

# Geometric Querying for Dynamic Exploration of Multidimensional Data

Olga Sourina

School of Electrical & Electronic Engineering,  
Nanyang Technological University, Block S1,  
Nanyang Avenue, Singapore 639798  
eosourina@ntu.edu.sg  
<http://www.ntu.edu.sg/home/eosourina>

**Abstract.** This paper describes a geometric query model for dynamic exploring multidimensional data. An application of the model for solving the problem of materials selection in product design is discussed. Data from database are interpreted geometrically as multidimensional points. A query window is a query solid of any shape specified by its location. The queries are formulated with geometric objects and operations over them. The geometric objects and operations are described with implicit functions. The process of query specification is visualized. The user poses the queries through graphics interface accessing dynamically multidimensional points, geometric primitives and applying geometric operations over them.

## 1 Introduction

Visual data exploration has recently attracted attention in both academics and industrial worlds. According to Keim [1], data to be visualized can be classified as one-dimensional data, two-dimensional data, multidimensional data, text and hypertext, hierarchies and graphs, algorithms and software. On the other hand, the data can be thought as geometric and non-geometric ones. In this paper, we will describe geometric query model for data that can be represented as multidimensional points.

Different techniques were proposed for visualization of multidimensional data. One group of techniques visualizes multidimensional data as two- or three-dimensional projections and sections in Euclidean space. In these cases, two or three-dimensional visualization tools are directly applied for data visualization [2]. These tools allow for drawing projections [2] and hyperslices [3]. DataSpace system takes step further utilizing the display space by placing panels with projections, possibly generated by different visual applications [4]. The three-dimensional visualization techniques are common for three- or multi-dimensional spatial databases. These systems incorporate visualization algorithms to interpret query results [5].

Another group of visualization techniques involves mapping of multidimensional points into two dimensions of the screen. These techniques can be geometry-based,

(e.g., well-known Parallel Coordinates), icon-based, pixel-oriented, and hybrid techniques [1].

Visualization techniques often employ interactive techniques for a better exploration of data. These interactive techniques have to help the user to query, browse, zoom, link, and brush information. Although visualization techniques are well-established area of research, the interactive techniques, especially querying multidimensional data, is still a very challenging topic. Recently, the need for such a study has become increasingly important because of the modern process of integrating visualization techniques, data mining algorithms, database systems and data warehousing to explore complex information. A specification of properties of multidimensional data can be done with dynamic querying, or interactive filtering. Recent works have reported the value of dynamic queries coupled with two-dimensional data representations for progressive refinement of user queries [6-8]. Polaris [8] is an interface for exploring large multidimensional databases that extends the well-known Pivot Table interface. The visual specification generates queries to the database that select data subsets for analyses. It then, filters, sorts and groups the results into panes, and finally groups, sorts, and aggregates the data before passing it to the graphics encoding process. In Magic Lenses [6], dynamic queries are extended by encoding each operand of the query as a Magic Lens filter. Compound or complex queries can be constructed by overlapping the lenses. Each lens includes a slider and a set of buttons to control the value of the filter function and to define the composition operation generated by overlapping the lenses.

Our approach combines visualization of multidimensional data and visualization of the query formulation employing very compact geometric query model. In this paper, we will show that our geometric model makes constructing full Boolean queries on numerical attributes more intuitive, and allows applying multidimensional query window of any shape.

As it has been mentioned, query modeling received relatively less attention in data visualization. In spatial databases, queries are classified into two types: those based on locations of objects, and those based on objects themselves [9]. We consider only queries based on objects location. In works [10-11], we proposed a query model for relational database that provides the user of scientific and engineering databases with geometric query tools. In this model, we considered the operations over relations as geometric operations over geometric objects that are  $n$ -dimensional points and a query solid. To implement the geometric query model in the most efficient way, it is important to propose an adequate mathematical model for geometric objects and geometric operations. The formal mathematical specification of the geometric query model is proposed with the function representation of geometric objects. Function based shape modeling is becoming increasingly popular in computer graphics [12-14]. The idea of this approach is that complex geometric objects can be produced from a "small formula". Thus, using the function representation named F-Rep, geometric shapes are defined with the inequality  $f(x,y,z) \geq 0$ , where the function  $f$  is positive for the points inside the solid, equal to zero on its border and negative outside the solid. Function representation lets us unify in one single model practically any geometric model provided it can be represented with a real function  $f(x_1, x_2, \dots, x_n) \geq 0$  with at least  $C^0$  continuity. We extend and further enrich this representation and thus come up with a very compact and easily extensible geometric

query model that is implemented in the geometric query system GEDA. The geometric query system prototype GEDA is implemented as a set of external attachments, treating the kernel relational database management system as a closed system. We use advanced computer graphics algorithms and visualization techniques to implement the geometric query model.

To demonstrate possible application of the proposed query model, we will apply the developed geometric query tools for solving the problems of materials selection in product design. Although there are other graphics systems for solving the problems in material engineering such as VISUAL [15] and the Cambridge Materials Selector [16] system, we propose a novel and efficient way of displaying and accessing materials properties data using the developed geometric query tools.

## 2 Geometric Algebra

In this section, we describe our geometric query model. It is defined as  $\{G, \Phi\}$ , where  $G$  is a set of geometric objects and  $\Phi$  is a set of geometric operations in  $n$ -dimensional Euclidean space  $E^n$  ( $n=1,2,\dots,N$ ). Two classes of geometric objects are introduced:  $n$ -dimensional points mapped from the database, and  $n$ -dimensional primitive geometric objects for a construction of general query solids with geometric operations.

Geometric operations  $\Phi$  are geometric functions of geometric arguments. The result of a geometric operation is also a geometric object from the set  $G$ , which can be applied operations from the set  $\Phi$  again. The operations can be unary and binary. The unary operation over  $G_1 \in G$  means the operation  $G_2 = \Phi_i(G_1)$ , where  $G_2 \in G$  and  $i=1,\dots,k$ . The binary operation over  $G_1, G_2 \in G$  means the operation  $G_3 = \Phi_i(G_1, G_2)$ , where  $G_3 \in G$  and  $i=1,\dots,k$ . In the basic set of geometric operations, we include set-theoretic union, intersection, difference, complement, Cartesian products, and orthographic projection. Introducing this set, we keep in mind the correspondence of the geometric query model operations to a minimal set of relational algebra operations. To increase an expressive power of the proposed geometric query model, we also introduce affine transformations (rotation, translation and scaling). Set-theoretic operations, projection and affine transformations have the same sense as in constructive solid geometry and geometric modeling.

We will also use a *point/solid classification predicate*. Let  $P$  be a  $n$ -dimensional point,  $G_1$  be a solid,  $bG_1$  be a boundary of  $G_1$  and  $iG_1$  be an interior of  $G_1$ . Then a point/solid classification is described by a 3-valued predicate:

$$S_3(P, G_1) = \begin{cases} 0, & \text{if } P \notin G_1 \\ 1, & \text{if } P \in bG_1 \\ 2, & \text{if } P \in iG_1 \end{cases}$$

As it was discussed in work [11], geometric interpretation of relational algebra selection operation can be phrased as follows: "find out the points that belong to the solid." In our model, the query solid can be a complex geometric solid. The complex query solid can be created with union, intersection, and other operations over

primitive solids that are hyperhalfspaces, hypercuboids, hyperellipsoids, etc. Selection operation of relational algebra can be found in geometry as point/solid classification predicate. Thus, in order to pose a query, we first generate the complex geometric solid, and then apply the point/solid classification predicate to query the points in the database. The result of the query is a new set of points. Different range and intersection queries can be implemented with this query model. For example, a range query can be phrased as "Retrieve all points or solids contained within the intersection of 3-D box, cylinder and ellipsoid."

### 3 Formal Mathematical Specification of the Geometric Query Model

Let us introduce the formal mathematical specification of the geometric query model with the implicit function representation of geometric objects.

Geometric object can be a set of points  $\mathbf{P}=\{[x_1, x_2, \dots, x_n]\}$  in the  $n$ -dimensional Euclidean space  $E^n$ , and primitive solid objects defined with real functions  $f(x_1, x_2, \dots, x_n) \geq 0$ . The implicit function  $f(x_1, x_2, \dots, x_n) \geq 0$  can be defined analytically or procedurally. Such functions define closed  $n$ -dimensional objects in  $E^n$  space under the following conditions:  $f(\mathbf{X}) > 0$  for the points inside the object,  $f(\mathbf{X}) = 0$  for the points on the object boundary,  $f(\mathbf{X}) < 0$  for the points outside the object, where  $\mathbf{X} = \{x_1, x_2, \dots, x_n\}$  is a position vector of a point in  $E^n$ .

We can consider our set of objects as multidimensional geometric points and multidimensional geometric objects defined with functions  $f(x_1, x_2, \dots, x_n) \geq 0$ . The zero set of these functions provides surfaces, and the values that are greater or equal to zero define multidimensional geometric solid objects. This approach should allow us to achieve a more compact model, the uniform implementation of operations for different types of objects, and, as a result, a compact and simple geometric query system based on this model.

Several examples of the function representation of primitive geometric solids that could be used for construction of geometric criteria follow.

#### Halfspace:

$$G_1: f_1(\mathbf{X}) = f_1(x_1, x_2, \dots, x_i, \dots, x_n) = (x_i - a) \geq 0$$

where  $a$  is some real number ( $a \in R$ ).

#### Hyperellipsoid:

$$G_1: f_1(\mathbf{X}) = 1 - ((x_1 - x_{0,1}) / a_1)^2 - ((x_2 - x_{0,2}) / a_2)^2 - \dots - ((x_n - x_{0,n}) / a_n)^2 \geq 0$$

where  $x_{0,1}, x_{0,2}, \dots, x_{0,n} \in R$  and  $a_1, a_2, \dots, a_n \in R$ .

#### Hypercone:

$$G_1: f_1(\mathbf{X}) = ((x_1 - x_{0,1}) / a_1)^2 - ((x_2 - x_{0,2}) / a_2)^2 - \dots - ((x_{n-1} - x_{0,n-1}) / a_{n-1})^2 \geq 0$$

where  $x_{0,1}, x_{0,2}, \dots, x_{0,n-1} \in R$  and  $a_1, a_2, \dots, a_{n-1} \in R$ .

**Common algebraic solid:**

$$G_1: f_1(\mathbf{X}) = \sum_{i,j,k} a_{lms} x_l^i x_m^j x_s^k \geq 0$$

where  $a_{lms} \in R$ ,  $i=1, \dots, n$ ,  $j=1, \dots, n$ , and  $k=1, \dots, n$ , while integer  $l, m$  and  $s \in 0$ .

Here, we assume that these functions exist and can be used in the mathematical model on the same conditions as analytical functions defining other primitive solids. We can move further by declaring that our model is open for any type of objects that can be defined with some function  $f(x_1, x_2, \dots, x_n) \geq 0$  with at least  $C^0$  continuity. It allows us to avoid solving the problem of a minimum set of primitives and to change this set depending on the application problem to be solved and/or the actual data mapped to our geometric model.

Next, we describe the implementation of geometric operations. We apply them to primitive geometric objects to obtain complex geometric shapes. The analytical definition of set-theoretic operations is implemented in the form proposed by Ricci [13], where operations over implicit functions are considered. We also need to add affine transformations (translation, rotation and scaling) for dynamic querying implementation.

Mathematically, **Union**:  $G_3 = G_1 \cup G_2$  of two objects  $G_1 \subset E^n$  and  $G_2 \subset E^n$  with the descriptive functions  $f_1$  and  $f_2$  will be defined as  $f_3 = f_1 \vee f_2 = \max(f_1, f_2) \geq 0$ , where  $G_3 \subset E^n$ . **Intersection**:  $G_3 = G_1 \cap G_2$  of two objects  $G_1 \subset E^n$  and  $G_2 \subset E^n$  with the descriptive functions  $f_1$  and  $f_2$  will be defined as  $f_3 = f_1 \wedge f_2 = \min(f_1, f_2) \geq 0$ , where  $G_3 \subset E^n$ . **Complement**:  $G_2 = \neg G_1$  of object  $G_1 \subset E^n$  with the descriptive functions  $f_1$  will be defined as  $f_2 = -f_1 \geq 0$ . **Difference**:  $G_3 = G_1 \setminus G_2$  between objects  $G_1 \subset E^n$  and  $G_2 \subset E^n$  with descriptive functions  $f_1$  and  $f_2$  will be defined as  $f_3 = f_1 \wedge (-f_2) = \min(f_1, -f_2) \geq 0$ , where  $G_3 \subset E^n$ . **Cartesian product**:  $G_3 = G_1 \times G_2$  of objects  $G_1 \subset E^k$  and  $G_2 \subset E^m$ ,  $k+m=n$ , with descriptive functions  $f_1$  and  $f_2$  will be defined as  $f_3 = f_1 \wedge f_2 = \min(f_1, f_2) \geq 0$ , where  $G_3 \subset E^n$ . **Translation**:  $G_2 = T(G_1)$  of object  $G_1 \subset E^k$  with descriptive functions  $f_1$  by  $a_1, a_2, \dots, a_n$  will be defined as  $f_1(x_1-a_1, x_2-a_2, \dots, x_n-a_n) \geq 0$ . **Rotation**:  $G_2 = R(G_1)$  of object  $G_1 \subset E^k$  with descriptive functions  $f_1$  of angle  $\alpha$  about some axis will be defined as  $f_1(x'_1, x'_2, \dots, x'_n) \geq 0$  where  $[x'_1 \ x'_2 \ \dots \ x'_n \ 1] = \mathbf{R}^{-1} [x_1 \ x_2 \ \dots \ x_n \ 1]$  and  $\mathbf{R}^{-1}$  is an inverse matrix of rotation. **Scaling**:  $G_2 = S(G_1)$  of object  $G_1 \subset E^k$  with descriptive functions  $f_1$  in  $s_1, s_2, \dots, s_n$  times will be defined as  $f_1(x_1/s_1, x_2/s_2, \dots, x_n/s_n) \geq 0$ . **Projection**: It is considered only for objects consisting of points as a simple elimination of one or several coordinates from the position vectors of points. Projections of solid objects are done as well but only for illustrative purposes and are implemented in a geometric query system prototype with the basic graphics programming tools.

With the implicit function representation of the primitive geometric solids, we can also describe possible formulas of query solids corresponding to relational selection operation predicates. In work [11], the formulas of query solids for relational selection operation using the implicit function representation of geometric objects were given.

In Section 2, we introduced a point/solid classification predicate. Let  $P$  be a  $n$ -dimensional point,  $G_1$  be a solid,  $bG_1$  be a boundary of  $G_1$  and  $iG_1$  be an interior of  $G_1$ . Then a point/solid classification is described with the function representation of the geometric object  $G_1$  by a 3-valued predicate:

$$S_3(P, G_1) = \begin{cases} 0, & \text{if } f_1(x_1, x_2, \dots, x_n) < 0 \quad P \notin G_1 \\ 1, & \text{if } f_1(x_1, x_2, \dots, x_n) = 0 \quad P \in bG_1 \\ 2, & \text{if } f_1(x_1, x_2, \dots, x_n) > 0 \quad P \in iG_1 \end{cases}$$

Furthermore, the uniform description of geometric solids with real functions allows us to introduce any complex geometric solids for posing a query and use the uniform algorithm for searching multidimensional points. In this paper, we do not address the problems of point access method we applied.

## 4 The Graphical User Interface

The proposed geometric query model is meant to be the user's model as well as the formal foundation for the implementation of the geometric query system GEDA. We had no intention to develop a geometric query language. We developed the graphical user interface based on the geometric algebra. We use the geometric concepts to pose location, range or intersection queries and apply visualization techniques to interpret the querying process. The points mapped from the database, the query formulation, and the resulting geometric objects are visualized. To get an initial impression of the data, the data are visualized either as 2- and 3-dimensional points clouds. After the data are visualized, a complex geometric query solid with union, intersection or other operations over primitive geometric solids is generated. These primitive query solids currently are line, cuboid (box), ellipsoid, cone and cylinder. In addition, any point belonging to the visualized solid can be located and identified in the database.

By choosing any geometric object from the list of objects implemented in the system (line, box, cylinder, etc.) and operations over them (union, intersection, complement, difference, etc.), the user can formulate any complex query. After the execution of the searching, the user can retrieve any field of the records founded in the relation database system. By including user-defined geometric objects, we can formulate any geometric query. By default, the first geometric query object is empty and the first operation is a union. For example, a query solid is a box. Sizes of the box can be changed using a mouse. We can use a variable light source to make a picture clearer by moving the light in any direction. The data can be drawn opaque or transparent. Query solids are always drawn transparent. We employ visualization techniques and advanced computer graphics algorithms for the implementation of the user interface. We apply either ray tracing or polygonization followed by fast rendering to visualize the reconstructed solid or its surface. For polygonization, we use the well-known marching cubes algorithm.

## 5 Application of Geometric Querying for Solving the Problem of Materials Selection

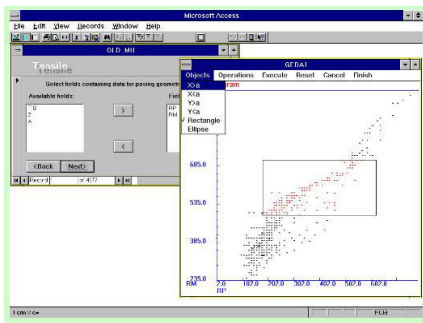
To demonstrate the materials selection procedure, we will explore a file of tensile properties in the structural materials database described in [17]. The file of tensile

properties stores data on the tensile properties of the structural materials under specified conditions (test temperature, neutron fluence, irradiation temperature). The file contains information on ultimate strength ( $A_1$ ), yield strength ( $A_2$ ), total elongation ( $A_3$ ), uniform elongation ( $A_4$ ), reduction of area ( $A_5$ ), proportional limit ( $A_6$ ), Brinell ( $A_7$ ), Rockwell ( $A_8$ ), Vickers hardness ( $A_9$ ) and microhardness ( $A_{10}$ ). Material selection problem can be described as follows. Each material has a set of attributes: its properties. A designer seeks a specific combination of these attributes: a property profile. The materials selection procedure seeks the subset of materials, which have the property profile, which maximizes performance. In another words, design requirements do not usually prescribe a material. They prescribe a performance profile that candidate materials must meet. It is rare that the overall performance is maximized by satisfying only one criterion. Meeting the performance profile requires the designer to identify the material which best satisfies several criteria. In the system prototype GEDA, the user works in  $k$ -dimensional space where  $k$  is a number of material selection criteria. Materials are selected through 2-, 3-dimensional automatic projections of  $k$ -dimensional selection space. Complex range queries of any shape can be applied. As it was noticed in Section 3.1, the set of primitive solids can be easily extended depending on the application problems to be solved and the data nature. We have extended a list of primitives with the cone primitive.

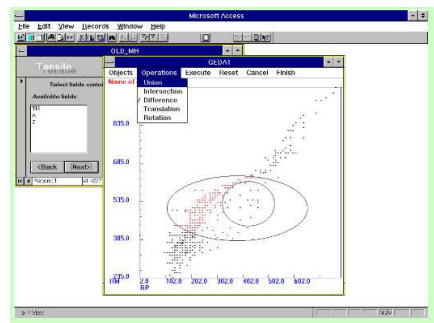
Let us describe some queries. Note, that most of the queries can not be formulated directly using SQL statements.

**Query 1.** Retrieve materials with values of ultimate strength and uniform elongation in the intervals from  $C_1$  to  $K_1$  and from  $C_4$  to  $K_4$  respectively.

In geometric terms, the problem is "to find all points inside the rectangle." With the geometric query system tools, we display all materials that are mapped from database as clouds of 2-dimensional points for properties  $A_1$  and  $A_4$  (see Fig. 1 a).



a)



b)

**Fig. 1.** Examples of 2D geometric queries.

Then, we pose the query drawing a rectangle in the coordinates of the properties. Using the picture-diagrams of the properties, the user can make a decision based on the mutual location of different material points on the diagram. The user can request a

large number of diagrams with the values, narrowing down the search, and in this case, visualization of querying can give adequate help.

The developed geometric query tools allows the user to formulate spatial queries that cannot be realized directly using the selection operation of relational algebra.

**Query 2.** *To retrieve materials with ultimate strength ( $A_1$ ) and uniform elongation ( $A_3$ ) properties located inside the difference between the ellipse  $G_1$  and the circle  $G_2$ , where a center of  $G_1$  is at the point with coordinates  $(a_1, a_3)$  and semiaxes are  $c_1, c_3$  on properties  $A_1, A_3$ , a center of  $G_2$  is at the point with coordinates  $(b_1, b_3)$  and radius is  $r$  on properties  $A_1, A_3$ .*

This geometric query can be posed directly using the proposed geometric query model only. The query specification process is shown in Fig. 1b. First, we choose two properties of a material. Then, we choose an ellipse from the menu, then difference operation, and another ellipse. After that, the user chooses “execute”. The resulting points of found materials are pictured in red. The user can list names of found materials or to continue the search.

The developed geometric query tools allow the user to phrase more complex queries to perform materials selection procedure. In Fig.2, examples of the following geometric queries are shown.

**Query 3.** *To retrieve materials with the properties of ultimate strength, yield strength and test temperature located no further than  $a_1, b_1, c_1$  distances from the material with properties values  $(x_1, y_1, z_1)$  or materials with these properties located no further than  $a_2, b_2, c_2$  distances from the material with properties values  $(x_2, y_2, z_2)$  or materials with these properties located not far than  $a_3, b_3, c_3$  distances from the material with property values  $(x_3, y_3, z_3)$ .*

This query can be also phrased as follows:

*To retrieve materials with properties located inside the union of the ellipsoids.*

The union of the ellipsoids is shown in Fig. 2a. This type of query is used when the user needs to define groups of materials.

**Query 4.** *To retrieve materials with the properties of ultimate strength, yield strength and test temperature located no further than  $r_1$  distance and further than  $r_2$  distance from the materials with the ultimate strength and yield strength values  $(x_1, y_1)$ .*

This query can be also phrased as follows:

*To retrieve materials with properties located inside the difference between the two cylinders with the center  $(x_1, z_1)$  and radiuses  $r_1$  and  $r_2$  respectively.*

In Fig. 2b, the complex query solid is shown as a difference between two cylinders. The resulting points corresponding to selected materials are shown in Fig. 2c.

**Query 5.** *To retrieve materials with properties of ultimate strength, yield strength and test temperature located inside the intersection between the cone and the box, where the two box planes correspond to the test temperature values  $z_1$  and  $z_2$ , and an ultimate strength, yield strength values on the planes are inside the circles with the center  $(x_1, y_1)$  and radiuses  $r_1$  and  $r_2$  respectively.*

First, three properties of materials (total elongation, ultimate strength and test temperature) are mapped from the database. Then, the user chooses a cone primitive from the menu (see Fig. 2d). After that, the user chooses an intersection operation, a

box primitive and “execute” (see Fig. 2e). The user can list names of the selected materials or he/she can continue the search. In Fig. 2f, the possible query solids are shown.

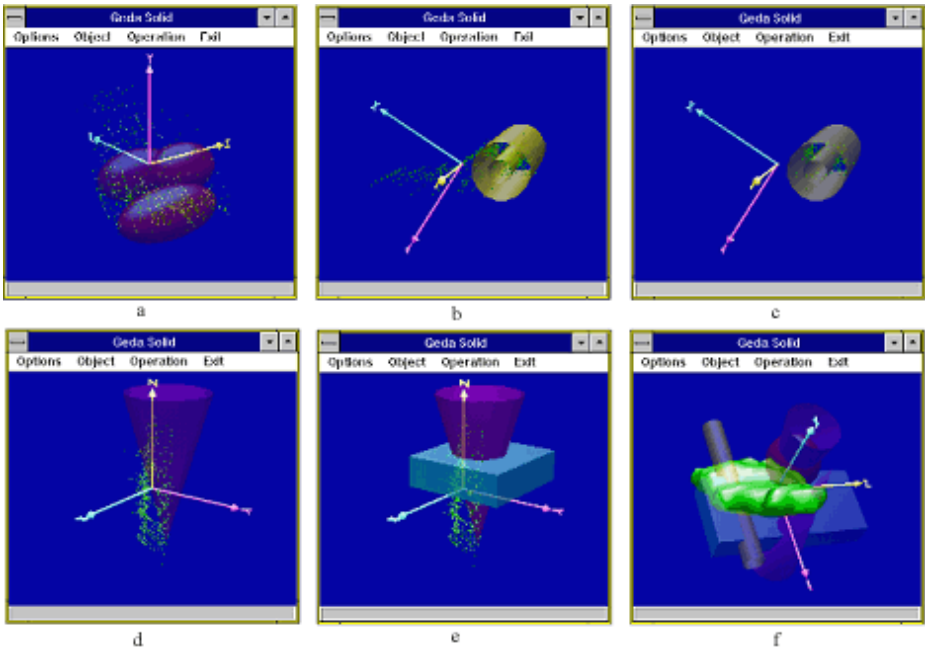


Fig. 2. Geometric querying with complex query solids

## 6 Conclusion and Future Work

We have presented a geometric query model with the implicit functions that was implemented as a demonstration prototype system GEDA. We conclude that GEDA is potentially valuable tool for any data visualization if the data could be interpreted as multidimensional points. Other visualization systems could also benefit from applying the proposed geometric query model. We have many plans for future work in extending this system. Visualization of the querying process has been implemented with the advanced computer graphics tools, and we are planning to improve the interface with Virtual Reality (VR) tools using stereo glasses and a haptic device to make it even more intuitive. We are also going to add more graphics like adaptive labels, legends, sliders, etc. Another area of future work is to design the system for large databases. The human visual system is the most experienced in the interpretation of realistic representations. The application of advanced computer graphics algorithms and visualization techniques for the dynamic query formulation and representation of the query results could help to explore the data through more intuitive interface employing modern VR tools.

## References

1. Keim, D.A.: Information Visualization and Visual Data Mining. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 8, 1 (2002) 1-8.
2. Furnas G.W., Buja A.: Prosections Views: Dimensional Interface through Sections and Projections. *J. Computational and Graphical Statistics*, Vol.3, 4 (1994) 323-353.
3. van Wijk J.J., van Liere R.D.: Hyperslices. *Proc. Visualization '93*, (1993) 119-125.
4. Anupam V., Dar S., Leibfried T., Petajan E.: DataSpace: 3-D Visualization of Large Databases. *Proc. of the Conf. on Information Visualization INFOVIS'95*, (1995) 82-88.
5. Sheu P.C.-Y., Silver D.: Extending object-oriented databases with problem solving and visualization, *Comput. & Graphics*, Vol. 17, 4 (1993) 447-455.
6. Fishkin K., Stone M.C.: Enhanced Dynamic Queries via Movable Filters. *Proc. Human Factors in Computing Systems CHI'95 Conf.*, (1995) 415-420  
<http://www2.parc.com/istl/projects/MagicLenses/DynamicQueries.html>.
7. Ahlberg C., Shneiderman B.: Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. *Proc. Human Factors in Computing Systems CHI'94 Conf.*, (1994) 313-317.
8. Stolte C., Tang D., Hanrahan P.: Polaris: A System for Query, Analysis, and Visualization of Multidimensional Relational Databases. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 8, 1 (2002) 52-65.
9. Yazdani N., Ozsoyoglu M.: Feature-Based Indexing in Spatial Databases, A Tech. Report, SWRU, (1994).
10. Sourina O., Boey S.H.: Geometric Model of Relational Database for Material Selection in Product Design. *Proc. of Third International Conference on Robotics and Computer Vision (1994)* 1643-1647.
11. Sourina O., Boey S.H.: Geometric Query Types for Data Retrieval in Relational Databases, *Data & Knowledge Engineering*, Elsevier Science B.V., Vol. 27, 2 (1998) 207 – 229.
12. Bloomenthal J., *An Introduction to Implicit Surfaces*, Morgan-Kaufmann, 1997
13. Ricci A.: A constructive geometry for computer graphics. *The Computer Journal*, Vol. 16, 2 (1973) 157-160.
14. Shapiro V.: Real functions for representation of rigid solids, TR CPA91-10, Cornell University, (1991).
15. Ozsoyoglu G., Ozsoyoglu Z.M., Vadaparty K.: A Scientific Database System for Polymers and Materials Engineering Needs. *Proc. of Seventh International Working Conference on Scientific and Statistical Database Management*, Virginia, (1994) 138-148
16. Cambridge Materials Selector, University Engineering Department, Cambridge, CB2 1PZ, (1992).
17. Nekrashevich J.G, Nizametdinov S.U., Polkovnikov A.V., Rumjantzev V.P., Sourina O.N., Kalinin G.M., Sidorenkov A.V., Strebkov J.S.: Information and Computer-aided System for Structural Materials. *Journal of Nuclear Materials*, 191-194, (1992) 1042-1045.