenter before the simulation. Other notable systems are MAP, developed at General Electric, Inc., MESA VERDE system [22], and the products of SILMA, Inc and Deneb Robotics, Inc. [7].

In addition to their contributions to mechanical engineers, schematic representations have important applications to computer graphics and animation researchers. These scientists are interested in animation, and in simulation and control of objects having many degrees of freedom, particularly models of human and animal figures. For example ([2], [21], [3], [4]), human figures are often modeled as "skeletons" made up of segments linked at joints. A segment is usually modeled as an elementary geometric shape, such as a cylinder or a block, or simply as a line segment. Physical properties of each segment are calculated from their geometric shapes directly ,or specified by estimated values by a designer. This kind of representation is suitable not only for traditional parametric key-frame animations, but also for kinematic algorithmic animations and dynamic-based techniques. An animation can first be generated cheaply and efficiently in schematic format in order to visualize the movement of an actor at a coarse level. After a satisfactory animation sequence is obtained, the surfaces or volumes of each part of the body are superimposed onto the skeletons in each frame for final production. Recent research has introduced multi-layer representations for modeling and simulating deformation of skin in motion and contact [5], [17], [19], [11]. Bones are modeled as rigid "skeletons", while surrounding skin or muscles are modeled by finite element meshes or various geometric patches to support deformation manipulations.

In summary, from the dynamic point of view, the point mass representation is a standard abstraction of rigid bodies in the formulation of motion equations for dynamic simulation and analysis. From the kinematic point of view, the kinematic relationships between components of a system are defined by abstract constraints imposed by schematic joints and links. From the geometric point of view, various kinds of schematic representations have been used in different systems depending on the application desired. In some systems, geometric shapes are independent

of dynamic analysis and used merely for graphical display and drafting purposes. In other systems, elementary geometric shape/volumes may be used to approximate rigid bodies such that their dynamic properties are derived from their geometries. These shapes are then used to handle contacts and collisions. Integration of more sophisticated geometric modelers that support a wider range of dynamic analyses are anticipated.

2.2 A Schematic Model

Schematic representations provide high-level abstractions of properties of physical objects, and allow analysis of these objects in their corresponding physical domains to be performed effectively and efficiently. We plan to develop a set of schematic primitives that will allow us to:

(a) model many physical systems dealing with mechanical engineering and robot manipulations,

(b) model geometric shapes of physical objects at various levels,

(c) specify a minimum number of physical properties of objects while maintaining their functionality, and

(d) extend the given set of schematic primitives with a user's own primitives in a systematic way.

Abstractions of dynamic properties of rigid bodies are well defined in past research. Dynamic properties include the center of mass, mass, and moment inertia, which are all essential for dynamic analysis. These properties can be defined either explicitly by user specifications or implicitly by their corresponding geometries.

Previous systems using geometry abstractions also exist, but these systems focus on particular types of representations that are suitable to particular applications. For example, [4] uses line segments to model human body in studying human walking, and [21] uses blocks to model human body in studying body motion. they usually lack a general systematic definition of schematic models.

We propose a system of schematic abstractions that is based on reductions in dimension. An object with complicated geometric shape can be approximated by volumes with simple shapes. A volume can then be shrunk into a surface, and further into a curve, and finally a point. We want to be able to model an object at different levels of abstraction, such that an appropriate level can be chosen by experimenters according to their particular applications. Furthermore, it will allow different part of a system been modeled at different levels based on the considerations of suitability and efficiency. The above classification provides a guideline for defining schematic models. We have designed and implemented a preliminary version of a schematic modeler that

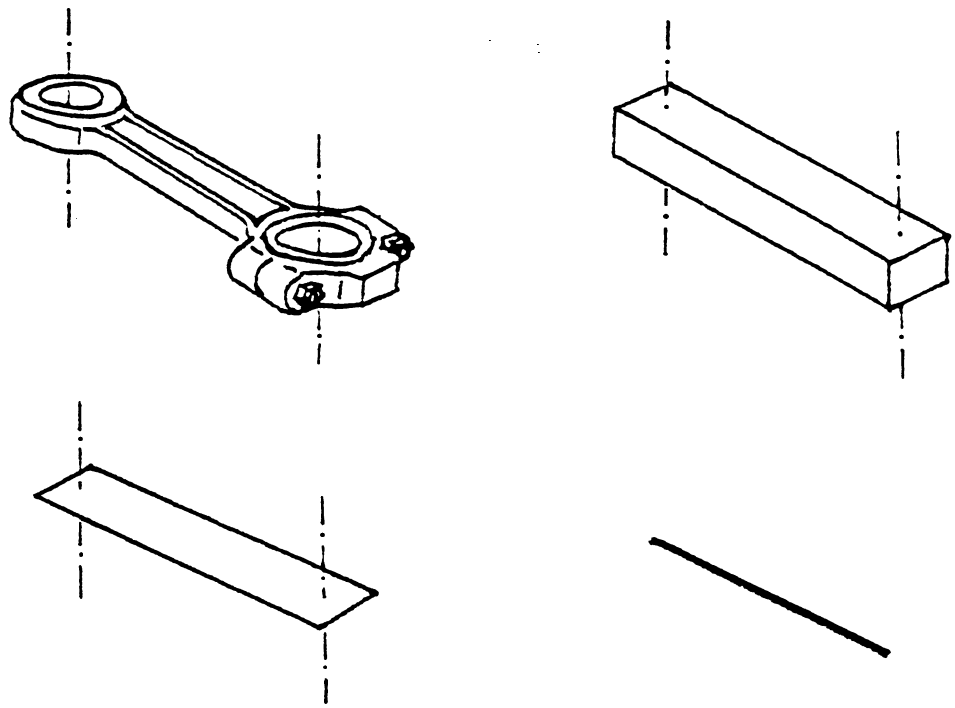


Figure 2: Schematic representation of a rod at different levels of abstraction

modeling system consists of a small set of schematic primitives. Though the set contains only a few, simple primitives, but is powerful enough to model many interesting mechanical systems.

We classified out schematic representations into four levels:

(a) elementary volume

(b) elementary surface

(c) elementary curve

(d) point mass.

First we will describe the definition of each level. Then we will discuss the relationships between different levels.

(a) Elementary Volume - Three-dimensional Schematic Primitive

A three-dimensional rigid object with a complex geometric shape can be approximated by some simple, elementary three-dimensional volume, such as a rectangular block, sphere, cylinder, etc. Though there is no reduction in dimension in this case; the entities to be specified by a user to define the geometry of the object can be reduced significantly. For example, a rod in an engine is shown in Figure 2(a). A complete specification of its geometry would require a definition of

each part and a method of attaching different parts together. In many situations, one can replace the complete geometry by a coarse approximation such as a rectangular block, as shown in Figure 2(b), which is roughly similar in shape, mass and moments of inertia to the rod.

Computer graphic studies of human and animal motions often model body parts as rectangular blocks or cylinders that are linked together at joint positions (see for example, the illustrations in [3] [9], and [21], etc.), and are called "skeletons".

Since we have volumes at this level, a user needs to provide only the volume densities, and the dynamic properties of the object can be derived automatically. Also, because we provide the shape and volume of an object, surface contact and collision handling are supported.

(b) Elementary Surface - Two-dimensional Schematic Primitive

A two-dimensional schematic primitive is an elementary surface. Elementary surfaces include planar surfaces with simple boundaries such as rectangles, triangles or circles; and some simple, elementary algebraic surfaces, such as spherical or cylindrical patches. Planar surface representation is often used in planar mechanics, but is rarely seen in existing simulation systems. Elementary surfaces are actually suitable abstractions for physical objects in many situations. For example (Figure 11), a floor can be represented by a rectangular plate and wheels rolling on the floor can be represented by disks. In our rod example, we can further abstract it as a rectangular plate as well (Figure 2(c)).

Because surfaces do not have volume sizes, dynamic properties of surfaces cannot be simply derived from conventional definitions. If we define the material density value to be the mass of unit area of a surface, the dynamic properties of a surface can be easily derived from the geometry of a surface. Note also that, surface contact computation are also supported at this level. For example, in Figure 11, rolling constraint controls the relative motion between the disks and the floor under the assumption that the coefficient of friction at the contact points between disks and floor are infinite.

(c) Elementary Curve - One-dimensional Schematical Primitive

A one-dimensional schematic primitive is an elementary curve. Elementary curves include line segments and sections of simple algebraic curves, such as circles and arcs. Again, the rod in Figure 2(a) can be further abstracted as a line segment, shown in Figure 2(d). Though it seems to be an over simplified representation, it still provides certain important geometrical information about the rod, such as length and approximate shape. If we use this representation in a simulation, we will still be able to visualize the kinematic behavior of the rod. Another example of this representation

is shown in the simulation of human walking [4], in which each part of the body is represented by a line segment of appropriate length and is linked to others at joint positions.

Similar to the two-dimensional case, a curve does not have a volume size. Thus, in order to allow definition and automatic derivation of dynamic properties from a curve specification, we define the material density of a curve to be the mass of unit length of the curve. Contact between a curve segment and a surface or a volume can also be effectively computed.

(d) Point - Zero-dimensional Schematic Primitive

Finally, any rigid body can be modeled as a point with mass concentrated at its center of mass. Though no geometric information can be derived from this representation, it still provides positional information of a rigid body in a simulation.

In order to use point representation for dynamic analyses, a designer must fully specify mass and moment of inertias.

Choosing the type of schematic primitive to be used for a particular application depends on what you want to learn from your analysis, and what you want to see from the graphic simulation. For example, if you want to see the motion of a rigid body, a simulation with point representation can give you a satisfactory idea about how the body moves in space. For the types of linkages often seen in mechanical mechanisms, a line segment is a good abstraction. Kinematic relationships between the joined linkages and system configurations are clearly expressed in such representations. It is often true that "simpler is clearer". Object contact problems may require surface modeling

2.3 Compatibility and Consistency

Since we have schematic representations at various levels of abstraction, to use them together in a single system, we must consider (1) How the abstractions at various levels can be compatible? and (2) How we maintain consistency if a component of a system in one representation is replaced by another at a different level?

The structure of primitive definition in our system is:

```
primitive type-name (argument-list)
begin
    schematics: schematic-type-name (parameters);
    material-property: (color, density, ..);
    dynamic-property: (mass, inertias);
    joint-positions: { joint-name: position-in-local-coordinates; }*
```

```
    transformations: local-to-global-transformation-matrix[4][4];
    ...
end;
```

A local Cartesian coordinate system is defined with every rigid body. The center of mass is assumed to be at the origin. Each schematic primitive is defined in a fixed orientation with respect to the local coordinate. Joint positions are defined in local coordinates as well.

A kinematic or dynamic analysis is carried out in a global coordinate system. All local coordinates of each component are transformed into the global coordinate system by its transformation matrix.

For each schematic representation, a point representation is always defined inherently. This provides a common base for the formulation of motion equations in dynamic analyses. The dynamic properties of objects, such as mass and moment of inertias, are derived uniformly, giving the meaning of "material density" described in the last subsection.

Joint positions are defined independently of schematic types. That is, no matter what type of schematic representation is chosen for an abstract geometric shape of a rigid component, its relevant joint positions are consistently defined in its local coordinate system. Hence, kinematic inter-relationships between components of mechanisms joined together at the defined joint positions will also be consistent.

2.4 A Small, Preliminary Set of Schematic Primitives

To test the model we proposed, we have designed a small set of schematic primitives and implemented them on *Newton*. The schematic primitives are described below, and their implementation is discussed in the next section.

(a) Elementary Volumes

Several elementary volumes exist in *Newton*, such as the cuboid, sphere, and cylinder. They are defined in the local coordinate system as shown in Figure 3.

(b) Elementary Surfaces

Only two primitives, the rectangular plate and disk, are defined at this level (Figure 4).

(c) Elementary Curves

A line segment is the only primitive defined at this level (Figure 5).

(d) Point

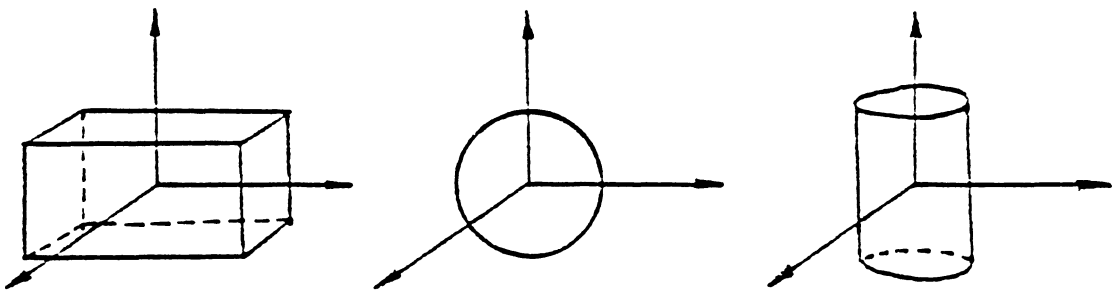


Figure 3: Elementary volumes

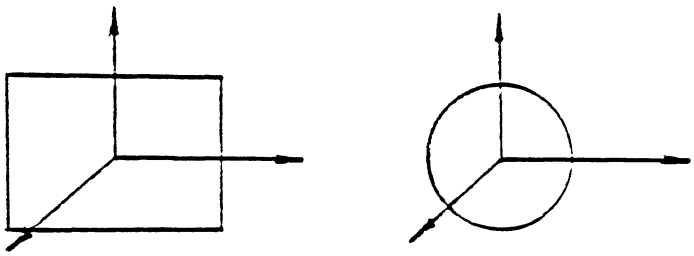


Figure 4: Elementary surfaces

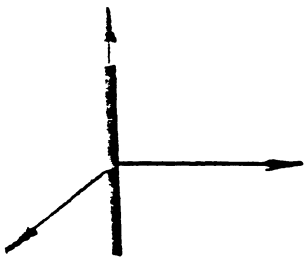


Figure 5: Elementary curves

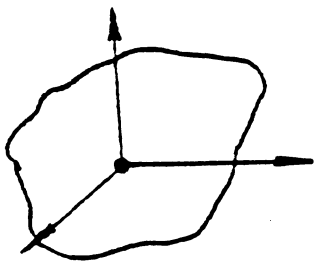


Figure 6: Point mass

A point mass with associated moments of inertia is the highest level of abstraction for an arbitrary rigid body (Figure 6).

3 A Preliminary Implementation

We implemented a small set of schematic primitives on *Newton*. Its system architecture is described below; more detailed descriptions can be found in [14] [8].

3.1 Newton Architecture Overview

Newton has three main components: the definition and representation module, the analysis module, and the report system, as depicted in Figure 7.

Newton's definition language is used to describe a variety of simulation entities, including objects, hinges, constraints, models, equations, and quantities. Simulated objects are represented as instances of the object class. A class is either a primitive type, corresponding to single indivisible bodies, or a composite type, representing collections of objects related by constraints. The components of composite objects can be either composites or primitives. Thus, descriptions of complex mechanisms are made simpler by breaking the description down into natural part- component hierarchical relationships.

Quantities include: positions, velocities, accelerations, masses, volumes, coefficients of restitution and friction, forces and torques, and so on. Models are data structures used to partition information about an object according to particular domains. We have added several detailed descriptions of abstract and geometric models to *Newton's* existing language.

Our abstract model of a primitive object consists of a name and a property list containing material, density, color. etc. From the specified properties, other characteristics are derived. For instance, by using density and volumetric properties of the geometric model, we can determine the mass and inertia of a primitive object. Other properties can be added as needed when we enlarge the number of modeling domains. For instance, elasticity properties will be needed when modeling deformable bodies. A composite object is represented by a graph whose edges represent relations between those objects. Examples of relations are touching, rigidly connected, and hinged in a particular way. These properties may be interrogated by other subsystems.

The geometric modeling subsystem represents both the shape and the position/orientation of objects in three-space. In the definition phase, the subsystem is used to construct the shape of primitive objects and to position primitive components relative to a global coordinate frame.

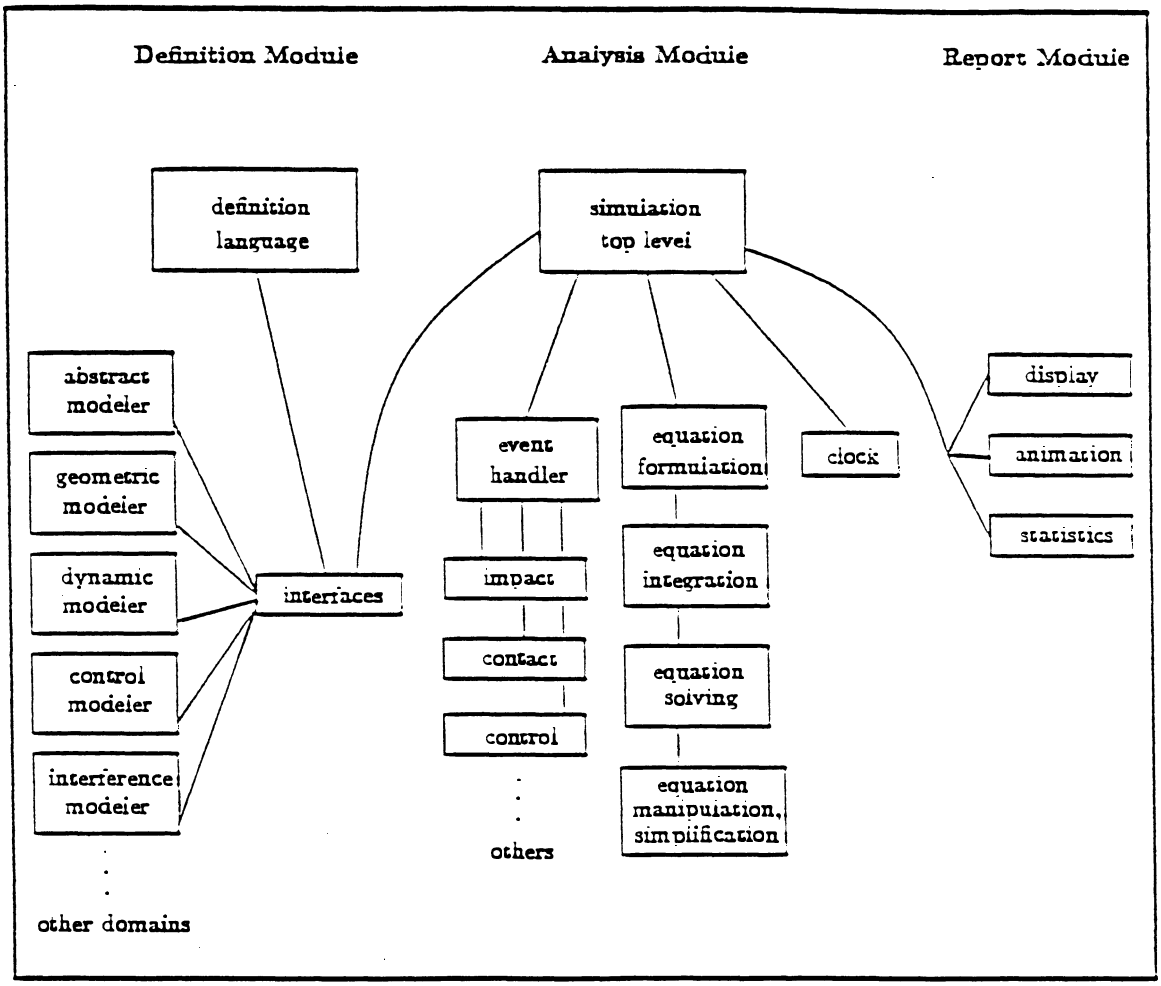


Figure 7: The basic Newton architecture diagram

During a simulation, the usual operation is to calculate new position/orientation of an object during intervals of time by using numerical integration algorithms.

Our analysis component implements the top-level control loop of simulations. The major computations in the loop are integration and exceptional events handling. The integration subsystem first collects and processes the set of motion equations representing the current dynamics state, and then sets up and solves a linear system to produce the accelerations and hinge reaction forces and torques. Then new positions and velocities for all objects are produced from their old positions, velocities, and accelerations by integrations. The event handler is responsible for detecting and handling collisions, contact formations and breakages, control algorithm state changes, and other events that yield discontinuities in object velocities or accelerations.

The report system handles generation of graphical and statistical feedback to users during simulations as well as recording relevant information for later regeneration of animations.

3.2 Definition Language

In order to easily incorporate our schematic primitives in *Newton*, we extended the definition language of *Newton*. Several extensions were made in the interface module that correctly translate our extended definition language into the corresponding system submodels.

A scenario to be simulated is called a world. A world is defined by describing its objects and by placing these objects into an initial configuration. An object is declared as an instance of a type. We demonstrate definitions of primitive and composite types by an example of one-cylinder-engine. A primitive object type definition contains several subsections specifying domain specific aspects. A rod abstracted as a line segment is described by

```
PRIMITIVE rod (length, material_density) BEGIN
    properties: ( density: material_density );
    geometry: linesegment (length)
        where begin
            top: (0, length/2, 0);
            bottom: (0, -length/2, 0);
        end; dynamics:;
END
```

Here the abstract properties density is identified, the geometric shape is defined as a line segment with specified length, and certain features of this shape are named, such as "top" and

"bottom" points of the rod. Feature x of object a is referred to as $a.x$, and the names of features are inherited. E.g., if a is a component of composite object b, then feature x in a is referred to as $b.a.x$. The dynamic model is constructed automatically, so no specification is needed, but its inclusion must be indicated by stating the keyword "dynamics" followed by an empty description.

A piston is also defined as a line segment:

```
PRIMITIVE piston (length, material_density) BEGIN
    properties: ( density: material_density );
    geometry: linesegment (length);
        where begin
            top: (0, length/2, 0);
            bottom: (0, -length/2, 0)
        end; dynamics:;
END
```

A real crankshaft has a complicated shape. We use a highly abstract schematic primitive - again, a line segment - to represent its rotation axis with associated approximations of mass and moment of inertias. The arm position is declared as a primitive feature.

```
PRIMITIVE crankshaft (length, arm_radius, material_density, Ixx, Iyy, Izz) BEGIN
    properties: ( density: material_density );
    geometry: linesegment (length, (inertia: Ixx, Iyy, Izz))
        where begin
            end1: (0, length/2, 0);
            end2: (0, -length/2, 0);
            arm: (arm_radius, 0, 0);
        end
    dynamics:;
END
```

Note the direct specification of inertia parameters in this definition. This option is included for users who may not be satisfied with the approximated inertia values automatically derived from the given geometry. For example in the case of a crankshaft, the moment of inertia with respect to its local y-axis derived from the geometry of line segment is zero. It is often useful to provide users with options that override system-derived values.

A crankcase is the body of an engine that is a frame for supporting and assembling the components defined above. The axis of the cylinder and positions of bearings supporting the crankshaft

are defined as features in the crankcase. A prismatic hinge joins the piston to the cylinder such that the motion of the piston is constrained to be a linear translation along the cylinder. A revolution hinge joins the crankshaft to the bearings so that the crankshaft can only rotate about an axis defined by the two supporting bearings. The engine is either stationary relative to the ground in our environment world or mounted on a vehicle. In both kinematic simulations and a dynamic analyses of an engine, the shape of the crankcase is less important. Thus, we define the engine as a point type with estimated mass and moment of inertias.

```
PRIMITIVE crankcase (mass, Ixx, Iyy, Izz) BEGIN
    properties: ();
    geometry: pointobject(mass, Ixx, Iyy, Izz);
        where begin
            bearing1: (0.1, 0, 0);
            bearing2: (-0.1, 0, 0);
            cylinder_top: (0, 0.25, 0);
            cylinder_bottom: (0, 0.15, 0);
        end;
    dynamics:;
END
```

Then, a one-cylinder-engine is defined as a composite object in the following way:

```
COMPOSITE one-cylinder-engine BEGIN
    COMPONENTS
        crankcase: crankcase (20, 40, 60, 40);
        piston: piston (0.1 20);
        rod: rod (0.1, 10);
        crankshaft: crankshaft (0.2, 0.05, 25, 8, 4, 8);

        cylinder_piston_hinge: prismatic;
        crankshaft_crankcase_hinge: pin;
        piston_rod_joint: ball.and.socket;
        rod_crankshaft_joint: ball.and.socket;

    STRUCTURE BEGIN
        JOIN crankshaft TO crankcase WITH crankshaft_crankcase_hinge MATCHING
            begin (end1 bearing1); (end2 bearing2) end;
        JOIN piston TO crankcase WITH cylinder_piston_hinge MATCHING
            begin (top cylinder_top); (bottom cylinder_bottom) end;
```

```
        JOIN piston TO rod WITH piston_rod_joint MATCHING (bottom top);
        JOIN rod TO crankshaft WITH rod_crankshaft_hinge MATCHING (bottom arm);

    END
END
```

In this mechanism, the piston can move up and down along the cylinder and it drives the crankshaft through a connecting rod. Figure 8 shows a sequence of simulations of this engine.

3.3 Implementation of Schematic Primitives

The implementation of our schematic primitives on *Newton* is relatively easy. The handling of schematic primitives are added in three places to the system.

(1) Translation of definition to system submodels

The data structure of system submodels are translated from the definition languages. We can now derive dynamic parameters, such as mass and moment of inertias. For geometric models of objects in em Newton, the volume of the object can be easily computed; then the mass and moment of inertias are automatically derived by providing material properties of the object. This type of computation is extended systematically to handle our schematic primitives in various dimensions. For a two dimensional primitive, we first compute the area; mass and moment of inertias are then derived by providing surface density properties. In the case of one dimension curve primitives, the length of a segment of curve is computed and the curve density property is needed.(is this the right word, or should it be "derived"?) In the case of point mass, no computation is necessary and user specified quantities in the definition language are simply copied into corresponding system's submodels.

As mentioned above, it is sometimes useful for a user to directly specify dynamic parameters in the definition language when those derived from the schematic primitive are unsatisfactory. In such cases, the user-specified quantities always override the corresponding values derived by the system.

(2) Collision detection

Collision detection in the current system is facilitated by the geometric representation of objects. In our current implementation, we only allow this computation to be made between volumes, volume and surface, and surfaces. If one of a pair of objects is a one-dimensional primitive or a point mass, the pair is skipped for collision computation.

(3) Reporting

Once new primitives are included, a set of low level graphical subroutines are written to generate graphical display of the new schematic primitives. Those subroutines are called from the main routine of the reporting system.

4 Simulation with Schematic Primitives

The following examples demonstrate several encouraging implementations of our schematic representation system.

Example 1: one-cylinder-engine

We discussed our definition language in the last section by describing a one-cylinder-engine mechanism with highly abstract schematic primitives. Now we want to simulate the starting process of the engine. The force acting on the top of the piston is primarily a simplified force approximating combustion, exhaust, intake, and compression strokes. During the combustion stroke, the force applied to the piston is

$$force_{max} \times \left(1 - \frac{|x_{top} - x|}{|x_{top} - x_{bottom}|}\right)$$

where x is the position of piston respect to the local coordinates of cylinder (precisely speaking is to that of crankcase). Thus x_{top} and x_{bottom} are the top and bottom positions of the piston in the same coordinates. The force during the combustion stroke is linearly inverse-proportional to the displacement of the piston from its top position. In exhaust and intake strokes, the acting forces on the piston are relatively small and ignored. In compression strokes, the resistant force applied on the piston is similar to that in combustion, except that $force_{max}$ is replaced by $resistance_{max}$. During simulation, this acting force is treated as an external force since it comes from "outside" the defined mechanism. We begin our process by declaring a named control force for the piston in the definition language and providing a lisp function with the associated name. This action will implement the above simplified combustion model that is invoked from the simulation main loop in every cycle (refer to [8] for details of control definition in Newton system). A starting-torque applied to a crankshaft is defined in a similar way. Obviously, more sophisticated combustion models can be used to obtain more accurate analyses.

Figure 8(a) show nine frames of a simple simulation. The graphs in Figure 8(b) depict some simulation data, such as the force acting on the piston, the y-coordinate of position of piston in the local coordinate frame of crankcase, and the angular velocity of the crankshaft. We can derive

more interesting data from the simulation if needed.

Example 2: four-cylinder-engine

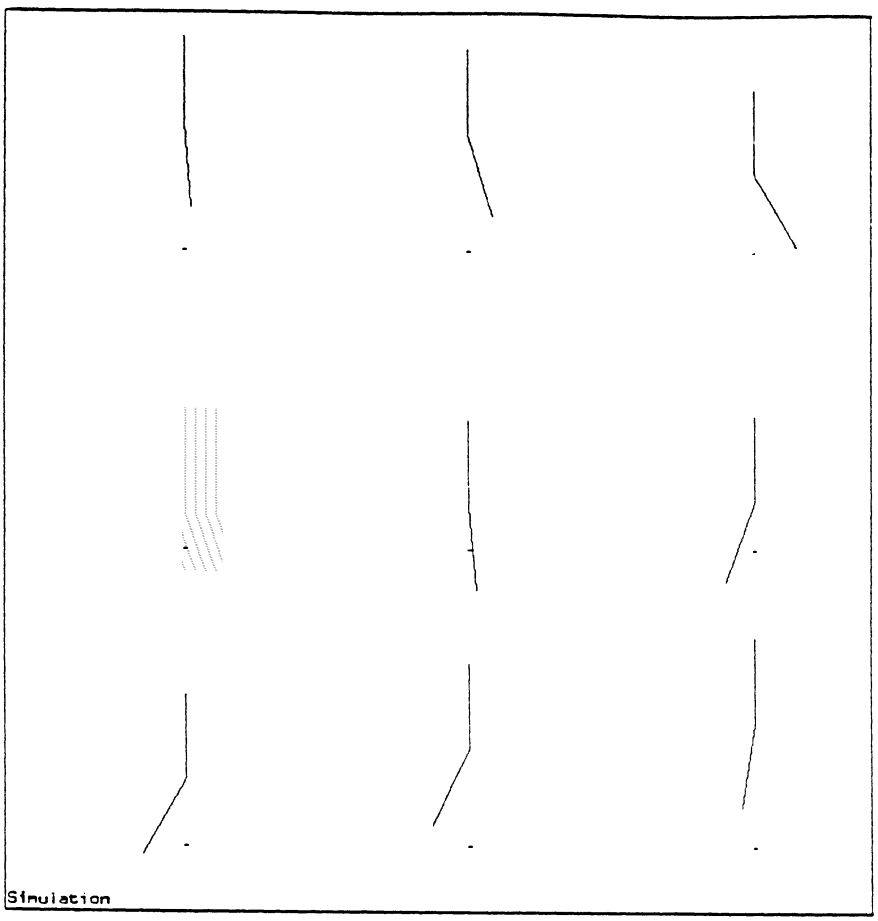
We can easily create a four-cylinder-engine with our schematic model. This highly abstract model demonstrates the simplicity and clarity of schematic representation in contrast to full geometric representation. Figure 9 and 10 show a simulation sequence and some simulation data with the four-cylinder-engine respectively.

Example 3: car-base

By studying a car base, we demonstrate that handling of contacts between volumes and surfaces (in this example between surfaces) is supported by schematic representations at surface and volume levels. Figure 11 depicts a schematic model of a car-base (properly called a dolly). A wheel is represented as a disk. An axis is represented as a line segment. A pair of wheels is rigidly connected to each end of the axis respectively. The base consists of four "steel" frames rigidly connected into a rectangle. Then two wheel assemblies are "mounted" onto the car-base through revolution joints. Contacts between the wheels and ground are governed by a rolling constraint [10]. In this simulation, we assume that the coefficient of friction between contact surfaces is infinite, thus there is no slip between them. During the simulation, the car-base has an initial velocity and no external force is applied. Five frames from this simulation are shown in Figure 11.

Example 4: car

Finally, we mount the four-cylinder-engine onto the car-base (Figure 12). A real transmission system of a car is complicated. In our model, we define a highly abstract joint, called a differential hinge, that constrains the ratio value of angular velocities of two arbitrarily oriented rotational axes to be a constant. This is a good example of abstraction for constraints. Modeling a differential mechanism consisting of a pair of gears at the geometry level would be prohibitive. Our abstraction provides a good approximation to this mechanism. Note that this abstraction is also suitable for U-joints, belt transmissions, etc. We use the simplified combustion model described in Example 1 for each piston. A force applied on the "top" of a piston is transmitted to the crankshaft through its linking rod. The torque of the crankshaft is then transmitted to the rear axis of the car through a differential hinge that drives the rear wheels. The rolling constraint between the wheels and ground controls the relative movements of wheels with respect to the ground surface. Finally, we observe the car accelerating on the ground. Figure 13 shows a sequence of frames from the simulation and selected simulation data in the graphs. The model consists of 21 rigid primitive components and 50 constraint joints. Our example demonstrates that complicated systems with many inter-connecting



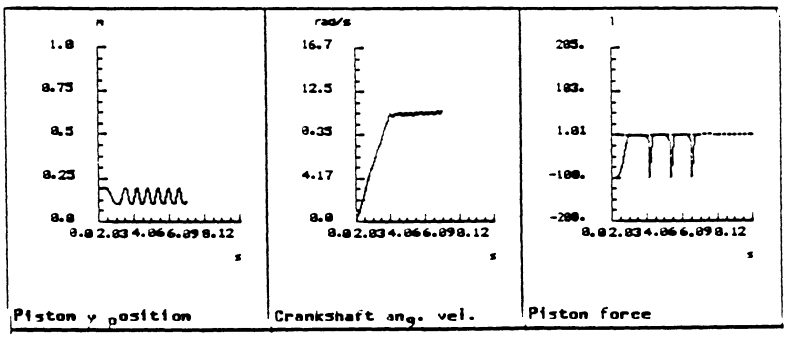


Figure 8: Simulation of one-cylinder-engine

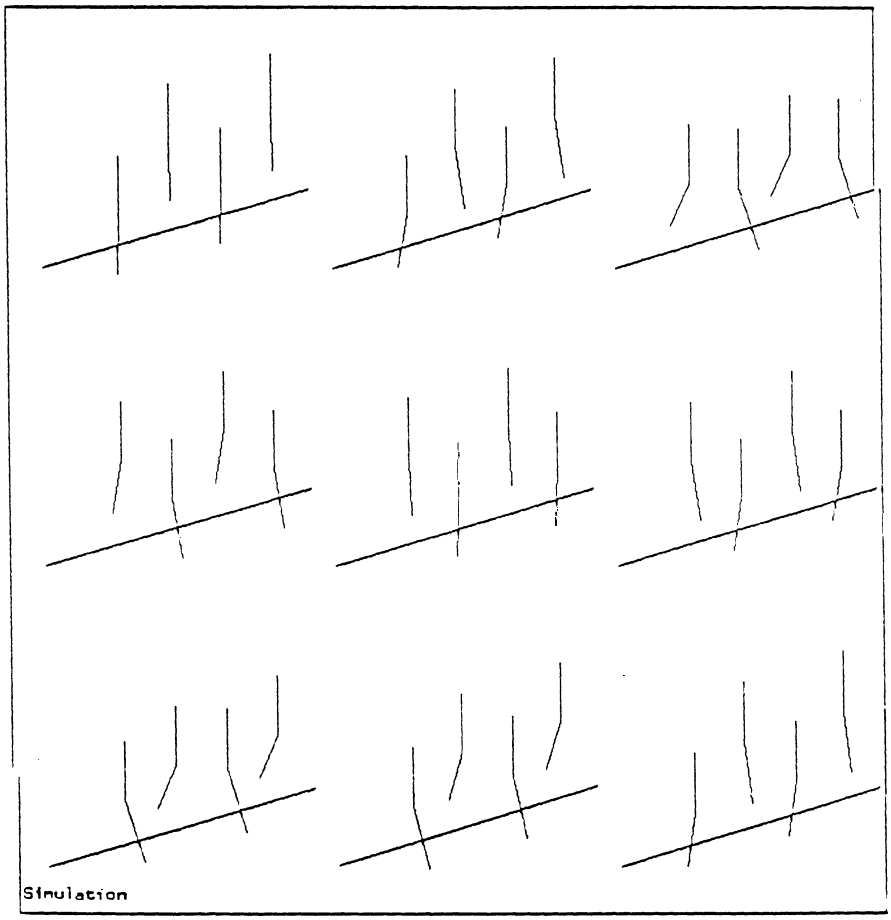


Figure 9: Simulation of four-cylinder-engine

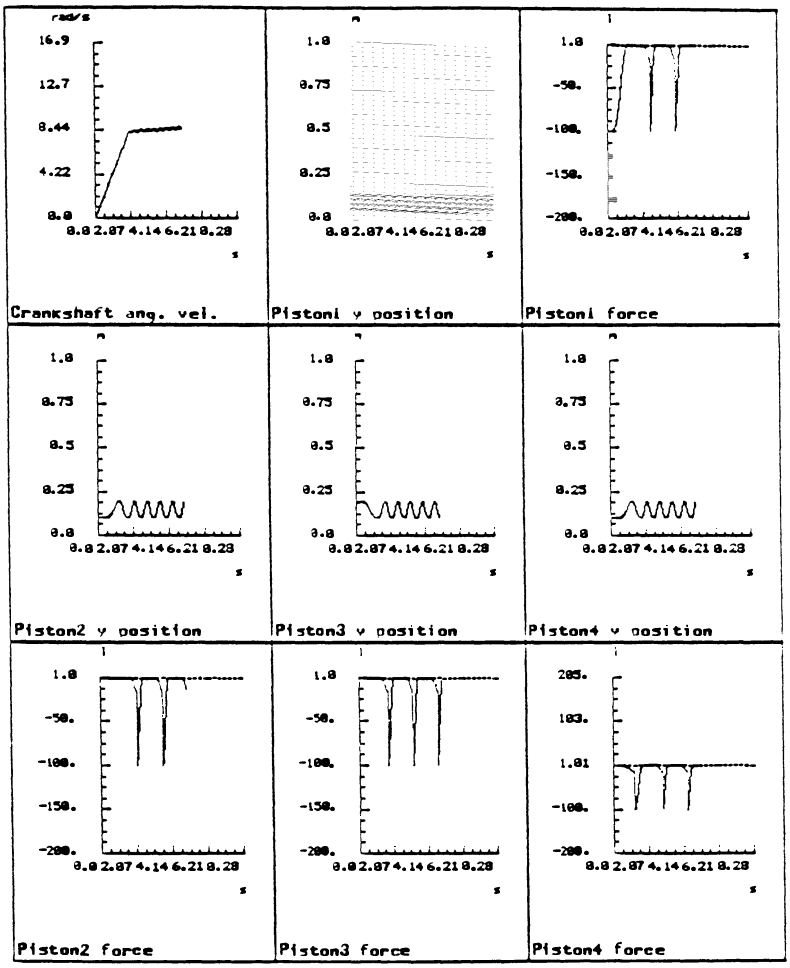


Figure 10: A set of simulation data of four-cylinder-engine

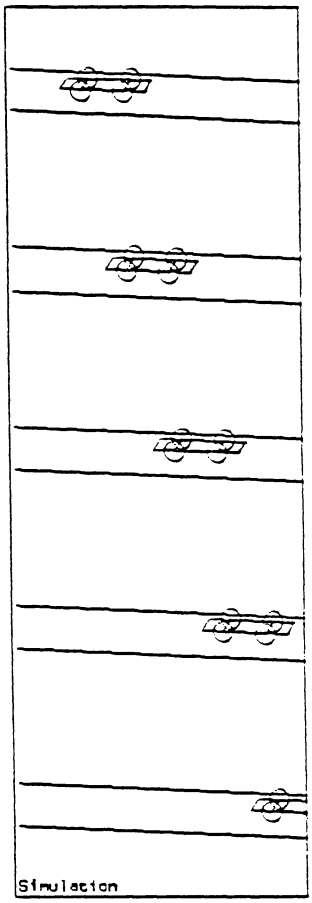


Figure 11: Simulation of car-base

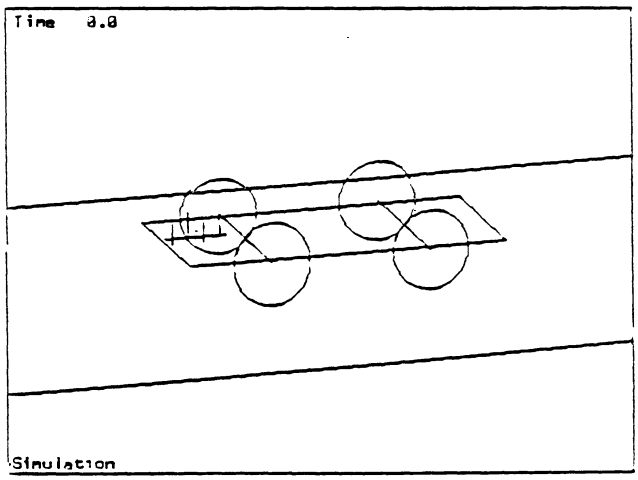


Figure 12: A schematic model of a car for dynamic simulation

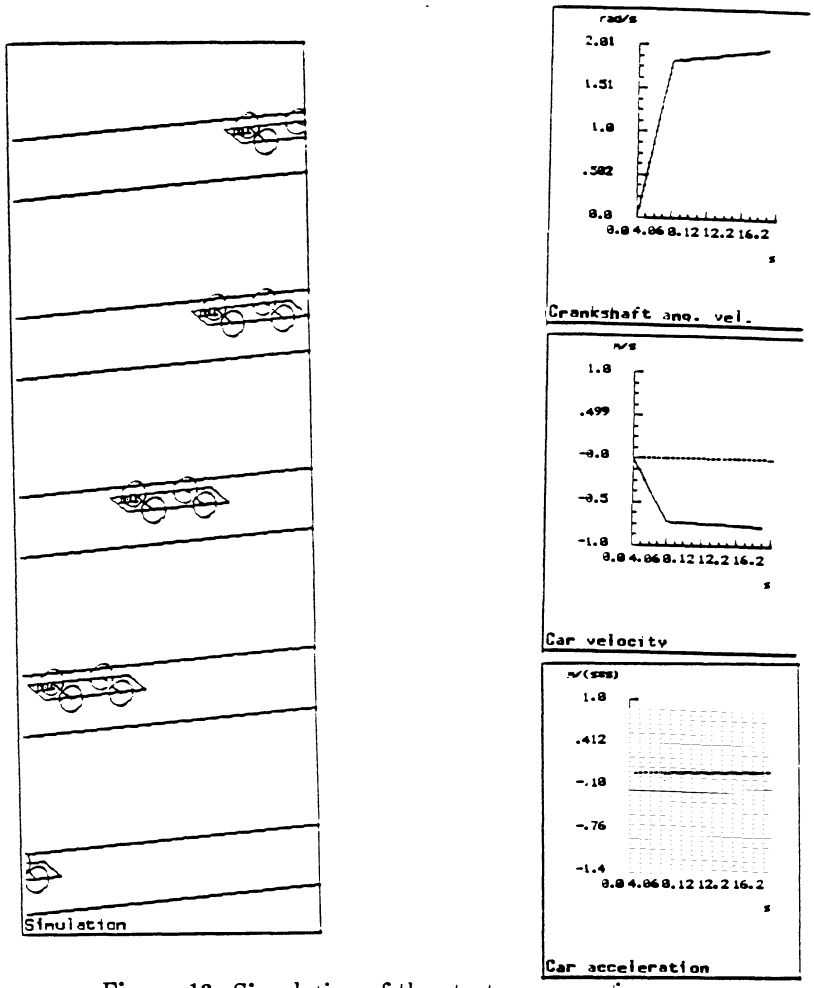


Figure 13: Simulation of the start process of a car

constraints can be schematically modeled to produce kinematic and dynamic simulations.

Example 5: bicycle

Our last example is a simulation of the steering stability of a bicycle. The problem can be difficult to be solved analytically. However, we can design and implement a schematic representation on *Newton* within half an hour, change parameters and modify an existing model within minutes. The simulations produce a variety of analysis results. Figure 14 shows a schematic model of a strangely-shaped bicycle. The dimensions of the components are from [12] and provided by Suresh Goyal of Cornell University. The bicycle consists of front and rear wheels, front and rear body frames, and three revolution constraint joints (one for joining two body frames for steering, and two others for mounting the two wheels to frames respectively). The rolling constraint described above applies to the contacts between the wheels and ground.

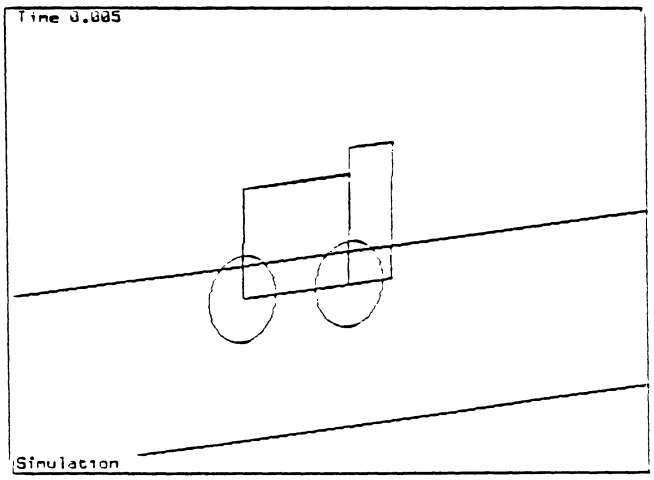


Figure 14: A schematic model of a bicycle

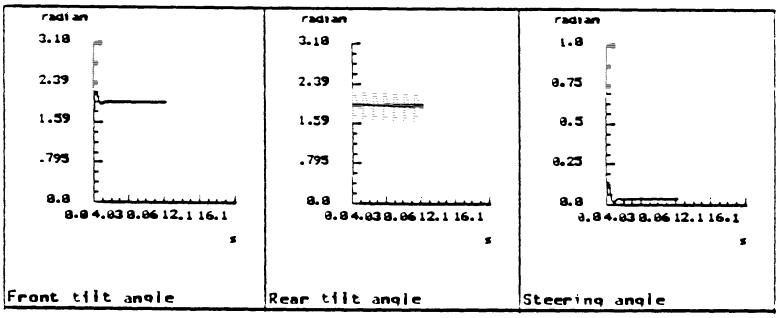


Figure 15: Simulation of the bicycle under a small disturbance

We give a forward velocity of about 10 meters per second to the bicycle and a side-push to the body of the bicycle in order to study its stability under turbulence. Figure 15 shows a selected set of simulation data. The curve in the first graph plots the tilt angle of the front wheel as the function of time, the second curve plots the tilt angle of the rear wheel, and the third curve plots the steering angle. The simulation shows that the bicycle wiggles around initially and converges to a stable state.

5 Discussion

We proposed a method of classifying schematic representations to guide our design of a small, experimental set of schematic primitives at different levels of abstraction in a systematic manner. Our initial work has been restricted to the modeling of rigid components for kinematic and dynamic simulations. We demonstrated that by using just a few schematic primitives, we can model and analyze many physical systems. With abstract schematic representations, we can approximate rigid components of complex shapes in a rather simple format. Thus, the efforts to create models that simulate physical systems is reduced significantly, and effective modeling and simulation of large, complicated physical systems becomes relatively easy.

The advantages of schematic models will become more evident as we develop more sophisticated, general-purpose graphic interfaces and integrate them with analysis modules in our system. So far, we have restricted our applications to problems dealing with kinematics and dynamics of rigid components. Low-level schematic models have been used elsewhere, such as in electric systems, electro-magnetic machinery, electronic circuits etc. However, they are usually restricted to a limited range of problems, or are applicable to large systems but with very expensive implementations. Therefore, their applications are quite limited. High-level abstract schematic models are more popular. Extending our schematic model to include other physical domains such as electrical, electro-magnetic, etc. should be exciting. However, many problems remain to be studied, such as the ability to provide the appropriate schematic model for different physical domains, and studying relations and interactions between different domains.

Acknowledgements

Discussions with Prof. John Hopcroft, Jim Cremer, Suresh Goyal, Anne Neirynck, and Richard Zippel are valuable and helpful. This work is supported in part by NSF grant DMC 86-17355,

ONR grant N00014-86K-0281, and ONR grant N00014-88K-0591, and Sandia Contract 75-6562.

References

[1] Ambler, A., Robotics and solid modeling: a discussion of requirements robotics applications put on solid modeling systems, Robotics Research, The Second International Symposium, pp.361-367, MIT Press, 1985.

[2] Armstrong, W.W. and Green, M.W., The dynamics of articulated rigid bodies for purposes of animation, The Visual Computer, Vol.1, No.4, pp.231-240, 1985.

[3] Badler, N.I., Manoochehri, K.H. and Walters, G., Articulated figure positioning by multiple constraints, IEEE CG&A, Vol.7, No.6, pp.28-38, June 1987.

[4] Bruderlin, A. and Calvert, T.W., Goal-directed, dynamic animation of human walking, Computer Graphics, Vol.23, No.3, pp.233-240, July 1989.

[5] Chadwick, J.E., Haumann, D.R. and Parent, R.E., Layered construction for deformable animated characters, Computer Graphics, Vol.23, No.3, July 1989.

[6] Chace, M., Modeling of Dynamic Mechanical Systems, Presented at the CAD/CAM Robotics and Automation Institute and International Conference, Tuscon, Arizona, February 1985.

[7] Craig, J.J., Issues in the design of off-line programming systems, in Robotics Research, The Fourth International Symposium, pp.379-389, 1988.

[8] Cremer, J.F., An Architecture for General Purpose Physical System Simulation – Integrating Geometry, Dynamics, and Control, Ph.D. Thesis, Dept. of Computer Science, Cornell University, Ithaca, New York, April 1989.

[9] M. Girard, Interactive design of 3D computer-animated legged animal motion, IEEE CG&A, Vol.7, No.6, June 1987.

[10] Goyal, S., Second order kinematic constraint between two bodies rolling, twisting and slipping against each other while maintaining point contact, Tech Rep TR 89-1043, Computer Science, Cornell University, NY, October 1989.

[11] Gourret, J., Thalmann, N.M. and Thalmann, D., Simulation of object and human skin deformation in a grasping task, Computer Graphics, Vol.23, No.3, pp.21-30, July 1989.

[12] Hand, R.S., Comparisons and stability analysis of linearized equations of motion for a basic bicycle model, M.S. Thesis, Dept. of Theoretical and Applied Mechanics, Cornell University, May 1988.

[13] Hill, D.D. and Coelho, D.R., Multi-level simulation for VLSI design, Kluwer Academic Publishers, Boston, 1987.

[14] Hoffmann, C.M. and Hopcroft, J.E., Simulation of Physical Systems from Geometric Models, IEEE Journal of Robotics and Automation, Vol.RA-3, No.3, pp.194-206, June 1987.

[15] Hopcroft, J.E., Electronic Prototyping, Computer, Vol.22, No.3, pp.55-57, March, 1989.

[16] Haug, E.J., editor. Computer Aided Analysis and Optimization of Mechanical System Dynamics, NATO ASI Series. Series F: Computer and Sciences, Vol.9, Springer-Verlag, 1984.

[17] Komatsu, K., Human skin model capable of natural shape variation, The Visual Computer, No.3, 1988.

[18] Magnenat-Thalmann, N. and Thalmann, D., The Direction of Synthetic Actors in the Film Rendez-vous a Montreal, IEEE CG&A, Vol.7, No.12, pp.7-19, 1987.

[19] Magnenat-Thalmann, N., Laperriere, R. and Thalmann, D., Joint-dependent local deformations for hand animation and object grasping, Proc. Graphics Interface'88, Edmonton.

[20] Paul, B., Analytical dynamics of mechanics - a computer-oriented overview, Mechanism and Machine Theory, 10:481-507, 1975.

[21] Wilhelms, J., Using dynamic analysis for realistic animation of articulated bodies, IEEE CG&A, Vol.7, No.6, pp.12-27, June 1987.

[22] Wittenburg, J. and Wolz, U., The program mesa verde for robot dynamics simulations, in Robotics Research, The Third International Symposium, ed. O.D. Faugeras and G. Giralt, pp.197-204, MIT Press, 1986.