# INTEGRATED MODELING, SIMULATION, AND OPTIMIZATION OF MULTIBODY SYSTEMS

**P. EBERHARD and D. BESTLE**

*Institute B of Mechanics, University of Stuttgart, D–70550 Stuttgart, Germany,*
*email: pe@mechb1.fertigungstechnik.uni–stuttgart.de*

**Abstract.** Design of nonlinear dynamic systems is a challenging task. Computers may help to analyze and optimize virtual prototypes in an earl design phase, but there is a lack of design tools. In this paper, a design concept on the basis of a multibody system approach is described. Programs for modeling, simulation, sensitivity analysis, optimization, and animation are operating under a common administration which ensures model consistency. The designer is supported by a graphical user interface, distributed computation improves efficiency.

**Key Words.** Multibody Systems, Optimization, Sensitivity Analysis, Concurrent System Design, Distributed Computation

## 1. INTRODUCTION

Design of mechanical systems is changing from an intuitive product development on the basis of experimental studies to a simulation–based approach. Such virtual prototyping helps to shorten development cycles since problems can be already detected and avoided in an early design phase which is a requirement of concurrent engineering.

Virtual prototyping has to be based on mathematical models. Especially, the multibody system approach has been successfully used for analyzing the dynamic behavior of systems in vehicle dynamics, robotics, machine dynamics, and biomechanics. Although there have been developed several programs for simulating the dynamic behavior of multibody systems (Schiehlen, 1990), there is still a lack of tools for synthesizing nonlinear dynamic systems systematically.

Computer–aided design has to be more than making intuitive design changes on the simulation model instead of the hardware prototype. It is the aim of this paper, therefore, to show how results and methods from several disciplines can contribute to an integrated modeling and design approach.

## 2. INTEGRATED DESIGN APPROACH

An integrated design approach for dynamical systems has to support all steps from problem formulation to problem solution by optimization, Fig. 1. Firstly, the technical system to be optimized has to be transformed to a mathematical model. Modeling techniques like the multibody systems method or the finite element method may be used for this task. Then, design goals have to be defined which is often difficult since technical requirements and human wishes are sometimes hard to formulate as mathematical functions. Beside the complexity of the models this is maybe one of the reasons why even integrated design methods cannot substitute the design engineer, but support by software systems will help to make design faster. In order to improve technical systems, design changes have to be made. Therefore, parameters of the model have to be classified either design variables whose values can be chosen within given bounds or as system constants whose values are fixed during optimization.

Mainly, the behavior of dynamical systems is described by systems of differential or differential–algebraic equations. To evaluate the values of the performance criteria, these equations have to be solved by numerical integration over time. Since this is rather time consuming, one has to use effi-

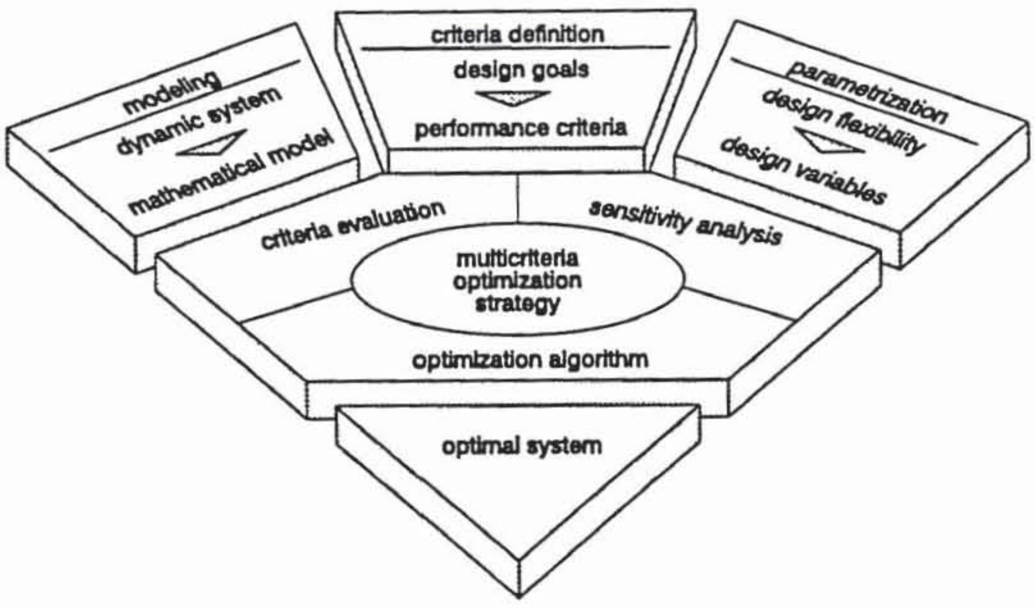


*Fig. 1. Components of an optimization problem*

cient optimization algorithms requiring gradient information. For multibody systems, sensitivity analysis can be performed by the adjoint variable method (Bestle and Eberhard, 1992).

In general, design of dynamic systems with respect to several specifications leads to a multicriteria optimization problem. The designer then has to develop strategies to overcome the problem of conflicting criteria, and to find optimal trade-offs. With concepts like scalarization or hierarchization it is possible to transform the multicriteria optimization problem to a single or a sequence of scalar nonlinear programming problems which can be solved by general purpose optimization codes like sequential quadratic programming or stochastic simulated annealing algorithms. However, the solution of the reduced optimization problem will yield only single points of the whole set of Edgeworth–Pareto optimal solutions of the original multicriteria design problem. The transformation procedure, therefore, has to be part of an iterative design process.

Optimization will always show the weakest elements of the problem formulation enforcing changes in the model or criteria. In order to avoid inconsistencies resulting from such changes, a problem formulation without redundancies has to be used. This can be achieved by an integrated approach including modeling, simulation, sensitivity analysis, animation and optimization. The program system AIMS developed by Bestle and Eberhard (1993) is aimed to support design of nonlinear dynamical systems on the basis of the multibody system approach.

## 3. SOFTWARE CONCEPT

In the beginning of the development of a program system like AIMS, the software engineer has to make several decisions concerning the design guidelines for the whole project. AIMS has been designed to be modular in each component, the portability is supported by avoiding hardware-dependent extensions. Wherever possible, existing reliable software components like integration or optimization routines, communication libraries or plot routines have been used. The smooth integration of such standard software is guaranteed by a well documented interfaces (Bestle and Eberhard, 1994).

On the one hand, AIMS is used in several industrial companies and, therefore, has to be usable with low insight in the internal structure and little mathematical background. On the other hand, AIMS is a research software which allows to exchange or extend modules easily, in order to have a platform for developing and testing new ideas and algorithms. Therefore, a couple of specialized, cooperating programs with limited functionality has been created instead of a single large program with global functionality.

Important modules are the multibody system package NEWEUL (Kreuzer and Leister, 1991) for generating the symbolical equations of motion, and the package MAPLE (Char et al., 1992) for generating symbolical equations to perform sensitivity analysis with the adjoint variable method (Bestle and Eberhard, 1993).

Executing simulation and optimization programs in a concurrent and distributed environment requires the use of a common database for the problem–specific equations, the design variables, and the system constants. Consistency is ensured by shared datafiles and problem defining include files.

The modules of AIMS are operating under common administration. Simple, reliable and efficient management of the software have lead to following requirements:

- **Graphical user interface:** The user is supported by a graphical user interface. A portable X–Windows/Motif based Interface called NEWOPT has been created where it is possible to manipulate all the necessary data without detailed knowledge of the internal structure. This allows to perform simulation and optimization runs after a rather short training period. Since the user interface is completely independent of the numerical procedures, the whole functionality is still available on simple ASCII–terminals.
- **Automatic detection of dependencies:** In order to achieve high performance it is not advisable to use interpreted code. Therefore, the software modules have been divided into three groups:
  (i) Problem–independent routines containing e.g. numerical methods. Such routines must be compiled only once during the installation and remain unmodified during normal use.
  (ii) Problem–dependent routines, e.g. wrapper for the inclusion of the system–describing equations. These routines must be compiled for each problem, but the underlying code also remains unchanged during normal use.
  (iii) Problem–specific, automatically generated include files containing all the information about the system and the criteria. For each problem or model to be investigated this code has to be generated.

  In order to release the design engineer from caring about software dependencies, necessary actions after changing the model or criteria are controlled and supported by UNIX–makefiles. Thus, the user always works with consistent data and unambiguous models.
- **Concurrent and distributed computation:** Simulation and optimization of dynamical systems is a rather time consuming task. Therefore, it is advantageous to distribute the computation load to

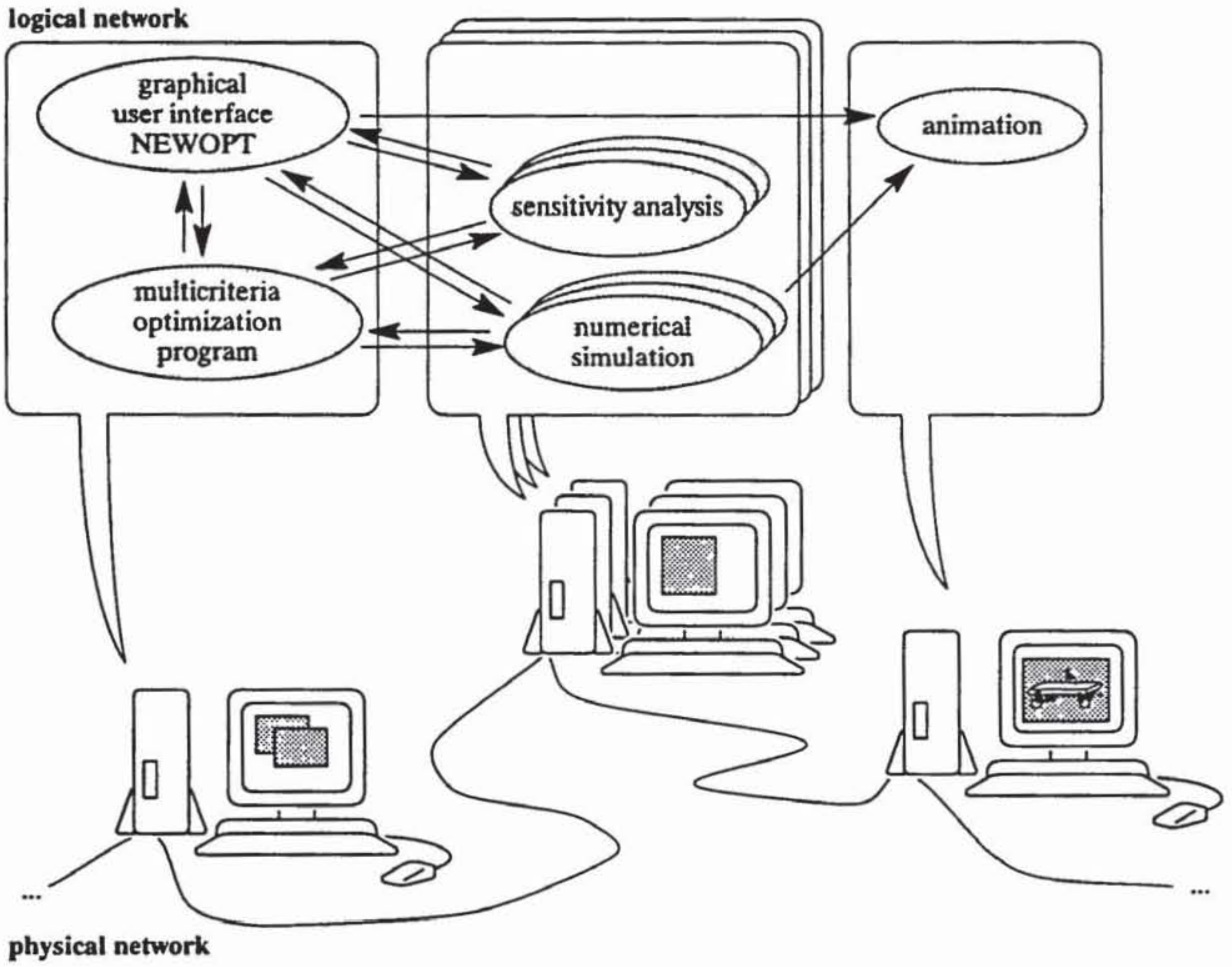


*Fig. 2. Computer network for distributed computation*

different computers and to do the calculations concurrently, i.e. in parallel. Due to network capabilities of modern workstations, workstation clusters may be used for this purpose.

A typical part of a computer network is shown in Fig 2. Distributed programs running on several workstations at the same time to fulfil a common purpose then offer the computing power of supercomputers for a small percentage of their costs.

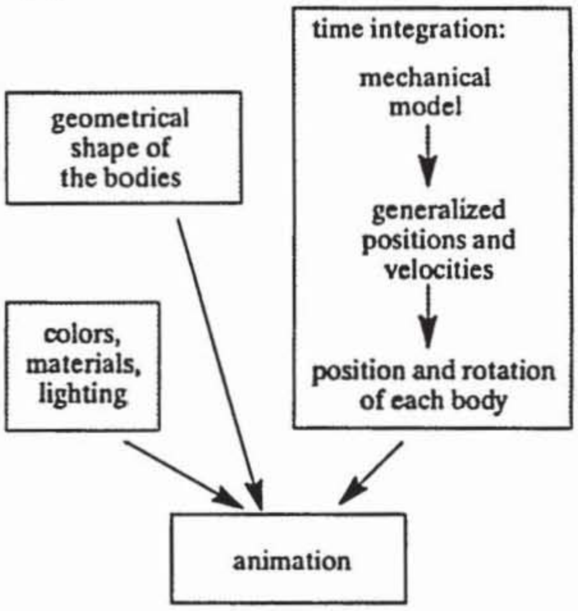
A necessary condition for using such a concept is the possibility to find a suitable partitioning for the whole work to do. Some partitionings are rather natural, e.g. performing the animation on specialized graphic workstations, or the execution of administration programs and numerical calculations on different computers. Some other distributions require a deeper analysis of the algorithms. Often, numerical algorithms have a structure that allows execution in parallel after some changes.

Beside the existence of several computers and a fast network connecting them, the existence of computer libraries to open or close connections between independent programs, to send or receive messages, to pack and unpack data or to synchronize programs is necessary.

This can be achieved by communicating processes exchanging data and results over network protocols. Tools like PVM (Parallel Virtual Machine) can support this task (Beguelin et al., 1994). PVM is available via anonymous ftp from the netlib server for almost every UNIX–computer. Some companies like CRAY and IBM also offer optimal suited versions for their architectures.

- **Animation:** Since it is difficult for people to analyze the large amount of data which is obtained as result of simulation and optimization, graphical representations of the data have to be provided. Beside the traditional xy–plots, bar–charts, etc., the representation by animated moving pictures of the technical system is very useful to check the correctness of the simulation results and will give better insight in the system behavior.

The use of animation software requires some additional data, Fig 3. From the numerical integration one may automatically compute the position and rotation for each part and time–step. Additionally, the geometrical shape, the colors and materials have to be described by preprocessors or –for more complicated bodies– by CAD–programs. Such additional information has to be computed only once if rigid body motion is investigated. All transformations and drawings are done in the animation software itself.



time integration:
mechanical model
generalized positions and velocities
position and rotation of each body
geometrical shape of the bodies
colors, materials, lighting
animation

*Fig. 3. Data for computer animation*

The animation package included in AIMS offers some special features like output synchronization, space–mouse support, or stereo viewing with special 3D–glasses. Fig. 6 shows a screen-copy of a vehicle animation.

## 4. MULTIBODY SYSTEM DESIGN

The multibody system approach can be successfully applied to mechanical systems where the individual parts undergo large translational and rotational displacements whereas the deformation of the parts themselves can be neglected. The basic elements of a multibody system model are rigid bodies, coupling elements like springs, dampers or active force elements, and joints like bearings or ideally position controlled elements, see Fig. 4.

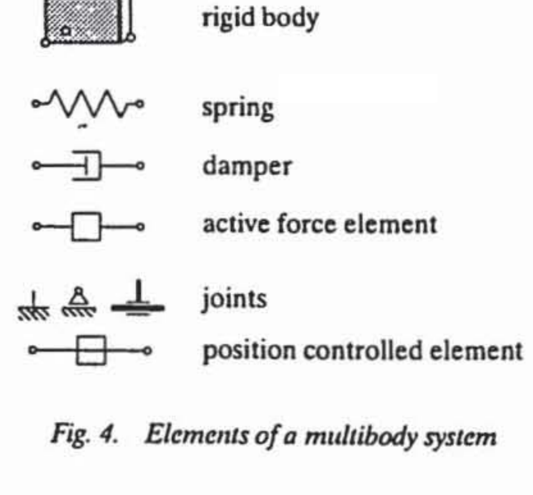


*Fig. 4. Elements of a multibody system*

Applied to vehicle dynamics a spatial model may look like the one given in Fig. 5. The model has

$f = 11$ degrees of freedom, its motion can be described by three angular and three translational coordinates for the car body, a translational coordinate for the driver's seat, and angles for the wheel displacements, respectively. These generalized coordinates are summarized in a vector $y \in R^f$. Analogously, the translational and angular velocities of the individual bodies are described by generalized velocities $z \in R^g$ where we have $g = f$ for holonomic multibody systems. Position and velocity is then given by initial conditions and differential equations of motion

$$\begin{aligned} \dot{y} &= v(t, y, z, p), \\ M(t, y, p)\dot{z} + k(t, y, z, p) &= q(t, y, z, p) \end{aligned} \qquad (1)$$

which can be found from Newton's and Euler's law and d'Alembert's principle (Schiehlen, 1986). In AIMS, the generation of these differential equations is supported by the formalism NEWEUL (Kreuzer and Leister, 1991).

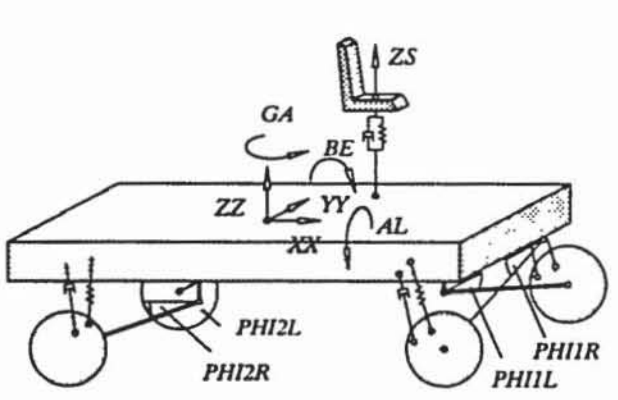


*Fig. 5. Spatial vehicle model*

Modeling technical systems as multibody systems already involves an implicit parametrization. The dynamic behavior of the model is completely determined by parameters like the mass and moments of inertia of each body, geometrical dimensions, and damping and stiffness coefficients of coupling force elements. The parameters which can be changed within given ranges for optimizing the dynamical behavior are considered as design variables $p \in R^h$.

Examples for design goals in vehicle dynamics are ride comfort and ride safety. Both performance measures can be expressed as functionals like

$$\psi(p) = G^1(t^1, y^1, z^1, p) + \int_{t^0}^{t^1} F(t, y, z, \dot{z}, p)\, dt \qquad (2)$$

where $[t^0, t^1]$ is a time interval of interest. The vector functions $F$ and $G^1$ evaluate the performance of the dynamic system within the given time interval and at the final time $t^1$, respectively. According to the dynamic behavior, the value of $\psi$ is completely determined by choosing special values for the design variables $p$.

Especially in vehicle dynamics it is useful to simultaneously investigate different models with common parameters or single models with different driving manoeuvres or excitations. The design variables and criteria of these subproblems can then be combined to a general multicriteria optimization problem. These problem structures fit very good into the concept of distributed concurrent engineering.

For the use of gradient–based optimization algorithms, gradients of the performance function (2) with respect to the design variables $p$ have to be computed. Since the state $y$, $z$ and thus $\psi$ is not given as explicit functions of $p$, this cannot be performed by direct differentiation methods. Purely numerical methods like finite differences also fail due to inaccuracy of numerical integration of the equations of motion. The adjoint variable method resulting in additional differential equations has proven to be highly efficient and reliable for computing the total derivatives of $\psi$ with respect to the design variables $p$ (Bestle and Eberhard, 1992, Bestle, 1994). Therefore, it has been implemented in the program package AIMS. Required partial derivatives of the functions given in the equations of motion (1) with respect to the state and the design variables are performed automatically by MAPLE (Char et al., 1992). For very complicated functions one can also use programs for Automatic Differentiation, e.g. ADIFOR (Bischof et al. 1992) to compute the partial derivatives.

Within AIMS, optimization results can immediately be made visible by animation, Fig. 6. In the case shown, the front vehicle has been optimized with respect to maximum ride comfort for driving over a bump. To make the difference more evident, the rear vehicle has been optimized with respect to to minimum ride comfort showing move higher amplitudes.

## 5. CONCLUSIONS

Increasing computing power and network capability of modern workstations enables design engineers to do computer–aided design with respect to dynamical as well as geometrical aspects. However, there is a need for integrated modeling and design software. Mechanical engineering as well as computer science and the optimization community can contribute to create flexible, modular concepts to support modern product design.

## 6. REFERENCES

Beguelin, A. et al. (1994). *A User's Guide to PVM: Parallel Virtual Machine*, Technical Report ORNL/TM–12187. Oak Ridge National Laboratory, Oak Ridge.

Bestle, D. (1994) *Analyse und Optimierung von Mehrkörpersystemen*. Springer, Berlin.

Bestle, D. and Eberhard, P. (1992). Analyzing and Optimizing Multibody Systems. *Mech. Struct. and Mach.*, **20**, 67–92.

Bestle, D., and Eberhard, P. (1993) *Optimierung in der Fahrzeugdynamik*, Forschungsbericht FB–19. Institute B of Mechanics, University of Stuttgart, Stuttgart.

Bestle, D. and Eberhard, P. (1994). Automated Approach for Optimizing Dynamic Systems. In: *Computational Optimal Control*, (R. Bulirsch and D. Kraft, Eds.), pp. 225–235. Birkhäuser, Basel.

Bestle, D. and Eberhard, P. (1994). *NEWOPT / AIMS 2.2 Ein Programmpaket zur Analyse und Optimierung von mechanischen Systemen*, Anleitung AN–35. Institute B of Mechanics, University of Stuttgart, Stuttgart.

Bischof, C. et al. (1992). ADIFOR Generating Derivative Code from FORTRAN Programs. *Scientific Programming*, 1(**1**), pp. 11–29.

Char, B.W. et al. (1992). *Maple V, First Leaves*. Springer, New York.

Kreuzer, E. and Leister, G. (1991). *Programmsystem NEWEUL'92*, Anleitung AN–23. Institute B of Mechanics, University of Stuttgart, Stuttgart.

Schiehlen, W. (1986). *Technische Dynamik*. Teubner, Stuttgart.

Schiehlen, W. (Ed., 1990). *Multibody Systems Handbook*. Springer, Berlin.

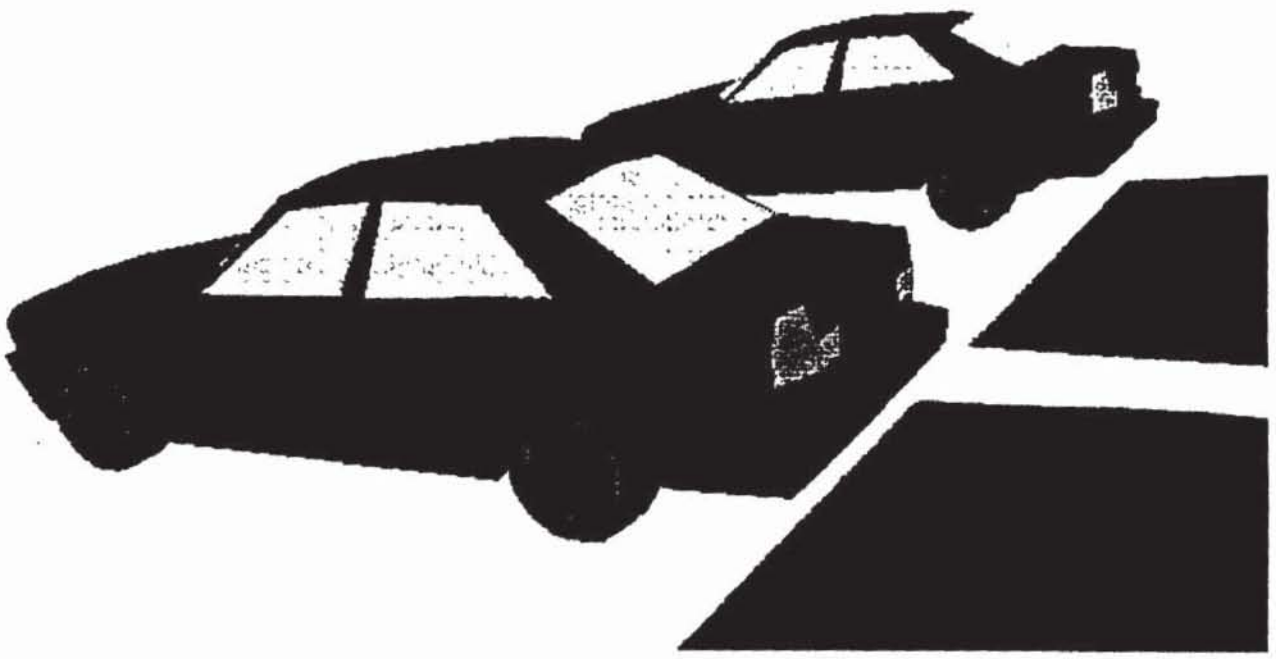


*Fig. 6. Animation*