University of Glasgow | School of Computing Science

Honours Individual Project Dissertation

Level 4 Project: An Investigation into Importance of Different Geometric Components when Extending a Cross Section of Four Dimensional Objects

Joe Subbiani
March 18, 2022

Abstract

Four dimensional space is a mathematical concept that cannot be visualised in its entirety by humans. Several paradigms exist to visualise four dimensional objects in 3D, but a viewer will only be able to see a fraction of a 4D object at one time.

Ray marching over signed distance functions provide a quick method of rendering 3D cross sections of a wide range of primitive 4D shapes. 3D cross sections of 4D objects provides a platform to develop extended visualisations to provide a user with more information about a 4D object. Such extensions have been explored in other literature. This paper aims to evaluate the effectiveness of a series of extensions to 3D cross sections of 4D objects; each representation providing the user with more information in order to help them interpret and manipulate the object they are handling.

TODO: Results of Experiment It was found that ...

Acknowledgements

I would like to thank my supervisor, Dr. John Williamson for the constant support and advice throughout the project, within and outside of the weekly meetings. Thanks to his guidance and support, I was able to take this project as far as I have, and fully explore the complexities of the fourth dimension.

Education Use Consent

Signature: Dominic Joe Subbiani Date: 24 January 2022

Contents

1 Introduction 1

1.1 The Fourth Dimension 1

1.2 Opportunities for Exploration 1

1.3 Motivation 1

1.4 Aims 2

2 Background 3

2.1 Representations of the Fourth Dimension 3

2.1.1 Extensions to the 3D Cross Section 3

2.2 Building a 4D Object 4

2.2.1 Mesh Based Rendering 4

2.2.2 Ray Marching Shaders 5

2.3 Rotating a 4D Object 5

2.3.1 The Problems with Rotation 5

2.3.2 Geometric Algebra and The Rotor 6

2.4 Interaction and Direct Manipulation 6

2.4.1 Methods of Interaction in 3D 6

2.4.2 Methods of Interaction in 4D 6

2.5 Teaching and Evaluation 7

3 Analysis and Requirements 8

3.1 Problem Specification 8

3.2 Limitations 8

3.3 Prioritisation Using MoSCoW 8

4 Design 9

4.1 Tutorial 9

4.2 Experiment 9

5 Implementation 10

5.1 Building 4D Objects With Ray Marching 10

5.1.1 Derivation of 4D Shapes 10

5.2 Extending 3 Dimensional Cross Sections - Representing 4D Objects 10

5.3 Rotation of a 4D Vector 10

5.4 Methods of Rotation 10

5.4.1 Swipe Based Input - Rotation About The Global Axes 10

5.4.2 4D Grab Ball - Rotation About The Local Axes 10

5.5 Texturing Ray Marched Objects 11

5.6 Building the Experiment 11

5.6.1 11

5.6.2 Data Collection 11

6 Evaluation 12

6.1 Aims 12

6.2 Experimental Design 12

6.2.1 Preliminary Research 12
6.2.2 Repeated Preliminary Experiments 12
6.2.3 Tasks and Parameters 12
6.2.4 Limitations 12
6.3 Results 12

7 Conclusion 13
7.1 Guidance 13

Appendices 14

A Appendices 14

Bibliography 15

1 Introduction

1.1 The Fourth Dimension

The world is described as three dimensional Euclidean space, conventionally referred to as $\mathbb{R}^3$. Three axes, conventionally labeled x, y and z, define the three degrees of translational freedom an entity can move within. All three axes that make up Euclidean space exist perpendicular to each other. Four dimensional space, conventionally referred to as $\mathbb{R}^4$, is the mathematical concept where a fourth perpendicular axes, conventionally labeled w, exists perpendicular to all of the other three dimensional axes. It is impossible to visualise four dimensions in its entirety and often has to be abstracted to an analogues 3D - 2D example to reason and understand the behaviour of four dimensional entities.

1.2 Opportunities for Exploration

Handling four dimensional space presents a number of opportunities and challenges that come with the desire to explore, interpret and manipulate higher dimensional objects. The most immediate challenge is that of being able to manipulate a four dimensional object. Rotating an object is a non trivial challenge and the most common method of rotation in 3D cannot be extended to higher dimensions. There are several ways in which users can interact with 3D objects. Another non-trivial challenge is finding intuitive ways in which a user can manipulate a 4D object through a two dimensional user interface.

To understand the orientation of an object, a user needs to be able to differentiate similar faces of that object from each other. Colour, patterns and textures can be applied to an object in order to assist in comprehending the current status of an object in comparison to an otherwise visually similar counterpart. An example of this is a cube rotated 90 degrees ($\frac{\pi}{2}$ radians) and the same cube rotated by 270 degrees ($\frac{3\pi}{2}$ radians).

In this paper several methods will be applied in an attempt to provide more information about a four dimensional object. Such methods include: viewing the object from other angles, attempts to visualise the object in its entirety, and abstractions of the rotation of an object in order to help understand higher dimensional rotations.

1.3 Motivation

Three dimensional space is very neat. There are three degrees of translational freedom expressed by the three axes x, y and z. There are a further three degrees of rotational freedom that are most commonly expressed as rotations about each axis. This supposed tidiness has lead to misconceptions that are so heavily rooted in the average persons understanding of geometry such that nearly every digital 3D system is built upon a mathematically impure foundation. In the 3D world, these misconceptions do not matter, and the mathematics is still sound. Quaternions, the most common method of controlling or interpreting rotation, is heavily used in gyroscopic devices, interactive digital 3D environments, animation and robotics. More abstract practices such as mathematics and physics, however, need to consider geometry in higher dimensional spaces.

A variety of fields within science and engineering utilise the visualisation of higher dimensional spaces to tackle a variety of problems. For any problem defined in n dimensional space, to understand it more intuitively, it is often suitable to reduce the dimensionality of the problem. However, when a problem is reduced to $\mathbb{R}^3$ or $\mathbb{R}^2$ it may become somewhat trivial. Visualisation of 4D objects can often server as the "bridge from the 'trivial cases' to the 'nontrivial cases'", (Zhou 1991). Such problems that utilise the visualisation of four dimensional space include: understanding differential geometry, collision detection, analysis of 3D objects in motion, and scalar-fields in 3D space; among others. (Zhou 1991)

TODO: what is the benefit of trying to teach people about it?

1.4 Aims

As per the motivation for this project; an effective tutorial must be compiled together in order to explain the foundations of four dimensional geometry. A system to view and interact with 4D objects must be developed and have the capability to showcase a wide variety of 4D shapes. Research into intuitive methods of user interaction should be explored, and a system to rotate an object without gimbal lock should be employed.

An evaluation will be conducted into what aspects of geometry are most important when trying to interpret higher dimensional spaces. This will be evaluated through a series of extensions to a three dimensional cross sections of four dimensional objects; each of which will focus on a different aspect of geometry. These representations will be experimentally evaluated using a series of tests that each focus on a different aspect of geometry, and how a cross section of a four dimensional object changes under rotation, as well as a users ability to manipulate the shape.

TODO: Research Questions
This project aims to answer

2 Background

2.1 Representations of the Fourth Dimension

There are many ways to represent the fourth dimension. Arguably the most intuitive understanding is that of a 3D cross section. As with how a sphere in $\mathbb{R}^3$ can be considered a a stack of infinitely thin circles that vary in size from top to bottom, along the z axis; a fourth dimensional sphere can be considered a stack of 3D spheres that vary in size as you move across the w axis.

The 3D cross section is limited given that only a slice of the shape can be seen at any given time. Stereographic projection attempts to rectify this by allowing a viewer to visualise as much of the shape as their field of view permits. Stereographic projection in $\mathbb{R}^3$ takes a 3D sphere and maps its geometry across a 2D plane. To map another object, for example a cube, its geometry must be projected beforehand onto the surface of a sphere (Radiya-Dixit 2017). Stereographic projection in $\mathbb{R}^4$ takes the spherical projection of a 4D object, and maps it across a 3D environment. A viewer can stand in the center of the 3D environment and observe the mapped geometry. This approach of interpreting four dimensional space is well suited to virtual reality, however it is complex to interpret in a meaningful way, and abstracting an understanding to lower dimensions is nontrivial.

2.1.1 Extensions to the 3D Cross Section

Through simple discussion with peers, a 3D cross section of a four dimensional object appears to be a far more intuitive concept to grasp compared against stereographic projection. TODO: tie in with aims to teach people (answering the questions) given it's relative simplicity in using lower dimensional abstractions to assist in understanding higher dimensional geometry.

An extension to a 3D cross section involves providing the viewer of the object with more information about the object they are dealing with. Several such extensions have already been explored conceptually or even fully developed.

Showing several cross sections besides one another, each offset along the w axis an increased amount is a simple means of trying to provide more information to the user about the entire shape of the object. For example: displaying several 2D circles that circularly increase and decrease in size besides one another provides a viewer with enough information to estimate that the object being displayed is likely a 3D sphere.

TODO: Figure of several cross sections

Kageyama (2015) proposed such a system that displays several cross sections of a 4D object along an ovular path, in order to try and provide a viewer with more information about the object. By displaying the several cross sections in an ovular path, compared to a linear path, Kageyama was able to display a far greater number of cross sections, allowing for a more detailed view of the four dimensional object in its entirety. Each cross section, as they move further away from the central slice, decreases in size. This avoids potential confusion in conveying the idea that travelling across the fourth dimension wraps around, forming a circular path. Scaling each cross section however, can produce some immediate confusion when first viewing an object. For example, taking several cross sections of a 4D cylinder that is extended along the *w* axis should produce several 3D spheres of the same size. Kageyamas visualisation will have each of these 3D spheres decrease in size such that a user could more likely interpret the shape to be a 3D cross section of a 4D sphere given it exhibits the same scaling behaviour. If a user interacts with the object using the appropriate keys they could observe the lack of change in the size of the 3D cross section and draw the correct conclusion. Whilst the ovular display does provide a great deal of extra information to a viewer, it is limited by its potential to be obscure when it comes to objects with similar cross sections.

Another interesting extension which arguably provides the user with more information about an object, is being able to view said object from other angles. For example: take a cone that is extended upwards to a point. From the top, a cross section of the object will appear as a circle. However from a side on perspective, the object will appear as a triangle.

TODO: Figure of multi-view

Matsumoto et al. (2019) proposed *Polyvision*, a virtual reality system to view four dimensional objects from multiple angles to quickly and effectively understand the shape being interacted with.

TODO: Limitations by matsumoto (VR is cool, doesnt use 4D rotation?, only used wireframes)

2.2 Building a 4D Object

Rendering three dimensional shapes on a 2D screen is an art that has evolved over time and ranging several disciplines. When Tron released in 1982, computers were not powerful enough to render triangle meshes in real time; as is the standard for 3D rendering nowadays. Similar films of that era, such as Dune 1984 or Star Wars the Original Trilogy often used a mix of practical effects and physically modifying the film. The team behind tron, however, opted to take advantage of machines to do some heavy lifting. In-spite of the technological limitations the team were able build a system to combine and render simple but smooth primitive objects. They used a system called Ray Marching (Sheppard 2010).

Today, it is standard to use triangle meshes to build and even render complex models in real time; and not just for film, but for real time interactive media such as video games. A mesh consists of vertices, edges and faces. A vertex being a point on the surface of an object that edges connect to. A face is formed by a closed loop of edges. Generally, a face will be made up of three vertices and three edges, creating the most simple 2D shape: A triangle; which form the surface of a 3D mesh.

2.2.1 Mesh Based Rendering

Both Bosch (2020) and Tianli (2018b) have developed real time interactive systems which display four dimensional objects using mesh based rendering. As illustrated by Tianli (2018b), 4D shapes are made up of cells, similar to how a 3D mesh is made up of faces. The most basic cell is the tetrahedron. A tetrahedron is made up of four triangles defined by faces, vertices and edges. The pentachoron, also known as a hyper-tetrahedron or a 5-cell, is the most primitive 4D object besides the hyper-sphere. The mesh of a pentachoron of course consists of 5 tetrahedral sub-meshes (cells).

Mesh based rendering allows the creation of very complex higher dimensional objects. Bosch (2020) built a novel game engine implementing four dimensional rigid body dynamics, as well as the rendering of a four dimensional world. On the other hand, Tianli (2018b) used the readily available Unity game engine to construct a hyper-cube and hyper-tetrahedron, demonstrating the flexibility of existing software built for two and three dimensions.

There are two main drawbacks to mesh based rendering: The time taken to construct 4D objects, and the complexity of translating four dimensional geometry to 3D. Simple 4D Platonic solids (Parker 2014) such as the hyper-tetrahedron (5-cell), hyper-cube (8-cell) and hyper-octahedron (16-cell) are fairly straightforward. Developing complex shapes such as a the hyper-dodecahedron (120-cell) or the hyper-iscosahedron (600-cell) becomes a complex and time consuming task. This complexity only increase when trying to develop curve faced shapes such as a hyper-sphere, torus or cone, which are smooth. Smooth geometry generally requires hundreds of vertices as well as smooth surface shading that only add to computational load and development time.

TODO: The complexities revolve around cross section (Tianli 2018a)

2.2.2 Ray Marching Shaders

Despite of advancements in computational power and rendering capabilities, ray marching has not lost its place in the world. Signed distance fields are still heavily used in volumetric rendering meta-ball geometry, collision detection and video game shaders, with ray marching. Whilst less intuitive and approachable than mesh based rendering, ...

TODO: quilez actively exploring ray marching for change
TODO: similar techniques used for volumetric rendering
TODO: foundation for ray tracing which is becoming increasingly popular

(Quilez) (The_Art_of_Code 2018)

TODO: Not novel to ray march 4D, but equations for objects would have to be derived

2.3 Rotating a 4D Object

In $\mathbb{R}^3$ there are three degrees of rotational freedom. As there are three axes, the method of rotating an object has commonly been considered as rotating about an axis. However, in $\mathbb{R}^4$ there are six degrees of rotational freedom, despite there only being four axes. Instead, rotation must be considered as a rotation about a plane formed by two axes. Therefore the six planes of rotation in $\mathbb{R}^4$ would be xy, xz, yz, xw, yw and zw.

2.3.1 The Problems with Rotation

Rotating an object in $\mathbb{R}^2$ is somewhat trivial. Given a vector v_{xy} rotating about an origin point within the xy plane, the vector will follow a circular path dictated by the following 2D rotation matrix:

$$v'_{xy} = v_{xy} \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}$$

Rotating an object in a space greater than two dimensions immediately becomes a nontrivial problem due to its non-commutativity. In dimensions greater than $\mathbb{R}^2$, rotation matrices can be composed into a homogeneous rotation matrix. Multiplying a vector with a such a matrix will produce a rotated vector, but doing so will likely encounter gimbal lock. Gimbal lock occurs when rotating an object considers each plane of rotation as independent from one another. As a result, if a particular plane becomes parallel with another, a degree of rotation is lost and the vector being rotated will not change as expected.

Introducing the quaternion: The quaternion is an extension of the complex number system. A quaternion has 4 components: x, y and z which describe the axis of rotation, and w which describes the amount of rotation. Quaternions do not suffer from gimbal lock and can be applied to each other to perform several rotations in series.

Quaternions have a problem: They consider rotations as a rotation about an axis. As discussed by Bosch (b), axial rotations are not appropriate as a generalised method of rotation across dimensions. For example, you would not consider a 2D object rotating about the z axis. It makes far more sense to stay within $\mathbb{R}^2$ and rotate about the xy plane.

2.3.2 Geometric Algebra and The Rotor

Geometric Algebra, often referred to as Clifford Algebra, is a field of mathematics describing vector space. Vector space is populated by multivectors, graded by their vector components. Most notably, multivectors are defined by their associative, distributive and anti-commutative properties (Baker). A grade zero multivector is a scalar number. a grade one multivector is just a vector. Grade two multivectors are known as bivectors, which build the foundation for the rotor. A trivector is a grade 3 multivector forming a parallelepiped from 3 vectors. Finally in $\mathbb{R}^4$ there is what is known as a pseudo-scalar.

A bivector b_{ab} is made up of two vectors a and b, and has, alongside its orientation in space defined by it's vector components, two important properties: The direction of rotation, from a to b, and an area, dictated by the parallelogram formed by its two vector components as illustrated by slehar (2014). A bivector in the reverse direction, i.e from b to a fulfills the anti-commutative property such that $b_{ba} = -b_{ab}$.

As mentioned above, a rotation should be considered as a rotation about a plane. A bivector defines the plane for a vector to rotate about. As demonstrated by Mathoma (2017), a rotor can rotate a vector following the principle that a double reflection forms a rotation (Mathoma 2016).

In a given vector space greater than $\mathbb{R}^2$, a multivector greater than grade zero can be projected onto the vector space basis blades. A basis blade is a the plane formed parallel to two perpendicular axes. The xy blade is often referred to as e_{xy} or e_{01}. Therefore a bivector in $\mathbb{R}^3$ can be decomposed into three projections onto the e_{xy}, e_{xz} and e_{yz} planes; and a bivector in $\mathbb{R}^4$ can be decomposed into 6 elements.

It is possible to multiply multivectors using the geometric product; a combination of the inner and outer product. Multiplying a vector v by a rotor R and it's reverse $R^\dagger$ will rotate vector about the bivector component of the rotor.

$$v' = RvR^\dagger$$

As shown by (Bosch a), the quaternion and the 3D rotor are very similar in concept. The rotor, unlike the quaternion, has the advantage of not having to leave the dimension it belongs in order to rotate a vector, allowing it to be used in n dimensions.

2.4 Interaction and Direct Manipulation

2.4.1 Methods of Interaction in 3D

local, grab ball driven

global, swipe gestures (often multi-touch)

(Shoemake 1994)

2.4.2 Methods of Interaction in 4D

(Murata and Hashimoto 2000) (Kageyama 2005)

2.5 Teaching and Evaluation

(saf 2012)

3 Analysis and Requirements

self defined project with the goal of teaching people about abstract mathematical concepts

3.1 Problem Specification

teach people, and develop something suited to teaching people

3.2 Limitations

time to develop

time per user to experiment
- quantity of quality

3.3 Prioritisation Using MoSCoW

explanation of MoSCoW

Must Have

be able to render multiple 4D shapes

multiple extensions to a 3d cross section of 4d space

create a tutorial video to teach and explain 4d - must cover the geometry - must cover the rotation

Should Have

be able to interact with them in 4D directly

multiple tests to evaluate a persons understanding

Could Have

multiple methods of direct interaction/manipulation

What is the problem that you want to solve, and how did you arrive at it?

4 Design

4.1 Tutorial

requirements for teaching

- explain 4D space, why it cannot be visualised - explain 4D rotation in relation to 2D and 3D rotation

4.2 Experiment

requirements for testing

- be able to understand the shapes - be able to interpret the rotation of a shapes - be able to manipulate a shape

5 | Implementation

5.1 Building 4D Objects With Ray Marching

shapes defined with signed distance functions

relative to the camera - slice is based on W coordinate of camera based

algorithm

5.1.1 Derivation of 4D Shapes

5.2 Extending 3 Dimensional Cross Sections - Representing 4D Objects

timeline: simple - ellyptical version - ellypitcal adds more information, but not as intuitive as straight line
onionskin - similar to timeline, but not very intuitive for users unfamiliar with 4D
multi-view: polyvision
abstraction of 4D rotation using 3D rotation - does not extend well to n dimensions unlike all other propersitions

5.3 Rotation of a 4D Vector

(Bosch 2011) (Bosch a)

initial manual derivation

sympy and galgebra

equations

5.4 Methods of Rotation

5.4.1 Swipe Based Input - Rotation About The Global Axes

intuitive
does not define "how rotated" 4D axes are relative to the local coordinates of the shape

5.4.2 4D Grab Ball - Rotation About The Local Axes

idea
progress
why it failed

5.5 Texturing Ray Marched Objects

normal based projection

5.6 Building the Experiment

5.6.1

5.6.2 Data Collection

SimpleJSON

Email - challenges
Copy & Paste - challenges

6 Evaluation

6.1 Aims

6.2 Experimental Design

6.2.1 Preliminary Research

what is the best way to explain concepts
try teach some friends the basics and see what explanations work best

the best ways to teach people about geometry
- reference that paper about 5 stages or something

between users experiment - test user with every representation

6.2.2 Repeated Preliminary Experiments

build system
test it
improvements

Need to provide clear explanations of concepts early on

provide time to play with object before tests to learn controls and behaviour

6.2.3 Tasks and Parameters

shape match
- all shapes
- diffuse texture - patterns give it away

rotation match
- not diffuse texture and no sphere - no surface imperfections and can be impossible to tell if certain rotations occuring
- 1-2 4D rotations and occasionally 3d rotations

pose match
- randomly oriented match shape

random order of representations

6.2.4 Limitations

6.3 Results

7 Conclusion

Summarise the whole project for a lazy reader who didn't read the rest (e.g. a prize-awarding committee).

7.1 Guidance

- Summarise briefly and fairly.
- You should be addressing the general problem you introduced in the Introduction.
- Include summary of concrete results ("the new compiler ran 2x faster")
- Indicate what future work could be done, but remember: **you won't get credit for things you haven't done**.

A Appendices

Typical inclusions in the appendices are:

- Copies of ethics approvals (required if obtained)
- Copies of questionnaires etc. used to gather data from subjects.
- Extensive tables or figures that are too bulky to fit in the main body of the report, particularly ones that are repetitive and summarised in the body.
- Outline of the source code (e.g. directory structure), or other architecture documentation like class diagrams.
- User manuals, and any guides to starting/running the software.

TODO: PDF of Google Forms Questionnaire

Don't include your source code in the appendices. It will be submitted separately.

7 Bibliography

The van Hiele Model of Geometric Thinking. pages 72–75, Praha, 2012. Matfyzpress. ISBN 978-80-7378-224-5. URL https://www.mff.cuni.cz/veda/konference/wds/proc/pdf12/WDS12_112_m8_Vojkuvkova.pdf.

M. J. Baker. Maths - Clifford Algebra - Arithmetic. URL https://www.euclideanspace.com/maths/algebra/clifford/algebra/index.htm.

M. t. Bosch. Code Template for Rotor3 and Quaternion in C++, a. URL https://marctenbosch.com/quaternions/code.htm.

M. t. Bosch. Let's remove Quaternions from every 3D Engine, b. URL https://marctenbosch.com/quaternions/.

M. t. Bosch. 4D Rotations and the 4D equivalent of Quaternions, May 2011. URL https://marctenbosch.com/news/2011/05/4d-rotations-and-the-4d-equivalent-of-quaternions/.

M. t. Bosch. N-dimensional rigid body dynamics. *ACM Transactions on Graphics*, 39(4), July 2020. ISSN 0730-0301, 1557-7368. doi: 10.1145/3386569.3392483. URL https://dl.acm.org/doi/10.1145/3386569.3392483.

A. Kageyama. Keyboard-based control of four-dimensional rotations. *Journal of Visualization*, (19):319–326, Sept. 2005. URL https://link.springer.com/article/10.1007/s12650-015-0313-y#ref-CR4.

A. Kageyama. A visualization method of four-dimensional polytopes by oval display of parallel hyperplane slices. (19):417–422, Nov. 2015. URL https://link.springer.com/article/10.1007%2Fs12650-015-0319-5.

Mathoma. Geometric Algebra in 2D - Two Reflections is a Rotation, Nov. 2016. URL https://www.youtube.com/watch?v=Hy2gbdbrJZ8&list=PLpzmRsG7u_gqaTo_vEseQ7U8KFvtiJY4K&index=7.

Mathoma. Geometric Algebra - 3D Rotations and Rotors, Apr. 2017. URL https://www.youtube.com/watch?v=iwQlrgAduMg&list=PLpzmRsG7u_gqaTo_vEseQ7U8KFvtiJY4K&index=15.

K. Matsumoto, N. Ogawa, H. Inou, S. Kaji, Y. Ishii, and M. Hirose. *Polyvision: 4D Space Manipulation through Multiple Projections*. PhD thesis, The University of Tokyo, Kyoto University, Kyushu University, 2019. URL https://www.math.kyoto-u.ac.jp/~kaji/papers/SIGGRAPH2019_Polyvision_final.pdf.

M. Murata and S. Hashimoto. Interactive Environment For Intuitive Understanding of 4D Object and Space. In *Multimedia Modeling*, pages 383–401, Nagano, Japan, Oct. 2000. WORLD SCIENTIFIC. ISBN 978-981-02-4489-7 978-981-279-199-3. doi: 10.1142/9789812791993_0024. URL http://www.worldscientific.com/doi/abs/10.1142/9789812791993_0024.

M. Parker. *Things to make and do in the fourth dimension*. Particular Books, 2014.

I. Quilez. Distance Functions. URL `https://www.iquilezles.org/www/articles/distfunctions/distfunctions.htm`.

A. Radiya-Dixit. Visualizing the Fourth Dimension, Apr. 2017. URL `https://researchblog.duke.edu/2017/04/26/visualizing-the-fourth-dimension/`.

S. Sheppard. TRON Legacy: Oh how far we've come in computer rendering, Dec. 2010. URL `https://labs.blogs.com/its_alive_in_the_lab/2010/12/tron-legacy.html`.

K. Shoemake. *Arcball Rotation Control*. PhD thesis, University of Pennsylvania, Philadelphia, 1994. URL `https://research.cs.wisc.edu/graphics/Courses/559-f2001/Examples/Gl3D/arcball-gems.pdf`.

slehar. Clifford Algebra: A visual introduction, Mar. 2014. URL `https://slehar.wordpress.com/2014/03/18/clifford-algebra-a-visual-introduction/`.

The_Art_of_Code. Ray Marching for Dummies!, Dec. 2018. URL `https://www.youtube.com/watch?v=PGtv-dBi2wE`.

Z. Tianli. 2D Cross-Section in Unity, May 2018a. URL `https://luicat.github.io/2018/05/19/2D-clipping-in-unity.html`.

Z. Tianli. *4D Rendering: Projection & Cross Section*. PhD thesis, KTH Royal Institute of Technology, May 2018b. URL `https://luicat.github.io/2018/05/23/4D-rendering.html`.

J. Zhou. *Visualization of Four Dimensional Sace and Its Applications*. PhD thesis, Purdue University, Dec. 1991. URL `https://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1921&context=cstech`.