Collision Detection of Functionally Defined Objects for Constant Time

Sergei I. Vyatkin, Boris S. Dolgovesov
Institute of Automation and Electrometry, SB RAS, Novosibirsk, Russia
sivser@mail.ru, bsd@iae.nsk.su.

Abstract

The problem of collision detection of functionally defined objects based on perturbation functions for constant time is considered. The collision detection algorithm of different complexity during a constant time is discussed. Recursive object space subdivision algorithm is proposed. In the perturbation function representation, collision detection becomes trivial due to the inside/outside property of the functionally defined surface.

Keywords: *Collision Detection, Perturbation Functions, Recursive Object Space Subdivision.*

1. INTRODUCTION

The geometric concept of virtual environment modeling using *F*-rep can be described as an algebraic system [1]

$$(M, \Phi, W), \quad (1)$$

where M is the set of geometric objects, Φ is the set of geometric operations, and W is the set of relations on the set of objects. Geometric objects are considered as closed subsets of n-dimensional Euclidean space E^n with the definition:

$$f(x_1,x_2,\ldots,x_n) \geq 0, \quad (2)$$

where f is a real continuous function defined on E^n. In work [1] authors call f a defining function. The inequality (2) is called a function representation (or *F*-rep) of a geometric object.

An example of the relations is collision detection for objects. The binary relation is a set of the set $M^2=M\times M$. It may be defined as

$$S_i: M\times M \rightarrow I \quad (3)$$

Collision detection is a complicated problem solved in various computer programs [2]. This means that for each animation frame, one should test whether any two or more objects collided (see Figure 1).

Figure 1: Collision detection of functionally defined objects based on perturbation functions.

The ideal case is collision detection of any complexity between two arbitrary objects in the minimal time. Since the control of collisions between all pairs of objects is a resource-consuming process, such tests are usually done only for part of objects. The detection algorithm can be simplified prior to testing the presence of the given point (belonging to one of the objects), e.g., inside the cube confining the second object. The problem of simulating the behavior of interacting bodies having irregular shape arises in some applications such as dynamics of body collisions and celestial mechanics, molecular dynamics, graphics simulations for the problem of nano-assembly automation and its application in medicine using collective robotics [3], computer games and haptic interactions.

Figure 2: Collision of two objects: the worst case.

Particularly in calculating motions of many objects that move under changing constraints and frequently make collisions, one of the key issues of dynamic simulation methods is calculation of collision impulse between rigid bodies [4]. A fast algorithm for calculating contact force with friction by formulating the relation between force and relative acceleration as a linear complementary problem was equally demonstrated and this model was based on solving the linear complementary problem [5]. Baraff's algorithm has achieved great performance for real-time and interactive simulation of two-dimensional mechanisms with contact force, friction force and collision impulse, although friction impulse at collision was not completely covered in such a model. In geometric haptic rendering models, collision detection is not trivial to compute. One of the most popular collision detection algorithms in geometric haptic rendering is H-Collide [6]. It uses a hybrid hierarchy of spatial subdivision and OBB trees. The simplest algorithms for collision detection are based upon using bounding volumes and spatial decomposition techniques. Examples of bounding volumes include bounding

spheres, bounding boxes, convex polyhedrons. Examples of bounding boxes include axis-aligned bounding boxes and oriented bounding boxes. In work [7] authors used a bounding sphere hierarchy to detect collisions. However such approaches cannot define collision of objects represented on figure 2. Spatial decomposition techniques based on subdivision are used to solve the interference problem. Recursive subdivision is robust but computationally expensive. In particular, Hahn [8] used a subdivision based collision detection algorithm.

For curved objects, Herzen and etc. [9] have described an algorithm based on subdivision technique. A similar method using interval arithmetic and subdivision has been presented for collision detection by Duff [10]. However, for commonly used spline patches computing and representing the implicit representations is computationally expensive [11]. In [12], Pentland and Williams used implicit functions to represent shape and the property of the "inside-outside" functions for collision detection. But this algorithm has a drawback in terms of robustness, as it uses point samples. Thus, the most popular collision detection algorithms are extremal distance and extremal points (Barraf: four nonlinear equations solving), testing sample points (Pentland, Gascuel: accuracy of sampling using huge memory), interval methods (Duff, Snyder: interval bounds on the output of functions with time-consuming). The detailed explanation of main problems is described in [10]. The main problems are procedurally defined functions, time-dependent surfaces and surfaces of high complexity. Surgery simulation and entertainment technology require fast deformable models and efficient collision handling techniques. Efficient collision detection algorithms are accelerated by spatial data structures, bounding volume hierarchies, distance fields, etc. Such data structures are commonly built in a pre-processing stage. But pre-processed data structures are less efficient for deforming objects. Collision detection for deformable objects introduces additional challenging problems [13].

As a result of work of the known algorithm of collision detection of functionally defined object [14], the collision is not always detected and, moreover, detection of different collisions requires a greatly different time.

The goal of this paper is to prove that using the proposed algorithm the object collision is detected in a constant and less time for collisions of different complexity, and the detection of events is absolutely ensured.

2. PREVIOUS WORK

A functionally defined object is completely defined by means of the real-valued describing function of three variables (x_1, x_2, x_3) in the form of $F(X) \geq 0$, then the objects are treated as closed subsets of the Euclidean space E^n, defined by the describing function $F(X) \geq 0$, where F is the continuous real-valued function and X= (x_1, x_2, x_3) is the point in E^n, defined by the coordinate variables. Here $F(X) > 0$ defines points inside the object, $F(X) = 0$ defines points on the boundary, and $F(X) < 0$ defines points that lie outside and do not belong to the object. Hence, applying the Boolean intersection operation and calculating the first point belonging to the intersection, one will easily detect whether the objects collided or not.

The collision detection algorithm described in [14] is based on the relation of object intersection and uses the Sobol's quasirandom sequences and the spiral quadratic search for detecting nonnegative values of the function defining the intersection. In so doing, body, confining spheres are used to define the region of search.

We will consider the known algorithm of collision detection between functionally defined objects [14]. Let the objects G_1 and G_2 be defined as $f_1(X) \geq 0$ and $f_2(X) \geq 0$. The intersection (interference, collision) relation is defined by the bivalued predicate:

$$S_C(G_1,G_2) = \begin{cases} 0, if\ G_1 \cap G_2 = \varnothing \\ 1, if\ G_1 \cap G_2 \neq \varnothing \end{cases} \quad (4)$$

G_1: $f_1(\boldsymbol{X}) \geq 0_1$

G_2: $f_2(\boldsymbol{X}) \geq 0$. Is intersection empty?

A function $f_3(X)= f_1(X)\&f_2(X)$ defining the result of the intersection can be used to evaluate Sc (3). It can be stated that $Sc=0$ if $f_3(X)<0$ for any point of the space E^n. In so going, spheres confining the bodies are used to define the region of search. The main disadvantage of this algorithm is that the time of functionally defined object collision detection depends greatly on the relative position of collided objects and parts of their surfaces. This algorithm not always can detect collisions of objects. We will consider the algorithm in more detail.

1. Define the admissible domain D for two given objects.
   Admissible domain D:
1) bounding boxes are projected onto three coordinate planes;
2) projections of intersections are detected in each plane;
3) rectangular domain is detected in the space.

2. Find the argument p*, where $f(p^*)=\max(f(p))$ in D (if $f(p^*)<0$, then collision is not detected, and if $f(p^*)\geq 0$, then p* returns the collision point coordinates).

Search for an external point:

1) quasirandom LPτ points (Sobol's sequences are generated):
- the points are placed randomly in the nodes of a rectangular grid.
- N points guarantee $N^{-1/3}$ accuracy of point detection;

2) start of search by a quadratic spiral from the random point:
- succeeds in one-dimensional quadratic searches,
- a quadratic interpolant by three uniformly space points;
- it is required that the function $f(p)$ is of C^1continuity;

3) stop trials in the following cases:
- zero or a positive value of the function $f(p)$ is found,
- the assigned number N of trials is exceeded,
- argument p* is found with the given accuracy.

Example

Deformable noisy spheres

$f(x,y,z,t)=R^2-x^2-y^2-z^2+noise(x,y,z)$,

where x=x(t), y=y(t), z=z(t), and noise is the so-called solid noise function (Gardner's series).

The worst case is when the objects have no collision point:

1.07 seconds for 1000 random points.

The event of collision: 0.07 seconds for 106 trials.

We should note that the time tests were done on a parallel multiprocessor computer system (OCCAM-2) based on transputers T805.

This algorithm does not always result in collision detection, i.e., the algorithm does not ensure detection of event as stated by authors [14]. Moreover, a drastically different time is required for different collisions.

3. COLLISION DETECTION FOR FUNCTIONALLY DEFINED OBJECTS BASED ON RECURSIVE OBJECT SPACE SUBDIVISION

We propose another way of collision detection without using any bounding volumes around each object and pre-processing stage. This way of collision detection is based on the relation of object intersections, function representation with perturbation functions [15-17] and on the recursive object space subdivision [18, 19] for search the contact point of the objects.

3.1 Set of geometric objects

It is proposed to describe complex geometric objects by defining the function of deviation (perturbation function of the second order) from the basic triangles [17], planes and quadrics [15-17]. So, we proposed to describe geometric objects by defining the perturbation function from the basic quadric in the form:

$$F(x,y,z) = A11x^2 + A22y^2 + A33z^2 + A12xy + A13\,xz + A23yz + A14x + A24y + A34z + A44 \geq 0 \quad (5)$$

In particular, the freeform is a composition of the basic quadric and the perturbation $F'(x,y,z) = F(x,y,z) + R(x,y,z)$, where the perturbation function $R(x,y,z)$ is found as follows:

$$R(x,y,z,)=Q^2(x,y,z) \text{ if } Q(x,y,z)>0 \quad (6)$$

0 otherwise,

where Q is the perturbing quadric. A perturbed quadric (freeform) can be also considered as Q. In other words, the composition of the basic quadric and the deviation function is a new perturbation function, i.e., a derivative for another basic quadric. Since $\max[Q + R] \leq \max[Q] + \max[R]$, this means that to estimate the maximum Q on some interval, we must calculate the maximum perturbation function on the same interval. The surface obtained will be smooth (see Figure 3), and a small number of perturbation functions will be necessary to create complex surface forms.

Figure 3: Functionally defined objects based on perturbation functions.

3.2 Set-theoretic operations

Let the objects G_1 and G_2 be defined as $f_1(X) \geq 0$ and $f_2(X) \geq 0$. The binary operation of the objects G_1 and G_2 means operation $G_3=\Phi_i(G_1,G_2)$ with the definition

$$f_3=\psi(f_1(X),f_2(X)) \geq 0, \quad (7)$$

where ψ is the continuous real function of two variables.

The geometric model should allow designing of objects and their compositions of infinite complexity. This is primarily achieved by means of Boolean operations of uniting $A \cup B$ or (A+B) and intersection $A \cap B$ or (AB) (see Figure 4).

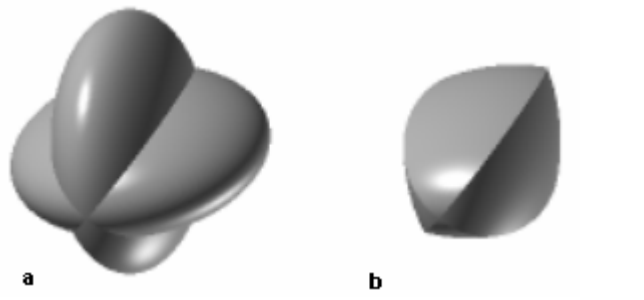

Figure 4: Set-theoretic operations on objects: (a) union, and (b) intersection.

The whole scene is a kind of a tree. Each node of the tree is an object constructor performing logical operations its descendants, and vertices of the tree are primitives. When the object constructor is queried while rendering, the object addresses its descendants, transforms the obtained results, and gives the answer to the query. The descendant may be a primitive or another object constructor. While applying the geometric operations, rotations, displacements, and scaling to the object constructor it performs all these operations on its descendants, and in addition changes its Boolean function in the case of inversion.

3.3 Recursive object space subdivision based collision detection algorithm

After calculation of the intersection, i.e., application of the Boolean operation of intersection, the search for the contact point of collided objects is done by means of recursive subdivision of the object (model) space. Hence, it is sufficient to find at least one point (or more) belonging to intersection.

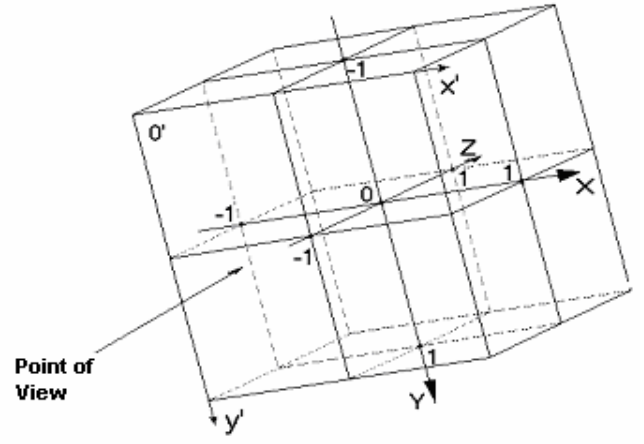

Figure 5: The model coordinate system (M) in which the space inside the cube is subdivided.

We consider the object-intersection $A \cap B$ ($A = F_3(x,y,z) + R_3(x,y,z)$; $B = F_2(x,y,z) + R_2(x,y,z)$) (see Figure 4 (b)) that has the property of answering the request on intersection with a rectangular parallelepiped or a bar. The negative answer guarantees that the object-intersection is not intersected and has no common points belonging to the intersection is done by recursive subdivision of the space inside the cube defined by boundaries of ±1 along each coordinate (see Figure5).

The center of the cube matches the origin of the model coordinate system M whereas the plane Z= –1 coincides with the screen plane. At the first step of recursion, the initial cube is subdivided into four smaller subcube in the screen plane. At the stage of subdivision of space along the quaternary tree, 2-times compression and transfer by ±1 along two coordinates are performed. Assume, that domain of point search is a cube in which embed our object-intersection.

Then recursive subdivision of the domain applied: domain cuts by 2 planes, that perpendicular to the screen plane XY, into 4 bars. For each bar intersection test are executed. If the object intersects with given bar, then bar subdivides further. Otherwise, we exclude bar from subdivision. This corresponds with exclusion of the square areas in the screen, on which given bar (and therefore, object- intersection) are projected (see Figure 6). Intersection test for base quadric is performed as follows.

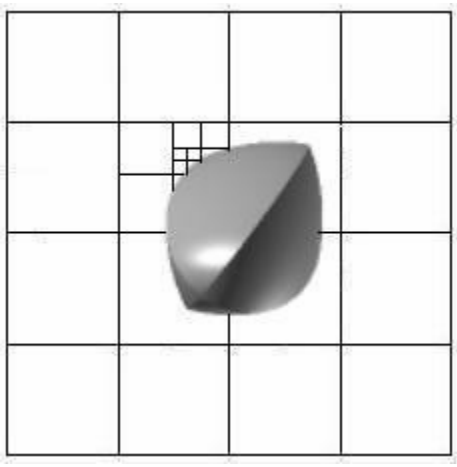

Figure 6: The screen coordinate system (P) and quaternary subdivision.

If in the equation of quadric Q(x, y, z) = 0 (4) the values of the variables x, y, z vary within the length [-1, 1], then

$$\max[\ |Q(x, y, z) - A44|\ ] \le \max F =$$

$$|A11|+|A22|+|A33|+|A12|+|A13|+|A23|+|A14|+|A24|+|A34| \quad (8)$$

We should note that if $|A44| \le \max[\ |Q(x, y, z) - A44|\ ] \le \max F$, then, probably, a point $M_0 = (x_0, y_0, z_0)$ ($-1 < x_0, y_0, z_0 < 1$) exists such that $Q(x_0, y_0, z_0) = 0$. If $\max F < |A44|$, then such points do not knowingly exist, and the sign of the coefficient A44 distinguishes location of the bar inside or outside with respect to the quadric surface Q=0 (if $A44 \ge 0$, then the subbar is inside the quadric). Using results of this test, we perform subdivision of subbars that fall within the quadric completely or, probably, partially, and the knowingly external subbars are eliminated from processing. A test for intersection of subbars with freeforms is somewhat different. For the basic quadric the test for intersection looks as follows:

if ((A44+R) < 0) && (|A11|+|A22|+|A33|+|A12|+|A13|+|A23|+|A14|+|A24|+|A34| < -(A44+R)) then the subbar is outside. (9)

Here R is the maximum perturbation function on the current interval; Aij - are the coefficients of quadratic function. The following test is performed for the perturbation function:

if (|A11|+|A22|+|A33|+|A12|+|A13|+|A23|+|A14|+|A24|+|A34| < |A44|) then the subbar is outside of the range of definition of perturbation, (10)

where Aij - are the coefficients of the quadratic perturbation function, and a value of R is additionally calculated and added to the basic function.

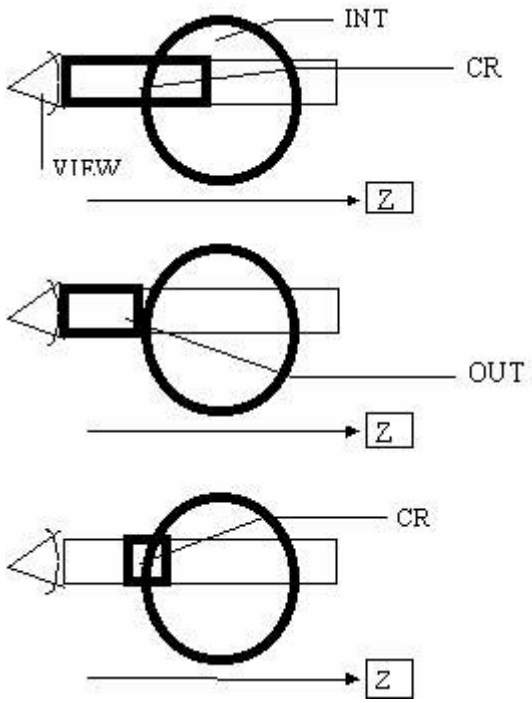

Figure 7: The binary subdivision.

On some recursion level we obtain the bars with one pixel-wide footprint (slices). If for "*a*" steps crossing is not found, search stops, where $2^a x 2^a$ is the screen resolution. If crossing is found, then we start subdivision of slices in depth, i.e. along Z-axis (see Figure 7).

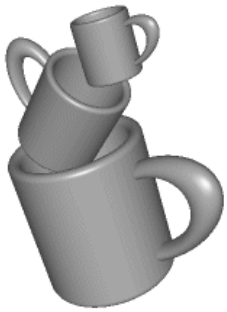

Figure 8: Collision of objects, collision is detected.

Therefore, for each slice we determine first point, which contains object-intersection (the test for crossing is similar above described, but accordingly, with smaller number of coefficients). In other words, we traverse scene space (unit cube) by quad-tree, leafs of which (slices) is a roots of the binary subtrees. Coordinate system in which the algorithm subdivides cubic volume is called work or model space and is denoted M (see Figure 5). Coordinate system with camera (viewer) in its origin and viewing frustum is denoted P (see Figure 6).

To calculate the time of collision detection, we tested objects having different complexity (shape) and different form of collision (we mean collisions by different sides and parts of objects) (see Figure 8). The collision calculation time was real on PC (Intel Pentium 4 Processor, 2800 MHz), and the time spread during testing was below 1% of the given time.

Thus, for the functional object definition using the perturbation functions [15-17] the test for intersection is similar to the test described in [18, 19] for rendering. Next we prove this statement.

3.3.1 Test for zeros of analytical function in the δ-vicinity of the origin of coordinates

In the rendering algorithms by recursive subdivision of an arbitrary-dimension space, the most important question is about intersection of an object defined as $f(x,y,z) \geq 0$, with a subdivision cell (a square, a bar neighborhood $B(P, \delta)=\{X_i: |X_i-P|\leq\delta\}$).

It is clear that the exact solution of this problem (the system of inequalities) is possible only for rather simple functions $f(x,y,z)$. Even for a function expandable into polynomials in $B(P, \delta)$ to the nth degree the exact solution is unsuitable because it will involve roots to the nth degree.

However, we do not need the exact solution for our application. It is necessary and sufficient to answer the question whether there are contour points in the vicinity because the complete overlapping or the absence of object intersection by a cell, are solved by simple inequalities. It is noteworthy that the approximate solution should take into account the case of potential intersection and be asymptotically accurate.

Thus, we will give another formulation of the problem as a definition of the set of points belonging to $B(0, \delta)$ and at the same time satisfying the equation $f(x,y,z) = 0$.

Definition. The set of points $B(P, \delta) = \{X: \text{dist}(X,P) \leq \delta\}$ is called a δ-area of the point P in the metric space. Further considerations are concerned with the case of three-dimensional space because this is precisely the case of our interest, and the Manhattan metric $\text{dist}(X,Y) = \max\{|X|,|Y|\}$, although these results are valid also for an arbitrary dimension and metric.

Finding the solution. Let $f(x,y,z)$ be an analytical function in a three-dimensional space. Then in $B((0,0,0), \delta)$ it can be expanded into the Taylor's series. Discarding terms with power higher than some d, yields

$$f(x,y,z) = \sum_{0\leq i,j,k;i+j+k\leq d} f_{ijk} x^i y^j z^k \quad (11)$$

Write expression (10) in the form

$$f(x,y,z) = \sum_{h=0}^{d} f^h(x,y,z), \quad (12)$$

where

$$f^h(x,y,z) = \sum_{h=i+j+k} f_{ijk} x^i y^j z^k \quad , \quad (13)$$

Here we assume that the following condition is fulfilled:

$$\left|f(0,0,0)\right| > 0 \quad , \quad (14)$$

because otherwise the contour involves the origin of coordinates, i.e., the trivial case.

Let us consider the inequality of triangle:

$$\left|f(x,y,z)\right| \geq \left|f^0\right| - \sum_{h=0}^{h=d} \left|f^h(x,y,z)\right| \quad (15)$$

and the function $f^i(x,y,z)$:

$$\left|f^i(x,y,z)\right| \leq \sum_{i+j+k=h} \left|f_{ijk} x^i y^j z^k\right| \leq \left\{\sum_{i+j+k} \left|f_{ijk}\right|\right\}\left\{\max\left|x^i y^j z^k\right|\right\} \leq F_h \varepsilon^h \quad (16)$$

, where

$$\varepsilon = dist\big((x,y,z);(0,0,0)\big) = max\{|x|,|y|,|z|\} \quad (17)$$

Thus, writing the test in the practically suitable form we have the following inequality (if it is true, then $f(x,y,z)$ has zeros in $B((0,0,0), \delta)$:

$$F(\delta) = F_0 - \sum_{h=1}^{d} F_h \delta^h \geq 0 \Leftrightarrow$$

$$\left|f_0\right| - \sum_{h=1}^{d} \left(\left|f_{0,0,h}\right| + \left|f_{0,1,h-1}\right| + \ldots + \left|f_{h,0,0}\right|\right)\delta^h \geq 0 \quad (18)$$

Since the vicinity is usually a unit vicinity, $B((0,0,0), \delta=1)$, the formula is reduced to comparison of the modules of the free term of the equation of the figure (a line or a surface) with the sum of modules of the rest of coefficients.

4. CONCLUSION

The problem of collision detection or contact determination between two or more objects is fundamental to computer animation, computer simulated environments, robot motion planning, physical based modeling and molecular modeling as well.

In the proposed paper, we have analyzed the algorithm for detection of collisions of objects defined by analytical perturbation functions by means of recursive object space subdivision. We have shown the advantages of the algorithm over the known functionally defined object collision detection algorithm.

We may conclude that in the proposed functionally defined object collision detection algorithm, the collision is always detected and does not depend on the relative position of collided objects and parts of their surfaces, i.e., such an algorithm guarantees detection of the event, which has been proved both experimentally and theoretically; it is required to have equal number of levels of object space subdivision and, therefore, equal computation time; the time of computation by means of the proposed collision detection algorithm was real on PC (Intel Pentium 4 Processor, 2800 MHz); the object collision was detected in a constant time for collisions of different complexity and the time spread in the tests was below 1% of the given time; in the offered collision detection algorithm we didn't use any bounding volumes around each object.

5. REFERENCES

[1] Pasko A.A., Adzhiev V.D., Sourin, A.I. et al. "Function representation in geometric modeling: concepts,

implementation and applications"// The Visual Computer.1995. 11(6). P. 429.

[2]http://www.cs.unc.edu/~geom/papers/subject.shtml#COLLISION

[3] M.C. Lin, "Efficient Collision Detection for Animation and Robotics", PhD thesis, Dept. of Electrical Eng. and Computer Science, University of California, Berkeley, USA, 1993

[4] D. Baraff, "Analytical methods for dynamic simulation of non-penetrating rigid bodies", in Computer Graphics Proceedings, ACM SIGGRAPH, vol. 23, pp. 223-232, 1989

[5] D. Baraff, "Fast contact force computation for nonpenetrating rigid bodies", in Computer Graphics Proceedings, Annual Conf. Series. ACM SIGGRAPH, pp. 23-34, 1994

[6] A. Gregory, M. Lin, et al. "H-Collide: A Framework for Fast and Accurate Collision Detection for Haptic Interaction". IEEE Virtual Reality, 1999

[7] D. C. Ruspini, K. Kolarov, and O. Knatib. "The haptic display of complex graphical environment". *Proceedings of SIGGRAPH 97,* vol. 1, pp. 295-301, August 1997

[8] J. K. Hahn. "Realistic animation of rigid bodies". *ACM Computer Graphics*, 22(4):pp. 299-308, 1988

[9] B.V. Herzen, A.H. Barr, and H.R. Zatz. "Geometric collisions for time-dependent parametric surfaces". *ACM Computer Graphics*, 24(4), August 1990

[10] T. Duff. "Interval arithmetic and recursive subdivision for implicit functions and constructive solid geometry". *ACM Computer Graphics*, 26(2):pp. 131-139, 1992

[11] C. Hoffmann. "Geometric and solid modeling". Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1989

[12] A. Pentland and J. Williams. "Good vibrations: modal dynamics for graphics and animation". *ACM Computer Graphics*, 23(3):pp. 185-192, 1990

[13] M. Teschner, S. Kimmerle, G. Zachmann, B. Heidelberger, Laks Raghupathi, A. Fuhrmann, Marie-Paule Cani, Frangois Faure, N. Magnetat-Thalmann, W. Strasser. "Collision Detection for Deformable Objects". Proc. Eurographics State-of-the-Art Report. Pages 119-139. 2004/ http://www-evasion.imag.fr/Publications/2004/TKZHRFCFMS04

[14] Savchenko V.V., Pasko A.A. "Collision detection for functionally defined deformable objects": The First International Workshop on Implicit Surfaces (Grenoble, France, April 18-19, 1995) /Eds. B.Wyvill and M.P. Gascuel: Eurographics-INRIA, pp. 217-221, 1995

[15] http://www.cgg.ru/febr/vjat/pivweb.html#a).

[16] Vyatkin S.I., Dolgovesov B.S., Yesin A.V. "Non-Polygonal Graphics and Volume-Oriented Rendering": Proceedings of the IASTED International Conference on Automation, Control, and Information Technology (Novosibirsk, Russia, June 10-13, 2002) // ACTA Press. Anaheim, California, USA, pp. 447-451, 2002

[17] Vyatkin S., Dolgovesov B. "Surfaces and Patches of Arbitrary Forms Based on Scalar and Analytical Perturbation Functions" // Graphicon' 2002: *12th International Conference on Computer Graphics,* Nizhny Novgorod, 2002

[18] Vyatkin S., Dolgovesov B., Yesin A., et al., "Voxel Volumes volume-oriented visualization system", *International Conference on Shape Modeling and Applications* (March 1-4, 1999, Aizu-Wakamatsu, Japan) IEEE Computer Society, Los Alamitos, California, pp. 234-241. 1999

[19] Vyatkin S., Dolgovesov B., Guimaoutdinov O., "Synthesis of Virtual Environment Using Perturbation Functions", pp 350-355, volume III (Emergent Computing and Virtual Engineering), *World Multiconferenceon Systemics, Cybernetics and Informatics Proceedings*, Orlando, FL, USA, July 22-25, 2001

About the authors

Sergei I. Vyatkin (Ph.D.) is a scientific researcher of Synthesizing Visualization Systems Laboratory at Institute of Automation and Electrometry SB RAS.

His contact email is sivser@mail.ru,

Boris S. Dolgovesov (Ph.D.) is a head of Synthesizing Visualization Systems Laboratory at Institute of Automation and Electrometry SB RAS.

His contact email is bsd@iae.nsk.su