УДК 004.925.83

С.И. ВЯТКИН
Институт автоматики и электрометрии СО РАН, Новосибирск, Россия
С.А. РОМАНЮК, С.В. ПАВЛОВ, А.А. ДУДНИК
Винницкий национальный технический университет, Украина

АРХИТЕКТУРА ГРАФИЧЕСКИХ ПРОЦЕССОРОВ НА БАЗЕ ФУНКЦИЙ

Для визуализации высокореалистичных изображений реализуется переход от сканирования двумерного пространства в трехмерное. Используются три типы свободных форм объектов наряду с полигональной: 1. Неявные функции возмущения. 2. Скалярные функции возмущения. 3. Трехразмерные данные (массив вокселей). Предлагается растеризация пронумерованных примитивов без разделения их на полигоны.

Ключевые слова: функции возмущения, архитектура графических процессоров, рей кастинг

S.I. VYATKIN
Institute of Automation and Electrometry SB RAS, Novosibirsk, Russia
S.A. ROMANYUK, S.V. PAVLOV, O.O. DUDNYK
Vinnitsa National Technical University, Ukraine

FUNCTION-BASED GPU ARCHITECTURE

For visualizations of high-realistic expressing is realized transition from the scan of two-dimensional space to three-dimensional. Task three types of the free forms of objects alongside with polygonal: 1. Implicit perturbation functions. 2. Scalar perturbation functions. 3. 3D-data of volume (voxel's array). Rasterization of enumerating primitives without partitioning them on polygons is proposed.

*Keywords: perturbation functions, GPU **architecture**, raycasting*

1. Introduction

Real-time computer graphics oriented to 3-D scene visualization has attained appreciable success nowadays. Though a sufficiently high realism of real-time scene imaging has been attained, some problems are still present (e.g., imaging of large terrain regions), where it is necessary to store and visualize scenes containing a greater number of polygons than it is implemented in the present-day systems. For instance, the problem of imaging of mountains requires for initial description hundreds of thousands of polygons. On the other hand, exact modeling of shapes of the car, airplane, submarine frames requires thousands of spline-surfaces (curvilinear areas defined by polynomial functions) whereas the case of definition by polygons will require tens and sometimes hundreds of thousands of polygons. Moreover, polygonal 3-D graphics with scanning of polygons in the image plane is not three-dimensional in the full sense of the word. Information presented to the user in such an approach is incomplete. The main point is the absence of information on object depth. This case implies not the absence of the Z-coordinate of the surface point but the absence of information about the beam passing through the object.

We present results of some investigations concerned with modeling of a system in which it is proposed, to use along with the polygonal representation, object representation by free forms in the form of real and scalar functions and volume spaces of voxel arrays. The possibility of freeform volume visualization is investigated. A recursive algorithm of rendering with object space division with regard to perspective is proposed for visualization.

2. Function-based objects

The characteristic feature of the proposed freeform representation is: firstly, the fact that the main primitives are presented by second-order surfaces, i.e., quadrics. A primitive-quadric is the basis for constructing the rest of objects. The quadric is defined by a real continuous descriptive function of three variables (x1, x2, x3) E^n in the form $F(X) \geq 0$. Quadrics are considered as closed subsets of the Euclidean space E^n defined by the descriptive function $F(X) \geq 0$ where F is the continuous real function; X= (x1,x2,x3) - is the point at E^n specified by the coordinate variables. Here $F(X) > 0$ specifies points inside the quadric, $F(X) = 0$ – the points at the boundary, and $F(X) < 0$ - the points lying outside and not belonging to the quadric. Using the quadrics, the first class of free forms is constructed with the use of real perturbation functions. This type of free forms is suitable in generation of artificial (man-made) objects. The second class of free forms is proposed with the use of scalar perturbation functions with respect to the basic plane or the quadric, for example, to generate a terrain or sculpture models.

2.1 Perturbation functions in the implicit form.

It is proposed to describe complex geometric objects by defining the function of deviation (of the second order) from the basic quadric [1]. Free forms are constructed on the basis of quadrics. The freeform is a composition of the basic quadric and the perturbation perturbation function. 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. The surface obtained will be smooth, and a small number of perturbation functions will be necessary to create complex surface forms. Thus, the problem of object construction reduces to the problem of quadric surface deformation in a desired manner rather than to approximation by primitives (polygons or patches represented by B-spline surfaces). In addition, while solving the descriptive function in the form of inequality $F(X) \geq 0$, we can visualize not only the surface but also the internal structure of the object [2].

2.2 Perturbation functions in the scalar form.

It is proposed to describe complex geometric objects by defining (in the scalar form) the second-order function of deviation from the basic surface or (in the simplest form) from the basic plane. A terrain is a particular case of such objects; it is defined by means of the basic plane and the perturbation function defined in an infinitely long parallelepiped. Values of the perturbation function are specified at the parallelepiped cross-section by a 2-D height map. As a basic surface we may use a plane, and then the direction of the carrier plane normal must match the longitudinal direction of the parallelepiped - the region of perturbation function definition.

A terrain model is coded as differential height map, i.e. the carrier surface is defined by algebraic means and only deviation from this basic surface is stored in the each node. Such modeling method simplifies creation of smooth detail levels and shading. The data of height grid is not subject to geometry transformations as the triangle vertices do. The geometry transformations are only required for the carrier surface. During the recursive voxel subdivision on each level we project the centers of the voxels onto some plane. The computed coordinates, as well as in the case of ordinary RGB texture map, will define address in the so-called "altitude map" or "shape texture" [3]. We calculate the altitude corresponding to this address and a level of details, and use it to modify coefficients of the plane or quadric equation. As a result we will obtain a smooth surface of arbitrary shape modulated with the values from the altitude map. But the problems solved by this algorithm require much more complicated methods within the traditional approach. Indeed, the common way to present terrain with polygons requires an abundance of polygons. Besides, the number of additional problems arises such as high depth complexity, hidden polygons removal, priorities, switching between levels of detail, clipping polygons by the pyramid of vision, etc. Such problems do not appear in the proposed method. The Geometry Processor works with the single plane. The corresponding traversing of the tree and the set of masks provide the right priority order. The backside of a terrain is rejected automatically. The clipping terrain by the pyramid of vision becomes unnecessary since sampling of just required altitudes from the altitude map is provided automatically by the rendering algorithm. To switch between levels of detail the same procedure is used as for the ordinary texture.

3. Rendering technique.

We used the multilevel ray casting algorithm [3], which performs efficient search for volume elements - voxels which participate in image generation. At the first step of recursion, the initial viewing pyramid is divided into four smaller sub pyramids in the screen plane. At the stage of division of space along the quaternary tree, 2-times compression and transfer by ±1 along two coordinates. If the intersection is determined, then the sub pyramid is subject to the next recursion level. Sub pyramids that do not intersect with the object are not subject to further immersion to recursion, this corresponds to elimination from consideration of square screen spaces which the sub pyramid (and, consequently, the object surface) is not mapped to. The viewing pyramid is subdivided until reaching the maximal set level of recursion. The technique has an advantage that it allows discard of large parts of empty space at an early stage. While searching for voxels containing the imaging object surface areas, the pyramidal space is traversed along the quaternary tree whose leaves are roots of binary trees. The multilevel ray casting technique allows us to determine effectively and quickly belonging of rays of different levels (pyramids) to surfaces, and discard space regions outside the objects.

While visualizing the surfaces a test is verified for belonging of only intersected voxels (unit volume elements), external and internal voxels are discarded. To improve imaging realism and extend the class of objects imaged (translucent structures with internal density distribution, 3-D textures), it is necessary to image the internal translucent object structure. For this purpose, not only voxels lying on the surface but also voxels inside the object must participate in imaging. Therefore, while dividing the volume the internal object parts are not discarded, for them algorithm recursion is performed further. Scanning of the scene along the Z-coordinate, which corresponds to scanning of the volume through the depth, is not interrupted upon meeting the surface but is continued until the volume is scanned completely or a certain value of transparency higher than a threshold value is stored. To reduce the computation time the algorithm is adapted to quick passage of homogeneous spaces of objects, for which it is unnecessary to scan the volume completely reaching the last recursion level, it is necessary to "skip" empty or homogeneous spaces along the Z-coordinate and immediately calculate the color and the total transparency. Since ray passage through empty space makes no contribution to the final image, then the skip of the empty space is able to make the processing substantially fast and does not affect the image quality.

4. Motivation of choice of architecture

Depending on the type and amount used processor elements voxel processor can have a different architecture. Time of processing the primitives on i level forms from two values:

$Ti <= 4Tcr + Tmod$, for quaternary tree,

$Ti <= 2Tcr + Tmod$, for binary tree,

Where Tcr is a time of the searching crossing the primitives with sub pyramids; Tcr is a time of modification of factors of equations of primitives.

Production is valued number of primitives of area S, processed in the unit of time. Voxel Processor Production as a computing pipeline is production of cascades of pipeline with a most time of processing a primitive of area S: $p = P/Pmax$. Average loading a voxel processor or factor of efficiency a multiprocessors to realization possible to define as $L = P/PsN$, where N is a number of processor elements; Ps is one processor element power with the stack (stack); P is given architecture power.

On the figure 1 brought graphs of dependency of relative power and boots the different architectures.

The canonical polygonal graphics system can be parallelized by replicating all components. Unfortunately,

the replicated pipelines cannot work independently of each other. Two possible interconnects are used. The first interconnect is used to route primitives to the appropriate rasterizers. The other possibility is, to route fragments after rasterization to the appropriate memory location. Fortunately, the replicated pipelines can work independently of each other in our approach. The main merit of our approach is the reduction of the load on the geometry processor and decrease of data flow from it to the video processor. The best architecture of polygon's channel is pipeline variant 2, table 1), but the best architecture of voxel-based surface's channel is variant 7 (table 1). From graphs on the Fig. 1 seen as this variant wins beside the variant 2 (hatched part) for objects more 20 pixels, when equipping are spared in 5 once (table 1).

Table 1

	Equipment	**Output**	**Load**	*** Notice**
1). Processor with stack	minimum equipment (1 p.e.)	minimum output $P \geq \frac{1}{\sum ti}$	maximum loading 100%	output is defined by total volume of work p.e.-processor element
2). Pipeline of processors	average equipment (10 p.e.)	average output $P \geq \frac{1}{t_n}$	loading 20-30%	output is defined by last cascade output pipeline from 10 processor elements exceeds output stack in 1.5-3 times
3). Tree of processors	maximum equipment ($\approx 3.5 \cdot 10^5$) $n=3.495 \cdot 10^5$ p.e.	maximum output $P \geq \frac{1}{t_1}$	loading 0.1-0.01%	output is defined by time of processing on first faking level
4). Pipeline + tree	equipment above the average (13p.e or 28p.e)	output above the average $P \geq \frac{1}{ti+1}$	loading 40-50%	1) output given hybrid exceeds output stack in 5-14 once 2) output pipeline + tree is defined faking level numbers, defining root of partial tree
5). Stack + tree	equipment below the awerage (8 p.e.)	output above the average	loading 60-80%	
6). Multistack	equipment average (9p.e.)	output above the average	loading 90-100%	
7). Stack + 1 pipeline	Equipment small (2p.e.)	output average	loading 80-100%	

The first stage of the Function-Based GPU (F-B GPU) is a processor of geometric transformations. The second stage is a pyramid traversal processor, which consists of pipelines (terrain pipeline, quadric pipeline, polygon pipeline) of uniform cellular processors. Rendering, the process of coloring in the surfaces supplied by the traversal subsystem to create the final image, takes place in the Pixel Processor (PP).

Primitives may be planes, or quadrics with scalar perturbation functions. The primitives are then rasterized. Given microchip is intended for processing the objects of the free forms with scalar perturbation functions and voxel-based terrain, which is a private event of the free forms.

Primitives may be quadrics, or quadrics with implicit perturbation functions. The primitives are then rasterized. Given microchip is intended for processing the objects of the free forms with analytical perturbation functions and quadrics, which are a private event of the free forms.

On the first step of recursion source pyramid of visibility is divided into four smaller sub pyramids in screen planes. Test is executed for each new sub pyramid on intersection with the object. On the grounds of results of this test is realized fission of sub pyramids, which lie inwardly of surface wholly or, possible, partly, but undoubtedly external of sub pyramid are excluded from processing. Fission of pyramid of visibility leads until it is reached greatly stated recursion level.

To raise realism of expressing and increase a class of displayed objects (translucent structures with internal sharing density, 3D textures) it is necessary to display an internal translucent structure of object. For this in imaging must participate not only voxels, which rest upon surfaces, but in the same way and that, which inhere inwardly object. Consequently, when doing a volume internal parts of the object are not rejected, for they are conducted further recursion of algorithm. At the scan a scene on Z coordinate that corresponds the scan a volume in the depth, scan is not broken when meeting with the surface, but works hereinafter, while completely does not scan a volume or will not be accumulated definite sign of transparency of greater certain threshold value. For reducing a time of calculations an algorithm is adapted to the quick passing of uniform areas of objects. For which at all unnecessary completely scan a volume, getting to the last recursion level, but follows 'race through started or uniform area on Z coordinate, and immediately calculate a color and general transparency.

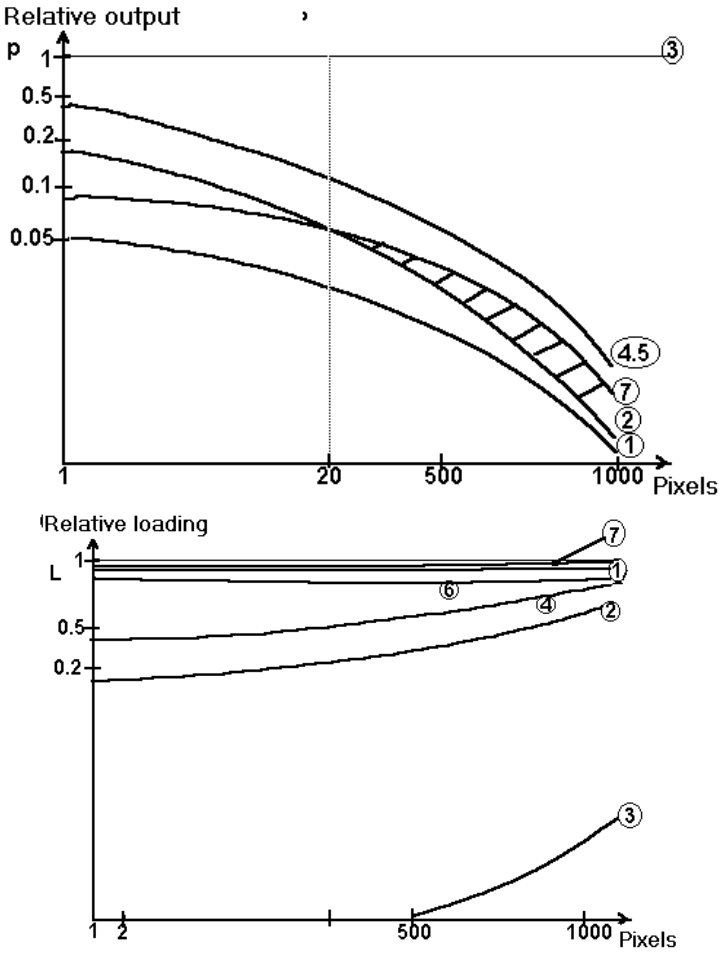

Fig. 1. Graphs of dependency of relative power and boots the different architectures

To raise realism of expressing and increase a class of displayed objects (translucent structures with internal sharing density, 3D textures) it is necessary to display an internal translucent structure of object. For this in imaging must participate not only voxels, which rest upon surfaces, but in the same way and that, which inhere inwardly object. Consequently, when doing a volume internal parts of the object are not rejected, for they are conducted further recursion of algorithm. At the scan a scene on Z coordinate that corresponds the scan a volume in the depth, scan is not broken when meeting with the surface, but works hereinafter, while completely does not scan a volume or will not be accumulated definite sign of transparency of greater certain threshold value. For reducing a time of calculations an algorithm is adapted to the quick passing of uniform areas of objects. For which at all unnecessary completely scan a volume, getting to the last recursion level, but follows 'race through started or uniform area on Z coordinate, and immediately calculate a color and general transparency.

Primitives may be points, lines, or triangles. The primitives are then rasterized. Texture, shading, and fog are applied to each pixel or pixel fragment. Finished pixels are stored in the frame buffer along with their depth, which is used to resolve visual priority. Given microchip is intended for processing the triangles, removing the invisible surfaces, texturing and shading, with provision for the mist and without it.

Many of today's PC architectures employ the "universal memory" architecture. Main memory is used for program execution, database storage, and texture storage. While this approach minimizes cost, it is difficult to guarantee a sustained fill rate because the available memory bandwidth is constantly being spread over these various non-uniform processes. To complicate matters, even the largest system memories are too small to accommodate modern terrain databases that are completely covered with geospecific photo texture. An end-to-end system solution is necessary to handle the texture requirements of the simulation environment. To compensate for the varying demands on system memory, a texture memory cache is usually located on the rendering engine. But caches that are sufficiently large enough to smooth out the variations use precious gates in the core of the rendering chipset. A texture memory capable of storing relatively large amounts of texture information, say 16-64MB, attached to the chip helps keep the gate count down at the expense of board real estate. Frequently used textures such as those used for special-effect animations and common database features may be locked into part of this memory. The remaining space can be used to hold the terrain texture in the vicinity of the eye point. Real-time software pages terrain texture into this space from secondary storage (usually hard disks) based on the eye point's geographical location within the database. Obviously, the various system busses and secondary storage access times must be sized to handle the expected paging rates. Texture compression helps relieve the bandwidth demands and, depending on where the texture is decompressed, may permit the use of a smaller texture memory.

In the channel architecture base of processing the polygonal figures prescribed idea of recursive procedure of doing a screen, localizing internal area of polygonal figures in the process of rasterization. Main distinctive devils of method are a polygon description by kits of direct, getting through ribs, and recursive fission of screen on hutches, which area decreases in 4 with each step of fission times. For eliminating the defects, in accordance with discrete television raster, determination an accessories of element of expressing to the polygon is realized on the increased resolution of raster: each pixel is divided into 16 sub pixels. As a result is formed sub pixel code of fragment, which is compared to the code of mask of hutch, formed from earlier processing polygons.

Several types of parallelism are used in the channel of processing the polygons:

Pixel level and primitive level realize word aligned.

To the account asynchronous functioning (working) the pixel channels occurs pixel aligned, which gives a most effect for 2D array and ensures plural-primitive parallelism level (rasterization of several primitives or several fragments of one big primitive parallel).

Is used technology of rarefying raster with the apperture.

Advantages of given approach:

1. Discriminating characteristic of given approach is concluded in that that exists nearly single-line dependency of speedup factor Ssqt from the amount of processors, this is a feature of optimum system architecture.

2. Square organization a frame-buffer strategy reaches speedups for any orientation vectors, in that time, as a single-line organization capable to reach a parallelism for horizontal or vertical vectors only.

As can be seen from graph {2} on the Fig. 1 for the channel of processing the polygons the most optimum architecture - a pipeline, as far as under the greater amount of polygons their size decreases, but pipeline the most effectively processes polygons a size less 20 pixels.

5. Possibilities of proposed technology for practical use, points for further consideration, output in sale (compromise result)

Order of different type development of microcircuits and output them in sale

Reasonable first go develop a microchip of Pixel Processor, possible variant of using a ready chip, oriented on triangles, with the following expansion before two pixel channels. Functions of geometric processor can undertake Host Processor, but later it is necessary to develop and geometric processor or use a suitable available chip, this it is necessary for polygons mainly, as far as for two other pipelines (Terrain Pipeline & Quad Pipeline) geometric processor not actual. Hereinafter necessary minimum must be increased by additional functions and possibilities, and following microchip for the development and fabrication could become Terrain Pipeline. Accelerator Cost could vary depending on the type and amount of chips, but from its functions and possibilities consequently.

6. Points for further consideration and development

In the first place it is necessary to enter aside from such traditional primitives as polygons else and voxel-based surfaces, to he pertain quadrics, free forms on the base an quadrics with analytical perturbation functions, terrain and free forms with scalar perturbation functions on the base of algorithm of building voxel-based terrain, equivalent Volume-Splines. On the base of enumerating surfaces possible to generate and amounts (amorphous objects including). 3D Texture and direct rendering volumes better to postpone for the future, when comes a necessary technology.

7. Attitude to generally accepted technologies

Absolutely new technology is a visualization of the free forms as analytical, so and scalar type (way of task of the free forms with analytical perturbation functions and rasterization three types of the free forms). On the other hand has been an association of proposed technology with known, such as polygons, as well as in this case it is offered new and efficient method to visualizations the polygons in the comparison with known.

Conclusion

Our investigation in the volume-oriented visualization technology have made it possible to reveal some advantages in both the scene representation technique and the rendering algorithm oriented to real-time implementation. The change over from rasterization in the image plane “back-end” or the image-space end of the graphics pipeline) to volume rendering, (“fronted” or the object-space end of the graphics pipeline) in combination with the proposed object definition techniques, though increases the amount of real-time computation, as a whole, nevertheless it results in some merits improving the scene imaging realism. The main merits of our approach are the following:

- Reduced number of surfaces for describing curvilinear objects (representation of objects by freeform surfaces reduces 100 times and more the database description compared with their representation by polygons)
- Reduction of the load on the geometry processor and decrease of data flow from it to the video processor
- Sufficiently simpler construction of terrain because the preliminary surface triangulation and the viewing pyramid clipping are unnecessary (to change the level of detail we use a mechanism similar to the usual texture sampling)
- The computation time in terrain generation is practically independent of the height map resolution and depends only on the screen resolution
- The possibility to process voxel arrays bounded by freeform surfaces

- Simple animation and morphing of scenes.

A statistics will bring for the example for voxel-based terrain. Where is not required triangulation of terrain, in the given technology main, on than it is based, is a hierarchical mechanism by control levels of detail. Which practically reminds MipMapping usual texture of color, as well as in the event of the texture of color a time of calculations does not depend on permits of height map If conduct a get fat analogy with the texture, all its positive characteristics are inherited given technology, to which possible refer including and simplicity an animation such surface without computing expenses on geometric transformations of tops of triangles, if such surface was tessellated.

Possible also metamorphosis of non-homeomorphic objects that without breakups impossible and following splicing in the event of polygonal surface. Another feature of the proposed method is the possibility to define dynamic objects like waves or surface movements by defining time dependent function, which alters scalar field values. As a result, one can visualize propagating waves, surface deformations (explosions, waterspouts of navy battle, sinking of a submarine in a wavy sea with semi-transparent water, cloud shape deformations, etc.). Objects can move and penetrate within each other, can change shape and size.

All these merits of our approach form the ground for creating a new class of computer visualization systems for various applications. The proposed algorithm of rendering along with the possibility to visualize arbitrary surfaces of free forms and inhomogeneous volume spaces offers a wide scope of application.

References

1. Vyatkin S.I., Dolgovesov B.S. Synthesis of virtual environment using perturbation functions // volume III World Multiconference on Systemics, Cybernetics and Informatics Proceedings, Orlando, FL, USA, July 22-25, 2001, P.350-355.

2. Vyatkin S.I., Dolgovesov B.S. A 3D Texture-Based Recursive Multi-Level Ray Casting Algorithm // Proceedings of the Second IASTED International Multi-Conference on Automation, Control, and Information Technology - ACIT`2005. "Software Engineering" ((Novosibirsk, Russia, June 20-24, 2005). Ed.: Yu.I. Shokin, O.I. Potaturkin. ACTA Press. P. 92-97.

3. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin 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, 1999, pp. 234-241.

Литература

1. Вяткин С.И., Долговесов Б.С. Синтез виртуальной среды с использованием функций возмущения // Том III Всемирная Мультиконференция на Системности, Работы по кибернетике и математике, Орландо, Флорида, США, 22-25 июля, 2001, ст.350-355.

2. Вяткин С.И., Долговесов Б.С. Многоуровмевый алгоритм рейкастинга на основе 3D текстур// Труды Второй IASTED международной мультиконференции по автоматизации, управлению и информационных технологиях - ACIT`2005. "Программная инженерия" ((Новосибирск, Россия, 20-24 июня, 2005). Ред .: Ю. Шокин, О. И. Потатуркин. ACTA Press. ст. 92-97.

3. Вяткин С.И. Объемно-ориентированная система визуализации воксельных моделей / Долговесов Б.С., Есин А.В. - Международная конференция по Shape моделированию и приложениях (1-4 марта 1999 года, Айдзу-Вакамацу, Япония) Компьютерное общество IEEE, Лос-Аламитос, Калифорния, 1999, стр. 234-241.

Рецензія/Peer review : 8.1.2015 р. Надрукована/Printed :25.1.2015 р.
Стаття рецензована редакційною колегією