# Processing and Visualisation Applet for Multidimensional Objects

Slava M. Yordanova [1]

*Abstract:* **In this issue are presented the methods for visualization of multidimensional objects (images). It is developed an applet, based on the Java language for demonstration of the visualization process. The N-dimensional space is divided into 4-D objects.**

*Keywords:* **Visualization, 2-D, 3-D, 4-D, Images, Translation, Rotation, etc.**

## I. INTRODUCTION

The main purpose of this investigation is developing of a new method for visualization of multidimensional objects. This problem become more popular with the penetration of the computer processing of the multidimensional objects in the communications, television and internet. Is used the Java language for presentation of the visualization process and N-dimensional space is divided into 4-D objects.

## II. VISUALIZATION METHODS OF MULTIDIMENSIONAL OBJECTS

### *A. By cutout.*

This method is based on the building of range of cutouts with N-dimensional plane. Obviously the cutout will be (N-1)-dimensional "volume".

### *B. By projection.*

Stereographic projection:

1. The sphere touches the plane.
2. The touch point is fixed and the opposite point (the north pole) is sent in the infinity.
3. Any other point of the sphere is projected in exact point from the plane.

### *C. By unfolding*

If a multiface has for faces regular multiangles , so it can be divided through the edges and can be unfolded that all his faces will lay in one plane.

## III. MULTIDIMENSIONAL GEOMETRY

Here are described the principles for developing of 3-D objects based on 2-D images. The investigation is concretized to 4 dimensions, then is made summary for N-D images through mathematical induction. The word dimension means here geometric dimension and the Euclid's spaces are only used.

### *A. Frame of reference.*

N – dimensional frame of reference is build on the base of (K-1)-dimensional frame of reference and is appended N-th axis perpendicular to all available N-1 axis [2, 3, 4].

### *B. Vectors.*

The vector P presents a vector with beginning the center of the frame of reference and end the point with coordinates (x, y). Length of the vector means (also called module of the vector) the distance between the center of the frame of reference and the point (x, y). 3D vector is defined analogically as 2D vectors. The 3D vector describes point in the 3-D space that has 3 coordinates – x, y, z, P(x, y). The vector in the N-dimensional space describes point with coordinates ($X_1$, $X_2$, …, $X_n$) and is a vector with beginning the center of the frame of reference and end the above mentioned point. [1, 2, 3]

Vector product in 4-dimensional space – conditions for its realization:

1. If the vectors – operands are linear independent, then the vector product is a zero vector.
2. If the vectors – operands are linear independent, then the result vector has to be orthogonal to each of them.
3. The vector product does net change the scale. For each constant K is true the following state:

$$k.X_4\left(\overrightarrow{U_0},\overrightarrow{U_1},\overrightarrow{U_2}\right)= X_4\left(k\overrightarrow{U_0}.\overrightarrow{U_1}.\overrightarrow{U_2}\right)= X_4\left(\overrightarrow{U_0}.k\overrightarrow{U_1}.\overrightarrow{U_2}\right)= X_4\left(\overrightarrow{U_0}.\overrightarrow{U_1}.k\overrightarrow{U_2}\right)$$

4. The change of position of two of the operands changes only the sign of the result vector.

### *C. Matrices.*

[1] Slava M. Yordanova, Technical University,
Faculty of Computational technique and electronics,
Varna 9000, Bulgaria,
E-mail: sl@windmail.net

For description of the transformations in the 3-D space is necessary 4x4 matrix. For description of the transformations in the N-d space will be necessary matrices with dimensions (N+1)x(N+1).

Lets take a vector product in the 3-D space. It can be examine as determinant of a matrix 3x3 with following values:

$$X_3\left(\vec{U},\vec{V}\right)=\begin{vmatrix} i & j & k \\ U_0 & U_1 & U_2 \\ V_0 & V_1 & V_2 \end{vmatrix}$$

Here the U and V are operands and i, j and k are the components of the result vector. Using that

$\vec{i}\left(U_1.V_2-U_2.V_1\right)-\vec{j}\left(U_0.V_2-U_2.V_0\right)-\vec{k}\left(U_0.V_1-U_1.V_0\right)$

the 4-dimensional matrix may be presented as 4-D vector product:

$$X_4\left(\vec{U},\vec{V},\vec{W}\right)=\begin{vmatrix} i & j & k & l \\ U_0 & U_1 & U_2 & U_3 \\ V_0 & V_1 & V_2 & V_3 \\ W_0 & W_1 & W_2 & W_3 \end{vmatrix}$$

The identification matrix for N-dimensional space looks like that:

$$\begin{vmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ . & . & . & \dots & . \\ 0 & 0 & 0 & \dots & 1 \end{vmatrix}$$

The calculation of 4-dimensional product is 2x2 determinants that are used more than once, so they are kept using a special algorithm in our program.

*D.Operations with vectors and matrices.*

The 3-dimensional space is defined by 3 main vectors called basic. The vector product in 3-D space is a binary operation. The N-dimensional space is defined by the N basics vector, so the vector product in this space will need (N-1) linear independent N-dimensional vectors that define N-th orthogonal to them N-dimensional vector [1, 2, 3].

1. If the vectors – operands are linear independent, then the vector product is a zero vector.
2. If the vectors – operands are linear independent, then the result vector has to be orthogonal to each of them.
3. The vector product does net change the scale.
4. The change of position of two of the operands changes only the sign of the result vector.

The calculation of 4-dimensional product consists of 2x2 determinants that are used more than once, so it is useful to be kept in the memory.

*E. Coordinates transformation.*

If exists a vector that describes a point in the referred space and this vector is multiplied with the corresponding transformation matrix, so the result vector will describe the transformed point.

All transformation matrix could be summarized in the so called global transformation matrix. In the 2-dimensional space the axis that the body rotates is perpendicular to this space, it means that it isn't in the space. The rotation in 3D is not a rotation around an axis but rotation parallel to a 2D plane. Exist 6 basic rotation matrices in the 4-D space that correspond to rotation around corresponding planes XY, YZ, ZX, XW, YW, and ZW [1, 2, 3]

## IV. 3-D SPACE VISUALIZATION.

1. It is necessary to be defined the view point. This is made by marking a point in the 3-D space from wich the observation will be made. This point is called fromview or viewpoint.
2. Definition of the line of the view. This is made by defining the vector by the line of the view or by defining a seeing point from the scene. This point is called to-point.
3. Scene orientation definition. This parameter is needed for correct view (because the view may be seen upside down)

To be done this, is defined a vector called up-vector. The up-vector defines the orientation of the watcher towards the line of the view so it can't be parallel to it.

If a perspective is used then it is necessary to be defined the scale factor that is used for final image generation.

## V. VISUALIZATION IN 4-D.

The bringout of common visualization model for $N^{th}$ – dimension is difficult task. Have to be concretized the study of definite amount of dimensions. The saturation of lines in design of higher dimension objects prevents the real study. That's why we pay attention of the 4-dimension space, where the number of the edges that are visualized over a 2-dimensional screen are a few and we can take idea of the studied object [2, 3].

The visualization of 3-dimensional space is reduced to developing of 3-dimensional scene over 2-dimensional screen. The visualization of 4-dimensional scene is a process of developing of a 4-dimensional scene over some 3-dimensional region that from other site may be presented with one of the known methods. The visualization parameters for the projection of 4D in 3D are similar to these of 3D in 2D.

*A. Visualization of 4D wireframe objects in 2D.*

The projection of 4D space in 2D inserts additional projection. As well as 3D projection, the additional 4D projection may be controlled with independent composition of visualization parameters [1, 2].

The first step in visualization of 4D wireframe objects is the process of developing of 4-dimensional peaks from the 4-dimensional space in intermediate 3D region.

The next step is to take already projected peaks in 3-dimensional space and to project them over a 2-dimensional

screen. This projection on the other hand is defined with 3-dimensional visualization parameters and also may be orthogonal or parallel.

*B. Description of projection 3D → 2D.*

The method that is used and expanded in this research is developed by Foley. It includes difference and product of 3-dimensional vectors and matrix 3x3 for each projected point. The first step in projection of a 3D point is the transformation of its absolute coordinates in relative (relative towards the observer). It's necessary to determine a vector with center the eye of the observer and points toward observed point. This vector difference has to be rotated in way that the observed point to lay on Z axis of the new coordinate system, and the up – vector on Y axis. This is made by multiplying the vector difference with the transformation matrix. The transformation matrix has 3x3 dimensions and its rows are A, B and C that correspond to X, Y and Z of the new coordinate system.

The algorithm for calculation of transformation matrix is:

Algorithm:
***Dot3(U, V)*** – returns the scalar product of two 3-dimensional vectors
***U*** and ***V*** – are parameters.
NumVerts: **Integer**// number of 3D peaks
Radius3: **Real** // Radius of the surrounding area around the peaks Va, Vb, and Vc
**Vector3** //Vectors that are situated in the columns of the transformation matrix
Vangle3: **Radians** //3D angle of observation
VertList: **array of Vertex** // massive of 3D peaks
**function** ProjectToScreen (Cx, Cy, Lx, Ly: **Real**)
S, T: **Real**
**V: Vector3** //temporary vector
**begin**
**if**
S=I/Radius3
**else**
T=I/tang (Vangle3/2)
**for** i=1 to NumVerts
V=VertList[i].Position3-Form3
**if**
S=T/Dat3(V, Vc)
tList[i].Screen[x]=Cx+(Lx*S*Dot3(V, Va))
VerList[i].Screen[y]=Cy+(Ly*S*Dot3(V, Vb))
**end** //if
**end** ProjectToScreen
norm=Norm3 (Vc)
**If** norm==0
Error (Up3 is parallel to the line of look)
Va=Va/norm //calculates Vb
Vb=Cross3 (Va, Vc)
**end** Calc3Matrix

As the vectors A, B, and C are calculated, so the coordinates of all 3D points can be transformed in the new coordinates. What we want to be the result is a projection that describes the points that lay over the cone of observation in rectangular region [-1, +1]x[-1, +1]. This rectangular region then is shown on the monitor.

*C. Rotation of 4D wireframe objects*

The program, written for this investigation, uses the movement of the mouse horizontally. The rotation of the view is realized by rotation of the view point. So we avoid rotation of all peaks from the scene. When the 3D view is rotated, it is not necessary to be recalculated the projections 4D → 3D. More effective is to be kept the values of the projected in 3D peaks and to be recalculated only the projections 3D → screen [2,4].

Basic steps in rotation of 3D viewpoint:

1) Rotation of 3D point *From* toward 3D point *To* in one of the planes.
2) Recalculation of 3D projection transformation matrix.
3) Calculation of the relative 3D coordinates of all peaks.
4) Representation of each wireframe peak.
5) Rotation of 4Dview toward 4D point To in some plane.
6) Recalculation of 4D projection transformation matrix.
7) Projecting all 4D points in 3D region.
8) projecting all 3D points to the screen
9) Representing all peaks.

## VI. PROGRAM REALISATION.

The Java programming language is selected because:

- Platform independence of Java.
- Easy code transportation written in Java (between platforms).
- Web orientation of Java.
- The popularity of the language make the source readable to wide range of specialists.
- For developing is used Java SDK 1.3 of Sun Company and as IDE update, RealJ editor

It is possible to be used programming language as C or C++. The problem with Java is the presence of virtual machine that slows down the process of visualization. In our case the delay is not irritating because wireframe objects are visualized, but if make a try of rendering or raytracing, the delay will be obviously and the study of the object will be impossible, because of loss of dynamics in communication with the user [1, 4].

## VII. RESULTS

The applet is realized for visualization with regular multifaces and regular polytopes via the projection method. The regular polytopes are 4-dimensional analog of regular multifaces. They are 6 types: simplex (analog of 3D tetrahedral), hypercube (analog of 3D cube), hyperoctahedral (analog of 3D octahedral), 24-volume, hyperdodecahedral. The number of the peaks, edges and faces are subordinated to the Oilier's formula.

V-E-F-C=0

Where: V – number of peaks
E – number of edges
F – number of faces
E – number of cells (3-dimensional)

In the common case the principle of applet's action is following: in the entrance is brought a range of 4-dimensiona

coordinates that describe a certain polytope and after corresponding processing is realized 4D perspective projection.

TABLE 1
REGULAR POLYTOPES, REALIZED WITH THE APPLET

| Body | Peaks | Edges |
|---|---|---|
| Simplex | 5 | 10 |
| Hypercube | 16 | 32 |
| Hyperoctahedral | 8 | 24 |
| 24-volume | 24 | 96 |
| Hyperdodecahedral | 600 | 1200 |
| Hypercosahedral | 120 | 720 |

The Euclid's space with dimensions higher than 4 there are 3 different types of regular multivolumes.: simplex (hypertetrahedral), hypercube, hyperoctahedral.

The general work of this applet is as follows: at the entrance is submitted amount of 4-dimensional coordinates that are describing the given polytope. After processing by corresponding rules 4D perspective projection is realized. The code of the program is divided to a few Java files that make the task more readable and correctional.

- AboutDialog.java – contains class (class About Dialog) that initializes the whole dialog window.
- FourD.java – contains class that manages the scroller for determination of the viewpoint (class MyScrollbar), class that manages the drawing process (class DrawPanel) and class that manages the user controls (class DrawControls).
- Matrix.java – contains class (class Matrix) that manages all matrix actions.
- Model.java – contains class (class Model) that describes visualized models except 120-cell and 600-cell models (they are too big so are separated in single files).
- Model 120.java
- Model 600.java
- Window.java – contains the class My Window that manages the entire communication.

## CONCLUSIONS.

The applet makes visualization of regular multifaces and polytopes. The following options are available: Selection of body for visualization and Rotation of visualized object toward current selected plane.

The plane is chosen from buttons on the working screen. The rotation could be started by left clicking with the mouse on the visualization screen and could be stopped by clicking one more time over it. Step by step rotation towards one of the planes could be made by pressing the corresponding button many times. Highlighting a face – The option "Faces Shown" must be switched on and the faces could be selected by pressing the buttons "Previous" and "Next". Zooming is made by the scrollbar at the left of the visualization screen.

## REFERENCES


[1] Allen I. "3D Coding BlackHole" http://pages.infinit.net/jstlouis/3dbhole/

[2] Jugert E., "Hyperspace structures- Exploring the fourth dimension http://www.lboro.ac.uk/departments/ma/gallery/hyper/

[3] Laine A. ahd Fan J. "Euler's formula in Higher Dimensions" state.edu/~fiedorow/math655/HyperEuler.html

[4] Lee,S. and Hsu,F.,"Viewing Four-dimensional Objects In Three Dimensions", Geometry.college Tue, 6 Sep 199