SURGICAL VIRTUAL REALITY – OPTIMIZATION OF COLLISION DETECTION

Dan CUSTURĂ-CRĂCIUN[1], Daniel COCHIOR[2], Corneliu NEAGU[3]

Just like flight and spaceship simulators already represent a standard, we expect that soon enough, surgical simulators should become a standard in medical applications. A simulations quality is strongly related to the image quality as well as the degree of realism of the simulation. Increased quality requires increased resolution, increased representation speed but more important, a larger amount of mathematical equations. A simulator executes one of the most complex sets of calculations each time it detects a contact between the virtual objects, therefore optimization of collision detection is fatal for the work-speed of a simulator and hence in its quality.

Keywords: surgical virtual reality, collision detection optimization, representation quality

1. Introduction

Traditionally, applying new surgical techniques involves certain interaction between the surgeon and his patient, more specifically his tissues (the barrier between the surgeon and his patient being represented at most by a surgical instrument) and gained handiness can be improved only by amplifying perceptions (by optical devices). The fusion between calculation technique, robotics, haptic devices, telecommunications and virtual reality (VR) makes possible a whole series of events such as: real-time virtual surgical intervention, allowing the surgeon to study various surgical procedures and to „replay" certain surgical-states or to complete his surgical training. Furthermore, VR makes „distance-surgery" accessible as well as providing robot-aid to the main surgeon during his surgical intervention. The first steps towards achieving this were already taken, meaning that surgical interventions were executed from outside the operating theaters, from another hospital or from another continent [1]. Using virtual reality, by digitally transposing the surgical procedures, shall bring a

[1] PhD candidate. General Manager at Best Soft Expert SRL from Bucharest Romania, email : custura_dan@yahoo.com

[2] MD, PhD, Principal Investigator I, „Titu Maiorescu" University, Department of Surgical Disciplines, Faculty of Medicine, Surgical Department Hospital CF 2, Bucharest, Romania

[3] Professor, PhD, Department of Manufacturing Technology, POLITEHNICA University of Bucharest, Romania, e-mail: neagu_corneliu@yahoo.com

revolutionary upgrade to surgery. It opens new possibilities of applying much more effective training methods; it allows experiment of new surgical techniques as well as it improves the surgeon's skills and abilities. During the past decade, we noticed a significant increase in the interest towards simulators development based on VR. In attempt to better understanding and increasing VR based simulators quality, research in subjects like virtual 3D geometry and computer graphics was intensely sustained and supported [2].

2. Collisions; Algorithms

For a good representation of virtual reality, we refer to the role of the geometric model which is to stock geometric properties of bodies in the virtual environment[3]. The geometric role model completed with the dynamic model will describe a body's or a group of bodies' behavior in movement tending to an equilibrium position inside a virtual environment[4,5,6]. Intervention coming from a human operator towards an object inside virtual space and usage of that object (instrument) in order to move, touch, deform or cut another object from the virtual space is based on collision detection, collision localization determination, the normal's and the reaction's local force determination[7,8,9].

In virtual reality, collision detection has a primordial role, being the base of virtual object-interaction representation process.

Hypothesis: Note an environment with n objects (at least 2) of which at least one is moving. (Fig. 1)

Problem: Will the objects collide? If yes, what forces will they generate?

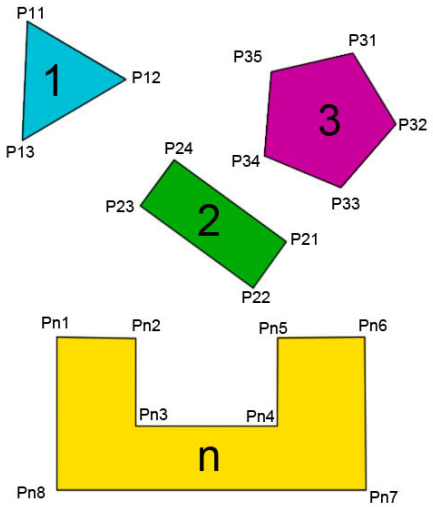

Fig. 1 Virtual environment with various objects

Solution:

To be started with object number 1, formed by points P1: we test if point P11, describing object 1, has become part of object 2, formed by P2 points and so on, so forth for point P12 and for all "n" objects[10].

To be noted that applying the upper algorithm, we should execute the following number of equations for the first object (number of objects "n" = 4):

$$P1 \times P2 + P1 \times P3 + P1 \times P4 = 3 \times 4 + 3 \times 5 + 3 \times 8 = 51 \quad (1)$$

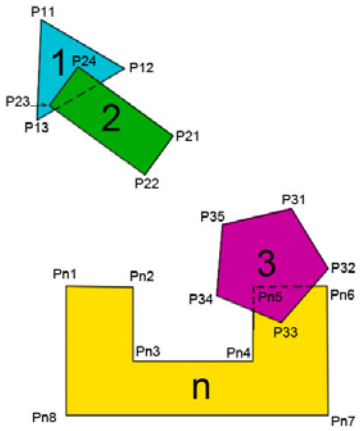

Fig 2 Collision of n objects

For a random number of objects (n) we can note:

$$pos1 = \sum_{k=2}^{n} (p_1 * p_k) \quad (2)$$

Where pos1 is the number of calculations to be executed to test collision for object 1, P1 is meaning the number of point of the first object, pK being the number of points of the current object, K being the number of the current object and n being the total number of objects.

For all objects we have:

$$pos = \sum_{m=1}^{n} (\sum_{k=2}^{n} (p_m * p_k)) \quad (3)$$

Where "pos" is the number of calculations to be executed in order to test collision, pm being the number of point of the first current object, pk being the number of points of the current object, m being the foreign object's number on which collision we test, k meaning the current object's number and n meaning the total number of objects.

For a number of 5 objects, each containing a number of 3 000 points, pos would be:

$$pos = 3000x3000x5x5 = 2.25 \times 10^8 \quad (4)$$

A computer having a 3GHz speed processor would execute these operations in 75 seconds. Unfortunately, other calculations are necessary and furthermore, the execution step should be faster than 25Hz = 1/25 seconds for an acceptable image quality. It is obvious that an optimization is required.

How is a collision test done?

In two dimensions, if object one is a square, object 2 is also a square and the objects' edges would be parallel with the axes, then collision cases are described in Fig. 3.

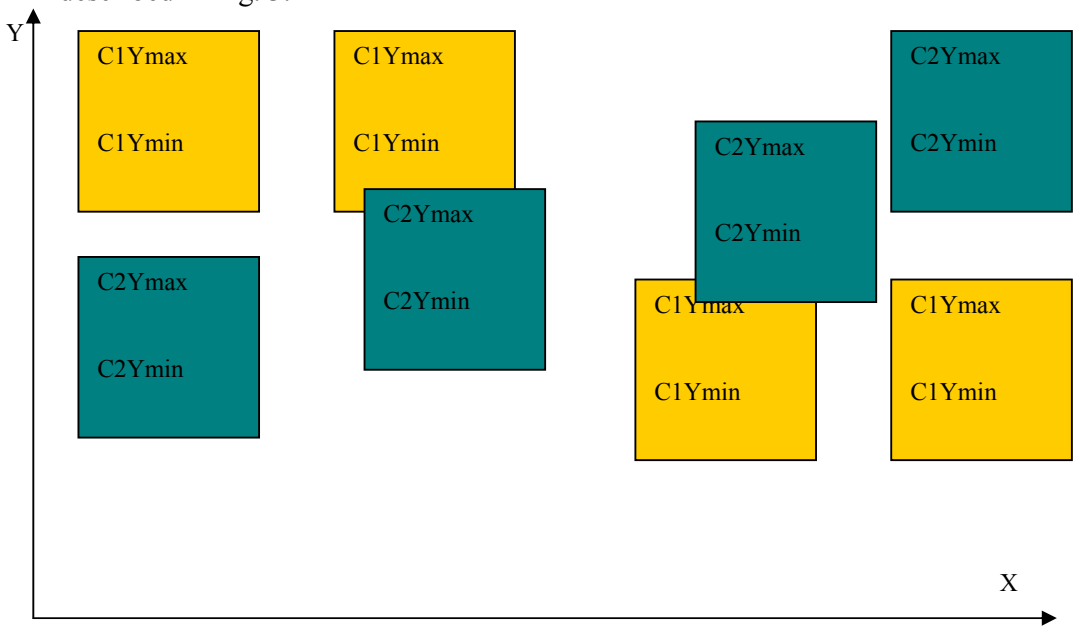

Fig 3 Collision cases for 2D objects

Collision test requires that the following relationships to be true, in the hypothesis used beforehand:

$$C1Ymin < C2Ymax \text{ and } C1Ymax > C2Ymin \quad (5)$$

Similarly, in 3D we have:

$$C1Xmin < C2Xmax \text{ and } C1Xmax > C2Xmin \quad (6)$$

And

$$C1Ymin < C2Ymax \text{ and } C1Ymax > C2Ymin$$

And

$$C1Zmin < C2Zmax \text{ and } C1Zmax > C2Zmin$$

Where C1Ymin is the smallest Y from all Y's values of object P1, respectively x, z. The presented test is named AABBs (axes aligned bounding boxes) [11]

In terms of volume, calculations must be performed by the following steps:

- Determination of Xmax for object 1
- determination Xmin for object 1
- determination Ymax for object 1
- determination Ymin for object 1
- determination Zmax for object 1
- determination Zmin for object 1
- ... determination X,Y,Z max and min for all the other objects
- Testing (6) on each pair of objects
- Choosing which pair of objects are in collision
- Contact surface/spot determination (GJK)
- Applied forces determination
- Distributed forces determination
- reaction forces determination
- deformation determination
- force feedback determination
- 3D environment representation
- 2D screen representation of visible parts in the environment

The presented case in the previous hypothesis is rather restrictive. In Fig. 4 we present the collision case of triangles.

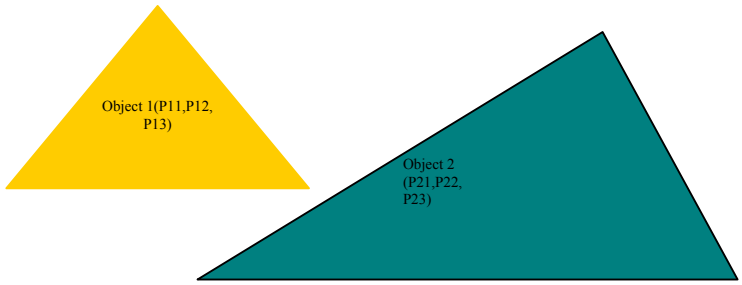

Fig 4 Collision cases with 2D triangles

To be noticed that applying algorithm AABBs, in the objects case presented in Fig. 4, the collision test is positive even though the objects do not touch.

Practically, as the objects approach each other, we should calculate the distance between them. When distance tends to 0, we could declare it a collision. One of the most efficient algorithms is the **Gilbert-Johnson-Keerthi** (GJK) algorithm[3,14].

This algorithm calculates the distance between a random convex object and an exterior point iteratively. Moreover, it can specify which of the object's apex, edge or face is the closest to a certain point. Even though the algorithm is

highly efficient, there is a probability that the solution to be found after 3 to 8 steps, each step requiring a series of calculations to be made.
Extrapolating, to tell if two convex objects are colliding, we should apply GJK algorithm to each object and moreover, to every point of the other objects.

To get closer to the real case, we observe that not all objects to be represented are convex. In Fig. 5 we present a case of collision between a concave object and a convex one.

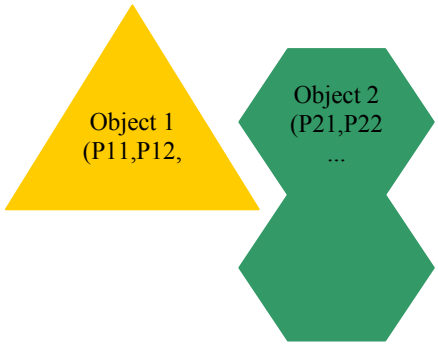

Fig 5 Collision case in 2D objects (concave-convex)

To properly approach this case, a new step should be introduced trough the algorithm, in which concave objects should be divided in subparts (smaller, new, convex objects).

During the simulation of a laparoscopic surgical intervention, in a virtual environment, the number of simulated objects may reach tens. For an acceptable simulation quality, the number of points to describe an object should reach thousands for simple objects and millions for more complex objects. Obviously, in terms of software, concave objects should be divided to subparts (smaller, convex objects).

In view of the situation presented above, it is self-evident that one or more optimizing solutions are required to reduce the number of calculations.
One of the most used algorithms of optimizations is ***Octree***. [12]

3. Octree

Octree is a data organizing concept according to which each node or knot, has at most 8 other sub-nodes (children) (Fig. 6).

Visually, if we divide a cube with 3 perpendicular planes at a distance of a/2, where "a" is the main cube's edge, 8 cubes with an "a/2" edge will be obtained. The larger cube is the "parent" of the smaller cubes (children). Dividing cubes will proceed to a certain level (resolution), to a certain eps accuracy (a_i< eps) or to attaining all given points.

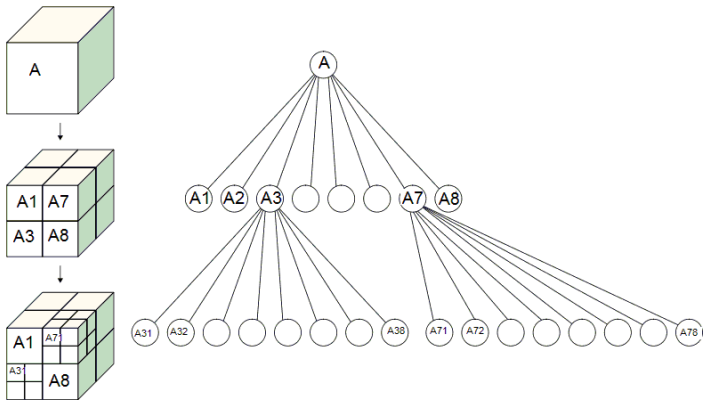

Fig 6 Building an octree from root A

Among the established terms and notions we note:

- root = cube with no further parents
- children, branches = subparts of a cube
- parent = the cube which was divided into subparts
- ancestors = the set of cubes which contain a certain subpart
- leaves = the set of subparts which have no other sub-subparts (which do not divide and have no other branches) [12]

In case of a random object, classic build of an octree starts from the extensive cube (the root) and continues sequentially on each level of branch, assigning points of the "to-be-represented" object to each subpart and thus to the whole structure. To be noticed that the structure may lack cubes if they do not contain any of the points which describe the object. The result will be represented by a list of nodes (points that describe the object) indexed based on a hierarchy.

This type of hierarchy is useful starting with collision detection and ending with deformation calculation (FEM) [4], reaction forces determination, etc. Compared to the AABBs algorithm presented in 3 figure, according to the collision detection algorithm of two objects described by points, one object is stationary (target) and the other is movable (the instrument), implies:

- choosing a point describing the instrument
- testing if that certain point is part of the target's root
- if the point is part of the root, the branches of the root are tested to determine which specific branch contains the point
- the process iterates to the leaf level, specifying exactly where the collision had place

Assuming a target structure such as in Fig. 6 and a device with a P1 describing cube A31, then the following tests shall be performed iteratively according to the algorithm above (Fig. 7):

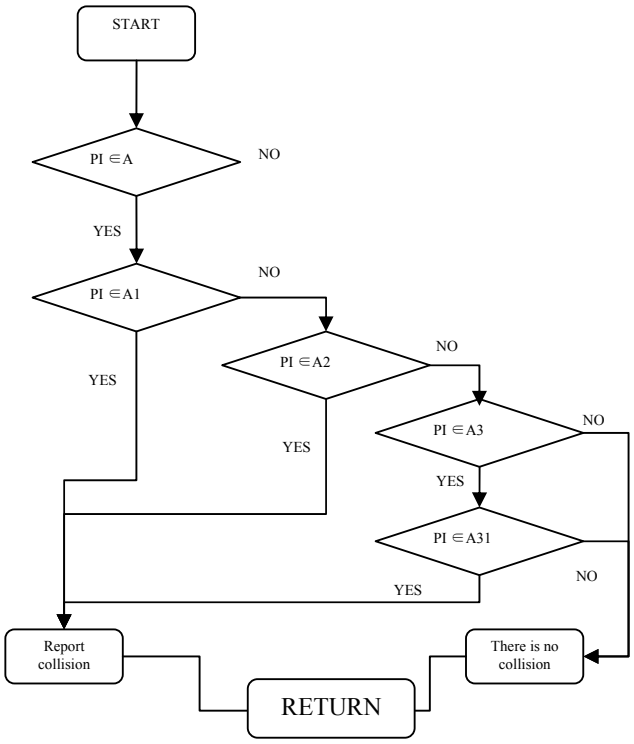

Fig 7 Algorithm AABBs based on octree

The analysis algorithm above notes that the tests for branches A4 to A8 were removed because branch A3 was found valid to the test; tests for branches A32 to A38 were also ruled out because branch A31 was found valid and even more important, testes for children of branches A4 to A8 were also removed.

To be noticed that, naturally, this algorithm is applied to every point of the studied object. Particular, using octree in order to describe objects of surgical virtual reality, the following should be noted:

In larger objects cases, objects described by thousands to millions of points which move under the influence of physic parameters or which deform significantly under the influence of mechanical parameters, the octrees should be recalculated after each movement or deformation.

Reconstruction of octrees implies, obviously, an increasing amount of calculations. In case of instruments described by hundreds to thousands of points, applying an algorithm to detect collision between a large object and that specific instrument implies, actually, applying the algorithm for each point describing the instrument.

4. Double - octree

Double-octree algorithm consists in applying the collision detection algorithm between object A (target) and object B (instrument) not by testing progressively object A's leaves compared to all points of object B, but compared progressively with object B's octree (Fig. 8).

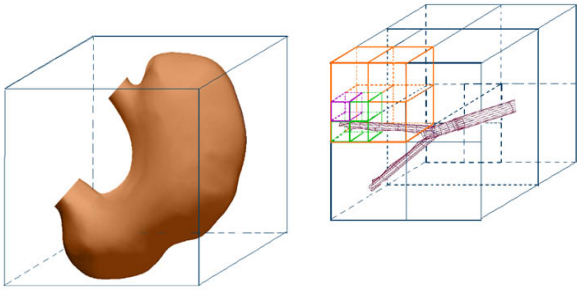

Fig 8 Collision detection

Moreover, in laparoscopic virtual reality, instruments have a specific property: "They do not actually fit inside a cube!" because in most of cases they are long and thin. The structure's tree building principle in this case is not spatial distribution, but position along the axis of the instrument. Therefore, for instruments it is suggested a structure formed by a set of root-cubes (slices) which divide the instrument into a number of parts of the order of tens (Fig. 9).

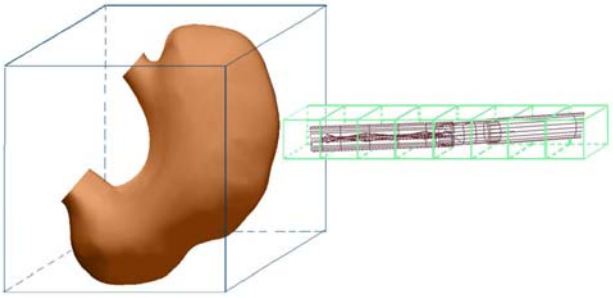

Fig 9 Collision detection with an instrument formed by a set of root-cubes

Indexing root-cubes that divide an instrument into slices will be done according to the most likely place of collision (from apex to handle). Assigning object points to the slices, being a characteristic of that instrument, will be executed during the programming. Each octree's edge values are, obviously, different after the instrument has been moved (rotated, translated). But if

assigning points has already been done, recalculating Xmin, Xmax, Ymin, Ymax, Zmin and Zmax only for the cube used for the test, will reduce the required time to fulfill these calculations because the other slices will not be tested. The efficiency of the algorithm will significantly increase, ruling out 90% of the previously executed calculations for instruments with at least 10 root-cubes and 10.000 points.

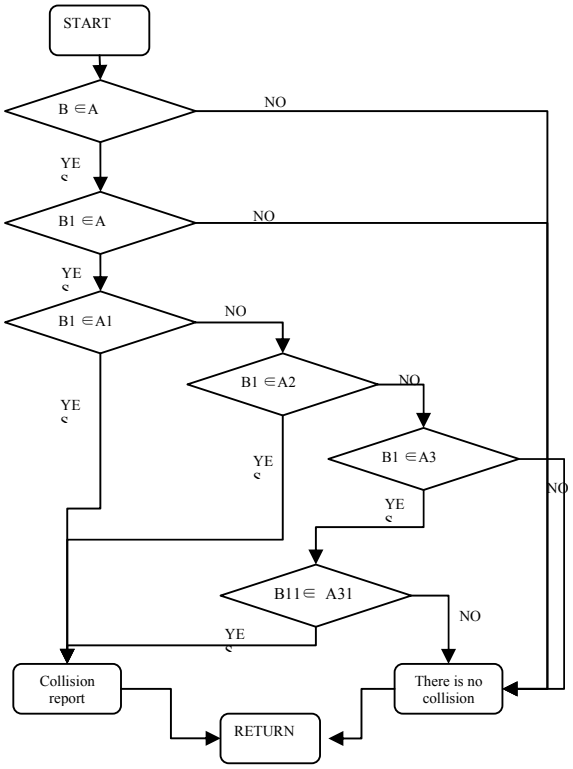

Fig 10 Double-octree algorithm

Thereby if we apply the double octree algorithm upon a virtual environment with 5 objects, each containing 3 000 points, of which a tool-type object is in collision with an organ-type object, it is concluded that test [6] is to be applied only 5 times if there was no collision!

In Fig. 6, the level inside of the octree is S = 3. Actually S depends on the size of the object and it is calculated so as the representation of a cube from the last level (the leaves level) to be less than half the size of a pixel. Usually, a pixel is 0,3 mm high; thus, an object of maximum scale D = 10 cm requires a number of S levels for an exact graphic representation. S can be calculated based on the following formula:

$$s \geq \log_2\left(\frac{2 \times D}{h}\right) \tag{7}$$

Where "s" is the number of octree levels (s, obviously is an integer). D is the maximum scale of the object and h is the height of a pixel. For the previously presented case, S = 10. The number of equations to be calculates is estimated around 3S+5 times, namely about 35 times. The version lacking optimization (see equation 4) reaches 2.25 x 10^8 calculations to be completed. Therefore we can conclude that the algorithm shortens the calculation process ($2.25x10^8$)/35 = $6x10^6$ times, namely 6 million times.

5. Conclusions

Living in a society in which fighting disease is on a daily agenda, having good, well trained doctors, is a necessity not related to comfort but related to hope of life and life quality. To achieve this, a laparoscopic surgical simulator based on virtual reality is a step forward, it is a priority in Romania.

Since 2003, together with several collaborators, during project SIVILAP and subsequently project EVAMED, a laparoscopic surgical simulator based on virtual reality was brought to life. The simulator was completed and recently optimized based on the previously described algorithm obtaining a radical performance increase caused by collision detection, which is about 6 million times faster than the unoptimized version and 3 times faster than the simple octree version.

Dual-octree algorithm can be applied not only in informatics but as well as in any scientific field working with hierarchical data (such as in statistics, etc).

R E F E R E N C E S

[1] *D. Cochior, D. Custură-Crăciun, C. Ciobanu-Oprea*."Surgical virtual reality - usefull or needed ?" in romanian „Realitatea virtuală chirurgicală – utilă sau necesară?". Editura Electra, 167 p. ISBN 973-7728-24-6, 2005. Monograph

[2] *A. El Saddik, M. Orozco, M. Eid, J. Cha.* Haptic technologies bringing touch to multimedia. Hardcover, 218p. ISBN: 978-3-642-22657-1, 2011. Monograph

[3] *Zhou, Y., Chen, W., and Tang, Z.,* An Elaborate Ambiguity Detection Method for Constructing Isosurfaces within Tetrahedral Meshes. Computers & Graphics, 1995. 19(3): p. 355-364.

[4] *Miller, K., Joldes, G., Lance, D., et al.,* Total Lagrangian Explicit Dynamics Finite Element Algorithm for Computing Soft Tissue Deformation. Communications in Numerical Methods in Engineering, 2007. 23(2): p. 121–134.

[5] *Masutani, Y., Inoue, Y., Ishii, K., et al.* Development of Surgical Simulator Based on FEM and Deformable Volume-Rendering. in Medical Imaging 2004. 2004. San Diego, CA: The International Society for Optical Engineering.

[6] *James, D.L. and Pai*, D.K. ArtDefo: Accurate Real Time Deformable Objects. in The 26th Annual Conference on Computer Graphics and Interactive Techniques. 1999. Los Angeles, CA: ACM.
[7] *Ericson, C.,* Real-time Collision Detection. 2005: Morgan Kaufmann.
[8] *Cavusoglu, M.C., Goktekin, T.G., Tendick, F., et al.,* GiPSi: An Open Source/Open Architecture Software Development Framework for Surgical Simulation, in Medicine Meets Virtual Reality 12. 2004: Newport Beach, CA. p. 46–48.
[9] *Berkley, J., Turkiyyah, G., Berg, D., et al.,* Real-Time Finite Element Modeling for Surgery Simulation: An Application to Virtual Suturing. IEEE Transactions on Visualization and Computer Graphics, 2004. 10(3): p. 314-325.
[10] *Zachmann Gabriel,* Virtual Reality in Assembly Simulation —Collision Detection, Simulation Algorithms and Interaction Techniques, Phd Thesis, Dem Fachbereich Informatik der Technischen Universität Darmstadt, 2000
[11] *Coumans E.,* „Collision Detection and Rigid Body Dynamics", http://graphics.stanford, 2010
[12] *Meagher D.,* "Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer" , Rensselaer Polytechnic Institute (Technical Report IPL-TR-80-111), 1980
[13] *Custură-Crăciun D., Cochior D., Tecuceanu V., Ciobanu-Oprea C.,* SIVILAP- Simulator de chirurgie laparoscopică (hardware, software) (PNCDI): Program National INFOSOC SIVILAP ; 2004-2006
[14] *Gilbert, E., Johnson, D., and Keerthi,* S., A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space. IEEE Journal on Robotics and Automation, 1988. 4(2): p. 193-203.