We are IntechOpen,
the world's leading publisher of Open Access books
Built by scientists, for scientists

5,000
Open access books available

125,000
International authors and editors

140M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Virtual Prototyping Platform for Designing Mechanical and Mechatronic Systems

Cătălin Alexandru

Abstract

The chapter deals with the description of a virtual prototyping platform that facilitates the design process of the mechanical and mechatronic systems. The virtual prototyping stages are defined and then integrated in a block diagram, highlighting how the data are transferred between these stages in order to finally obtain a valid and optimal virtual model, close (as structure and functionality) to the real one. The whole process is guided by the basic principle for successful virtual prototyping: as complicated as necessary and as simple as possible. The real modeling case, the specific simplifying assumptions, and the validity (viability) fields of the simplifying assumptions are discussed with reference to the main components of a mechanical or mechatronic system (bodies, connections between bodies, actuating elements). The purpose is to manipulate the simplifying assumptions in a way that reduces the complexity of the virtual model, but without altering the accuracy of the results. The basic types of analysis/simulation are depicted by considering their particularities, highlighting their role in the process of designing mechanical/ mechatronic systems, and then the optimization is conducted by the use of parametric design tools. Finally, a case study is developed following those mentioned above.

Keywords: virtual prototyping, mechanical and mechatronic systems, modeling, simulation, optimal design

1. Introduction

In the process of product development, as in many other fields, using the computer is no longer just a useful alternative to classical instruments, but it has become a real need. The computer is currently used from the concept elaboration stage until the manufacturing and implementation. Designers now have access to very sophisticated and high-performance working tools, based on software solutions dedicated to the various stages of product design and development. The traditional computer-aided design (CAD) and computer-aided manufacturing (CAM) approaches are now being addressed through computer-aided engineering (CAE) integrating platforms, which allow the evaluation and improvement of the product at the system level and not separately on its parts or subsystems, such an approach being reflected in increasingly efficient and competitive products [1, 2].

As the complexity and the competitiveness requirements of the products (in this case, mechanical and mechatronic systems) increase, the design and development times must be reduced, conditions in which the development and testing of

physical prototypes become major impediments. Thus, it is necessary to implement design techniques based on modeling, simulation, and optimization in virtual environment, which can ensure a higher performance and quality of the products using only a fraction of the time and cost required in traditional approaches. Virtual prototyping is a computer-aided engineering-based discipline that entails modeling products, simulating and optimizing their behavior under real-world operating conditions. Through the use of various types of software solutions for evaluating the form, functionality, and durability of the products in an integrated approach, complex digital prototypes can be created and then used in virtual experiments (lab and field tests) in a similar way to the real cases [3–5].

In this context, the chapter proposes to present the integrated concept of modeling, simulation, and optimization of the behavior of mechanical and mechatronic systems through the use of a virtual prototyping software platform. The platform integrates specific software solutions for evaluating the form, assembly, functionality, and durability of mechanical and mechatronic systems. The components of the virtual prototyping platform are depicted by mentioning their particular role, as well as the mode in which they are integrated within the platform and communicate (data transfer) with each other. Then, the virtual prototyping stages are discussed starting from a flowchart reflecting the mode in which the data are transferred from one stage to another, to obtain a valid and optimal virtual prototype, these being the two attributes that the virtual prototype must have in order to be a truly useful/viable one. Finally, a case study is developed by considering a complex product, namely, a suspension system for motor vehicles, which is approached in mechatronic concept, by integrating the two main subsystems (the mechanical and actuating and control devices) at the virtual prototype level.

2. The software platform for virtual prototyping

In the general case, the virtual prototyping platform of the mechanical systems integrates three basic software solutions (**Figure 1a**): computer-aided design, multibody systems (MBS), and finite element analysis (FEA). In addition, in the case of mechatronic systems (mechanical systems with controlled actuation), the virtual prototyping platform integrates a design for control (DFC) software solution (**Figure 1b**), in the concurrent engineering concept (for the purpose of co-simulation).

Firstly, with the help of CAD software, the geometrical (solid) 3D model of the mechanical system is developed, with the purpose to determine the mass and inertia properties (moments and products of inertia) of the bodies (rigid parts). The 3D model is then transferred to the MBS software, which is intended to analyze and optimize the behavior of the mechanical system (in terms of kinematics, statics, and dynamics, by case). The data transfer from CAD to MBS is performed using
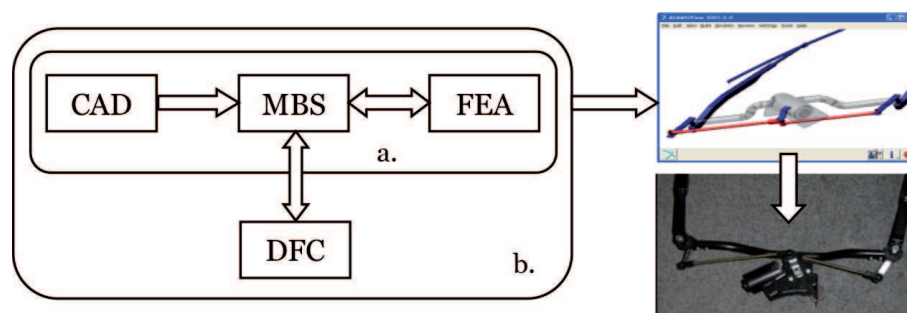

Figure 1.
The virtual prototyping software platform for mechanical (a) and mechatronic (b) systems.

standard geometry file formats, such as STEP, IGES, Parasolid, stereolithography, and others. From this point of view, there are no rules, but only certain recommendations of the software producers regarding the file format. For example, the recommended geometry transfer formats from the main CAD software to the MBS environment Automatic Dynamic Analysis of Mechanical Systems (ADAMS) of MSC Software (which is a global leader in virtual prototyping software and services) are presented in **Table 1**. With such file formats, the import into ADAMS is done through the general ADAMS/Exchange transfer interface. At the same time, specialized modules (interfaces) for geometry transfer were developed, which perform a customized transfer between the CAD and MBS ADAMS environments, as it is also presented in **Table 1** [6].

Initial Graphics Exchange Specification (IGES) format represents the first standard of interchangeability, being designed in American Standard Code for Information Interchange (ASCII) code. IGES reduces the CAD model to a list of entities, each entity being associated with a number. Drawing Exchange Format (DXF) is also based on graphical entities, for each data type, which is ASCII encoded, being allocated a line. Standard for the Exchange of Product Model Data (STEP) format describes the data at the product level and not the entity, through a specialized language (Express) that establishes the correspondence between the STEP file and the CAD model. Stereolithography (STL) format is a neutral format based on stereolithography, being used mainly in rapid prototyping devices (laser printing). Parasolid format is a geometric modeling kernel that allows transferring the entire 3D solid model through a single file, while in the case of the other formats, the transfer is done part by part (one file for each part).

The mass and inertia properties of the bodies are automatically calculated by the MBS software (ADAMS, in this case), depending on the 3D solid model imported from CAD and the associated material (defined by the well-known characteristics/properties: Young's modulus, Poisson's ratio, and density). Most MBS software solutions (including ADAMS) have their own solid modeling library, the modeling principles being the same as in CAD (elementary solids, composite solids using Boolean operations (union, extraction, and intersection), solids obtained by extrusion and, respectively, rotating surfaces), but for bodies with more complex geometry, the use of specialized CAD environment is required.

CAD software	File formats	Transfer interfaces
Unigraphics (UG)	Parasolid STL	UG/Mechanism
CATIA	STL STEP IGES	CAT/ADAMS
Pro/ENGINEER (currently Creo Elements/Pro)	STL IGES	MECHANISM/Pro
SOLIDWORKS	Parasolid STL IGES	Dynamic Designer
I-DEAS	STL IGES	Mechanism Design Mechanism Simulation
Mechanical Desktop	IGS STL DXF	Dynamic Designer

Table 1.
The file formats and transfer interfaces from CAD to ADAMS.

Based on the results of the dynamic analysis performed in the MBS environment, the system loads by forces and torques are determined, representing input data for the analysis with finite elements within the FEA software. Subsequently, the deformability state of the components is returned in the MBS software, thus making possible the dynamic analysis of the mechanical system with deformable (flexible) parts, which is more realistic (closer to reality) than the analysis with rigid parts [7, 8]. Through the analysis of compliant models, the stress and vibration states can be determined with the purpose to evaluate the functional and durability performances of the mechanical system. The data transfer from ADAMS to the main FEA environments (such as ANSYS, ABAQUS, or NASTRAN/PATRAN) is done through FEA Loads type format, from the general ADAMS/Exchange transfer interface. The FEA to MBS connection is usually made through the modal neutral file (MNF) format. In ADAMS, the data import from FEA is managed by the ADAMS/Flex interface. It should also be mentioned that ADAMS software package integrates a specialized module, called ADAMS/AutoFlex, which can be used for the conversion of rigid bodies into deformable equivalents, but with certain limitations in the case of more complex geometry bodies, for which it is still necessary to use specialized FEA software.

Finally, regarding the communication between ADAMS and the DFC software environments, in the case of mechatronic systems, the ADAMS package integrates the plug-in ADAMS/Controls through which the data transfer with one of the following DFC software is carried out: MATLAB/Simulink, MATRIX$_X$, and EASY5. Basically, ADAMS/Controls manages the input and output plants of the controlled process (as mentioned above, the outputs from the MBS are inputs into DFC and vice versa respectively), allowing to perform the co-simulation (in-parallel running/processing) of the two main subsystems of a mechatronic system, namely, the mechanical device and the actuating and control device. The information related to the input and output plants are saved in a specific file having the extension .m (for MATLAB) or .inf (for EASY5 and MATRIXX). At the same time, a command file (.cmd) and a data file (.adm) are generated, which are used during the co-simulation process. The files thus generated by ADAMS/Controls are then imported into the DFC application, where the ADAMS interface block is subsequently set and the control block model is designed. It should be mentioned that ADAMS provides some facilities for control system design, which are integrated into the Controls Toolkit module, but obviously not up to the level of complexity offered by dedicated DFC software.

By integrating the mechanical device and the control system at the virtual prototype level, the two models/subsystems are simultaneously tested and verified, thus simplifying the experimental testing process and eliminating (or at least minimizing) the risk that the control law is not accurately tracked (complied) by the mechanical device [9, 10]. Such a mechatronic concept approach is known as concurrent engineering. The simulation algorithm for mechatronic systems involves the following steps:

1. Within MBS software: modeling the mechanical device (including bodies, joints, actuating elements, other force generating elements), analyzing-simulating the MBS model, modeling the input (I) and output (O) plants in/from the MBS model, and exporting the MBS model for DFC

2. Within DFC software: importing the mechanical model, synthesizing the desired trajectories of the mechatronic system and modeling the input block diagram (reference signals synthesis), designing the control system block diagram, synthesizing the controller and the electrical interfacing circuits, and simulating the mechatronic system

The so described simulation process creates a closed loop, in which the controlled inputs of the control application affect the simulation in the MBS environment, while the outputs from the MBS simulation affect the level of the controlled signals in DFC.

3. The virtual prototyping process

A complete virtual prototyping process is defined by the following five stages (see also the workflow schematic representation in **Figure 2**): modeling, analysis, validation, refining, and optimization. During the modeling stage, the specific components of the mechanical or mechatronic system (such as bodies, connections between bodies, actuating elements, and other force generating elements) are created by using the software solutions shown in **Figure 1**. The output from modeling is the initial virtual model, which is then analyzed (simulated/tested) with the purpose to determine the behavior of the mechanical or mechatronic system, in terms of movement (linear or angular positions, velocities, and accelerations, by case) and reaction force states. The results obtained through the simulation in virtual environment (which are the analysis outputs) are then compared with the corresponding experimental results obtained by physical prototyping, in order to validate the virtual model. It should be mentioned that the physical prototyping is not a stage in itself of virtual prototyping, but a supporting process for this. By the comparative analysis of the virtual and experimental results, one of the following two cases can be reached: valid virtual model (when the virtual results fit with the experimental ones) and invalid virtual model (when the results obtained through the simulation in virtual environment do not match the experimental ones). In the first case, the last step of the virtual prototyping process will be the optimization, which aims to determine the optimal design of mechanical or mechatronic system (in terms of functionality, efficiency-energetic, or economic, by case). On the other hand, if the validation output is expressed by an invalid virtual model, the refining stage must be accomplished with the purpose to improve the fidelity of the virtual model by reference to the physical one. The refined virtual model is then analyzed (by simulation in virtual environment), followed by a new validation. In this way,
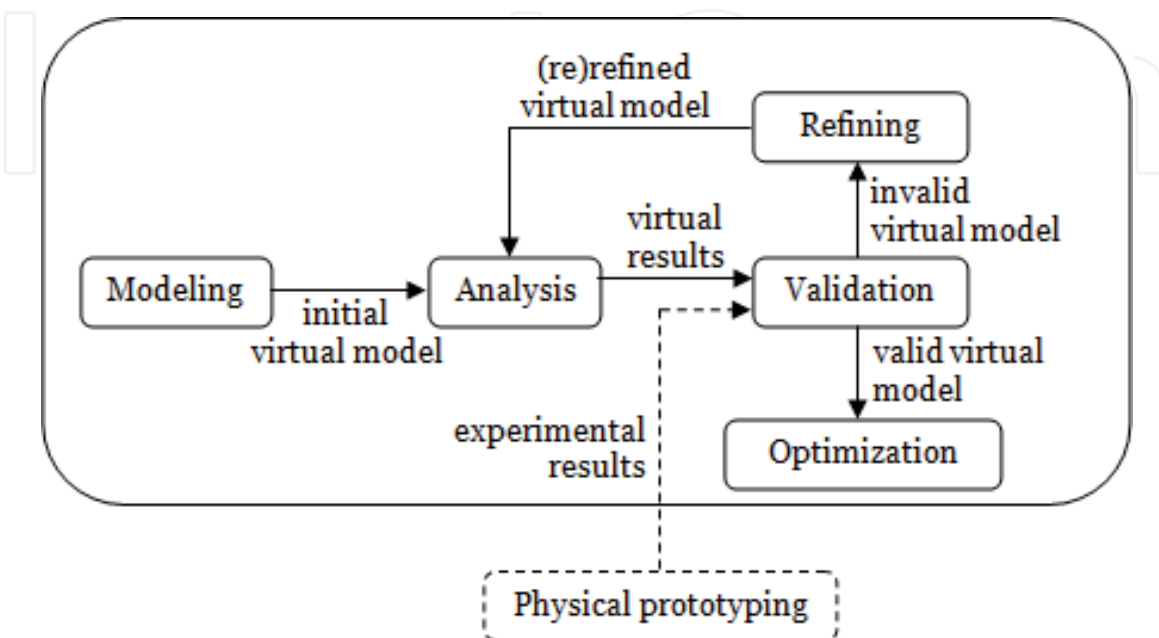

Figure 2.
The virtual prototyping workflow.

an iterative process (refining—analysis—validation) is carried out until a valid virtual prototype is obtained, which will be then the subject for optimization.

The basic principle for a successful virtual prototyping process can be formulated as follows: as complex as necessary and as simple as possible. This is in compliance with Einstein's statement: "A scientific theory should be as simple as possible, but no simpler." The idea is to manipulate the simplifying assumptions in a way that reduces the complexity of the virtual model (in order to make the real-time simulation), but without affecting/altering the precision of the results. In other words, a useful virtual prototype should be a trade-off between simplicity and realism. In the following, the implementation of this basic principle regarding the modeling and refining will be discussed for the basic components of a mechanical or mechatronic system, namely, bodies, connections between bodies, and actuating elements. For each of them, the real modeling case and the specific simplifying assumptions (hypotheses) are presented in **Table 2**.

In the real case, all the bodies are flexible (deformable), more or less, depending on the state of loading to which they are subjected, having constant mass (in most cases) and variable inertia properties (by changing the geometric shape). The simplifying assumptions for the modeling of the bodies are obtained from the real case by successively neglecting certain properties, as follows:

1. Rigid bodies: the shape of the bodies does not change during the analysis, so their inertial properties become constant.

2. Point masses: the shape is neglected by considering that the whole body mass is concentrated in a point (the center of mass), and in this way the inertia properties are not taken into account.

3. Bodies without mass (massless bodies): both the mass and the inertia properties are neglected as a consequence of the fact that the bodies are modeled by 2D elements/objects (such as lines, polylines, plane curves).

4. Composed restrictions: this is a special modeling case in which certain bodies are modeled as constraints between other bodies, such as constant distance or area constraints.

Components	Real case	Simplifying assumptions
Bodies	• Deformable (flexible) bodies	• Rigid bodies
		• Point masses
		• Bodies without mass (massless bodies)
		• Composed restrictions
Connections between bodies	• Deformable (flexible) contacts	• Rigid contacts
		• Joints
		• Constraint equations
Actuating elements	• Motors/actuators	• Motor forces or torques
	• Human operators	• Motion restrictions
	• External factors	

Table 2.
The modeling of the basic components.

The modeling of the bodies by composed restrictions is not possible for all the bodies in a mechanical or mechatronic system, but only in the following cases:

- The body is a mobile one, and not the fixed part of the system (which must remain the reference part to which the global reference frame is attached).

- The body is not input (by which movement is introduced into the system) or output part (from which the movement is collected), and this is because the movement can be introduced/collected only through/from bodies.

- No external forces or torques are applied to the body, or no force generating elements are connected to the body, and this is because the forces can only act on bodies.

A more detailed discussion on the modeling of the bodies as composed restrictions can be performed in correlation with the MBS models of the four-bar mechanism is schematically represented in **Figure 3**. So, the model shown in **Figure 3a** is a general one, with four bodies (three mobile parts, 1, 2, and 3, and the fixed part/ground, 0). Then, in **Figure 3b** and **c**, there models shown with three bodies (two mobile parts, 1 and 2 or 3, and the fixed part, 0) and one composed restriction (constant distance between the corresponding ends of the rod 2 and ground, respectively, of the crank 1 and rocker 3). Finally, in **Figure 3d**, the model is shown with a minimum number of bodies (the rod 2 and the fixed part) and two composed restrictions between the two bodies. The model with a minimum number of bodies is valid one only if the mobile body is at the same time input and output of the system, and this is possible because the body has three movements (two translations along the two axes of the representation plane and one rotation around the axis normal to this plane). The four MBS models shown in **Figure 3** are defined by the following numbers of generalized coordinates (movement parameters—6 per each mobile body): 18 (a), 12, (b and c), and 6 (d). Therefore, the model with a minimum number of bodies is the most convenient from the point of view of the complexity, which depends on the number of equations for determining the behavior of the system.

The connections between bodies (excepting the previously discussed composed restrictions, which are not connections with a physical equivalent) are nothing else than contacts between the geometric forms/shapes of the bodies. These contacts can be classified in two groups, with representative examples in **Figure 4**, as follows:

- Stationary (permanent) contacts (**Figure 4a**), where the connection between bodies is kept as in the initial state during the entire analysis range (throughout the system operation)

- Nonstationary contacts (**Figure 4b**), where the connection between bodies changes during the analysis, either the contact is lost or it occurs or the type of contact changes (e.g., from surface contact to linear or point contact)

Given that the real modeling case of the bodies is flexible (deformable) bodies, the real modeling case for the connections between bodies will be contacts between flexible bodies (or briefly, flexible contacts). Then, the first simplifying assumption for the modeling of the bodies (rigid bodies) is automatically transferred to the modeling of the connections, resulting rigid contacts (contacts between rigid
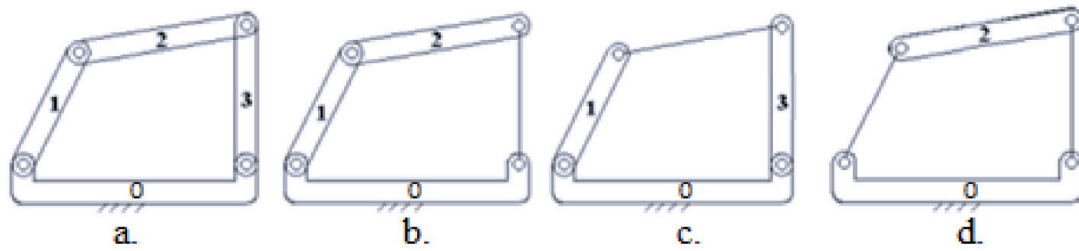
Figure 3.
MBS models for the four-bar mechanism: (a) 4-body model, (b, c) 3-body models and (d) 2-body model.
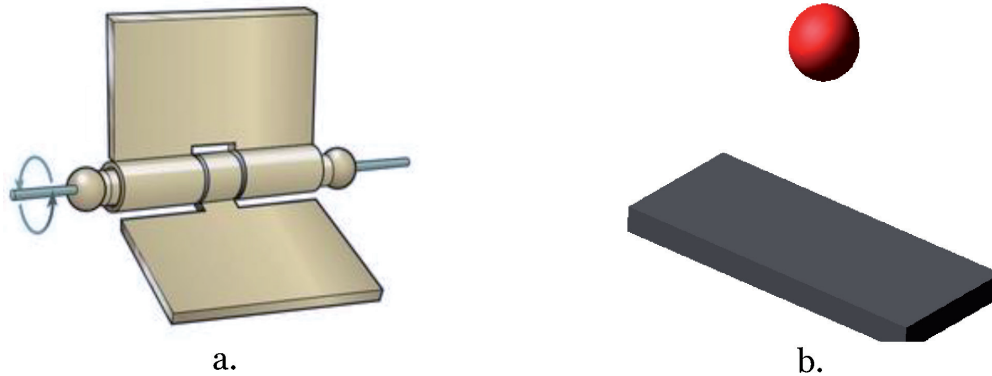

Figure 4.
Types of contacts (connections) between bodies: (a) stationary and (b) nonstationary.

bodies). Both in the case of flexible contacts and for rigid contacts, the connections do not restrict movements, but they introduce reaction forces and torques. The other simplifying assumptions for the modeling of the connections are the ones that restrict movements, namely, joints and constraint equations, which can be used only for the stationary connections (such as that shown in **Figure 4a**, where the contact between the two bodies of the hinge can be modeled as a revolute joint). It should be mentioned that the joint is a symbolical representation (like a modeling shortcut) in the software of the constraint equations, which can be classified in two categories: constraint equations generated by the software (through user-modeled joints), and constraint equations created by the user.

The actuating elements of the mechanical/mechatronic systems are usually found in the following categories (**Figure 5**): (a) rotary or linear motors/actuators, (b) external factors (such as the wind action for a wind turbine or the road irregularities for a vehicle suspension system), and (c) human operators. Whatever the case, the actuating elements generate mechanical power, which is defined by two components: force and movement. In these terms, the actuating element can be modeled by one of the two mentioned components: motor force/torque and motion restriction. The latter, by which the movement of the actuated (input) bodies is controlled, can be applied at the position, velocity or acceleration (linear or angular, by case) level, usually as in time variation laws. On the other hand, the motion restrictions can be applied in joints (thus controlling the relative motion between the adjacent bodies, as in the case of the jack mechanism shown in **Figure 5c**, where the relative motion between the crank and the fixed support can be controlled in the revolute joint between the two) or in points (thus controlling the spatial or planar positions of certain points of interest on the body, for example the point located in the end-effector extremity of the industrial robot shown in **Figure 5a**).
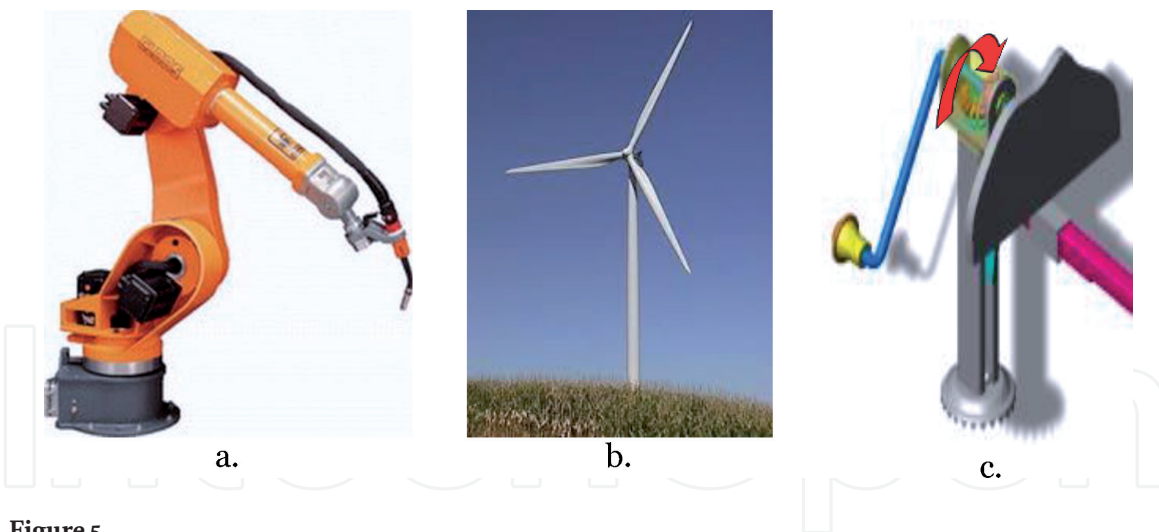
Figure 5.
Types of actuating elements of the mechanical/mechatronic systems: (a) motors, (b) external factor and (c) human operator.

4. The analysis flowchart

As it results from the ones presented in the second section of the paper, the central component of the virtual prototyping platform is the MBS software solution, which is the integrative solution used to simulate and optimize the behavior of the mechanical and mechatronic systems [11–15]. The analysis flowchart in MBS environment is schematically represented in **Figure 6**. The types of analysis can be performed separately or coupled in a certain sequence depending on the degree of freedom (DOF) of the mechanism, which expresses the number of uncontrolled (independent) movements, which take place under forces action. The basic components of a mechanical/mechatronic system can be structured in the following way: components that bring movement → mobile bodies; components that eliminate motion → connections between bodies (when they are modeled by joints or constraint equations); components that control motion → actuating elements (when they are modeled by motion restrictions). Thus, the number of degrees of freedom is given by the following equation (Gruebler's count) [16]:

$$DOF = 6n – \Sigma(r + r_m) \qquad (1)$$

where n is the number of mobile bodies, Σr is the sum of geometric restrictions (joints and/or composed restrictions), and Σr_m is the sum of motion restrictions.

To better understand the above, **Figure 7** shows three modeling cases for an open-loop system formed by two bodies (the mobile part and the ground) connected by a revolute joint, with the following particularities: (a) there is no actuating element; (b) the actuating element is modeled by a motion restriction, controlling in this way the angular position of the rotating body/crank; and (c) the actuating element is modeled by a motor torque applied on the rotating body. For the three cases, the following numbers of degrees of freedom are corresponding: (a) DOF = 6–5 = 1; (b) DOF = 6 – (5 + 1) = 0; (c) DOF = 6 – 5 = 1. Therefore, the second model (b) has no independent motion, the angular positions of the crank being controlled (imposed) by the motion restriction regardless of the mass and the inertial properties of the body. In the first (a) and the third (c) case, respectively, the model has one uncontrolled motion (the rotation of the crank), which is influenced by the action of the forces (mass and inertia forces in the both cases, and in addition the motor torque in the third case). Thus, the motion restrictions remove degrees of freedom, by controlling the motion, while the motor forces/torques do not remove degrees of freedom.
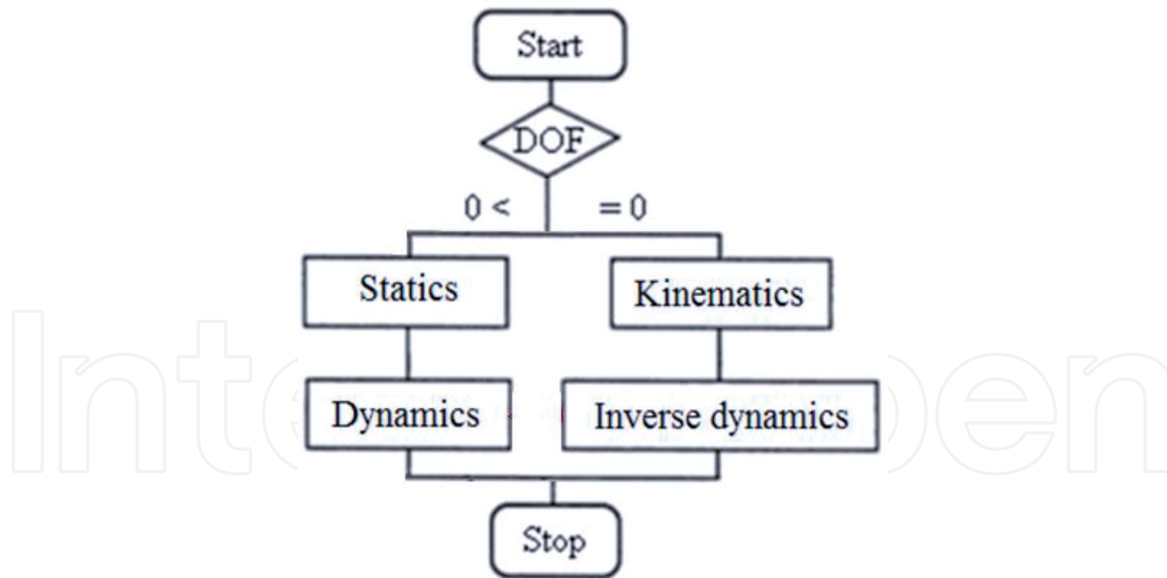
Figure 6.
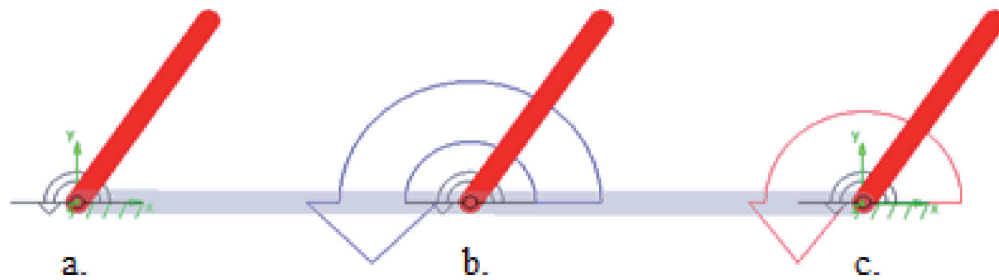*Analysis flowchart of the mechanical/mechatronic systems.*

Figure 7.
(b) Controlled vs. (a, c) uncontrolled movement.

The four types of analysis shown in **Figure 6** are defined by the following:

- Dynamics: inputs, the assembled configuration (bodies and connections), and the loads through forces and/or torques (all of them) outputs, the time histories of motion and reaction states

- Kinematics: inputs, the assembled configuration and motion restrictions (no forces/torques), and outputs, the time histories of motion

- Statics: inputs, the assembled configuration and the loads through forces and/or torques (excepting the forces that depend on velocity and acceleration, such as damping and inertia forces), and output, the equilibrium configuration

- Inverse dynamics: inputs, the same as in dynamics, but with the actuating elements as in kinematics, and outputs, the motor forces/torques

Considering the particularities of the simplifying assumptions for the modeling or refining of the basic components of the mechanical/mechatronic systems (as presented in the 3rd section of the paper) and those of the types of analysis mentioned above, **Table 3** shows the correlations between the simplifying assumptions and the analyses, which can be interpreted as validity fields for hypotheses (i.e., analyzss where the use of hypotheses does not generate errors).

Components	Simplifying assumption	Analysis
Bodies	Rigid bodies	Dynamics
		Inverse dynamics
	Point masses	Statics
	Massless bodies	Kinematics
	Composed restrictions	
Connections between bodies	Rigid contacts	Dynamics
		Inverse dynamics
	Joints/constraint equations	Kinematics
		Statics
		Dynamics
		Inverse dynamics
Actuating elements	Motion restrictions	Kinematics
		Inverse dynamics
	Motor forces/torques	Dynamics
		Statics

Table 3.
The validity fields of the simplifying assumptions.

The analysis methodology of the mechanical/mechatronic systems by using MBS software environment (ADAMS in this case) involves three stages: pre-processing (system modeling), processing (model running), and post-processing (processing results). The pre-processing stage involves indicating the input data, as follows: specifying information regarding the calculations to be performed, such as the type of analysis to be carried out, the units of measurement, the type of coordinate system (e.g., Cartesian), the gravitational acceleration vector, and the analysis time interval and modeling the components of the mechanical/mechatronic system (bodies, connections between bodies, actuating elements, and other force generating elements, such as springs or dampers, by case). The processing stage is performed automatically by the program, and consists from generating and solving the algebraic and differential equations that mathematically describe the system. The post-processing stage consists of processing the analysis results, by drawing variation diagrams/charts, generating tables with numerical values, and creating graphical animations, all of which providing an overview of how the mechanical/mechatronic system behaves.

5. Case study

Based on the above, a case study corresponding to a high complexity system, namely, a suspension system for motor vehicles, is presented below. The virtual model contains the front and rear wheel suspension subsystems, as well as the actuating subsystem. The prototype is used for simulating the passing over bumps dynamic regime under laboratory conditions, through the use of a virtual testing bench (**Figure 8**). The approach is a mechatronic one, in the sense that the actuating subsystem, containing four linear actuators that sustain/support and move the wheels, is controlled in such a way as to ensure the desired running path profile by the vertical displacements that apply to the wheels. The virtual model

of the mechanical device (including the two suspension subsystems, the car body, and the testing bench) was developed by using the MBS software environment ADAMS. The 3D solid model was developed with the help of the CAD software environment CATIA, the transfer to ADAMS being performed as described in the second section of the chapter.

The front wheels suspension subsystem (**Figure 9**) contains two independent Short-Long Arm (SLA) mechanisms, also called double-wishbone. The lower (long) and upper (short) arms of the mechanism are double-hinged to the car body by using bushings (compliant joints), while the other ends (outward) of the arms are connected to the wheel carriers by spherical joints. The same type of joints was used for the connections of the steering rods to the adjacent parts (wheel carriers and car body). The rear wheels suspension subsystem (**Figure 10**) ensures the guiding of the whole axle in the relative movement to the car body by a so-called 4S
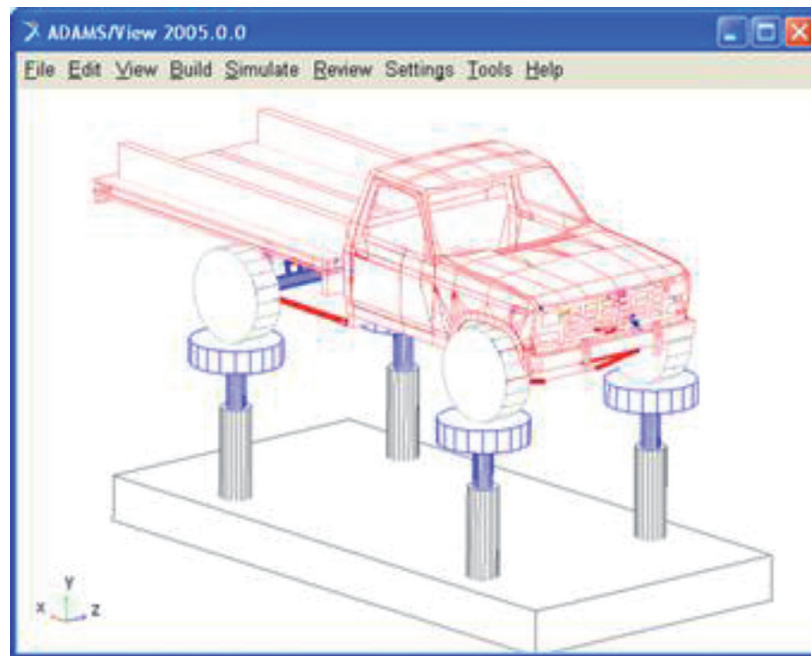
Figure 8.
The MBS virtual model of the vehicle.
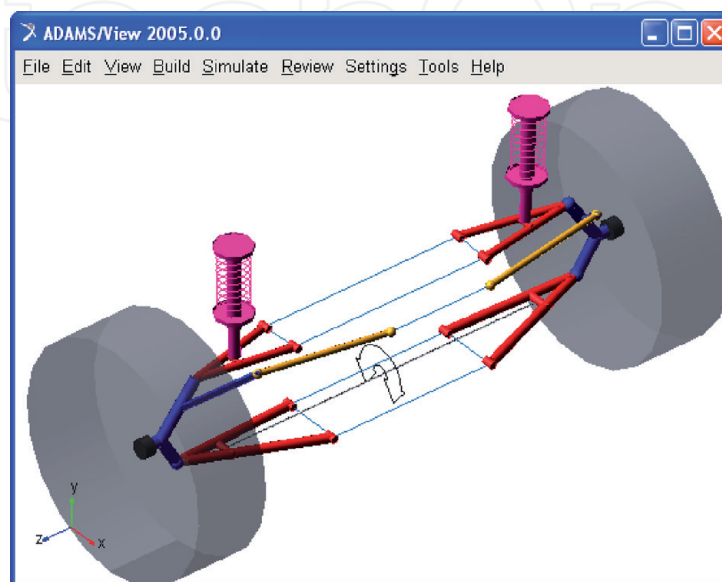
Figure 9.
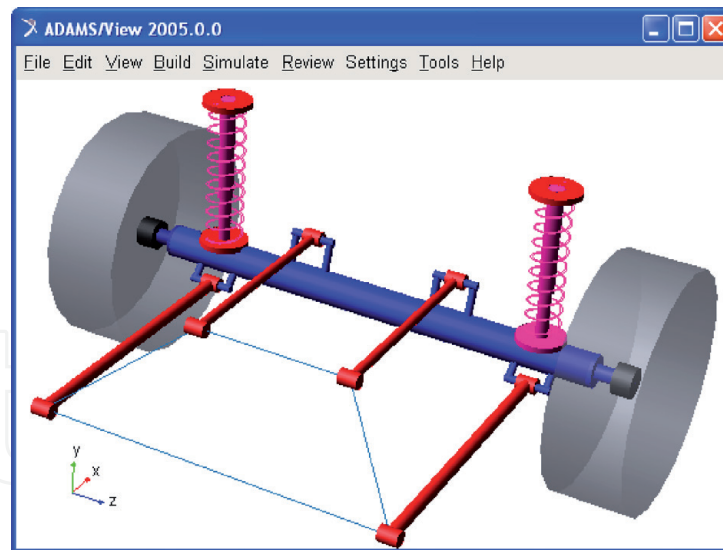*The front wheels suspension subsystem.*

Figure 10.
The rear wheels (axle) suspension subsystem.

suspension mechanism, with four longitudinal arms that are connected to the adjacent bodies by bushings. In the case of the front suspension, the spring and damper groups are mounted between upper arms and car body, while for the rear suspension, these elastic and damping elements are arranged between axle and car body. For the both suspension systems, the bumpers limiting the run (extension, and respectively compression), which are nonstationary elastic elements, are disposed inside the dampers, thus limiting the relative displacement between the two parts of the damper (cylinder and piston).

The connections between the wheels and the upper platters of the actuators were modeled by contact forces between the corresponding geometries, which allow considering the stiffness and damping properties of the tires, as well as the friction between the bodies. As mentioned, the vertical displacement of the actuator plates is controlled so as to simulate the passing of the vehicle over various types of obstacles (road irregularities), the movement being transmitted to the wheels and then, through the suspension mechanisms, to the car body. The control system for the actuating elements (**Figure 11**) was designed with the help of the DFC software solution engineering analysis systems (EASY5). In this model, the MBS mechanical device is referred by the ADAMS Mechanism block. It should be mentioned that ADAMS and EASY5 software solutions are produced—marketed by MSC Software Corp., so the compatibility between them (in terms of facilities for data transfer for the purpose of co-simulation) is very good.

The modeling of the actuation and control system was carried out in mechatronic concept, by integrating the mechanical model (**Figures 8–10**) and the control system model (**Figure 11**) at the virtual prototype level. Thus, the two models (MBS and DFC) are being tested—verified simultaneously, minimizing the risk that the control law not to be followed by the mechanical model. In this case, the mechanical and control models are connected and communicate one with other through the use of ADAMS/Controls. The communication scheme between the MBS model and the control system is shown in **Figure 12**. The inputs into the mechanical model (outputs from the control system) are the motor forces developed by the four linear actuators, while the outputs from ADAMS (inputs into EASY5) are the vertical positions of the wheel actuator plates.

The input and output plants were defined by using a set of ADAMS state variables. The input state variables (the motor forces) are defined in ADAMS by null values, going to receive their values from the control application. The input variable
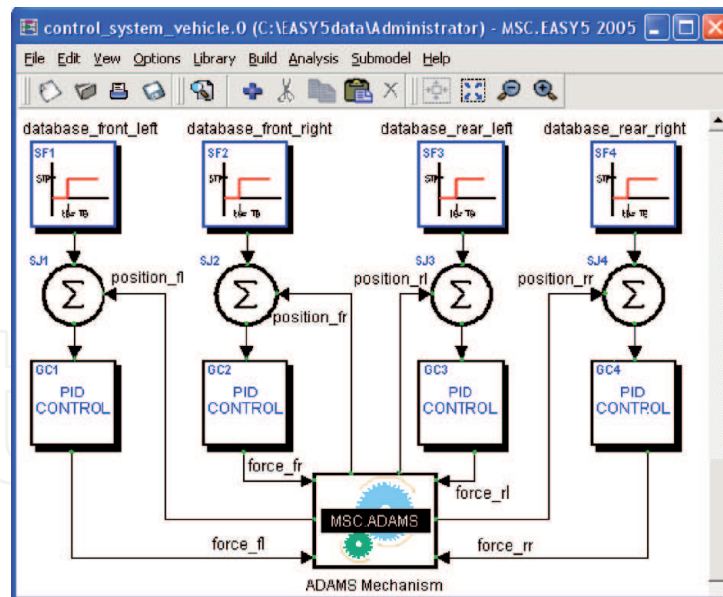
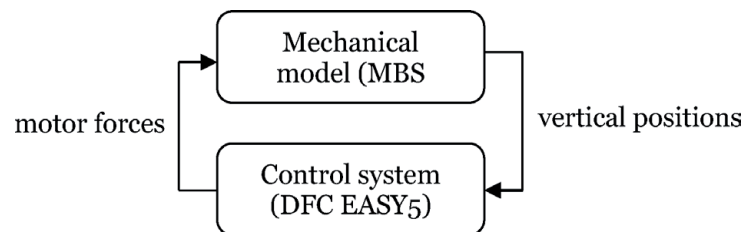**Figure 11.**
The DFC model of the control system.

Figure 12.
The input and output plants.

is called by using the predefined function VARVAL (variable), which returns the value of the variable. For the output state variables, the time functions return the linear displacements along the vertical axis (Y). The output variables are modeled by using the predefined function DY (To Marker, From Marker), where the markers represent coordinate systems belonging to the adjacent parts (actuator cylinder and piston respectively), placed in the translational joint between the two parts of the actuator. The input variables are reported by plant input, PIN1–4, and the output variables by plant output, POU1–4. Information related to the input and output plants are saved in a specific file for EASY5 (*.inf); at the same time, a command file (*.cmd) and a data file (*.adm) are generated for the subsequent co-simulation. In the ADAMS Mechanism interface block, the execution mode is then defined; in this case co-simulation, specifying also the interval with which ADAMS/Controls, writes the results to files and adapts the animation and the communication range between ADAMS and EASY5 [17].

As mentioned, the vertical positions of the wheel actuator plates are imposed to simulate the passing of the wheels over bumps (road irregularities). For the study presented in this work, it was considered the road profile shown in **Figure 13**, which includes four speed bumps with a height of 20 mm. The delays between the excitations of the wheels (front-rear and left-right, respectively) correspond to a vehicle speed of 20 km/h (the vehicle has the wheelbase, i.e., the distance between the front and rear axles, of 2.4 m). In the DFC (control system) model shown in **Figure 11**, the imposed movement laws were defined by using the step function generator blocks SF1–SF4. This type of input data block is defined by the time at initiation of step input (To_SF) and the step input value (STP_SF). The step is triggered when

time equals T0_SF and steps to a value of STP_SF, in accordance with the following conditional function:

$$\text{IF(TIME} < \text{T0\_SF) THEN S\_Out\_SF} = 0 \text{ ELSE S\_Out\_SF} = \text{STP\_SF,} \qquad (2)$$

where S_Out_SF is the step signal output, which is compared (by using a summing junction block) with the current position provided by the MBS model.

For each of the four linear actuators, PID controller was used as a control element. This controller corrects the difference between the imposed (desired) and current (measured) values of a specific parameter by computing and applying a compensatory measure that adapts the system properly [18, 19]. The optimal design of the controller can be achieved both by methods specific to the control theory (e.g., root locus, frequency methods), as well as by optimal parametric design techniques [20–22]. For this work, the optimization was performed by using the scripting capabilities integrated in EASY5 Matrix Algebra Tool (MAT), the control system model being managed as an EMX function. The optimization procedure is similar with that presented in [23]. The optimization goal is to minimize the difference between the imposed and current values of the actuator plate vertical position, while the design variables are the proportional (P), integral, (I) and derivative (D) factors of the controller.

In the conditions specified above, the time history variations of the vertical positions of the front and rear actuator plates are shown in **Figure 14**. The mechanical powers developed by the four linear actuators for generating the predefined
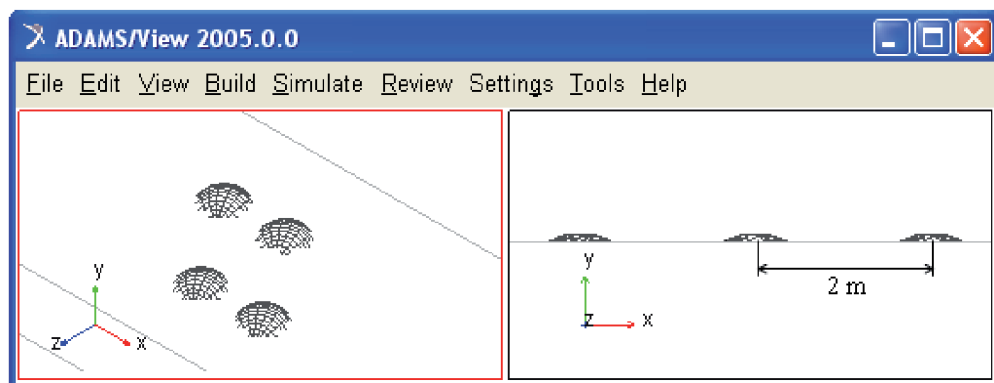

Figure 13.
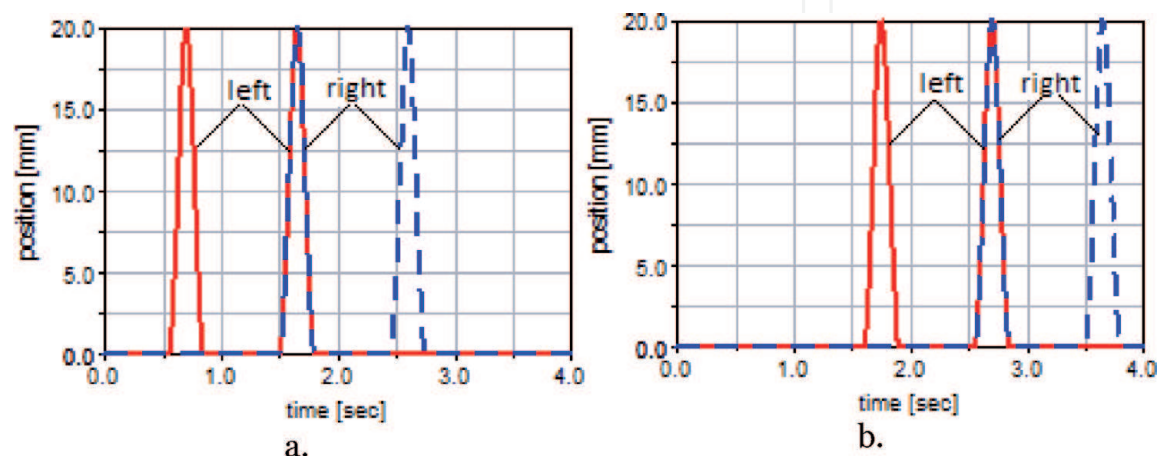*The road profile simulated by the virtual test bench.*

Figure 14.
The vertical displacements of the front (a) and rear (b) actuator plates.

movement laws, which were determined by multiplying the motor forces by the linear velocities of the actuator plates, are the ones shown in **Figure 15**. Some results that describe the dynamic behavior of the vehicle are presented in
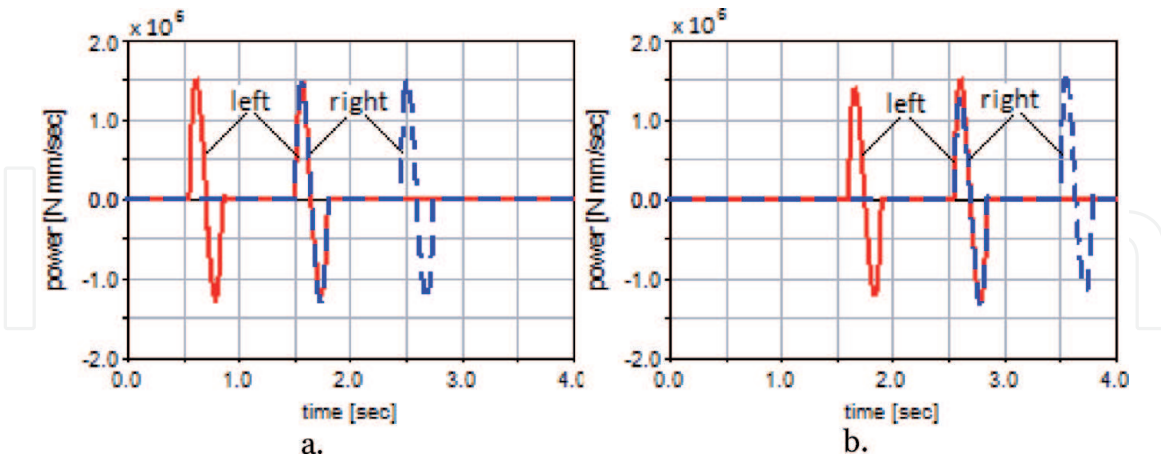

Figure 15.
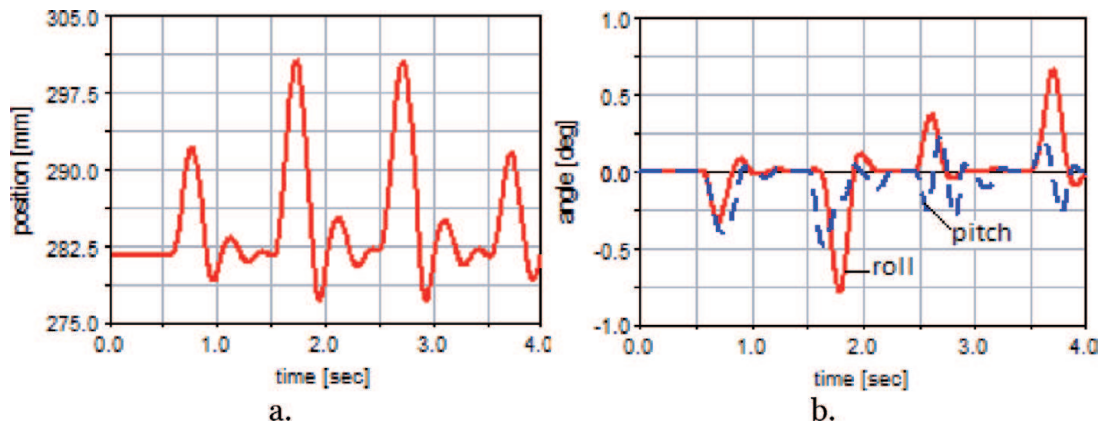*The power developed by the front (a) and rear (b) linear actuators.*

Figure 16.
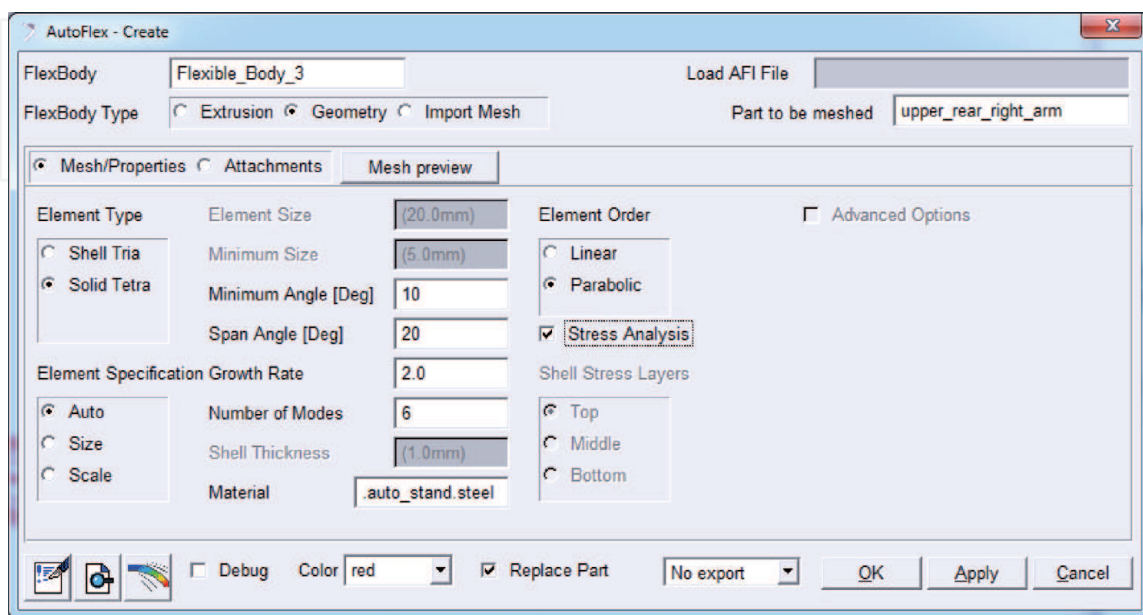*The main linear (a) and angular (b) oscillations of the car body.*

Figure 17.
Example of conversion from solid body to flexible body.

Figure 16, namely, the vertical displacement of the car body (a), which is measured in its center of mass and the roll and pitch oscillations/angles (b).

Further, the guiding arms of the front and rear suspension mechanisms, which were initially modeled as rigid bodies, were discretized into finite elements, for studying their deformability and stress state. The conversion from solid to flexible was achieved by using ADAMS/AutoFlex. For example, **Figure 17** shows the conversion window for the rear upper right arm, while in **Figure 18** the finite element model of this body, along with its first three vibration modes. It should be mentioned that ADAMS/AutoFlex allows viewing 18 vibration modes per flexible body, each mode of vibration being characterized by a modal frequency and a mode shape [24]. A dynamic simulation graphical frame for the compliant model (with flexible bodies), focused on the rear axle guiding mechanism, is shown in **Figure 19**, revealing the stress state of the guiding arms.

Many other results can be extracted from the simulations in virtual environment, for all the objects—components of the virtual model (e.g., bodies, connections between bodies, actuating elements, elastic and damping elements) and for any type of parameter (e.g., motion, force, energy), including results that cannot be measured experimentally for various reasons (such as lack of adequate sensors, hard to reach areas, high temperatures in the measuring area, and others). At the same time, by studying the influence of the various parameters that define the model (such as the global coordinates of the joint locations or the elastic and damping coefficients of the spring and damper assemblies) on the vehicle behavior, its kinematic and dynamic optimization can be simplified, by selecting the parameters which significantly influences the comfort, stability, or maneuverability of the vehicle.
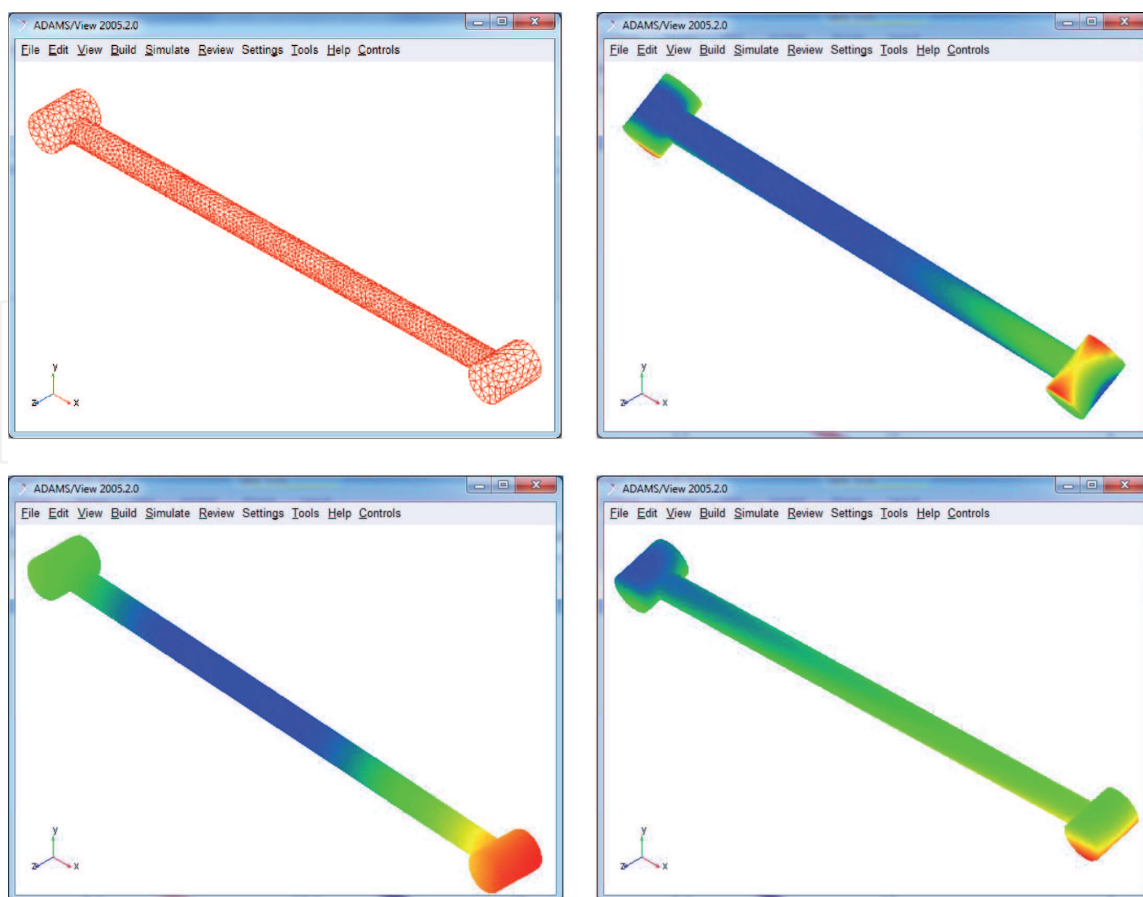

Figure 18.
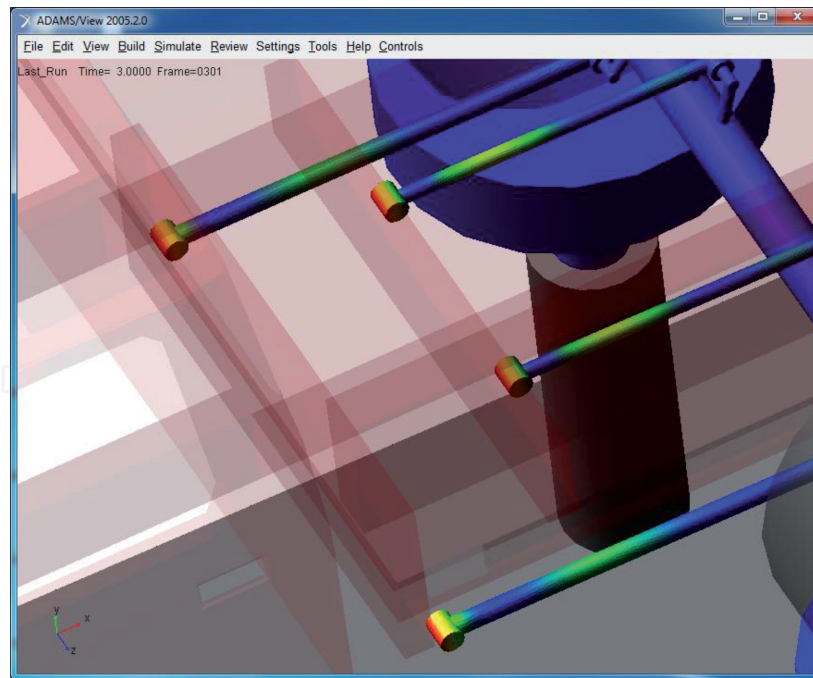*The FEA model of the rear upper right arm.*

Figure 19.
Dynamic simulation graphical frame.

6. Conclusions

The use of virtual prototyping software platforms in the analysis and optimization of the mechanical and mechatronic systems offers important benefits, which focus on reducing the costs, as well as the design and development time while increasing the quality (operational performances of the products). The virtual prototypes are not made from real materials (such as steel, aluminum or wood) that are generally expensive, but from bits, with which any type of material can be simulated. Other significant cost reductions result from the fact that virtual prototyping does not involve destroying prototypes during testing (e.g., in the real car crash tests), the virtual prototype being restored to its original state by a simple mouse click. Multiple design variations (in various parameter combinations) can be explored early, without going through expensive (and often superficial) physical prototyping cycles. The virtual prototyping technique allows the replication on computer of both the product itself and the specific operating (working) environment. Among the critical success factors regarding the successful implementation of the virtual prototyping platforms, we can point to well-defined process, system-level orientation, efficient setting of the goal, rapid dynamics of the simulation, and high-quality infrastructure (hardware and software).

Author details

Cătălin Alexandru
Transilvania University of Braşov, Romania

*Address all correspondence to: calex@unitbv.ro

IntechOpen

References

[1] Alexandru C, editor. Modeling and Simulation in Engineering. Rijeka: InTech Open; 2012. p. 298. DOI: 10.5772/1415

[2] Ryan R. Functional Virtual Prototyping - Realization of the Digital Car. Ann Arbor: Mechanical Dynamics Inc.; 2001

[3] Bernard A. Virtual engineering: Methods and tools. Journal of Engineering Manufacture. 2005;**219**(5): 413-421

[4] Haug EJ, Choi KK, Kuhl JG, Vargo JD. Virtual prototyping simulation for design of mechanical systems. Journal of Mechanical Design. 1995;**117**(63):63-70

[5] Mejia-Gutierrez R, Carvajal-Arango R. Design verification through virtual prototyping techniques based on systems engineering. Research in Engineering Design. 2017;**28**(4):477-494

[6] Getting Started Using ADAMS/View. Newport Beach: MSC Software Corporation; 2005

[7] Ambrósio JA, Gonçalves JP. Complex flexible multi-body systems with application to vehicle dynamics. Multibody System Dynamics. 2001;**6**:163-182

[8] Da Silva MM, Costa NA. Handling analysis of a light commercial vehicle considering the frame flexibility. International Review of Mechanical Engineering (IREME). 2007;**1**(4):334-339

[9] Alexandru C, Pozna C. The virtual prototyping of the windshield wiper systems in mechatronic concept. In: Proceedings of the 11-th European Automotive Congress (EAEC '07). Budapest: Scientific Society of Mechanical Engineers; 2007. pp. 58-69

[10] Enescu M, Alexandru C. Virtual prototyping of a spraying robotic system. Environmental Engineering and Management Journal. 2011;**10**(8):1197-1205

[11] Fischer E. Standard multi-body system software in the vehicle development process. Journal of Multi-body Dynamics. 2007;**221**(1):13-20

[12] Garcia de Jalón J, Bayo E. Kinematic and Dynamic Simulation of Multi-Body Systems. 1st ed. New York: Springer - Verlag; 1994. p. 440

[13] Haug EJ. Computer Aided Kinematics and Dynamics of Mechanical Systems. 1st ed. Needham Heights, Massachusetts: Allyn & Bacon; 1989. p. 500

[14] Schiehlen WO. Multi-body systems dynamics: Roots & perspectives. Multibody System Dynamics. 1997;**1**(2):149-188

[15] Shabana A. Dynamics of Multi-Body Systems. 4th ed. New York: Cambridge University Press; 2014. p. 393. DOI: 10.1017/CBO9781107337213

[16] Orlandea N. Design of parallel kinematic systems using the planar enveloping algorithm and ADAMS program. Journal of Multi-body Dynamics. 2004;**218**(4):211-221

[17] EASY5 Guide Help. Newport Beach: MSC Software Corporation; 2005

[18] Dorf RC, Bishop RH. Modern Control Systems. Boston: Pearson; 2016. p. 1032

[19] Haque HH, Hassan HM, Hossain SM. Comparison of control system using PLC & PID. In: Proceedings of the ASEE 2014 Conference. Bridgeport: American

Society for Engineering Education;
2014. p. 99(1-6)

[20] Alexandru C. Optimal design of the
controller for a photovoltaic tracking
system using parametric techniques.
Annals of the Oradea University:
Fascicle Management and Technological
Engineering. 2010;**IX**(1):2.1-2.9

[21] Alexandru C. Optimal design of the
mechanical systems using parametric
technique & MBS (multi-body systems)
software. Advanced Materials Research.
2012;**463-464**:1129-1132

[22] Alexandru C. Optimal design of
the dual-axis tracking system used
for a PV string platform. Journal of
Renewable and Sustainable Energy.
2019;**11**(4):043501(1-14)

[23] Alexandru C, Alexandru P. Control
strategy for an active suspension system.
World Academy of Science, Engineering
and Technology. 2011;**79**:126-131

[24] Blevins RD. Formulas for Natural
Frequency and Mode Shape. Riverside,
Connetticut: Krieger Pub; 2001. p. 506