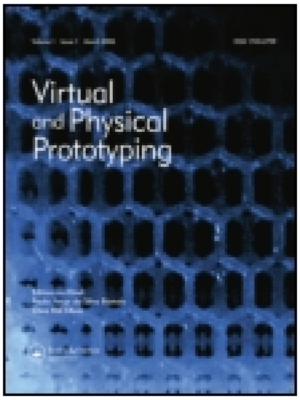

Virtual and Physical Prototyping
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/nvpp20

Natural interface in augmented reality interactive simulations

Pier Paolo Valentini [a]

[a] Department of Industrial Engineering, University of Rome “Tor Vergata”, Via del Politecnico, 1 - 00133, Rome, Italy

Available online: 20 Jun 2012

To cite this article: Pier Paolo Valentini (2012): Natural interface in augmented reality interactive simulations, Virtual and Physical Prototyping, 7:2, 137-151

To link to this article: http://dx.doi.org/10.1080/17452759.2012.682332

Virtual and Physical Prototyping, Vol. 7, No. 2, June 2012, 137–151

Natural interface in augmented reality interactive simulations

This paper demonstrates that the use of a depth sensing camera that helps generate a three-dimensional scene and track user's motion could enhance the realism of the interactions between virtual and physical objects

Pier Paolo Valentini*

Department of Industrial Engineering, University of Rome "Tor Vergata", Via del Politecnico, 1 – 00133 – Rome, Italy

(*Received 6 March 2012; final version received 1 April 2012*)

This paper deals with the description of a novel methodology in order to achieve realistic interactive simulations in an augmented reality environment. The challenge for achieving robust, real-time and accurate simulations is one of the most promising research areas in augmented reality. The key point is to increase the level of interaction between the user and the scene changing the user's role from spectator to actor. In the specific implementation, the interaction between the scene and the user is achieved with the use of an invasively-low device. The methodology is based on the use of a depth infra-red camera in addition to a standard image sensor in order to acquire the three-dimensional geometry of the scene and perform an efficient user tracking. Following this approach, it is also possible to manage occlusions between real objects and virtual ones. All these features have been integrated to a real-time dynamic solver in order to produce accurate manipulation of objects and dynamic simulations. The methodology is presented focusing on both hardware and software tools and some interesting examples have been discussed and the performances evaluated.

Keywords: augmented reality; user tracking; natural interface; occlusion; interaction

1. Introduction

Augmented Reality (AR) is an emerging field of the visual communication and computer technologies. It deals with the combination of real world images and computer generated data (Azuma *et al.* 2001). At present, 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 (Liarokapis 2007, Carmigniani *et al.* 2011). The level of details of the augmented scene has to be very realistic and the registration between real world and virtual contents has to be accurate in order to give the illusion of a unique real world (Bimber and Raskar 2005). A standard AR application is based on the use of one or two cameras which acquire an image stream from the real world. The stream is processed by a computer which interprets the scene and adds virtual contents, producing an augmented image stream which is projected again to the user by means of a blind visor or an external monitor.

The recent developments of both hardware and software performances suggest a desire to increase the level of interaction between the user and the scene changing the user's role from spectator to actor? The main idea to overcome this limitation is to use the AR methodologies for going beyond a mere visual or acoustical experience of pre-computed contents, including the capability of real-time

*Email: valentini@ing.uniroma2.it

Virtual and Physical Prototyping
ISSN 1745-2759 print/ISSN 1745-2767 online
http://www.tandfonline.com
http://dx.doi.org/10.1080/17452759.2012.682332

modifying and updating the contents of the scene. This combination of visualisation and interaction can be a solid base for developing specific computer-aided tools for supporting different activities from simple entertainment to architecture, technology and engineering (Valentini *et al.* 2010). The development of these methodologies and the design of specific devices for increasing the interaction between the user and the augmented scene is one of the most important topics of the research in the field of augmented and mixed reality (Kim and Dey 2010).

In a highly-realistic augmented scene, real objects coexist with virtual ones and it is often required that the user can interact with both. From this point of view, the augmented scene can be implemented with different levels of interaction.

Scientific literature reports several contributions dealing with possible methodologies for enhancing the interaction between the user and the augmented scene. Some of them (Piekarski and Thomas 2002, Pang *et al.* 2006, Valentini and Pezzuti 2010, Park 2011, Valentini 2011) involve the presence of patterned markers which are used for both computing the perspective transformation between the camera and the real world and for transferring information from the user to the scene. They are considered as communicators and the interaction is based on the computation of their relative position and attitude with respect to the camera and the other markers reference frame. These approaches do not require additional devices in order to manage the interaction but have some disadvantages. First of all, their precision is limited to that of the image acquisition system (Yamauchi and Iwamoto 2010). Moreover, the continuous and real-time tracking requires all the markers to be visible by the cameras at the same time and this occurrence reduces the working space for interaction (Azuma 1993).

Other contributions introduced the use of different sensors for tracking the position of the user in the scene and interpreting his intent (Vallino 1998). These methodologies are useful for both augmented reality and virtual reality applications. For this purpose, there are implementations concerning the use of optical trackers (Valentini and Pezzuti 2010), magnetic trackers (Valentini *et al.* 2010), mechanical trackers (Valentini 2011) and wearable gloves and sensors (Buchmann *et al.* 2004, Huagen *et al.* 2004, Valentini 2009). Optical tracking systems often make use of reflective markers (usually spheres for enhancing the visibility) whose position in the scene can be recognised by photogrammetric analysis using multiple cameras. Their precision is affected by the resolution of the cameras and suffers occlusion phenomena. The magnetic trackers are more precise, but they are influenced by electromagnetic perturbations caused by metallic parts in the scene. Mechanical trackers are even more precise, are not affected by occlusions and perturbations, but their capture volume is limited. They can be used in addition to haptic feedback in order to increase the level of realism and perception. Wearable sensors can be used for interpreting the user's intent (picking, grabbing, pushing, indexing, etc.) but they have to be used in addition to other tracking systems in order to compute the spatial position of the user in the scene.

According to all these contributions, the interaction between the user and the augmented scene is obtained by introducing sensors of differing type and nature whose information has to be real-time interpreted by the computer processor. The introduction of these sensors is somehow unreal because, in general, they are not naturally included in the real world and they may mine the overall realism of the scene. In some implementations they produce heavy, complex and difficult architecture requiring training and personal skill to be properly managed. For this reason, the more effective interaction would be achieved without the use of any sensor placed on the user. This way, the video stream processor should interpret the scene from outside without the inclusion of internal artefacts.

The motivations of the research come from all these considerations. The main idea concerns the development of a methodology for achieving a natural interface between the user and the scene in order to increase the realism of the augmented world.

This paper is organised as follows. In the first part, the outlines of the proposed methodology are presented, focusing on hardware description and software integration. In the second part, the details about the processing of the augmented scene are described, discussing the managing of occlusions, the natural user's interaction and the physical behaviour of virtual objects. In the last part some examples of implementation of the methodology are presented and discussed underlining the corresponding advantages and disadvantages in order to assess the system performances.

2. System integration and methodology overview

In the most implemented augmented reality applications, the acquisition of the real world is performed using a single camera or, for stereoscopic projection, two cameras (Vallino 1998). The role of these devices is to produce one or two Red-Green-Blue (RGB) image(s) that can be used for both image processing and for the definition of the background image of the final augmented scene projection.

In the presented methodology a different approach has been used. The real world is acquired by a Microsoft Kinect Sensor. It is a compound of an RGB camera, an infra-red (IR) projector and an IR depth camera (see Figure 1). The RGB video stream uses 8-bit Video Graphic Array (VGA) resolution (640×480 pixels) with a Bayer colour filter, while the monochrome depth sensing video stream is in VGA resolution (640×480 pixels) with 11-bit depth, which provides 2048 levels of sensitivity. The Kinect Sensor outputs video at a frame rate of 30 Hz. Although the

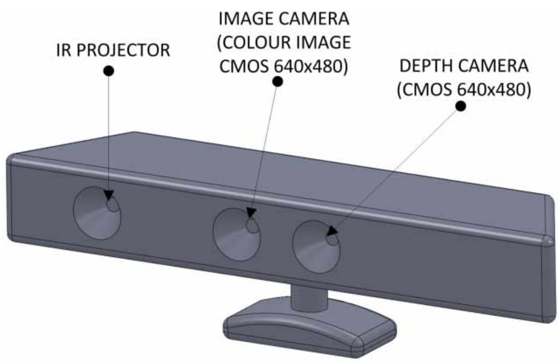

Figure 1. Layout of the Microsoft Kinect Sensor.

Kinect Sensor has been originally developed for home entertaining purposes, it has been also considered very interesting for technical and research activities due to the affordable cost, accuracy and personalisation. Interesting examples of application in the fields of robotics can be found at the internet site *http://www.ros.org/*.

The use of the Kinect Sensor allows the synchronised acquisition of an RGB image (as for the common AR applications) and a depth map of the same real scene. An RGB image is a data structure containing colour information of each acquired point (pixel). A depth map is a data structure containing the distance from the sensor of each acquired point (pixel) along a direction perpendicular to the image plane. In order to obtain the depth map, the IR emitter projects a light pattern of spots on the scene and the IR camera acquires the image of the pattern on the environment surfaces and computes the distance from the sensor (Freedman *et al.* 2010). The pattern is created by projecting optical radiation through a transparency containing the pattern. By combining the information of the depth map about the position *u, v* of a pixel on the image plane and its depth *d*, it is possible to compute the Cartesian coordinates *x, y* and *z* of the corresponding physical point in the scene with respect to the sensor.

The proposed methodology has been implemented using an Intel Core 2 Quad Q9550, 8 Gb RAM, and a Nvidia Quadro FX 3700. The operating system was Window 7 $\times$ 64 and the development suite for coding was Microsoft Visual Studio 2010. In order to acquire and process the data coming from the Kinect sensor the Prime Sense drivers have been used. These drivers are an alternative to the Microsoft Kinect SDK and are more suitable for processing raw data. They can be freely downloaded at *https://github.com/PrimeSense/Sensor*.

There are two data streams coming from the Kinect Sensor (see Figure 2). The first one, as in a traditional augmented reality application, is the RGB video stream. Each RGB frame can be processed in order to recognise the presence of patterned markers in the scene and to compute the perspective transformations between the camera and each marker $[T^{camera}_{maker,i}]$. These transformations are useful for a correct projection of the three-dimensional (3D) virtual contents in the rendered augmented scene. The processing of the RGB video stream can be performed using the standard ARToolKit libraries (Kato and Billinghurst 2011) which are freely downloadable at: *http://www.hitl.washington.edu/artoolkit/download/*. The ARToolKit has been suc-

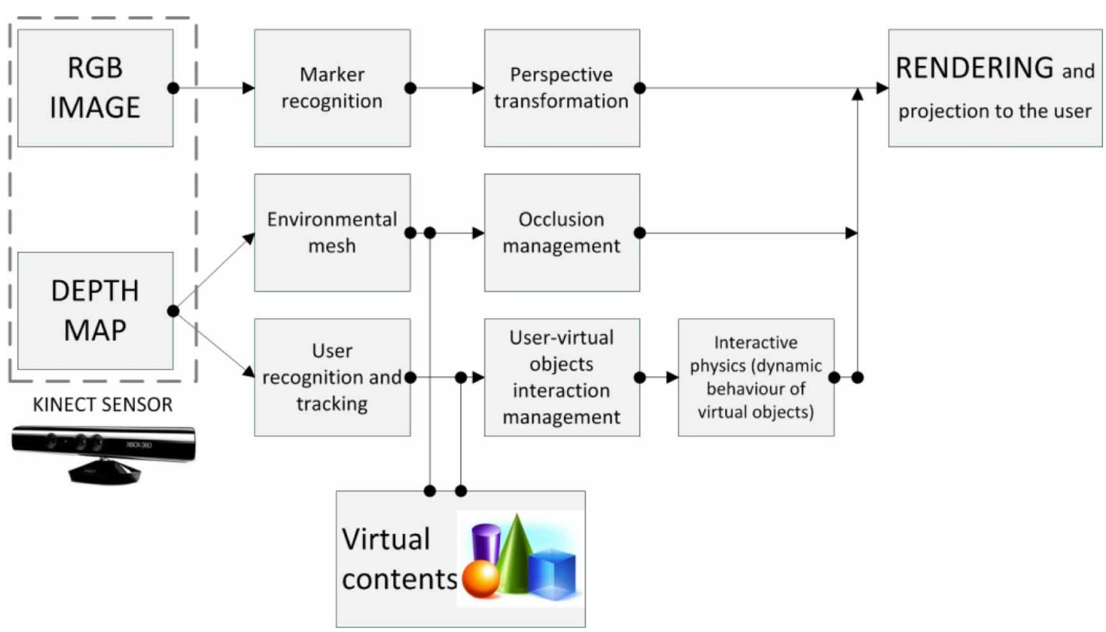

Figure 2. A schematic overview of the proposed methodology.

cessfully used in many previous works due to the accuracy, robustness and configurability. The only difference from a standard application is that the acquisition of an RGB image from the Kinect Sensor has to be performed using specific driver and video libraries which are herein discussed.

The processing of the depth map stream allows the inclusion of several enhancements useful for increasing the realism of the augmented scene and the level of interaction. Two are the main processes of the depth map. The first one concerns the computation of the environmental mesh which is a geometrical representation of the real world three-dimensional geometry.

Starting from the knowledge of the 3D coordinates of each point observed by the depth camera, it is possible to build a structured polygonal mesh describing the geometry of the surrounding environment.

This mesh can be textured with the colour information coming from the RGB camera. Since the two cameras include sensors that have different focal lengths and are located in different positions, a registration of depth map and RGB image has to be performed considering an offset and scaling. These transformations are intrinsic to the Kinect Sensor and can be performed automatically using the driver libraries.

An example of the environmental mesh reconstructed by the depth camera stream is depicted in Figure 3.

The environmental mesh has been computed using the OpenNI library freely downloadable at *https://github.com/OpenNI/OpenNI*. The OpenNI is a collection of routines for accessing and processing data from the Kinect Sensor using Windows or Unix operating systems.

Another interesting use of the depth camera stream is the possibility of tracking the users in the scene. For this purpose, it has been used with the OpenNI skeleton tracking tool for recognising and tracking the main body landmarks (physical joints) of a human user. Thanks to the complete scene 3D reconstruction of the depth camera, it is possible to compute the 3D coordinates of each body joint with respect to the sensor. It is interesting to note that the depth camera allows the tracking of the user without the use of any specific sensor. The body is recognised and tracked by just interpreting images and the position of body segments is computed. This is a crucial issue towards the development of a natural interface between the user and the augmented scene. Details about the user tracking will be provided in the next section.

After the computation of the environmental mesh and the position of the user in the scene, we can add the virtual contents (3D models and textual information).

The environmental mesh can be useful for managing occlusions between real objects and virtual ones. Standard augmented reality implementations neglect occlusions between the real and the virtual objects and the acquired image from the real world is considered as a background texture on which virtual objects and information are rendered. This means that a virtual object can occlude a real one but a real object cannot be superimposed on a virtual one. By using the information coming from the depth map sensor, occlusion of real objects with respect to the virtual one can be managed. Details will be provided in the next section.

Thanks to continuous user tracking, the interaction with virtual objects can be implemented as well. The user tracking routines allow the computation of the position of the user main body segments. This is useful for dealing with possible collisions and contacts or other specific interactions (grabbing, pushing, pinching, etc.). Details about possible interaction between the user and virtual objects are discussed in the next section.

Figure 3. An example of the environmental mesh of a portion of a room containing a sofa with a box on it computed from Kinect Sensor depth map acquisition. The image on the left has been generated considering raw depth data; the image on the right has been generated applying on the raw data a registered texture coming from the RGB image sensor.

In order to increase the level of realism, the objects in the scene have to behave according to physical laws. This suggests the use of a dedicated physical solver to simulate the dynamics of the object in the scene. Previous publications about the integration of dynamics simulation in augmented reality applications (Valentini and Mariti 2011) have revealed the interesting capability of the sequential impulse solvers.

3. Depth map acquisition and processing

Depth map raw data consist of an array containing each pixel's coordinates u, v on the image plane and the distance d along the normal direction to the image plane of the corresponding projected point. Neglecting the lens distortion and considering the focal properties of the camera and the pixel spacing, it is possible to convert these local coordinates to the global ones x, y, z which relate the position of each acquired point to that of the sensor.

With reference to Figure 4, the coordinate transformation can be written as:

$$\begin{cases} x = \left(\frac{u}{URES} - 0.5\right) \cdot d \cdot 2 \cdot \tan\left(\frac{FOH}{2}\right) \\ y = \left(0.5 - \frac{v}{VRES}\right) \cdot d \cdot 2 \cdot \tan\left(\frac{FOV}{2}\right) \\ z = d \end{cases} \quad (1)$$

where:

$URES$ is the horizontal resolution in pixels of the image plane (along the u direction);

$VRES$ is the vertical resolution in pixels of the image plane (along the v direction);

FOH is the horizontal angle of view (intrinsic parameter of the camera);

FOV is the vertical angle of view (intrinsic parameter of the camera).

In the specific implementation of this paper the above mentioned parameters have been set as:

$$URES = 640 \text{ pixels}$$
$$VRES = 490 \text{ pixels}$$
$$FOH = 1.0144686707507438 \text{ rad}$$
$$FOV = 0.78980943449644714 \text{ rad}$$

In order to correlate the information coming from the depth camera stream to that coming from the RGB camera, the point word coordinates in Equation (1) have to be registered to that of the RGB camera and this computation can be automatically performed using the OpenNI libraries.

4. Managing occlusions

The first advantage in using depth map information is the ability to manage occlusions between real objects and virtual ones.

A correct interpretation of the occlusion is one of the most challenging topics in augmented reality. Scientific literature includes some papers on the development and testing of methodologies for managing occlusion phenomena in mixed and augmented reality.

One of the most used strategies for managing occlusions is the use of phantoms (Breen *et al.* 1996, Fischer *et al.* 2004). According to this approach, real occluding objects are included in the scene as virtual replicas whose geometries have been modelled with sufficiently precise representations.

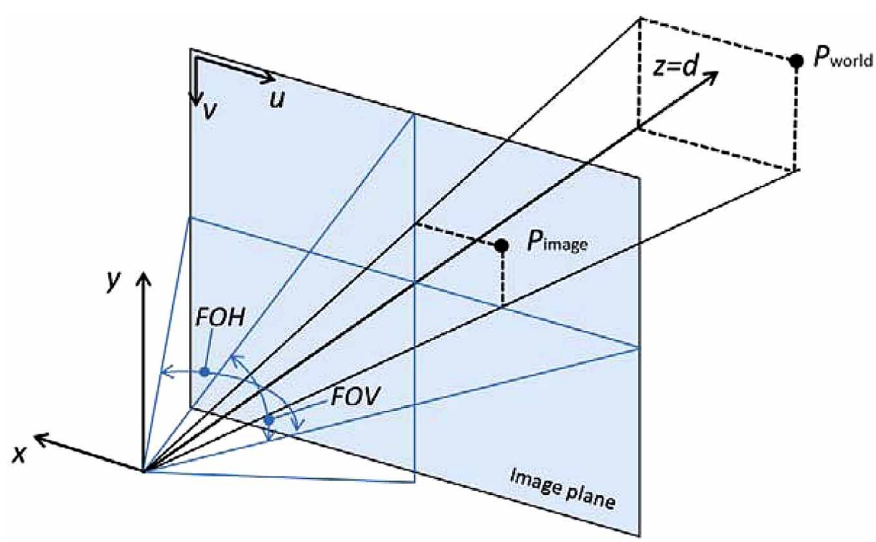

Figure 4. Relationship between image coordinates and real coordinates.

These virtual mock-ups are invisible to rendering engines but are used for performing depth comparison of physical objects for building an alpha map and limit the rendering portions of virtual occluded objects. Their use is simple but they require preliminary models of all the possible objects. Moreover, this methodology is more suitable for static objects, but more complicated for moving objects due to the necessity of continuous geometry tracking.

Another methodology is based on the evaluation of real occluding objects' shape by geometrical tracking using stereo matching (Hayashi *et al.* 2005, Fortin and Hebert 2006) or direct image segmentation (Tian *et al.* 2010, Zollmann *et al.* 2010).

More recently, the use of depth maps generated by stereo-cameras (Breen *et al.* 1995, Wolka and Anderson 1995, Koller *et al.* 1997, Kanbara *et al.* 2000) or time-of-flight cameras (Fischer *et al.* 2007) has been used for acquiring the spatial coordinates of the real object in the scene and to compute possible occlusion with the virtual ones. This last method seems very promising and is adaptable for integration in the present work due to the access to depth map information. Details about this integration are herein reported.

The occlusions among all the virtual objects in the scene can be easily managed using the z-depth (culling) check that is a very common procedure in computer graphics. When an object has to be rendered, the depth of a generated pixel (z coordinate) is stored in a buffer (called the z-buffer). If another object of the scene must be rendered in the same pixel, the render engine compares the two depths and chooses the one closer to the observer. The chosen depth is then saved to the z-buffer, replacing the old one. In the end, the z-buffer will allow the render engine to correctly reproduce the usual depth perception: a close object hides a farther one. In the present study we use the information coming from the depth camera in order to compute the z-coordinate for each pixel of the environmental mesh. For this reason, we decompose the overall background colour texture in smaller textiles. Each textile is then assigned to a small polygon of the environmental mesh whose vertices have x, y and z coordinates computed from depth image processing as shown in the previous section. Following this approach we populate a two-dimensional (2D) texture over a 3D polyhedral mesh (environmental mesh) as shown in the example of Figure 3. This mesh can be included in the z-depth check of the real object in the scene together with the other 3D virtual entities.

Figure 5 shows an example of implementation in an augmented scene. The background image acquired by the RGB camera is projected on the environmental polygonal mesh created from the depth camera acquisition. The scene contains a virtual cube (on a real table) and a virtual cylinder (on the floor). Both the objects can be occluded by other real objects in the scene (the table and the chair, in the example) and by the user's body during his movement.

The greatest problem in creating the environmental mesh from the depth camera acquisitions is due to the accuracy and resolution of the z-depth computation. The Kinect sensor has a depth resolution of some millimetres. It has been assessed that at a distance of 1.5 m from the sensor, the z-resolution is about 3?5 mm. Moreover, the x-y spatial resolution of the environmental mesh depends on the depth camera u-v resolution which is currently 640x480 pixels. This produces an environmental mesh with sharp polygons which causes some imprecision near the boundary when checking for occlusions (see Figure 6, on the left). In order to improve the quality of the environmental mesh and to achieve a more precise computation, a Catmull-Clark subdivision has been computed (Catmull and Clark 1978). This modification allows for a smoother and more detailed mesh improving occlusion effects. Figure 6 reports an example of an environmental mesh detail as acquired (on the left) and after the subdivision (on the right).

5. Managing the presence of the user in the scene

The tracking of the user's position in the spatial scene is a very important topic for building a highly interactive augmented environment. There are two main advantages of this computation. The first one concerns the possibility of capturing the three-dimensional position of the user's body segments in the scene in order to check impacts with virtual objects and prevent interpenetration. The second advantage is about the possibility of acquiring the user's pose (i.e. the position of limbs and hands) in order to interpret his intent in participating and modifying the augmented scene using simple and natural gestures (picking, grabbing, pushing, etc.). It is important to underline that the tracking of the user is not necessary for managing occlusions, because they are computed using the environmental mesh as described in the previous section. For this reason, in order to interpret the presence of the user in the scene and his intent, it is sufficient to track only some principal body landmarks, neglecting the exact segment shapes and geometrical details.

5.1 *User tracking*

The OpenNI library includes tracking routines for achieving a robust and precise tracking of the user's body main landmarks. Although the exact implementation of these algorithms is not open access, some useful information can be extracted from the related patent application (Berliner *et al.* 2010).

According to this approach, the tracking of the user is performed by processing the depth map. After its

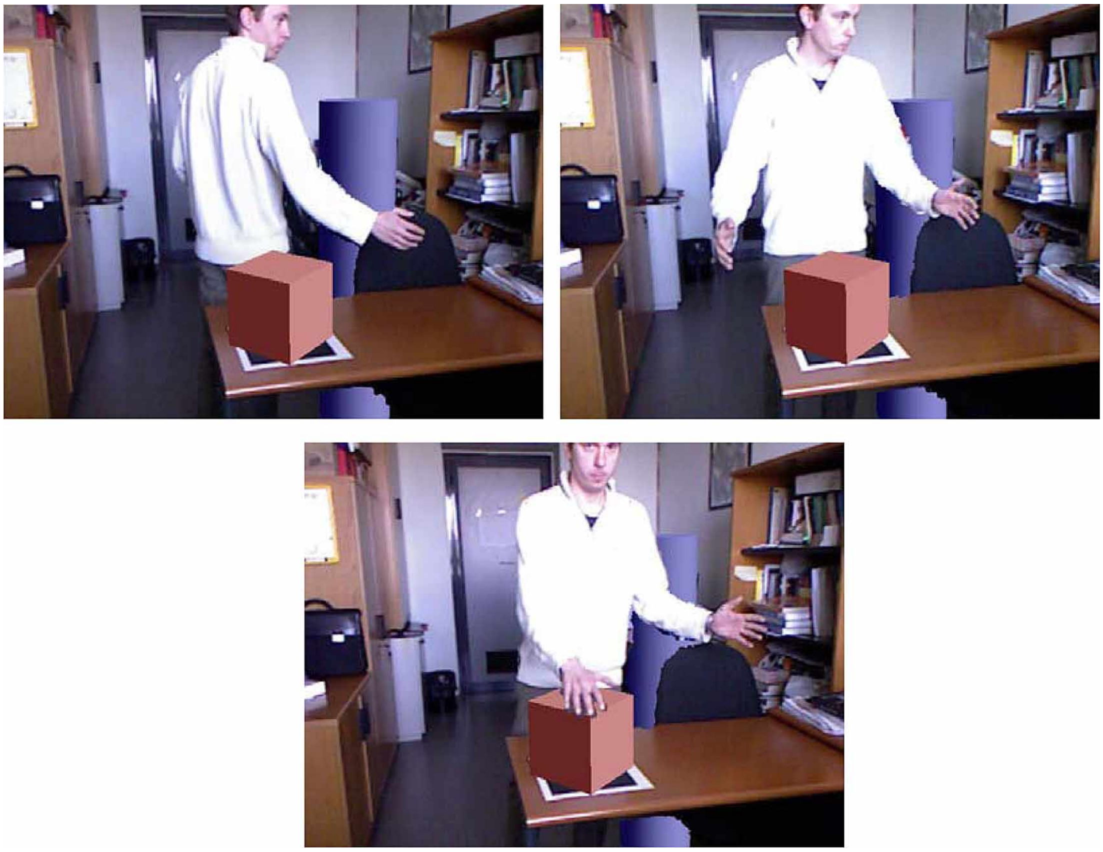

Figure 5. An example of the environmental mesh coming from depth map acquisition for dealing with static and dynamic occlusions.

acquisition, three main computational steps have to be performed. The first one concerns the removal of the background, in order to isolate the pixels that belong to the user body and with detection of body contours. The second is about the processing of these contours in order to identify main body parts as head, limbs and chest. The last one is about the computation of the medial axis of each part in order to output the spatial coordinates of the main body joints.

The OpenNI algorithm allows the tracking of position and orientation of the following 16 user's body landmarks (see Figure 7):

- Centre of the head;
- Centre of the neck;
- Right and left shoulder joints;
- Right and left elbow joints;
- Centre of right and left hand

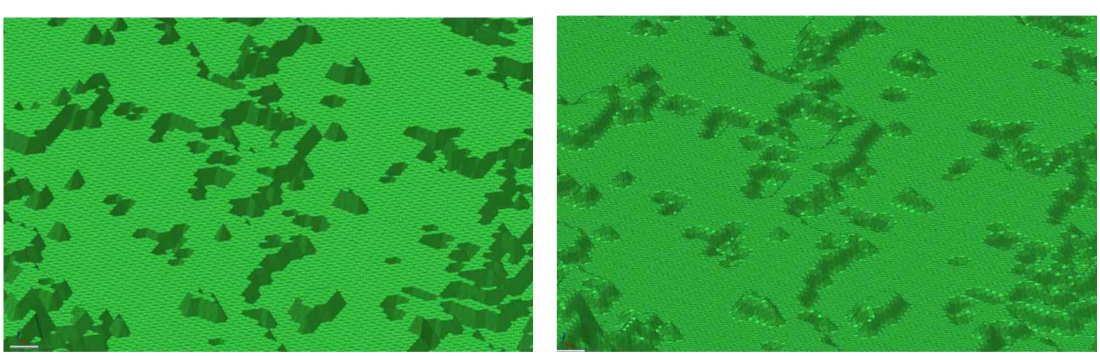

Figure 6. An example of Catmull-Clark subdivision for smoothing and up-sampling the environmental mesh (the portion of the mesh is zoomed).

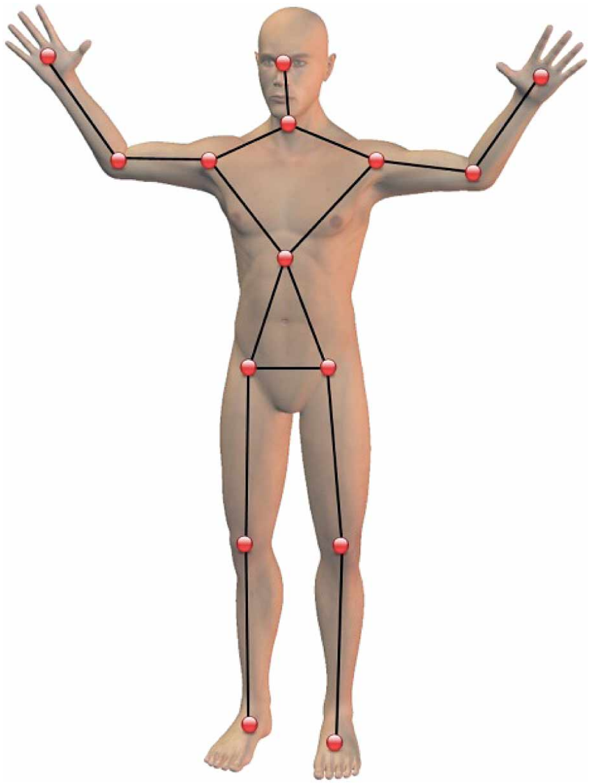

Figure 7. Traceable body landmarks and the corresponding stick skeleton.

- Centre of the chest;
- Centre of the abdomen;
- Right and left hip joints;
- Right and left knee joints;
- Centre of the right and left feet.

The tracking procedure requires a preliminary calibration that can be achieved by the user maintaining the "PSI" pose for some seconds. The "PSI" pose is depicted in Figure 7 and consists of standing in front of the depth camera, holding the body in the upright position with the upper limbs forming a Greek letter ψ appearance. The calibration position is necessary to store the anthropometric details of the body and for setting the predator-prey tracking algorithm initial condition. It is important to underline that the tracking algorithm is performed by the depth camera only, so the coordinates of the traced points can be easily transformed from the image plane and depth distance to the real three-dimensional space using the formulas in Equation (1).

The algorithm allows the tracking of several users at the same time. Obviously, each user is required to perform his own calibration procedure.

5.2 *Interaction between user and virtual objects*

There are two possible interactions between the user and the virtual objects that have been taken into account in the implementation. The first one is the possibility of impacts between the user and the virtual objects. This occurrence has to be detected in order to avoid physically incorrect interpenetrations and increase the level of realism. In this case, the detection of possible collision has to be monitored with respect to the entire body geometries.

The second interaction concerns the user's active will to modify the content of the augmented scene and it is related to the possibility of picking and moving objects. In this case, the interaction is limited to the hands which are the most important user's natural interfaces to the real world.

Let us now discuss in details how the two types of interaction can be implemented.

The detection of contacts between the user and the virtual objects in the scene can be implemented using a phantom simplified 3D model of the human body.

This model is a collection of solid primitives as spheres, cylinders and cones which approximates the geometry of the user's body. Their shapes are simple in order to apply robust and computationally efficient 3D contact detection algorithms (Coutinho 2001).

Each body segment solid primitive is considered attached to the user's stick skeleton in Figure 7. The spatial position and attitude of each shape is computed from the location of corresponding body landmarks.

The collision detection algorithm takes as input the position and the shape of the virtual body in the scene and computes (if the collision occurs) the contact points, penetration distance and the common normal vector.

Internal collisions among body segments have been excluded from the computation, since they cannot occur due to the physical presence of the user's body.

Figure 8 shows an example of phantom body segments rendered during user's body tracking.

The interaction between the user and virtual objects including picking is implemented in a different way.

For this purpose, the approach proposed in Valentini (2009) has been adapted for dealing with object manipulation and virtual assembling. The approach seems computationally efficient and also robust for implementation in the augmented environment of the present study.

The approach in Valentini (2009) is based on the use of geometrical checks and constraint equations between body landmarks and object geometrical features.

For this purpose, we can define a *Grasping Active Feature* (GAF) on the hand of the user. With reference to Figure 9, it is defined by a reference frame that is attached to the traced specific body landmark. For each pickable object, other active features have to be defined together with their

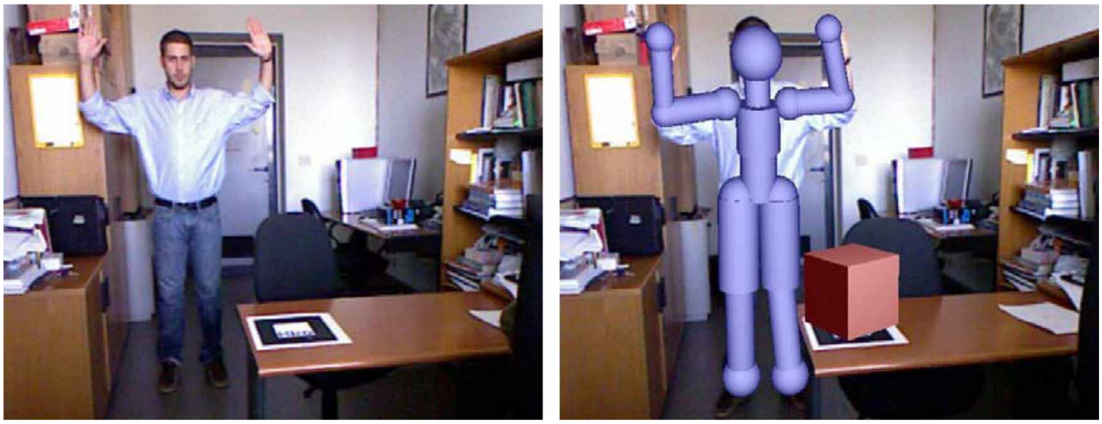

Figure 8. A collection of 3D solid primitives are attached to the user's skeleton computed from the tracking algorithm.

local reference frames. They will be referred to as a set of *Object Active Features* (OAF). All the mating relationships for picking and moving objects can be expressed by imposing constraint equations between these entities and other analogous entities on virtual objects. Both GAF and OAF represent sort of handles that can be used for imposing picking conditions. These constraints are included in the equations of the dynamic system to be solved. In order to manage small errors and imprecisions due to the implemented low-cost system, the constraint equations relating the two active features have been substituted by spring-damper elements (Valentini and Mariti 2011, Valentini and Pezzuti 2011).

Thanks to the OpenNI and NITE algorithms, we have sufficient information to locate the centre of the hand in the scene. In order to grasp a virtual object, two conditions must be fulfilled. The first one is that the GAF and OAF have to be sufficiently close. This condition controls the proximity between the user's body segment and the virtual object geometry. The second condition is that the user has to express the intent to pick the object, i.e. to impose grasping equations. If, and only if, the two conditions are fulfilled the grasping succeeds. The intuitive concept of grasping has to be expressed using algebraic equations. With reference to Figure 9, let us consider a virtual box and a generic OAF with its associated reference frame ($O_{Bi,j} - x_{Bi,j}y_{Bi,j}z_{Bi,j}$). Let us also consider a user with a GAF located at his hand, $O_{FT} - x_{FT}y_{FT}z_{FT}$.

The condition for grasping detection can be stated as:

$$\left\| O_{Bi,j}O_{FT} \right\| \leq tol \rightarrow \text{grasping may occur} \qquad (2)$$

Equation (2) means that the grasping can occur if the distance between the origins of the OAF and GAF reference frames is smaller than a specific tolerance.

After the detection of a possible grasping configuration using Equation (2), the user can confirm or dismiss the

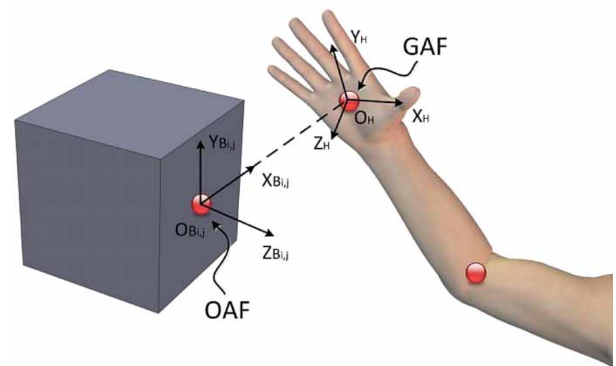

Figure 9. Grasping active feature (GAF) and object active feature (OAF).

action. This intent can be communicated via a voice command. If he confirms the intent, the object can be manipulated according to user movements, i.e. it moves fixed to the hand. Again, this condition can be implemented imposing several kinematic constraint equations on the movement of the virtual body or equivalent spring-damper elements as described in Valentini and Pezzuti (2010) and Valentini (2009).

Since the OpenNI and NITE algorithms are able to estimate not only the position of the centre of the hand but also its orientation (*i.e* the complete GAF reference system), complex constraints can be also established. The interested reader can find useful information about how to implement hand-object specific constraints (cylindrical, spherical and pinch) in Valentini (2009).

The picking phase ends when the user decides to release the grasping condition removing the constraints limiting the virtual body motion. Again, this occurrence can be communicated using a voice command.

6. Kinematic and dynamic behaviour of the virtual objects in the scene

In order to increase the level of realism, the objects in the scene have to behave in a physically correct way. In particular, they have to move according to Newton's laws. It means that the correct motion of the virtual object in the scene has to be computed solving specific equations of motion.

Given a collection of rigid bodies, the systematic deduction of the corresponding equations of motion and their solution can be performed using one of the multibody system dynamic approaches (Haug 1989, Korkealaakso 2007). For the specific integration in an augmented reality environment, not all the possible formulations can be used because a stable and real-time computation is required (Mirtich 1996).

Previous papers demonstrated that a convenient formulation for dealing accurate simulations of the motion of a rigid body collection involving impacts is that based on sequential impulses (Schmitt and Bender 2005, Schmitt *et al.* 2005). Due to its robustness, accuracy and velocity it is suitable for real-time processing and long simulation (Valentini and Mariti 2011).

Given a collection of rigid bodies, constrained by several equations and subjected to a set of external forces, the equation of motion can be deduced as,

$$\begin{cases} [M]\{\ddot{q}\} - \left[\psi_q\right]^T\{\lambda\} = \{F_e\} \\ \{\psi\} = \{0\} \end{cases} \tag{3}$$

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 generalised coordinates, velocities and accelerations, respectively;

$\{\psi_q\}$ is the Jacobian matrix of the constraint equations (differentiated with respect to the generalised coordinates);

$\{\lambda\}$ is the vector of Lagrange multipliers associated with the constraint equations;

$\{F\}_e$ is the vector containing all the elastic and external forces.

There are two main steps of 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:

$$[M]\{\ddot{q}\}_{approx} = \{F_e\} \tag{4}$$

This way, Equation (4) can be solved for $\{\ddot{q}\}_{approx}$ that represents the vector of approximated generalised coordinates.

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

$$\{\dot{q}\}_{approx} = h \cdot \{\ddot{q}\}_{approx} \tag{5}$$

$$\{q\}_{approx} = h \cdot \{\dot{q}\}_{approx} \tag{6}$$

where h is the integration time step.

In a second step, a sequence of impulses $\{P\}_{constraint}$ is applied to each body in the collection in order to correct the velocities $\{\dot{q}\}_{approx}$ according to the limitation imposed by the constraint. This second step is iterative. Each impulse is computed imposing the fulfilment of the constraint equations written in terms of generalised coordinates. From Newton's laws, the application of the impulses causes a variation of momentum:

$$[M]\left(\{\dot{q}\}_{corrected} - \{\dot{q}\}_{approx}\right)\{P\}_{constraint} \tag{7}$$

where

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

The corrected velocities can be computed from Equation (7) as:

$$\{\dot{q}\}_{corrected} = \{\dot{q}\}_{approx} + [M]^{-1}\{P\}_{constraint} \tag{8}$$

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

$$\{P\}_{constraint} = \left[\psi_q\right]\{\delta\} \tag{9}$$

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

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

$$\{\psi\} = \{0\} \Rightarrow \frac{\partial\{\psi\}}{\partial t} = \left[\psi_q\right]\{\dot{q}\} + \{\psi_t\} = \{0\} \tag{10}$$

$$\left[\psi_q\right]\{\dot{q}_{corrected}\} + \{\psi_t\} = \{0\} \tag{11}$$

Inserting Equation (9) into Equation (8) and substituting $\{\dot{q}\}_{corrected}$ into Equation (11), we can obtain:

$$\left[\psi_q\right]\left(\left\{\dot{q}_{approx}\right\} + [M]^{-1}\left[\psi_q\right]^T\{\delta\}\right)\{\psi_t\} = \{0\} \tag{12}$$

Equation (12) can be solved for $\{\delta\}$ obtaining:

$$\{\delta\} = \left(\left[\psi_q\right][M]^{-1}\left[\psi_q\right]^T\right)^{-1}\left(\left[\psi_q\right]\{\dot{q}\}_{approx}[\psi_t]\right) \tag{13}$$

Then, the impulses can be computed using Equation (9) and the corrected values of generalised velocities using Equation (8). Since the impulses are computed sequentially,

Figure 10. Snapshots of a natural manipulation of a cylinder into an augmented scene including occlusion management.

the global fulfilment of the constraint equations cannot be directly achieved. Some iterations are required.

It is important to underline that each impulse is applied independently from the others. In this way the constraint equations are not solved globally, but iteratively. The sequential impulse strategy is able to also manage initial transient conditions without requiring consistent initial condition on velocities which can be difficult to achieve due to limited precision in the image acquisition process. Complete details of the sequential impulse strategy go beyond the scope of this paper and an interested reader can refer to the referenced papers.

In the implementation described in this paper, the computation is performed using a fixed time step in order to synchronise the augmented scene. The value of this time step has to be an integer ratio of the frame acquisition. For all the proposed examples, considering a frame rate of 25 f.p.s., four time sub-steps have been performed for each acquisition (at 1/100 s). After the computation of all the sub-steps, the position and attitude of all the bodies in the scene are known and all the geometries can be rendered in the updated position.

In the augmented scene we can distinguish between three types of rigid bodies. The objects that cannot move (*i.e.* they are fixed) but can take part in collision can be defined as static bodies. For example, the surrounding environment (objects and scenario) can be defined using static rigid bodies. The objects that are movable, but where their motion does not depend on the integration of the equations of motion but is prescribed by the user, can be considered as kinematic rigid bodies. This typology is useful to describe the user's body phantom geometries. User's body segments have a prescribed motion (from an external source) that cannot be computed using Newton's laws. On the other hand, they can participate in the overall dynamic simulation by contacting other entities and altering their motion.

Finally, the objects whose motion has to be computed using Newton's laws and which are subjected to external forces and constraints are defined as dynamic rigid bodies. This description is useful for simulating all the free objects in the scene.

7. Methodology assessment and examples of application

In order to assess and validate the methodology and the system integration, several tests have been developed.

Figure 11. Snapshots of a simultaneous natural manipulation of two cylinders by two different users into an augmented scene.

7.1 *Preliminary assessments*

The first assessment concerns the test of robustness and precision of the tracking procedure. This preliminary investigation is very important since all the interaction strategies are based on the data acquired from the user's tracking. A group of 20 subjects with different anthropometrical features were asked to get calibrated and reach nine specific locations with their head and hands within a cube of 3 x 3 x 3 m. The coordinates of these specific locations have been measured with precision and have been compared to those acquired from the tracking algorithm. Error distances in tracking are always within 2 cm. The precision is higher (1 cm) in the x, y directions (the direction aligned to that of the image plane) and lower in the z (depth) direction (2 cm). The tracking accuracy seems not to be influenced by the user's body size. Due to the approximation in the recognition of the human body and the resolution of the device cameras, it can be considered a good result, suitable for the purpose of the proposed methodology. Moreover, the tracking has proved its robustness with 99% of valid data acquisition.

The second example concerns the manipulation of a virtual object for the assessment of the picking methodology and the correct managing of occlusions between real and virtual objects. The simulated scenario is about a user who moves in the scene, is able to grab a cylinder and moves it into the scene where another virtual entity is placed (a cube). In Figure 10 some snapshots taken during the simulation are reported. At the beginning of the simulation the user has to calibrate his pose. Then, he can grab a virtual cylinder by placing his hands near the upper and lower faces of the cylinder (where two OAFs) have been placed) and enforcing the constraints simulating the grasping with two linear springs. From this moment, the position of the cylinder is driven by the user. During the entire manipulation, the occlusions between real objects and virtual ones and among virtual objects are real-time computed, producing a realistic simulation. The results of the simulated scenario showed a very good management of occlusions (100% correctness) and an easy and efficient manipulation of the cylinder confirmed by all of the 20 test subjects.

The third example is similar to the second one and it was useful for assessing the performance of the system in the presence of more users (two in this case). Of course, each user needs his own calibration procedure, but the grabbing and manipulation of virtual cylinders can be performed

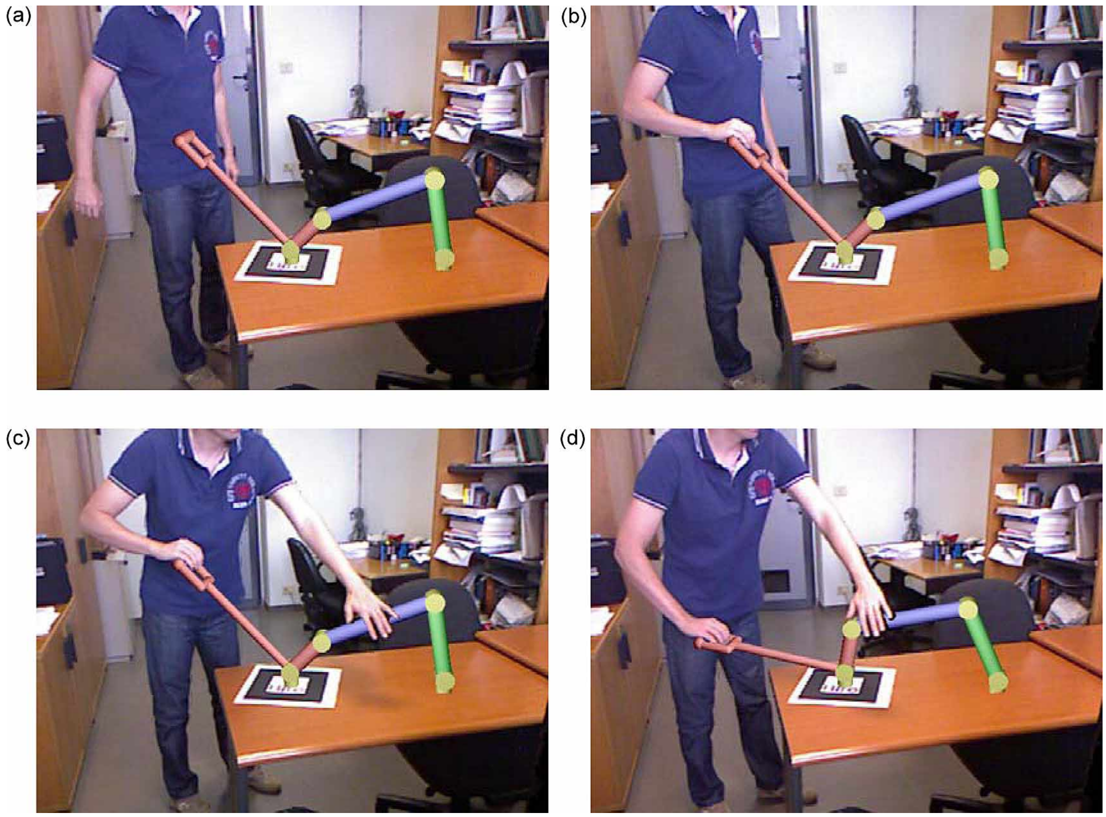

Figure 12. Snapshots of a natural manipulation and a dynamic simulation of a four-bar mechanism with a handle in an augmented scene.

simultaneously. The system is able to track each user independently. The tracking procedure was stressed by producing body overlapping and partial occlusions and the positive tracking of both users was more than 95%. Figure 11 shows three snapshots taken during the simulation. It is interesting to notice that all the possible occlusions are managed correctly (100% positive matching).

7.2 *An example of interactive simulation*

The fourth example concerns the simulation of a dynamic scene. The scene is populated with a four-bar linkage (three moving links and one fixed connected by four revolute joints). This mechanism is very common in industrial applications. One of the links is provided with a long bar and a handle with an Object Active Feature. After calibration, the user is able to grab it by placing his right hand near its geometrical centre. After the enforcing of the grasping constraint, the user's hand and the handle of the link are connected by a spring and they move accordingly. So, if the user pushes the handle downward, the first link rotates counterclockwise and the mechanism changes configuration, preserving its topology. It is interesting to notice that the error in the fulfilment of the constraint equations of the revolute joints is always lower than 0.5 mm and this is a very good performance for the precision in solving the equations of motion. Moreover, the management of the occlusions is performed throughout the simulation without errors (100% positive matching). Figure 12 shows three snapshots of the dynamic simulation.

8. Conclusion

In this paper, a methodology for implementing interactive simulations in an augmented reality environment has been presented. The interaction between the user and the scene is obtained by interpreting his pose without the use of any external tracking device. This leads to a natural interaction which can be very useful for realistic and immersive simulations.

The whole methodology is based on the use of a combined RGB/infra-red sensor which is able to compute a spatial map of the surrounding environment and track the user position. This equipment has been used for dealing with two different aspects. First of all, thanks to the capability in building an environmental mesh, it is possible to compute the three-dimensional position of each real object in the scene in order to check and manage possible occlusions between real and virtual objects. Moreover, by using image segmentation and body part recognition, it is possible to track main body landmarks of the user in the scene without applying any additional sensors.

This tracking methodology has been integrated with a constraint-based approach in order to implement a robust mathematical formulation for the manipulation of virtual objects in the scene (i.e. objects that can be picked and moved by the user).

All these features have been integrated with a dynamic solver based on the sequential impulse strategy. This combination allows us to perform both kinematic and dynamic simulation involving collision and impacts among bodies and to obtain an accurate and realistic behaviour.

Preliminary assessments demonstrate that the different aspects of the methodology are suitable for robust and reliable integration into an augmented reality environment and the required computational effort is compatible with real-time processing even in the presence of event-based simulations, long run-time and many users in the scene. The achieved precision in tracking, although not so high, is sufficient for correct implementation of interaction tasks and can be improved by using higher resolution sensors.

The proposed methodology is implemented using a low-cost architecture and the natural interface approach can be extended to more complex simulations involving other technical aspects such as structural mechanics, flexible body dynamics and interactive geometrical modelling.

Acknowledgements

The author wishes to acknowledge Simone Valentini for useful and motivating discussions.

References

Azuma, R.T., 1993. Tracking requirements for augmented reality. *Communications of the ACM*, **36** (7), 50–51.

Azuma, R.T., et al., 2001. Recent advances in augmented reality. *IEEE Computer Graphics*, **21** (6), 34–47.

Berliner, T., et al., 2010. Modelling of humanoid forms from depth maps. U.S. patent application US 2010/0034457 A1, 2010.

Bimber, O. and Raskar, R., 2005. *Spatial augmented reality: Merging real and virtual worlds*. Wellesley, MA, USA: A K Peters, Ltd.

Breen, D.E., et al., 1996. Interactive occlusion and automatic object placement for augmented reality. *Computer Graphics Forum (Proceedings of EUROGRAPHICS'96)*, **15** (3), 11–22.

Breen, D.E., Rose, E., and Whitaker, R.T., 1995. Interactive occlusion and collision of real and virtual objects in augmented reality. Technical report ECRC-95–02.

Buchmann, V., et al., 2004. FingARtips – Gesture based direct manipulation in augmented reality. *In:: Proceedings of Graphite '04: International conference on computer graphics and interactive techniques*, 2004, Singapore.

Carmigniani, J., et al., 2011. Augmented reality technologies, systems and applications. *Multimedia Tools and Applications*, **51**, 341–377.

Catmull, E. and Clark, J., 1978. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, **10** (6), 350–355.

Coutinho, M.G., 2001. *Dynamic simulation of multibody systems*. New York: Springer Verlag.

Fischer, J., Bartz, D., and Straßer, W., 2004. Occlusion handling for medical augmented reality using a volumetric phantom model. *In: Proceedings of ACM symposium on virtual reality software and technology (VRST)*, 2004, Hong Kong.

Fischer, J., Huhle, B., and Schilling, A., 2007. Using time-of-flight range data for occlusion handling in augmented reality. *Eurographics symposium on virtual environments (EGVE)*, 2007, Weimar.

Fortin, P. and Hebert, P., 2006. Handling occlusions in real-time augmented reality: Dealing with movable real and virtual objects. *In: Proceedings of the 3rd Canadian conference on computer and robot vision*, 2006, Quebec City, QC, Canada.

Freedman, B., et al., 2010. Depth mapping using projected patterns. U.S. patent application US 2010/0118123, 2010.

Haug, E.J., 1989. *Computer-aided kinematics and dynamics of mechanical systems*. Boston, MA, USA: Allyn and Bacon.

Hayashi, K., Kato, H., and Nishida, S., 2005. Occlusion detection of real objects using contour based stereo matching. *In: Proceedings of ICAT*, 2005, Christchurch, New Zealand.

Huagen,W., Shuming, G., and Qunsheng, P., 2004. Virtual grasping for virtual assembly tasks. *In: Proceedings of the third international conference on image and graphics (ICIG'04)*, 2004, Hong Kong.

Kanbara, M., et al., 2000. A stereoscopic video see-through augmented reality system based on real-time vision-based registration. *In: Proceedings of IEEE virtual reality conference*, 2000, New Brunswick, New Jersey, USA.

Kato, H. and Billinghurst, I., 2011. Inside ARToolKit, online paper available at http://www.hitl.washington.edu/artoolkit/Papers/ART02-Tutorial.pdf (accessed February 2011).

Kim, S. and Dey, A.K., 2010. AR interfacing with prototype 3D applications based on user-centered interactivity. *Computer-Aided Design*, **2**, 373–386.

Koller, D., et al., 1997. Real-time vision-based camera tracking for augmented reality applications. *In: Proceedings of the symposium on virtual reality software and technology (VRST-97)*, 1997, Lausanne, Switzerland.

Korkealaakso, P.M., et al., 2007. Development of a real-time simulation environment. *Multibody System Dynamics*, **17**, 177–194.

Liarokapis, F., 2007. An augmented reality interface for visualizing and interacting with virtual content. *Virtual Reality*, **11**, 23–43.

Mirtich, B.V., 1996. Impulse-based dynamic simulation of rigid body systems. Thesis (PhD). University of California, Berkeley.

Pang, Y., et al., 2006. Assembly feature design in an augmented reality environment. *Assembly Automation*, **26** (1), 34–43.

Park, J.-S., 2011. AR-Room: A rapid prototyping framework for augmented reality applications. *Multimedia Tools and Applications*, **55** (3), 725–746.

Piekarski, W. and Thomas, B.H., 2002. Using ARToolKit for 3D hand position tracking in mobile outdoor environments. *In: Proceedings of 1st international augmented reality toolkit workshop*, 2002, Darmstadt, Germany.

Schmitt, A. and Bender, J., 2005. Impulse-based dynamic simulation of multibody systems: Numerical comparison with standard methods. *In: Proceedings of Automation of Discrete Production Engineering*, 2005, Sozopol, Bulgaria.

Schmitt, A., Bender, J., and Prautzsch, H., 2005. On the convergence and correctness of impulse-based dynamic simulation. Internal report, 17, Institut für Betriebs und Dialogsysteme, Karlsruhe, Germany.

Tian, Y., Tao Guan, T. and Wang, C., 2010. Real-time occlusion handling in augmented reality based on an object tracking approach. *Sensors*, **10**, 2885–2900.

Vallino, J., 1998. *Interactive augmented reality.* Thesis (PhD). Department of Computer Science, University of Rochester, USA.

Valentini, P.P., 2009. Interactive virtual assembling in augmented reality. *International Journal on Interactive Design and Manufacturing*, **3**, 109–119.

Valentini, P.P., 2011. Interactive cable harnessing in augmented reality. *International Journal on Interactive Design and Manufacturing*, **5**, 45–53.

Valentini, P.P. and Mariti, L., 2011. Improving interactive multibody simulation using precise tracking and sequential impulse solver. *In: Proceedings of ECCOMAS multibody dynamics congress*, 2011, Brussels, Belgium.

Valentini, P.P. and Pezzuti, E., 2010. Interactive multibody simulation in augmented reality. *Journal of Theoretical and Applied Mechanics*, **48** (3), 733–750.

Valentini, P.P. and Pezzuti, E., 2011. Dynamic splines for interactive simulation of elastic beams in augmented reality. *In: Proceedings of IMPROVE 2011 international congress*, 15–17 June 2011, Venice, Italy.

Valentini, P.P., Pezzuti, E., and Gattamelata, D., 2010. Virtual engineering in augmented reality. *In: Computer animation*, New York: Nova Science, 57–83.

Wolka, M.M. and Anderson, B.G. 1995. Resolving occlusion in augmented reality. *In: Proceedings of the ACM I3D symposium on interactive 3D graphics*, 1995, Monterey, CA, USA.

Yamauchi, M. and Iwamoto, K., 2010. Combination of optical shape measurement and augmented reality for task support: Accuracy of position and pose detection by ARToolKit. *Optical Review*, **17** (3), 263–268.

Zollmann, S., et al., 2010. Image-based ghostings for single layer occlusions in augmented reality. *In: Proceedings of the international symposium on mixed and augmented reality*, 2010, Seoul, South Korea.