Efficiency and Precise Interaction for Multibody Simulations in Augmented Reality

Lorenzo Mariti and Pier Paolo Valentini

Abstract In this chapter, an enhanced methodology for interactive, accurate, fast and robust multibody simulations using Augmented Reality is presented and discussed. This methodology is based on the integration of a mechanical tracker and a dedicated impulse based solver. The use of the mechanical tracker for the interaction between the user and the simulation allows to separate the processing of the data coming from the position tracking from those coming from the image collimation processing. By this way simulation results and visualization remain separated and the precision is enhanced. The use of a dedicated sequential impulse solver allows a quick and stable simulation also for a large number of bodies and overabundant constraints. The final result of this work is a software tool able to manage real time dynamic simulations and update the augmented scene accordingly. The robustness and the reliability of the system will be checked over two test cases: a ten pendula dynamic system and of a cross-lift mechanism simulation.

1 Introduction

During the last years, many investigations focused on the increasing trend of using Augmented Reality (AR) [1] to support a variety of engineering activities and to develop interactive tools in design. Augmented Reality has been used for supporting geometrical modeling [2], reverse engineering [3], assembly simulation [4–7], analysis [8].

The augmented reality deals with the combination of real world images and computer generated data. Most AR research is concerned with the use of live video imagery which is digitally processed and augmented by the addition of computer generated graphics. The purpose of the augmented environment is to extend the visual perception of the world, being supported by additional information and virtual objects. One of the most important feature of AR is the possibility to embed an high level of user's interaction with the augmented scene [9].

L. Mariti · P.P. Valentini (✉)
Department of Industrial Engineering, University of Rome "Tor Vergata", via del Politecnico, 1, 00133, Rome, Italy
e-mail: valentini@ing.uniroma2.it

J.-C. Samin, P. Fisette (eds.), *Multibody Dynamics*,
Computational Methods in Applied Sciences 28,
DOI 10.1007/978-94-007-5404-1_8,

Some recent contributions showed an increasing interest in developing environments for simulating and reviewing physical simulations based on Augmented Reality [10–14]. They focused on the interactivity between the user and the augmented environment. By this way, the user is not a mere spectator of the contents of the augmented scene, but influences them. Interaction in dynamic simulations can concern both the definition of boundary conditions and initial parameters and the real time control of the process.

In a recent paper [15], P.P. Valentini and E. Pezzuti developed a methodology for implementing, solving and reviewing multibody simulation using augmented reality. According to their results, augmented reality facilitates the interaction between the user and the simulated system and allows a more appealing visualization of simulation results. For this reason, this approach has revealed to be suitable for didactical applications and teaching purposes as well.

On the other hand, the development of multibody simulations in augmented reality requires very fast solver in order to produce a smooth animation and an effective illusion. Valentini and Pezzuti proposed to use an optical marker to track the position of the user and interact with the objects in the scene. Moreover, they developed some simple examples to introduce the methodology.

Starting from the discussed background, this chapter aims to discuss two important enhancements of the work in [15] in order to improve the accuracy of user's interaction and to allow a robust simulation of large multibody systems. The accuracy in tracking the user is important to perform precise simulation that are required in many engineering applications. The use of optical marker is simple but in this case position tracking error highly depends on the resolution of the camera sensor and on the distance between the camera and the marker. Using standard USB cameras, this error can be some millimeters and can be unacceptable for precise applications, and unwanted flickering may occur due to the tracking algorithm limitations [16]. In order to improve the accuracy in tracking, in the present study we have included the use of a mechanical instrumented arm which is able to achieve a precision of about 0.2 mm in a working space of about 1.2 m of diameter. This enhancement is also important to separate the processing of the data coming from the position tracking from those coming from the image processing (for perspective collimation and visualization issues). By this way, higher precision in the analysis results can be achieved and only the graphical visualization is affected by optical imprecision.

The second enhancement which has been proposed and tested, is about the use of a dedicated solver for managing the integration of the equations of motion. The implemented solver makes use of the sequential impulse strategy [17] which allows a quick and stable simulation also for a large number of bodies, in presence of overabundant and unilateral constraints. According to some authors and applications (i.e. [18, 19]), this approach leads to a very fast and stable solution, but quite less accurate than global solution methods.

The chapter is organized as follows. First of all, a brief introduction about the state of the art of virtual engineering in augmented reality is presented, focusing on the aspects related to multibody simulations. Then, a description of hardware and software implementations is discussed, including the explanation of the iterative impulse solver and the details about the strategy for implementing interaction

between the user and the scene. In the second part, two examples are presented and discussed.

2 Multibody Simulations in Augmented Reality Environments

The first and basic implementation of multibody simulations in augmented reality is about the possibility to project on a real scenario the results coming from a pre-computed simulation. It concerns the rendering on the augmented scene of all the objects involved in the simulation whose position is updated according to the results of the simulation.

This implementation is similar to that of the common post-processing software for visualizing graphics results. The only difference is in the introduction of the simulated system in a real context. The advantage is to perceive the interaction with the real world and check working spaces, possible interferences, etc.

Although it can be useful, this approach does not unveil all the potential of AR [15]. A more powerful way to enhance multibody simulation is to introduce an high level of interactivity. It means that the user does not only watch the augmented scene, but interacts with it. In this case, the solution of the equations of motion has to be computed synchronously to the animation in order to populate the scene with quickly updated information.

With this type of interaction, the user is active in the scene and can change the augmented contents by picking, pushing and moving objects and controlling the provided information and the environment behaves according to realistic physics laws. The interaction is carried out with advanced input/output devices involving different sensorial channels (sight, hear, touch, etc.) in an integrated way [20]. In particular, in order to interacts with digital information through the physical environment, the environment has to be provided of the so called Tangible User Interfaces (TUIs) [21–23].

In the most simple implementations, the patterned markers used for collimating real and virtual contents are used also as TUIs. In advanced implementation visual interaction is achieved by dedicated interfaces (mechanical, magnetic, optical, etc.). By this way, the image processing for the computation of the camera-world perspective transformation can be separated from the acquisition and processing of the user intent. Thanks to this split computation, it is possible to achieve a very precise interaction and simulation and a less precise (and more efficient) visual collimation. This means that the results of the simulations can be accurate and suitable for technical and engineering purposes. On the other hand, the small imprecisions in the optical collimation are limited to graphics display.

Starting from these considerations, the generic integration algorithm between multibody simulations and augmented reality presented in [15] can be modified separating the contributions of interaction and collimation. For this reason, an high-interactive generic multibody simulation in augmented reality can be implemented following five main steps:

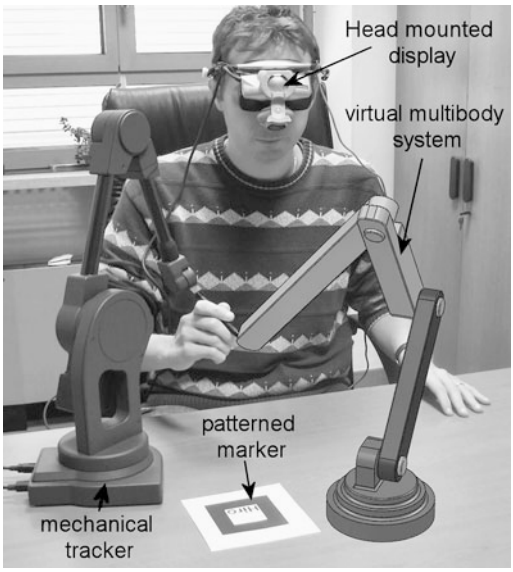

Fig. 1 Augmented reality hardware setup

1. Before the simulation starts, the geometries and topological properties (joints and connections) have to be defined (as for any multibody system);
2. The real scene has to contain information for collimating the real world to the virtual objects;
3. The real environment has to contain one or more TUIs for the acquisition of the user's intent of interacting with the scene;
4. During each frame acquisition, the user's intent has to be interpreted; the multibody equations have to be built and solved synchronously in order to compute the correct position of all the virtual bodies in the scene;
5. For each frame acquisition, virtual objects have to be rendered on the scene, after the numerical integration, in the correct position and attitude.

3 Implementation of the Augmented Reality Environment

3.1 Hardware Setup

For the specific purpose of this investigation, the implemented AR system (depicted in Fig. 1) includes an input video device Microsoft LifeCam VX6000 USB 2.0 camera, an Head Mounted Display (Emagin Z800) equipped with OLed displays, a Revware Microscribe GX2 mechanical tracker and a personal computer.

The Revware Microscribe is an instrumented arm (digitizer) which can be grabbed and driven by the user and possesses five degrees of freedom. It is able to acquire the real-time position of its tip stylus. The operating space is a sphere of about 1.2 m of diameter and the precision of tracking is up to 0.2 mm.

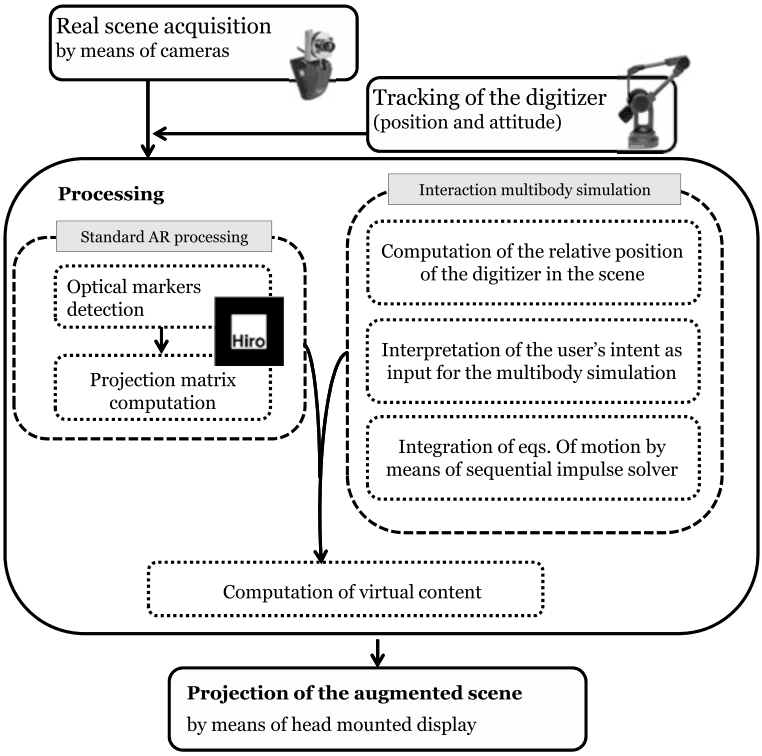

Fig. 2 Multibody simulation in augmented reality procedural scheme

3.2 Implementation and Interaction Strategy

In order to implement a AR environment suitable for multibody dynamics interactive simulation, we have chosen the following strategy (see Fig. 2). Before the simulation starts, the geometries and topological properties (joints and connections) have to be defined (as for any multibody simulation). Then, the real scene has to contain information for collimating the real world to the virtual objects. Usually this operation is performed by a patterned planar marker which is recognized by the processing units. For the scope it is necessary to perform the computation of the camera point of view and the corresponding perspective effect.

During each frame acquisition, the position of the user in the scene has to be acquired as well. This task can be performed using the mechanical tracker and computing the position and attitude of its end effector. This information can be interpreted and transferred as a spatial input for the simulation (user's intent to interact with the scene). Then, the multibody equations have to be solved in order to compute the correct position of all the virtual bodies in the scene, taking into account the user's intent.

After this computation, all the virtual objects have to be rendered on the scene in the correct position and attitude.

All the supporting software has been implemented using C++ programming language and Microsoft Visual Studio 2003 developing suite. Routines for image processing have been developed using the open source libraries named ARToolkit[1] which has been successfully used in many previous investigations. The libraries comprise a set of numerical procedures which are able to detect and recognize planar patterned markers in a video stream in real time. Using correlation techniques, the routines are also able to compute relative position and attitude between markers and camera with good precision for visual purposes. This computation is necessary for an accurate perspective collimation between virtual entities and real scene. The details about specific implementation and about the contents of the library can be found on the Internet site of the developers.[2]

The Microscribe GX2 has been integrated using the Microscribe SDK library that allows the real time access to position and attitude of each link of the instrumented arm. This library interprets the output coming from the rotation sensors of the five revolute joints of the mechanical arm and computes the position of the stylus tip by solving an open kinematic chain problem.

For managing complex geometries the OpenVrml library[3] has been included. All rendering tasks about virtual objects in the augmented scene have been performed using OpenGL library.

Details about the procedures for deducing and solving the equations of motion of the system under investigation have been provided in the next sections.

All these pieces of software have been integrated into a single simulation environment.

3.3 Collimation Procedure

The first step in the integration of the tracker in the augmented scene is the collimation between the information acquired by the instrumented device and that of the digital camera (see Fig. 3). The video stream acquired by the digital camera is elaborated by an image processing routine. It is able to recognize a patterned marker in the scene and to compute the corresponding transformation matrix between the camera and the real word. This matrix is used to project all the virtual contents in the augmented scene in the correct position and perspective.

The information acquired by the digitizer is concerned with the position and attitude of the end effector with respect to the reference frame fixed to the device itself.

In order to ensure the collimation between the data stream coming from the camera and that from the tracker, it is important to compute the relative transformation

[1]The ARToolkit libraries can be freely downloaded from the Internet site http://sourceforge.net/project/showfiles.php?group_id=116280.

[2]http://www.hitl.washington.edu/artoolkit/.

[3]The library can be freely downloaded from the Internet site http://openvrml.org/.

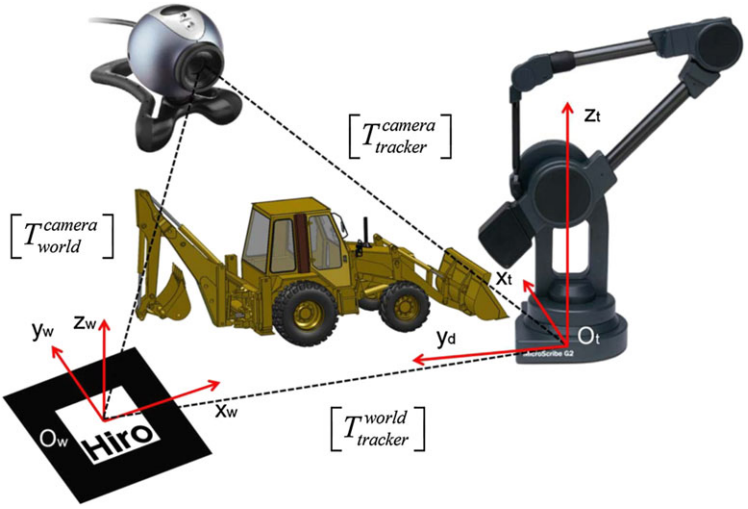

Fig. 3 Collimation between optical marker and mechanical tracking system

matrix between the tracker and the world (described by the marker). This calibration has to be performed only at the beginning of the application and it has to be repeated only if the relative position between the world marker and the digitizer changes.

The calibration procedure can be performed by picking with the tracker stylus a set not-aligned points (four no-coplanar points at least) at known positions with respect to the relative frame associated to the marker.

For expressing the coordinate transformation between points, it is useful to deal with homogeneous transformation matrices which include information on both rotation and translation parameters. A generic homogeneous transformation matrix can be expressed in the form:

$$[T] = \begin{bmatrix} [\text{Orientation}]_{3\times 3} & [\text{Position}]_{3\times 1} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \tag{1}$$

In the same way, a generic point can be expressed with the following coordinate vector:

$$\{P\} = \begin{Bmatrix} x & y & z & 1 \end{Bmatrix}^T \tag{2}$$

The coordinate transformation of a generic point P from the local coordinate system fixed to the digitizer to the world coordinate system attached to the marker can be written as:

$$\{P\}_{world} = \left[T^{digitizer}_{world}\right]\{P\}_{digitizer} \tag{3}$$

where:

$\{P\}_{world}$ is the vector containing the coordinate of the point P expressed in the world reference frame;

$\{P\}_{digitizer}$ is the vector containing the coordinate of the point P expressed in the local (tracker) reference frame.

Considering a collection of points $P_1\ P_2\ \ldots\ P_n$, we can build two matrices as:

$$[P]_{world} = \left[\{P_1\}_{world} \quad \{P_2\}_{world} \quad \ldots \quad \{P_n\}_{world}\right] \tag{4}$$

$$[P]_{digitizer} = \left[\{P_1\}_{digitizer} \quad \{P_2\}_{digitizer} \quad \ldots \quad \{P_n\}_{digitizer}\right] \tag{5}$$

In order to compute the matrix $[T^{digitizer}_{word}]$ we have to solve the following system of equations:

$$[P]_{world} = \left[T^{digitizer}_{world}\right][P]_{digitizer} \tag{6}$$

for the unknown elements of the matrix $[T^{digitizer}_{word}]$.

An homogeneous transformation matrix is defined by 6 independent parameters (three for the description of the rotation and three for the translation). For this reason, the system (6) has more equations than unknowns and the solution can be computed as:

$$\left[T^{digitizer}_{world}\right] = [P]^{-1^+}_{world}[P]_{digitizer} \tag{7}$$

where the $[P]^{-1^+}_{world}$ is the pseudo-inverse matrix of the $[P]_{world}$ matrix.

Due to numerical approximation or errors in acquisition, the orientation block of the computed matrix $[T^{digitizer}_{word}]$ can result not exactly orthogonal. Since it represents a rigid spatial rotation, it is important to correct this imprecision. For this purpose, we can operate a QR decomposition of this orientation block:

$$\left[\text{Orientation}^{digitizer}_{word}\right]_{3\times3} = [R_1]_{3\times3}[U_1]_{3\times3} \tag{8}$$

where (due to the QR algorithm):

$[R_1]$ is an orthogonal matrix representing the corrected rotation;

$[U_1]$ is a matrix whose upper band contains the errors of approximation and the lower band has only zero elements. In case of a pure rotation (orientation block without errors) $[U_1] = [I]$.

Finally, in order to compute the transformation matrix between the digitizer and the camera $[T^{digitizer}_{camera}]$, useful to collimate the acquired points to the visualized ones, a matrix multiplication has to be performed:

$$\left[T^{digitizer}_{camera}\right] = \left[T^{digitizer}_{word}\right]\left[T^{word}_{camera}\right] \tag{9}$$

Figure 4 shows some snapshots acquired during a calibration procedure. The reference points are picked using a reference cube of known dimensions (80 mm × 80 mm × 80 mm).

3.4 User's Interaction

Once the position and the attitude of the tracker are correctly recorded, we have to define the methods to interact with the simulation.

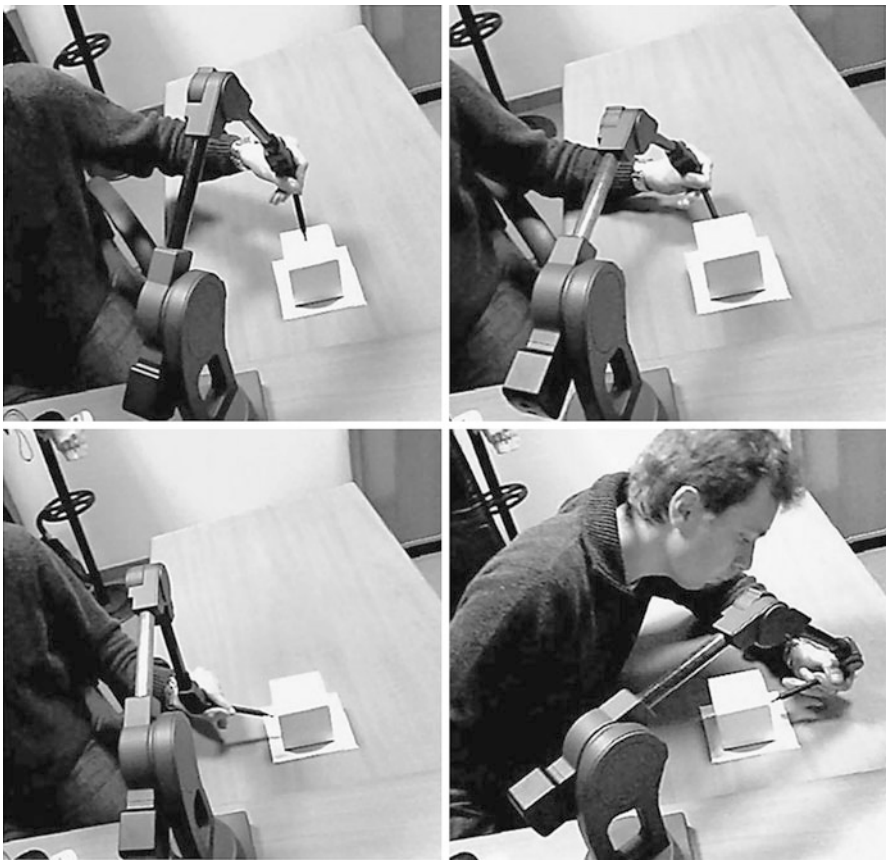

Fig. 4 Four snapshots taken during calibration procedure

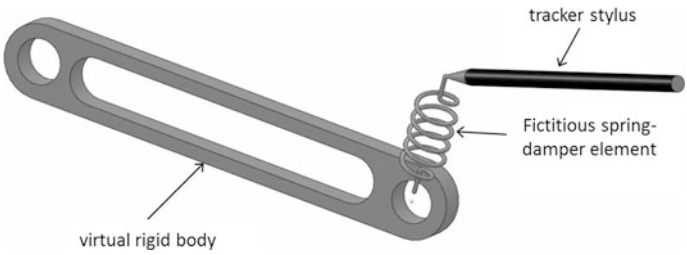

Fig. 5 The implementation of a fictitious spring for managing the user's interaction to the simulated bodies

A possible solution (see Fig. 5) is the use of a fictitious spring-damper element connected between the digitizer stylus tip and a point belonging to one of the rigid bodies in the simulation [24]. By this way, if the tip is moved in the scene, it affects the dynamics of that point that moves accordingly.

Mathematically, the use of a fictitious spring between the tracker stylus tip and a spline control point adds a term to the external force vector $\{F\}_e$ in equations of motion:

$$\{F\}_e = \{F\}_e - k_{f\_spring}\left\{d^{tracker\_tip}_{virtual\_body\_point}\right\} - c_{f\_spring}\left\{\dot{d}^{tracker\_tip}_{virtual\_body\_point}\right\} \quad (10)$$

where:

$\{d^{tracker\_tip}_{virtual\_body\_point}\}$ is the distance between the connected control point and the digitizer tip;
$\{\dot{d}^{tracker\_tip}_{virtual\_body\_point}\}$ is the derivative of $\{d^{tracker_tip}_{virtual_body_point}\}$with respect to time;
k_{f_spring} is the stiffness coefficient of the fictitious spring;
c_{f_spring} is the damping coefficient of the fictitious spring.

In order to prevent the dynamics of the system to be affected by the presence of an external (and not physical) elastic component the value of the stiffness parameter has to be chosen very stiff. Moreover, the use of a damping coefficient improves the stability of the system and prevents jittering in simulation.

4 Implementing the Sequential Impulse Solver

Given a collection of n_{body} rigid bodies, constrained by n_{joint} kinematic joints and subjected to a set of external forces, the equation of motion can be deduced following different approaches.

One of the most used multibody dynamics formulation is based on the Lagrange equations, obtaining a differential-algebraic system:

$$\begin{cases} [M]\{\ddot{q}\} - [\psi_q]^T\{\lambda\} = \{F_e\} \\ \{\psi\} = \{0\} \end{cases} \quad (11)$$

where:

$[M]$ is the inertia matrix of the collection of rigid bodies;
$\{\psi\}$ is the vector containing all the constraint equations;
$\{q\}$, $\{\dot{q}\}$ and $\{\ddot{q}\}$ are the vectors of absolute generalized coordinates, velocities and accelerations, respectively;
$[\psi_q]$ is the Jacobian matrix of the constraint equations (differentiated with respect to the generalized coordinates);
$\{\lambda\}$ is the vector of Lagrange multipliers associated to the constraint equations;
$\{F\}_e$ is the vector containing all the elastic and external forces (including the fictitious spring contribution).

In order to reduce the complexity of the solution, the constraint equations are often differentiated two times with respect to time and Eq. (11) is rearranged as:

$$\begin{bmatrix} [M] & [\psi_q]^T \\ [\psi_q] & [0] \end{bmatrix} \begin{Bmatrix} \{\ddot{q}\} \\ \{\lambda\} \end{Bmatrix} = \begin{Bmatrix} \{F_e\} \\ \{\gamma\} \end{Bmatrix} \tag{12}$$

where

$$\{\gamma\} = -\left([\psi_q]\{\dot{q}\}\right)_q \{\dot{q}\} - 2[\psi_{qt}]\{\dot{q}\} - \{\psi_{tt}\} \tag{13}$$

and the subscripts "q" and "t" denote the differentiation with respect to the generalized coordinates and time, respectively.

Both Eq. (11) and Eq. (12) allow to solve for the unknown generalized acceleration, velocities and positions taking into account all the constraint equations simultaneously. Of course, this approach can achieve accurate results with suitable DAE solver. On the other hand, a dynamic system with a lot of constraints increases the complexity of the problem and the computational effort to solve it. For this reason, the system in (12) can be rearranged for being suitable for the sequential impulse solution strategy.

There are two main steps in the impulse-based methodology. Firstly, the equations of motion are tentatively solved considering elastic and external forces but neglecting all the kinematic constraints. This produces a solution that is only approximated because the constraint equations are not satisfied. In a second step, a sequence of impulses are applied to each body in the system in order to correct its velocity according to the limitation imposed by the constraint. This second step is iterative. It means that a series of impulse is applied to the bodies until the constraint equations are fulfilled within a specific tolerance. It is important to underline that each impulse is applied independent from the others. By this way the constraint equations are not solved globally, but iteratively.

4.1 Solving the Equations of Motion

Following the approach introduced in the previous section, the sequential impulse formulation can be split into two main steps. The first one is about the solution of the equations of motion in (11) neglecting the constraint equations and constraint forces:

$$[M]\{\ddot{q}\}_{approx} = \{F_e\} \tag{14}$$

By this way, Eq. (14) can be solved for $\{\ddot{q}\}_{approx}$ that represent the vector of approximated generalized accelerations.

The values of the corresponding approximated generalized velocities and positions can be computed by linear approximation:

$$\{\dot{q}\}_{approx} = h \cdot \{\ddot{q}\}_{approx} \tag{15}$$

$$\{q\}_{approx} = h \cdot \{\dot{q}\}_{approx} \tag{16}$$

where h is the integration time step.

In order to correct the $\{\dot{q}\}_{approx}$ and fulfill the constraint equations, a series of impulses $\{P\}_{constraint}$ has to be applied to the bodies. Each impulse is computed imposing the fulfillment of the constraint equations written in terms of generalized coordinates. As well-known from the Newton's law, the application of the impulses causes a variation of momentum:

$$[M]\left(\{\dot{q}\}_{corrected} - \{\dot{q}\}_{approx}\right) = \{P\}_{constraint} \tag{17}$$

where $\{\dot{q}\}_{corrected}$ is the vector of generalized velocities after the application of impulses $\{P\}_{constraint}$.

The corrected velocities can be computed from Eq. (17) as:

$$\{\dot{q}\}_{corrected} = \{\dot{q}\}_{approx} + [M]^{-1}\{P\}_{constraint} \tag{18}$$

Considering that the impulses are related to the constraint equations, they can be computed as

$$\{P\}_{constraint} = [\psi_q]^T\{\delta\} \tag{19}$$

where $\{\delta\}$ is the vector of Lagrange multipliers associated to the impulses.

Since the effect of the impulses is to correct the generalized velocities and fulfill the kinematic constraints, the $\{\dot{q}\}_{corrected}$ has to satisfy the constraint equations written in terms of velocities:

$$\{\psi\} = \{0\} \quad \Rightarrow \quad \frac{\mathrm{d}\{\psi\}}{\mathrm{d}t} = [\psi_q]\{\dot{q}\} + \{\psi_t\} = \{0\} \tag{20}$$

$$[\psi_q]\{\dot{q}_{corrected}\} + \{\psi_t\} = \{0\} \tag{21}$$

Inserting Eq. (19) into Eq. (18) and substituting $\{\dot{q}\}_{corrected}$ into Eq. (21) we can obtain:

$$[\psi_q]\left(\{\dot{q}_{approx}\} + [M]^{-1}[\psi_q]^T\{\delta\}\right) + \{\psi_t\} = \{0\} \tag{22}$$

Equation (22) can be solved for $\{\delta\}$obtaining:

$$\{\delta\} = \left([\psi_q][M]^{-1}[\psi_q]^T\right)^{-1}\left([\psi_q]\{\dot{q}\}_{approx} + \{\psi_t\}\right) \tag{23}$$

Then, the impulses can be computed using Eq. (19) and the corrected values of generalized velocities using Eq. (18).

Since the impulses are computed sequentially, the global fulfillment of the constraint equations cannot be directly achieved. Some iterations are required. The computation of $\{\delta\}$, $\{P\}_{constraint}$ and $\{\dot{q}\}_{corrected}$ can be repeated till a tolerance on the fulfillment of Eq. (21) is reached or for a maximum number of time. Experience [17] shows that four or five iterations are sufficient to achieve an adequate tolerance. Since the constraints are imposed at velocity level, a stabilized formulation is required to control the constraints fulfillment at the position level. Details are provided in Sect. 4.3.

4.2 Computation of Reaction Forces

When simulating multibody dynamics, one of the most interesting results for engineers is the knowledge of the reaction forces, i.e. the forces exerted by the joints.

Using the sequential impulse formulation, the reaction forces cannot be computed directly but a preliminary computation is required. The problem is that impulses are applied sequentially, it means that each joint exerts more than one impulse during each time step and the various impulses have to be recollected.

In order to deduce a methodology for evaluating the reaction forces of the joints we have to introduce the concept of the accumulated impulse. It can be defined as the resultant impulse of each joint produced each time step and it can be computed as the sum of all the impulses exerted by the joint over the iteration.

In the solution of the equations of motion, the joint impulses are evaluated using Eq. (23) and Eq. (19). This computation is performed iteratively in order to reach a set of velocities congruent to kinematic joints. It means that at each iteration a new impulse vector $\{P\}_{constraint}$ is computed.

The accumulated Lagrange multipliers $\{\Delta\}$ of the impulses for each time step can be evaluated as:

$$\{\Delta\} = \sum_{iterations} \{\delta\} \tag{24}$$

The accumulated impulse $\{P_{tot}\}_{constraint}$ can be computed using Eq. (19) obtaining:

$$\{P_{tot}\}_{constraint} = [\psi_q]^T \{\Delta\} \tag{25}$$

The reaction forces $\{F\}_{constraint}$ exerted by the joints can be computed using the general relation between forces and impulses:

$$\{F\}_{constraint} = \frac{\{P_{tot}\}_{constraint}}{h} \tag{26}$$

4.3 Stability Issues

The use of the sequential impulse strategy is subjected to the use of constraint equations expressed in terms of generalized velocity. It means that the exact information about the kinematic joints may be lost during the integration process. In this case a position drift can be observed and stability problems may occur.

In order to enforce the constraints on position a stabilized formulation can be adopted. In this case, the constraint equations in Eq. (21) can be modified as:

$$[\psi_q]\{\dot{q}_{corrected}\} + \{\psi_t\} - \frac{\beta}{h}\{\psi\} = \{0\} \tag{27}$$

where β is a scalar chosen in the range 0–1.

Table 1 Geometrical and inertial parameters of the first example

Parameter	Value
Pendulum length	50 mm
Pendulum mass	0.1 kg
Pendulum principal moments of inertia	$[21, 21, 0.5]$ kg mm^2

5 Examples of Simulation

In order to test the proposed methodology and both hardware and software integration, in this section two examples of implementation are presented and discussed.

5.1 Ten Pendula Dynamic System

The first simulated scenario is about a collection of 10 rigid pendula that move under the effect of gravity. All the pendula have the same geometry and inertial properties which have been summarized in Table 1.

The first pendulum is pivoted to a fixed frame by means of a point-to-point 3 d.o.f. joint. The other pendula are sequentially connected by mean of revolute joints (hinges). The user can interact with the scene by connecting a fictitious spring between the tracker stylus and the free end of the last pendulum. In particular, the user can decide when the connection between the tracker and the last pendulum has to be activated (simulating the clipping) or deactivated (simulating the release). This scenario is also important to test the methodology with event based changes in the equations of motion.

In order to achieve stable and correct results, the solution strategy has to be able to manage rapid changes in force definition. The simulation has been performed with a fixed time step of 0.01 s.

Per each video frame, 4 integration steps are computed and the augmented scene is updated accordingly.

Figure 6 reports a series of four snapshots taken during the run of the simulation. The rigid body collection is real-time rendered along with the simulation. In the first part of the simulation (snapshot A of Fig. 6) the pendula are free to move and they are in an equilibrium position (aligned along the vertical direction).

Then, the user locates the tracker near the last pendulum tip and activates the fictitious spring connection (snapshot B). From this moment, the tip of the last pendulum moves subjected to this connection.

When the user moves the tracker, the tip of the pendulum follows it. It is important to notice that the motion of all the rigid body collection is continuously simulated according to the external action of the gravity and the driving force due to the user's presence.

When the user decides to release the fictitious spring connection (snapshot C), the collection of pendula moves subjected to gravity force only and it oscillates around the equilibrium position (snapshot D).

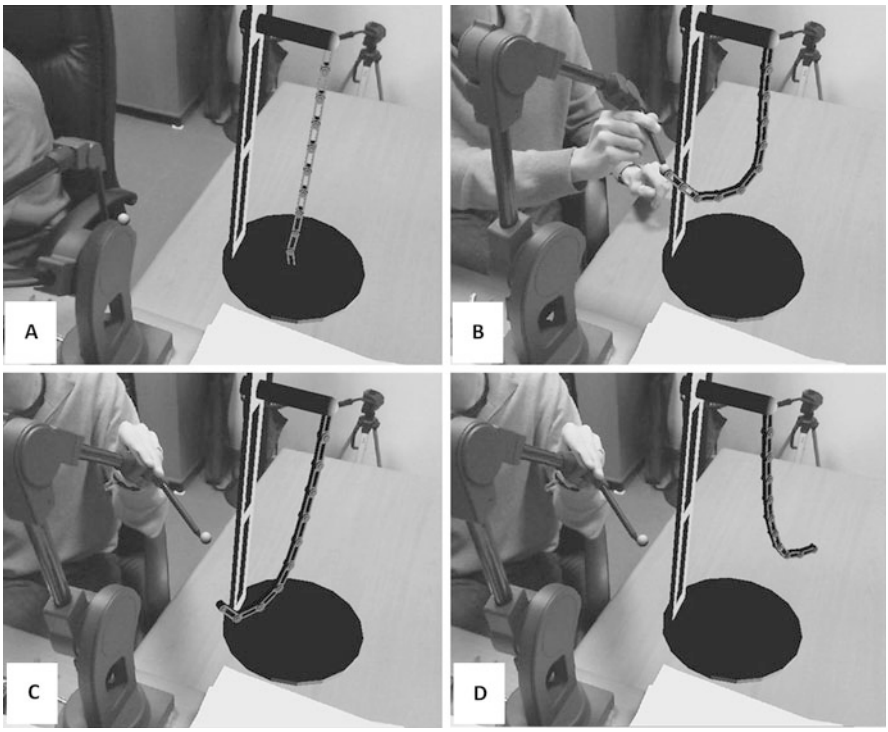

Fig. 6 Snapshots of the simulation of the first example

The accuracy and the stability of the simulation have been checked computing the error in the fulfillment of the constraint equations and the overall energy of the system (potential, kinetic and elastic). The norm of the constraint equations expressed using position variable is always lower than 10^{-6} m except for ten time steps when it reaches $1.6 \cdot 10^{-5}$ m. The variation of the overall energy of the systems (energy loss) is below 1 %. In the computation of the overall energy, the contribution of the external action of the user has been taken into account by evaluating the corresponding power as the dot product between the reaction force of the fictitious spring and the velocity of the digitizer stylus tip as:

$$Power_{user\_action} = \{F_{fictitious\_spring}\} \cdot \{v_{stylus\_tip}\} \tag{28}$$

where

$Power_{user\_action}$ is the power associated to the reaction force of the user's action;
$\{F_{fictitious\_spring}\}$ is the reaction force of the fictitious spring computed as in Eq. (10);
$\{v_{stylus\_tip}\}$ is the absolute velocity of the stylus tip.

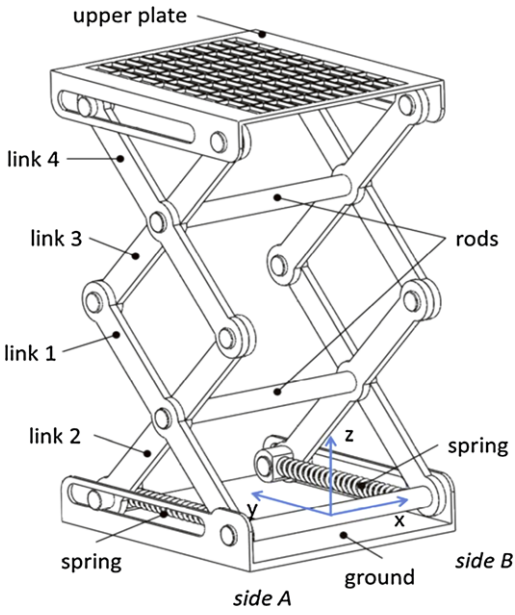

Fig. 7 Cross-lift device of the second example

5.2 Cross-Lift Dynamic Simulation

The second simulated scenario is about the dynamic motion of a cross-lift device which is comprised of 11 rigid bodies connected by 16 hinges and 4 slider joints (see Fig. 7). The links are connected by means of the following constraints:

- Two revolute joints between the frame and the first link on both side A and B of the mechanism;
- Two slider joints between the frame and the second link on both side A and B of the mechanism;
- Four revolute joints between adjacent links on each side of the mechanism (8 in total);
- A revolute joint between the third link and the upper plate on both side A and B of the mechanism;
- A slider joint between the fourth link and the upper plate on both side A and B of the mechanism;
- Four revolute joints connecting the two horizontal rods between the two side of the mechanism.

Two linear spring-damper elements act horizontally between the frame and the slider joint location of the second link on both side A and B of the mechanism. The gravity field acts downward along the vertical direction.

The example has been chosen in order to test the capabilities of the solver to deal with many overabundant constraints. Geometrical, inertial and elastic properties of the simulation have been summarized in Table 2.

Table 2 Geometrical, inertial and elastic parameters of the second example

Parameter	Value
Cross links length	300 mm
Distance between the two side of the mechanism (transverse rod length)	350 mm
Mass of the cross links	0.1 kg
Cross link principal moments of inertia	$[750, 750, 2]$ kg mm^2
Mass of the transverse rods	0.1 kg
Transverse rod principal moments of inertia	$[1021, 1021, 2]$ kg mm^2
Mass of the upper plate	0.1 kg
Upper plate principal moments of inertia	$[1021, 1021, 2]$ kg mm^2
Horizontal spring stiffness	40 N/mm

The user can interact with the scene by imposing a fictitious spring between the stylus tip and the middle point of the upper plate. In particular, the user can decide when the connection between the tracker and the upper plate has to be activated (simulating the clipping) or deactivated (simulating the release).

In the same way of the first example, the simulation has been performed with a fixed time step of 0.01 s. Per each video frame, 4 integration steps are computed and the augmented scene is updated accordingly.

Figure 8 reports a series of four snapshots taken during the run of the simulation. The cross-lift mechanism is real-time rendered along with the simulation.

In the first part of the simulation (snapshot A of Fig. 8) the mechanism is free to move and it is in an equilibrium position. Then, the user locates the tracker in the middle of the upper plate and activates the fictitious spring connection (snapshot B). From this moment, the upper plate vertical coordinates is controlled by the tracker and the mechanism moves subjected to this connection.

When the user moves the tracker, the upper plate follows it in the vertical direction (snapshot C), preserving the right connection with the other rigid bodies. It is important to notice that the motion of all the rigid body collection is continuously simulated according to the external action of the gravity, the springs and the driving force due to the user's presence.

When the user decides to release the fictitious spring connection, the lifting device moves freely subjected to gravity force and internal springs and it oscillates around the equilibrium position (snapshot D).

Also for this example, the accuracy and the stability of the simulation have been checked. Due to the presence of a lot of overabundant constraints, the norm of the constraint equations is higher than in the previous example. During the free motion (when the simulation run without the interaction of the user) it is lower than 10^{-6} m. When the user interacts by pushing and pulling the upper plate, the norm increases and reaches $6.1 \cdot 10^{-5}$ m. The variation of the overall energy of the systems is always below 2 %. As in the previous example, in the computation of the overall energy, the contribution of the external action of the user has been taken into account by

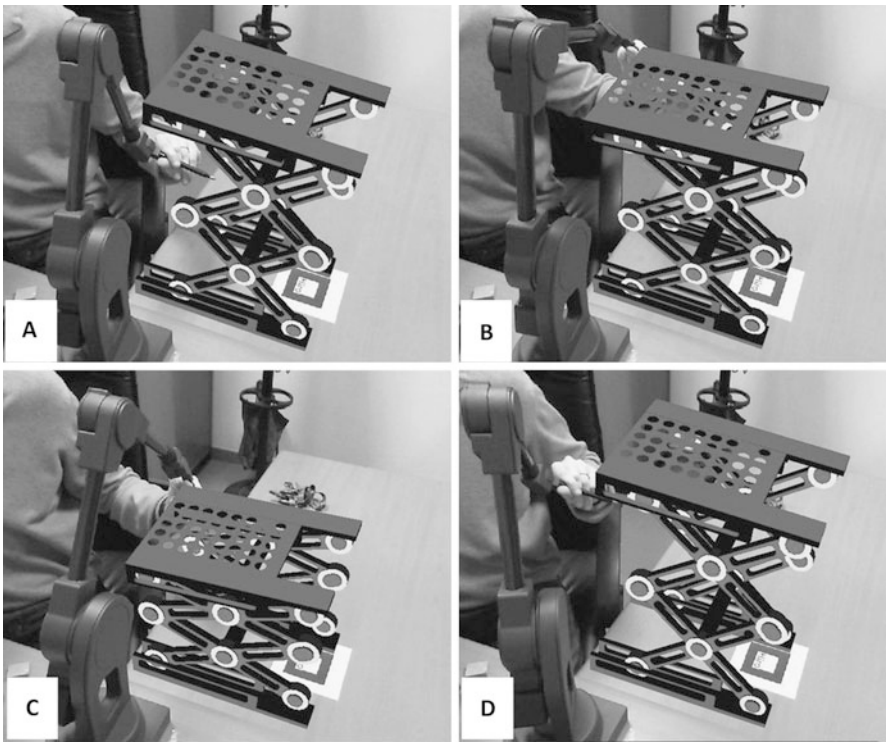

Fig. 8 Snapshots of the simulation of the second example

evaluating the corresponding power as the dot product between the reaction force of the fictitious spring and the velocity of the digitizer stylus tip using Eq. (28).

6 Conclusions

In this chapter, an enhanced methodology for interactive, accurate, fast and robust multibody simulations of mechanical systems using Augmented Reality has been presented and discussed. This methodology is based on the integration of a mechanical tracker and a dedicated impulse based solver.

In this context, the simulation of movement of mechanical systems in an Augmented Reality environment can be useful for projecting virtual animated contents into a real world. By this way, it is possible to build comprehensive and appealing representations of interactive simulations including pictorial view and accurate numerical results.

In particular, two important enhancements have been presented with respect to a previous implementations. First of all, it has been possible to improve the precision of the interaction between the user and the scene by means of a precise mechanical

tracking instrumentation. This constitutes an important improvement if compared with the use of simple optical markers for tracking the user in the scene. In the latter case, the precision in tracking was affected by the resolution of the camera, while with a mechanical device, it is possible to separate the processing of the data coming from the position tracking, from those coming from the image collimation processing. By this way, the simulation input is independent from the visualization input and output.

The second important enhancement is the use of a dedicated solver based on the sequential impulse strategy in order to perform a fast and robust simulation.

According to this approach, the solution is based on the less computational demanding solution strategy. Following the implemented algorithm, the equations of motion are firstly tentatively solved considering elastic and external forces but neglecting all the kinematic constraints. This produces a solution that is only approximated because the constraint equations are not satisfied. In a second step, a sequence of impulses are applied to each body in the collection in order to correct its velocity according to the limitation imposed by the constraint. This second step is iterative and involves the application of a series of impulses to the bodies until the constraint equations are fulfilled within a specific tolerance.

The final result of this work is a tool able to manage real time dynamic simulation and to update the augmented scene accordingly. The robustness and the reliability of the system have been checked over two test cases: a ten pendula dynamic system and the dynamics of a cross-lift mechanism.

According to the proposed methodology, the user can directly control the simulation by a smooth visualization on the head mounted display.

The integration among Augmented Reality, dedicated solver and precise input tracker can be considered an advantage for the future development of a new class of multibody simulation software. Moreover, this integrated simulation environment can be useful for both didactical purposes and engineering assessments of mechanical systems.

References

1. Azuma, R., Baillot, Y., et al.: Recent advances in augmented reality. IEEE Comput. Graph. Appl. **21**(6), 34–47 (2001)
2. Gattamelata, D., Pezzuti, E., Valentini, P.P.: A CAD system in augmented reality application. In: Proc. of 20th European Modeling and Simulation Symposium, Track on Virtual Reality and Visualization, Briatico (CS), Italy (2008)
3. Valentini, P.P.: Augmented reality and reverse engineering tools to support acquisition, processing and interactive study of archaeological heritage. In: Virtual Reality. Nova Publ., New York (2011)
4. Raghavan, V., Molineros, J., Sharma, R.: Interactive evaluation of assembly sequences using augmented reality. IEEE Trans. Robot. Autom. **15**(3), 435–449 (1999)
5. Pang, Y., Nee, A.Y.C., Ong, S.K., Yuan, M.L., Youcef-Toumi, K.: Assembly feature design in an augmented reality environment. Assem. Autom. **26**(1), 34–43 (2006)
6. Valentini, P.P.: Interactive virtual assembling in augmented reality. Int. J. Interact. Des. Manuf. **3**, 109–119 (2009)

7. Valentini, P.P.: Interactive cable harnessing in augmented reality. Int. J. Interact. Des. Manuf. **5**(1), 45–53 (2011)
8. Gattamelata, D., Pezzuti, E., Valentini, P.P.: Virtual engineering in augmented reality. In: Computer Animation, pp. 57–84. Nova Publ., New York (2010)
9. Valentini, P.P.: Enhancing user role in augmented reality interactive simulations. In: Human Factors in Augmented Reality Environments. Springer, Berlin (2012)
10. Buchanan, P., Seichter, H., Billinghurst, M., Grasset, R.: Augmented reality and rigid body simulation for edutainment: the interesting mechanism—an AR puzzle to teach Newton physics. In: Proc. of the International Conference on Advances in Computer Entertainment Technology, Yokohama, Japan, pp. 17–20 (2008)
11. Chae, C., Ko, K.: Introduction of physics simulation in augmented reality. In: Proc. of the 2008 International Symposium on Ubiquitous Virtual Reality, ISUVR, pp. 37–40. IEEE Comput. Soc., Washington (2008)
12. Kaufmann, H., Meyer, B.: Simulating educational physical experiments in augmented reality. In: ACM SIGGRAPH Asia (2008)
13. Irawati, S., Hong, S., Kim, J., Ko, H.: 3D edutainment environment: learning physics through VR/AR experiences. In: Proc. of the International Conference on Advances in Computer Entertainment Technology, pp. 21–24 (2008)
14. Mac Namee, B., Beaney, D., Dong, Q.: Motion in augmented reality games: an engine for creating plausible physical interactions in augmented reality games. Int. J. Comput. Games Technol. **2010**, 979235 (2010)
15. Valentini, P.P., Pezzuti, E.: Interactive multibody simulation in augmented reality. J. Theor. Appl. Mech. **48**(3), 733–750 (2010)
16. Azuma, R.T.: Tracking requirements for augmented reality. Commun. ACM **36**(7), 50–51 (1993)
17. Mirtich, B.V.: Impulse-based dynamic simulation of rigid body systems. Ph.D. thesis, University of California, Berkeley (1996)
18. Schmitt, A., Bender, J.: Impulse-based dynamic simulation of multibody systems: numerical comparison with standard methods. In: Proc. Automation of Discrete Production Engineering, pp. 324–329 (2005)
19. Schmitt, A., Bender, J., Prautzsch, H.: On the convergence and correctness of impulse-based dynamic simulation. Internal report 17, Institut für Betriebs- und Dialogsysteme (2005)
20. Jaimes, A., Sebe, N.: Multimodal human-computer interaction: a survey. Comput. Vis. Image Underst. **108**(1–2), 116–134 (2007)
21. Ullmer, B., Ishii, H.: Emerging frameworks for tangible user interfaces. IBM Syst. J. **39**(3–4), 915–931 (2000)
22. Fiorentino, M., Monno, G., Uva, A.E.: Tangible digital master for product lifecycle management in augmented reality. Int. J. Interact. Des. Manuf. **3**(2), 121–129 (2009)
23. Slay, H., Thomas, B., Vernik, R.: Tangible user interaction using augmented reality. In: Proceedings of the 3rd Australasian Conference on User Interfaces, Melbourne, Victoria, Australia (2002)
24. Valentini, P.P., Pezzuti, E.: Dynamic splines for interactive simulation of elastic beams in augmented reality. In: Proc. of IMPROVE 2011 International Congress, Venice, Italy (2011)