Abstract Aiming at applications to the scientific visualization of three-dimensional simulation data of dynamical systems, a keyboard-based control method to specify rotations in four dimensions is proposed. It is known that four-dimensional rotations are generally the so-called double rotations, and a double rotation is a combination of simultaneously applied two simple rotations. The proposed method can specify both the simple and double rotations by single key typing of the keyboard. The method is tested in visualizations of a regular pentachoron in four-dimensional space by a hyperplane slicing.

Keywords Four–dimension · Four-dimensional rotation · Four-dimensional visualization · Regular pentachoron · Regular polytope

1 Introduction

A three-dimensional (3D) computer simulation data with time evolution are regarded as a data field defined in a four-dimensional (4D) space. Spatio-temporal coherency in a 4D field can be found when the 4D data are appropriately rotated in the 4D space to mix the spatial and temporal coordinates, before being mapped to 3D or 2D images.

The rotations in four-dimensional euclidean space are generally the so-called double rotations. A double rotation has two independent angles with two fixed planes that are absolutely perpendicular each other (Cole 1890; Manning 1914). When one of the two rotation angles is zero, it is called simple rotation. The study of 4D rotations in the context of the visualization has a long history. Mathematical basis of multi-dimensional rotations and its applications in graphics can be found in Hanson (1995). Since a 4D rotation is represented by a composite of 3D rotations, sophisticated input methods for the 3D rotations, such as “rolling ball” (Hanson 1992) or “arcball” (Shoemake and Arceball 1992) can be used in 4D rotations. Various hardwares have been used to implement the 4D rotations, from joysticks (Banks 1992), mice (Hanson et al. 1999), haptic devices (Hanson and Zhang 2005), flight-controller pads (Sakai and Hashimoto 2011), head-tracked systems (Sakai and Hashimoto 2007), and to modern touch screens (Yan et al. 2012). Yan et al. developed a multitouch interface of figure gestures for 4D rotations (2012). Here we propose a much simpler approach to the 4D rotations that are based on key typings of the keyboard, rather than pursuing advanced input devices or methods.

Since a double rotation is equivalent to a product of two simple rotations (Cole 1890), it is reasonable to omit a specific user interface for the double rotations. However, one has to invoke two commands for simple rotations simultaneously for a double rotation, since an appearance of one step of a double rotation is not the
same as that of two simple rotations if the two simple rotations are applied step by step. Therefore, we assign in this paper a direct way to invoke the double rotations. One of interesting features of the double rotations is that a continuously applied double rotation has no period of rotation when the ratio between the two angles is irrational. It means that the rotated object never return to the original configuration even if the double rotation is continuously applied for ever. A simple rotation in 4D, on the other hand, has always a period 2π as in the rotations in 3D.

The final goal of this study is to visualize 4D data fields produced by time-varying, 3D numerical simulations. As Woodring et al. have shown (2003), mapping a 4D field into 3D fields by hyperplanes in 4D is effective to extract spatio-temporal coherency from the data. A hyperplane in 4D is written as

\[ c_0 + c_1x + c_2y + c_3z + c_4w = 0, \]  

where \( c_i \) for \( 0 \leq i \leq 4 \) are constants. In contrast to their “hyperslice” approach, in which general cases of the coefficients \( c_i \) are considered, we focus on the special cases of the type

\[ w = c_0. \]  

A slice of a 4D data field \( f(x, y, z, w) \) by the hyperplane of Eq. (2) is just a 3D field \( f(x, y, z, c_0) \), which is easily visualized by standard 3D data visualization techniques. Instead of considering general hyperplanes, we rotate the target 4D data in the 4D space under the fixed hyperplane. From this reason, we investigate extrinsic 4D rotations in which the target data are rotated about a fixed coordinates system. The keyboard-based control method proposed in this paper facilitates its user to keep track of the fixed planes.

We apply the proposed method to a visualization of a regular pentachoron which is the simplest figure in a 4D space, but a 3-sphere.

Before we describe the proposed method in the next section, here we briefly summarize some basic features of 4D rotations. In contrast to 3D rotations, there is no rotation “axis” for a 4D rotation. Take, for example, a 3D rotation around the \( z \) axis, \( R_z \), represented by the following matrix that transforms a point \( x_i = \{x_1, x_2, x_3\} = \{x, y, z\} \) to \( x'_i \):

\[
R_z(\alpha) = \begin{pmatrix}
\cos \alpha & -\sin \alpha & 0 \\
\sin \alpha & \cos \alpha & 0 \\
0 & 0 & 1
\end{pmatrix},
\]  

where \( \alpha \) is the rotation angle. It can be called a rotation in the \( x-y \) plane.

Let us consider a 4D rotation that transforms \( x_i = \{x_1, x_2, x_3, x_4\} = \{x, y, z, w\} \) to \( x'_i \):

\[
R_{xyzw}(\alpha, \beta) = \begin{pmatrix}
\cos \alpha & -\sin \alpha & 0 & 0 \\
\sin \alpha & \cos \alpha & 0 & 0 \\
0 & 0 & \cos \beta & -\sin \beta \\
0 & 0 & \sin \beta & \cos \beta
\end{pmatrix},
\]  

where \( \alpha \) and \( \beta \) are rotation angles. In this 4D case, the \( x \) and \( y \) coordinates are mixed with the angle \( \alpha \), and the \( z \) and \( w \) coordinates are mixed with the angle \( \beta \). A 4D rotation does not have a fixed “axis,” but it has a two fixed “planes” that are perpendicular each other in 4D space. A double rotation is a commutable product of two rotations. For example, \( R_{xyzw}(\alpha, \beta) = R_{xy}(\alpha)R_{zw}(\beta) \), where

\[
R_{xy}(\alpha) = \begin{pmatrix}
\cos \alpha & -\sin \alpha & 0 & 0 \\
\sin \alpha & \cos \alpha & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix},
\]  

and

\[
R_{zw}(\beta) = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & \cos \beta & -\sin \beta \\
0 & 0 & \sin \beta & \cos \beta
\end{pmatrix},
\]
that are called simple rotations.

As mentioned above, a slice of a 4D field \( f(x, y, z, w) \) by \( w = c_0 = \text{const.} \) generates a 3D field \( f(x, y, z) \). (Another major approach to the 4D visualization is the projection of 4D data into 3D space.)

This slicing procedure is a natural extension from the lower dimensions as shown in Fig. 1: The slicing of a 2D object (green circle) in the \( x-y \) space by a \( y = \text{const.} \) line (blue) makes a 1D object (magenta); the slicing of a 3D object (brown sphere) in the \( x-y-z \) space by a \( z = \text{const.} \) plane (blue) makes the 2D object (green); and the slicing of a 4D object (gray) in the \( x-y-z-w \) space by a \( w = \text{const.} \) hyperplane (blue) makes the 3D object (brown).

2 Interactive 4D rotations by keyboard

Here we propose a simple approach to 4D rotations based on the keyboard and factorization. We assign an integer to each of the four axes as follows:

\[
\begin{align*}
1 & \rightarrow x, \\
2 & \rightarrow y, \\
3 & \rightarrow z, \\
4 & \rightarrow w.
\end{align*}
\]

For a simple rotation \( R_{xy} \), for example, two integers 1 (for the \( x \) axis) and 2 (for the \( y \) axis) are multiplied and the key “2” \([= 1 \times 2]\) is pressed for the rotation. Similarly, for rotations in \( y-z \) plane, or \( R_{yz} \), is invoked by the key “6” \([= 2 \times 3]\). We use the hexadecimal notation, i.e., \( R_{zw} \) is invoked by the key “c” \([= 3 \times 4]\).

While a key is pressed, the assigned rotation to the key is continuously applied to the 4D object with a constant (usually small) angle \( \theta_0 \). When the Shift key is pressed, the inverse rotations, or negative angle rotations, are applied. For example, the rotation \( R_{zw} \), which is the inverse of \( R_{zw} \), is invoked by pressing Shift-c (or capital C) key.

The simple rotations are special cases when one of two angles in the double rotations is zero. There are three kinds of the double rotations under the rest frame: \( R_{xy,zw} \), \( R_{xz,yw} \), and \( R_{sx,zy} \). Focusing on the counterpart of the \( x \) axis, we assign three keys “y,” “z,” and “w” for these double rotations, respectively. The inverse rotations, such as \( R_{zw} = R_{zw}^{-1} \), are invoked again by pressing the Shift key.

The keys for the simple and the double rotations are summarized in Table 1.

The ratio between the two angles \( \alpha \) and \( \beta \) in the double rotations is also controlled by the keyboard. More precisely, when the “k” key is pressed, the angle \( \alpha \) is increased, keeping \( \alpha + \beta = \theta_0 \). Pressing the “j” key decreases the value of \( \alpha \), following the vi-editor’s convention. Either \( \alpha \) or \( \beta \) can be negative.

We will show, in the next section, examples of the proposed rotation method applied to a 4D object visualized by 3D slicing by a hyperplane. The hyperplane is \( w = c_0 \) in a four-dimensional \( x-y-z-w \) space.
The coordinate of the slicing hyperplane is $c_0 = 0$ in the default settings. The slice coordinate $c_0$ can be changed by pressing keys “l” or “h.” Following the vi-convention again, the “l” key increases $c_0$ and “h” key decreases it.

The above keys are summarized in Table 2.

### Table 1: The key assignment for specifying the simple and double rotations

| Simple rotation | Axes numbers | Key  |
|-----------------|--------------|------|
| $R_{xy}$        | $1 \times 2$ | “2”  |
| $R_{xz}$        | $1 \times 3$ | “3”  |
| $R_{xw}$        | $1 \times 4$ | “4”  |
| $R_{yz}$        | $2 \times 3$ | “6”  |
| $R_{zw}$        | $2 \times 4$ | “8”  |
| $R_{yw}$        | $3 \times 4$ | “c”  |

| Double rotation | $x$’s counterpart | Key  |
|-----------------|---------------------|------|
| $R_{xy,zw}$     | y                   | “y”  |
| $R_{xz,yw}$     | z                   | “z”  |
| $R_{xw,yz}$     | w                   | “w”  |

The inverse rotation for each of them is realized by Shift keys.

### Table 2: The key assignments for parameters changes. $\alpha$ is one of the two rotation angles for double rotations $R_{ijk}(\alpha, \beta)$

| Key   | Action                      |
|-------|-----------------------------|
| “k”   | Increase $\alpha$           |
| “j”   | Decrease $\alpha$           |
| “l”   | Increase $c_0$              |
| “h”   | Decrease $c_0$              |

The incremental angle for $\beta$ is automatically determined by $\alpha + \beta = \theta_0$, where $\theta_0$ is a constant angle for the simple rotations. The parameter $c_0$ specifies the slicing hyperplane at $w = c_0$.

The $w$ coordinate of the slicing hyperplane is $c_0 = 0$ in the default settings. The slice coordinate $c_0$ can be changed by pressing keys “l” or “h.” Following the vi-convention again, the “l” key increases $c_0$ and “h” key decreases it.

The above keys are summarized in Table 2.

### 3 Application to visualizations of a regular pentachoron

#### 3.1 A regular pentachoron

As a sample application of the 4D rotation, we visualize a 4D regular pentachoron. In geometry, the analogs in $n$ dimensions of the polyhedra in 3D space are called $n$-polytopes. Especially in 4D, a 4-polytope is called a polychoron. There are six regular polychora, one of which is the regular pentachoron. The regular pentachoron is represented as $\{3,3,3\}$ in the Schlafli symbol, indicating that it is a regular polychoron having three regular tetrahedra around each edge. Since it has five congruent tetrahedra, a regular pentachoron is also called as a 5-cell regular polychoron. The 4D coordinates $(x, y, z, w)$ of the vertices of a regular pentachoron used in the next subsection are

$$P_0 : \left( -\frac{a}{2}, -\frac{a}{2\sqrt{3}}, -\frac{a}{2\sqrt{6}}, -\frac{a}{2\sqrt{10}} \right),$$

$$P_1 : \left( \frac{a}{2}, -\frac{a}{2\sqrt{3}}, -\frac{a}{2\sqrt{6}}, -\frac{a}{2\sqrt{10}} \right),$$

$$P_2 : \left( 0, \frac{a}{\sqrt{3}}, -\frac{a}{2\sqrt{6}}, -\frac{a}{2\sqrt{10}} \right),$$

$$P_3 : \left( 0, 0, \frac{a\sqrt{3}}{2\sqrt{2}}, -\frac{a}{2\sqrt{10}} \right),$$

$$P_4 : \left( 0, 0, 0, \frac{2a}{\sqrt{10}} \right).$$

The inverse rotation for each of them is realized by Shift keys.

The $w$ coordinate of the slicing hyperplane is $c_0 = 0$ in the default settings. The slice coordinate $c_0$ can be changed by pressing keys “l” or “h.” Following the vi-convention again, the “l” key increases $c_0$ and “h” key decreases it.

The above keys are summarized in Table 2.
The center of gravity of this regular pentachoron is on the origin. These coordinates are constructed, for example, as follows.

Consider two points \( P_0 \) and \( P_1 \) on the \( x \) axis whose coordinates are \( p_0 = -a/2 \) and \( p_1 = a/2 \), respectively. See Fig. 2a. The distance between them is \( a \), and their center of gravity is on the origin.

We translate \( P_0 \) and \( P_1 \) in the negative \( y \) direction so that their coordinates are \( p'_0 = (-a/2, -y_0/2) \), \( p'_1 = (a/2, -y_0/2) \). We place the third point \( P_2 \) on the positive \( y \) axis at \( p'_2 = (0, y_0) \). See Fig. 2b. The center of gravity of the triangle \( P_0P_1P_2 \) is on the origin because \( \sum_{i=0}^{2} p'_{i} = 0 \) for any \( y_0 \). The distance between \( P_2 \) and \( P_1 \) is \( a \) if \( y_0 = a/\sqrt{3} \). From the symmetry, \( |P_2 - P_0| = |P_2 - P_1| \). The three points are on a circle of radius \( r_2 = y_0 \). The figure \( P_0P_1P_2 \) is a regular triangle with the edge length \( a \). A regular triangle is represented by \{3\} with the Schläfli symbol. It denotes a 3-sided regular polygon.

Keeping the above value \( y_0 = a/\sqrt{3} \), we translate the regular triangle \( P_0P_1P_2 \) in the negative \( z \) direction so that their coordinates are \( p''_0 = (-a/2, -y_0/2, -z_0/3) \), \( p''_1 = (a/2, -y_0/2, -z_0/3) \), and \( p''_2 = (0, y_0, -z_0/3) \). See Fig. 2c. We place the fourth point \( P_3 \) on the positive \( z \) axis at \( p''_3 = (0, 0, z_0) \). The center of gravity of thus constructed tetrahedron \( P_0P_1P_2P_3 \) is on the origin because \( \sum_{i=0}^{3} p''_{i} = 0 \) for any \( z_0 \). The distance between \( P_3 \) and \( P_i (0 \leq i \leq 2) \) is \( a \) if \( z_0 = \sqrt{3}/8a \). The four points \( P_i (0 \leq i \leq 3) \) are located on a sphere: with radius \( r_3 = z_0 \). The figure \( P_0P_1P_2P_3 \) is, therefore, a regular tetrahedron with edge length \( a \). See Fig. 2d. A regular tetrahedron is represented as \{3, 3\} with the Schläfli symbol, indicating that it is a regular polyhedron having three regular triangle faces around each vertex.

Finally, we construct a regular pentachoron in \( x-y-z-w \) coordinate space, following the same procedure. We translate the regular tetrahedron \( P_0P_1P_2P_3 \) in the negative \( w \) direction so that their coordinates are \( p'''_0 = (-a/2, -y_0/2, -z_0/3, -w_0/4) \), \( p'''_1 = (a/2, -y_0/2, -z_0/3, -w_0/4) \), \( p'''_2 = (0, y_0, -z_0/3, -w_0/4) \), and \( p'''_3 = (0, 0, z_0, -w_0/4) \). We place the fifth point \( P_4 \) on the positive \( w \) axis at \( p'''_4 = (0, 0, 0, w_0) \). The center of gravity of thus constructed pentachoron \( P_0P_1P_2P_3P_4 \) is on the origin because \( \sum_{i=0}^{4} p'''_{i} = 0 \) for any \( w_0 \). The distance between \( P_4 \) and other points \( P_i (0 \leq i \leq 3) \) is \( a \) if \( w_0 = \sqrt{2}/5a \). The five points \( P_i (0 \leq i \leq 4) \) are located on a hypersphere (3-sphere) with radius \( r_4 = w_0 \). \( P_0P_1P_2P_3P_4 \) is a regular pentachoron with edge length \( a \).

The regular pentachoron with center of gravity in Eqs. (8)–(12) is thus constructed.

3.2 A slice of a regular pentachoron by a hyperplane

There are two major approaches to the visualizations of 4D objects. One is to apply orthographic or perspective projections in 4D (Noll 1968; Sullivan 1991; Hanson and Heng 1991; Hoffmann et al. 1991; Hanson and Cross 1993; Hanson et al. 1999; Chu et al. 2009; Sakai and Hashimoto 2011). Another is to apply slicing with hyperplanes in 4D.

In the slicing approaches, a 4D object defined by a function \( f(x, y, z, w) = 0 \) is sliced by a hyperplane \( cx + cy + cz + cw = c_0 \), where \( c_x \) etc. are constants. For example, the slice of hyperplane \( w = 0 \) is a 3D object \( f(x, y, z) = 0 \) which can be observed in our 3D \( x-y-z \) space.

In contrast to the 4D projection method, the number of literatures on visualization of 4D objects by the hyperplane slicing is relatively small. Hausmann and Seidel developed a program to visualize four-dimensional regular polytopes (Hausmann and Seidel 1994). Their program can apply both the perspective projection and the hyperplane slices of the six regular polytopes in 4D. In their program, however, the slice is very restrictive: The hyperplane is always along one of symmetry axes of the polytope. A visualization of 4D object by a hyperplane slicing was investigated by Woodring et al. (2003) in which general cases for arbitrary slicing hyperplanes are investigated. Since the purpose of their study was to visualize time-varying, three-dimensional scalar field, the \( w \) axis corresponds to time and the 3D slices were visualized by the 3D volume rendering method.
Here we take the hyperplane slice approach to the 4D visualization of the regular pentachoron. The slicing hyperplanes are restricted to $w = 0$ and its parallels $w = \text{const}$. The 3D slice is visualized by OpenGL in C++.

Figure 3 shows a sample sequence of a simple rotation applied to the regular pentachoron $P_0 P_1 P_2 P_3 P_4$ in Eqs. (8)–(12) with $\alpha = 2$. The applied rotation is in the $x$–$w$ plane, or $R_{xy}(\theta)$, with $\theta = \pi/16$. The panels from (a) to (p) show slices of the regular pentachoron by a hyperplane $w = 0$. It is a 3D perspective view looking from $(x, y, z) = (0, 0, 5)$ to the negative $z$ direction. The panel (a) is the initial view of the 3D slice before rotation. It is a regular tetrahedron in 3D. The vertices, edges, and faces are shown by balls, bars, and semitransparent polygons, respectively. A vertex in 3D is a slice of an edge in 4D; an edge in 3D is a slice of a face in 4D; and a face in 3D is a slice of a tetrahedron in 4D. The panels (b) to (p) are slices by the same hyperplane $w = 0$ of the pentachoron when the simple rotation $R_{xy}$, by pressing “4” key, is continuously applied for every $\theta = \pi/16$ radian. After applying the $R_{xy}(\theta)$ rotations for 32 times, the regular pentachoron returns to the original configuration in 4D, whose 3D slice is shown in panel (a).
Figure 4 shows a sample sequence of a double rotation $R_{\text{xyz}}(a, b)$, where $a = \pi \sqrt{2/8\sqrt{3}}$ and $b = 1 - a$. The 4D regular pentachoron never return to the initial configuration, i.e., panel (a), in this double rotation.

Figure 4 shows a sample sequence of a double rotation $R_{\text{xyz}}(a, b)$, where $a = \pi \sqrt{2/8\sqrt{3}}$ and $b = 1 - a$. The 4D regular pentachoron never return to the initial configuration, i.e., panel (a), in this double rotation.

4 Summary

Various input devices and methods have been investigated to specify the rotations in 4D space. We have proposed a simple approach to 4D rotations based on the key typing of the keyboard. Both the simple and double rotations can be specified by single key typing. The rotations are supposed to be applied under a fixed frame of reference. The extrinsic rotation helps the user to keep track of the orientation of the object in the
4D space. We can swiftly apply arbitral rotations to the 4D object in a fully interactive way. We can also smoothly change the slice coordinate $c_0$ through the key typing when the hyperplane slicing is applied for the 4D visualization.

We have applied the proposed method to a visualization of a regular pentachoron by the hyperplane $w = 0$ and its parallels. The extension to the visualization of other regular polytopes in 4D is straightforward.

One of the problems to be solved in the present approach to the 4D visualization with the combination of the interactive keyboard rotations and the hyperplane slicing is that we cannot grasp a 4D structure, at once, under the fixed hyperplane $w = c_0$. A possible solution to this problem is to apply a lot of slices with slightly different $w = c_0$ coordinates at once and show them in a same window. This is a kind of an extension of the computed tomography to 4D. We are developing a program based on this approach and the results will be reported in future.

Acknowledgments This work was supported by JSPS KAKENHI Grant No. 20260052.

References

Banks D (1992) Interactive manipulation and display of surfaces in four dimensions. Proceedings of the 1992 symposium on interactive 3D graphics, pp 197–207

Chu A, Fu C-WW, Hanson AJ, Heng P-AA (2009) GL4D: A GPU-based architecture for interactive 4D visualization. IEEE Trans Vis Comput Gr 15(6):1587–1594

Cole FN (1890) On rotations in space of four dimensions. Am J Math 12(2):191–210

Hanson AJ (1992) The rolling ball. In: Kirk D (ed) Graphics gems III. Academic Press, San Diego, pp 51–60

Hanson AJ (1995) Rotations for N-dimensional graphics. In: Paeth A (ed) Graphics gems V. Academic Press, Cambridge, pp 55–64

Hanson AJ, Cross RA (1993) Interactive visualization methods for four dimensions. Proc Vis 93:196–203

Hanson AJ, Heng PA (1991) Visualizing the fourth dimension using geometry and light. Proc Vis 91:321–328

Hanson AJ, Ishkov KI, Ma JH (1999) Meshview: visualizing the fourth dimension. Computer Science Department, Indiana University, Technical Report, pp 1–5

Hanson AJ, Zhang H (2005) Multimodal exploration of the fourth dimension. Proc Vis 05:263–270

Hausmann B, Seidel HP (1994) Visualization of regular polytopes in three and four dimensions. Comput Gr Forum 13(3):305–316

Hoffmann CM, Zhou J (1991) Some techniques for visualizing surfaces in four-dimensional space. Comput Aided Des 23(1):83–91

Manning HP (1914) Geometry of four dimensions. Macmillan, New York

Noll AM (1968) Computer animation and the fourth dimension. Proceedings of the fall joint computer conference, part II (1968). ACM, pp 1279–1283

Sakai Y, Hashimoto S (2007) Interactive four-dimensional space visualization using five-dimensional homogeneous processing for intuitive understanding. Inf Media Technol 2:574–591

Sakai Y, Hashimoto S (2011) Four-dimensional mathematical data visualization via embodied four-dimensional space display system. Forma 26:11–18

Shoemake K (1992) Arcball: A user interface for specifying three-dimensional orientation using a mouse. Proceedings of the conference on graphics interface ’92 (1992), vol 92, pp 151–156

Sullivan JM (1991) Generating and rendering four-dimensional polytopes. Math J 1:76–85

Woodring J, Wang C, Shen HW (2003) High dimensional direct rendering of time-varying volumetric data. Proceedings of visualization ’03, pp 417–424

Yan X, Fu CW, Hanson AJ (2012) Multitouching the fourth dimension. Computer 9:80–88