Higher Dimensional Graphics:
Conceiving Worlds in Four Spatial Dimensions and Beyond

M. Cavallo

IBM Research, Yorktown Heights, NY

Abstract

While the interpretation of high-dimensional datasets has become a necessity in most industries, the spatial visualization of higher-dimensional geometry has mostly remained a niche research topic for mathematicians and physicists. Intermittent contributions to this field date back more than a century, and have had a non-negligible influence on contemporary art and philosophy. However, most contributions have focused on the understanding of specific mathematical shapes, with few concrete applications. In this work, we attempt to revive the community's interest in visualizing higher dimensional geometry by shifting the focus from the visualization of abstract shapes to the design of a broader hyper-universe concept, wherein 3D and 4D objects can coexist and interact with each other. Specifically, we discuss the content definition, authoring patterns, and technical implementations associated with the process of extending standard 3D applications as to support 4D mechanics. We operationalize our ideas through the introduction of a new hybrid 3D/4D videogame called Across Dimensions, which we developed in Unity3D through the integration of our own 4D plugin.

CCS Concepts

• *Human-centered computing* → *Interaction techniques;* • *Computing methodologies* → *Shape modeling;* • *Applied computing* → *Computer games;* • *Software and its engineering* → *Virtual worlds software;*

1. Introduction

Conceiving of and visualizing a reality characterized by more dimensions than a human can perceive represents a long-lasting problem that is both intellectually and computationally difficult to solve. Mathematicians have generalized tridimensional concepts, formulating the existence of higher dimensional geometries such as knotted spheres [Ban02], while physicists have theorized phenomena occuring in higher dimensional spaces (e.g. superstring and quantum theories) [McM08]. Literature, philosophy, and art have also contributed to the notion of extra spatial dimensions [Hen13], offering such conceptions as the neoplatonic concept of "cosmic consciousness" and Dante's hyperspheric cosmos in "La Divina Commedia" [Egg99], as well as theological literature suggesting the existence of divine entities in higher dimensions [Wil93]. Projection into two- and three-dimensional spaces is the standard technique for enabling humans to experience higher dimensional objects, but comes at the cost of introducing ambiguity and losing geometrical information such as symmetry and curvature [CD18]. Over the years, computer graphics and visualization research has attempted to enhance intuitive user experiences of these abstract geometric shapes through the introduction of visual cues (e.g. depth perception, colors, shadows), new interactive methods of manipulating 4D objects, and novel display methods [HC93, McM08, Yan15].

In this work, rather than trying to improve upon existing visualization or projection techniques for specific geometry, we aim to present a comprehensive approach of how standard 3D graphics pipelines can be extended as to conceive and interface higher dimensional, and more specifically, four-dimensional worlds. By "dimensions" we refer to the independent directions into which a geometrical space extends—not including *time*. We start by introducing a simple 4D rendering engine called *Continuum* (implemented as a Unity3D plugin), and we observe how basic concepts such as movement and interaction performed in higher dimensions manifest for users viewing projections of these apparently ineffable worlds. We then draw on the laws of physics to lay out possible ways of structuring a higher dimensional universe, and identify which content generation and authoring methods could be used to populate such a universe. We qualitatively evaluate our work through a new videogame called *Across Dimensions*, showcasing how 4D and 3D elements can be seamlessly combined in a concrete application. Finally, we conclude by discussing the challenges and potential benefits of applying higher dimensional concepts to existing 3D graphics applications.

2. Related Work

In the first part of this section we focus on the formulation of concepts of *higher dimensions* in the twentieth century, noting how this field lies at the intersection of science, art, and philosophy. Then we review more recent computer graphics and data visualization attempts at handling 4D geometry.

2.1. A Brief History of Higher Dimensions

During the last two decades of the nineteenth century, scientific innovations such as Röntgen's discovery of the X-ray, Rutherford's and Curie's work on radioactivity, Thomson's identification of the electron, and Hertz's studies on electromagnetic waves profoundly altered conceptions of matter and space, radically expanding our understanding of the physical world and acknowledging the limits of human perception. Theories on the existence of a suprasensible reality began to thrive [Hen13, Zöl82, Ste78], and our world was first described as a mere boundary of an unperceivable higher-dimensional reality [Ste78]. The first major popularization of a spatial fourth dimension can be found in Abbot's 1884 novel *Flatland* [Abb84], wherein the author describes how inhabitants of a fictional two-dimensional world might refuse to accept the possibility of higher dimensions. Distortion, denial of the three-dimensional perspective, multiple vanishing points, and high dimensional space continuity soon became recurring elements in the ineffable realities depicted by artists such as Picasso and Boccioni [Hen13]. Einstein's 1905 Special Theory of Relativity [Mil98] and Minkowski's 1908 formulation of the *space-time continuum* [Pet10], however, led to a gradual transformation of the concept of the fourth dimension from *space* to *time*. In the geometrical interpretation known as the *block universe* [Jea31], space and time are geometrically welded together in a four-dimensional volume, and human life simply progresses through 3D cross-sections of this higher dimensional structure. Playful speculation on four-dimensional geometry continued to appear in the work of artists such as Davis, Dalì and Duchamp [B*14], and evolved into the concepts of *hyperspace* and *cyberspace* in cinematography and science-fiction [Hei41, Nah01, You20, Kak16, Wer00, Ben91]. While a fascinating research direction, the interpretation of time as a geometrical dimension in its own right will not be explicitly covered in this work.

It was only towards the end of the twentieth century that the concept of a fourth dimension was reinterpreted as a suprasensible dimension of *space*, within which our 3D world would exist merely as a section or boundary. The first world conference on four-dimensional graphics (or "hypergraphics") was held in 1984 [Ban90b], on the occasion of the centennial anniversary of the publication of Flatland. During the same year, *superstring theory* emerged in physics as a possible "theory of everything", and attempted to achieve a grander unification scheme by coupling gravitation and electromagnetism, supporting the existence of up to eleven compactified dimensions of space [Pau84]. The advent of digital media and personal computers, with their increased graphics and visualization capabilities, were among the main contributors to this reignited interest in higher spatial dimensions. Michael Noll [Nol67] had already demonstrated in 1967 how a computer can easily construct higher dimensional content, while Arnold [Arn72] had proposed initial studies of how the human brain might retain 4D information [VF03]. The field of four-dimensional computer graphics flourished between the 1980s and the 1990s through the work of authors such as Banchoff and Strauss [BS78, Ban90a, Bri87], and was accompanied by emerging art, cinematic and architectural endeavors to depict an invisible, "higher" reality [Ruc12, Ruc14, RHR92, Nov98]. A decade later, superstring theory would be accompanied and later superseded by M-theory [Duf96], according to which extra-dimensions are compactified in curled-up membranes [Pas06].

2.2. Recent Attempts at 4D Geometry Visualization

All methods of representing 4D content to the human eye, either on a flat screen or in virtual environments, require reducing the dimensionality of the data. To compensate for the loss of visual information caused by this process, researchers have developed over time a set of dedicated visualization techniques. Early work on higher dimensional rendering focused on the use of color, texture, shadows and lighting to indicate 4D depth and create intuitively informative shaded images of mathematical 4D objects [HH92b, HC93, HS94]. Grid planes and hypervolume removal techniques were used to visualize vector fields [HH92a, BPRS98], whereas the visualization of silhouettes, auxiliary vectors, and surface normals was used to improve the understanding of spatial relationships in hypersurfaces [HZ91, Ban92, MNSH17]. Other experiments include attempts at "voxelizing" 4D geometry through hypercubes [BS07], constructing isosurfaces through convex hulls [BWC04], adding volumetric halos to hyper-surfaces [WYF*13], and visualizing trajectories of high-dimensional dynamical systems through 3D parallel coordinates [WLG97]. While parallel research focused on *time* as the fourth dimension, and attempted to visualize 3D movements or shape changes in a flat space-time continuum [SH07, OLS17a, OLS17b, OBL13], *time* was also incorporated into the understanding of 4D geometry through real-time, interactive manipulation of 4D objects [HC93]. This was progressively made possible thanks to the improvements in rendering performance and hardware optimization, and to the introduction of the first GPU-accelerated tetrahedron-based rendering pipelines [CFHH09].

Based on the notion that an understanding of shapes benefits from a combination of sight, touch, and other senses [ZWR14], various multimodal interactive manipulation approaches have been proposed. Examples include the use of multitouch interactions [YFH12, ZWR14], tangible interfaces [LRY*15], haptics and force-feedback [ZH07], standard and custom controllers [HIM99, Kag16a, SH11b], and even user positional input [Agu06]. Particular attention has been paid to visual and sensory cues allowing one to visualize 4D collisions and to identify smooth, non-intersecting hypersurfaces (e.g. knotted surfaces embedded in 4D), whose 3D projections are degraded by self-intersections [HZ05, ZH07]. Visualization of often semi-transparent, "ghost geometries" has also been used to preview the projection effects of 4D manipulations [LRY*15, Kag16b], together with the application of multiple simultaneous projections [MOI*19]. Several studies on spatial perception have also been performed to assess if and how the human brain could understand higher dimensions, and form mental images of such spaces [MSH15]. Ambinder et al. [AWC*09] have shown evidence that people can learn to make spatial judgments on the length of, and angle between, line segments embedded in 4D space, while Ogmen et al. [OSY19] studied the inherent capabilities and limitations of brain plasticity in building perceptual and sensorimotor 4D representations. One interesting experiment involved assessing whether a human could find an exit from an n-dimensional labyrinth [Jam09]—representing a rare attempt at creating a multidimensional computer game. Seeking to improve 4D perception through stereoscopic vision, researchers moved from using standard flat screens to the adoption of glass-free 3D displays [SH11b], immersive environments such as CAVE systems [CH94, Agu06], and augmented or virtual reality devices [LRY*15, Kag16b, MOI*19]. Recent work by M.T. Bosch has focused on physics applied to four-dimensional "toys" [Bos20], generalizing the rules of how objects bind together, slide, and roll to
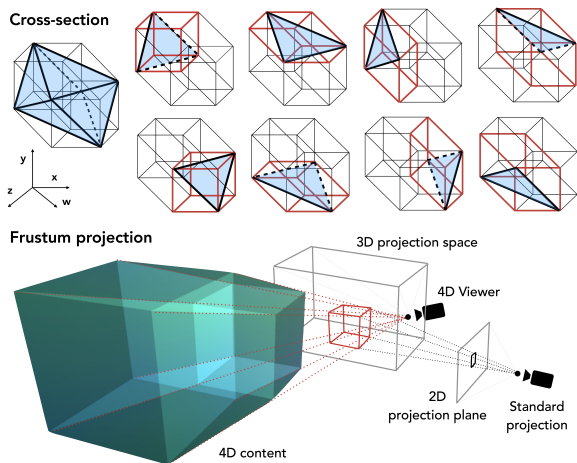
Figure 1: *Projection methods. Cross-section slices 4-dimensional objects based on their intersection with an arbitrary hyperplane. In this example, the cross-section of a rotated tesseract is constructed by merging the polyhedra generated by the intersection of its 4D geometry with a hyperplane. Frustum projection (or perspective) ensures objects further away from the viewer along w appear smaller. We implement this through the use of a 4D camera which generalizes the standard frustum projection, and generates the 3D meshes that will be later rendered by the usual 3D camera.*

higher dimensions. The videogame Miegakure [Bos], currently under development, leverages tetrahedral meshes and cross-section to provide a "hide and reveal" puzzle/adventure experience, and represents one of the possible use cases covered by our work.

Different from existing literature, this paper attempts to go beyond the visualization of isolated, abstract 4D objects, instead conceiving of a more general 4D universe, where multiple 4D and 3D entities can coexist and interact with each other in concrete use cases. We also note how prior work generally relies on either cross-section (mainly used for interactive manipulation) or frustum projection (more common in illustrations or educational applications), with very few examples [MSH13b, MSH13a] mentioning camera use and the control of vanishing points. Here we use both cross-section and frustum projection methods, and compare their usage in interactive applications where the *user*—not the 4D model anymore—is the protagonist.

3. Continuum

In this section we briefly introduce Continuum, our 4D rendering engine, which has been used to perform the experiments and to render the figures shown in this paper. The development of Continuum was not aimed at creating a fully featured, state-of-the-art, or high performance engine. Rather, it was conceived as very minimal, and focused on achieving our research goals. We implemented Continuum as a Unity3D plugin, which allowed us to leverage the tools already offered by the popular 3D game engine and make Continuum more accessible. Below we briefly describe how our engine extends the standard 3D approach to geometry definition, projection, and rendering, emphasizing how these newly introduced 4D concepts are perceived by the viewer. We refer our readers to existing literature for more specific 4D concepts and algorithms (e.g. [McM08, CFHH09, Bos20]).
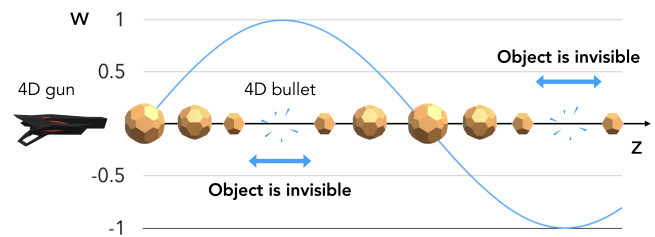
Figure 2: *4D Translation. When a 4D object moves along the fourth dimension, its geometry is differently sliced based on the current cross-section hyperplane, producing a visual effect similar to morphing. When there is no intersection with the hyperplane, the projection does not generate a mesh, making the 4D object "disappear." If we imagine shooting a bullet with a hypothetical 4D gun, generating a sinusoidal movement along w, the perceived effect would be a bullet continuously entering and exiting the cross-section (i.e. appearing and disappearing).*

3.1. Projection

Projection is the main method we use to reduce dimensionality of 4D objects and render them on screen. Our approach involves storing 4D geometry data in dedicated memory buffers as *tetrahedral* meshes, and applying a custom projection to obtain associated *triangle* meshes. At this point, objects can be fed through the standard 3D graphics pipeline and rendered on screen. This approach implies the existence of two independent viewers (or cameras): 1) our custom 4D projector, which transforms the input 4D geometry into 3D meshes, and 2) the standard 3D view frustum projection, which enables rendering 3D scenes on screen and is already available in Unity 3D. We enable the 4D to 3D projection step to be performed in two different ways: *cross-section* and *frustum projection*.

Cross-section. This method slices 4D objects based on their intersection with an arbitrary hyperplane. Let's consider as a 3D analogy the polygonal cross-section generated by the intersection of an arbitrary plane with a cube. Each edge of the polygon generated by such intersection lies in one of the six faces bounding the cube (which is why the polygon can have a maximum of six sides). Similarly, the intersection of a hyperplane with a tesseract generates a 3D polyhedron, whose polygonal faces each lie in one of the tesseract's eight bounding cubes (Fig. 1). Our cross-section implementation follows the same principle by finding the intersection of a hyperplane with the 4D geometry, defined using the tetrahedron (3-simplex) as a building block. Each intersecting tetrahedron results in a triangle or quadrilateral, and represents one face of the projected 3D mesh. If the intersection is empty, the projection will not produce any 3D output. In Continuum we define the hyperplane used in the cross-section through a 4D pose (translation + orientation), so that it is straightforward to associate it with the concept of a "4D camera".

Frustum projection (or perspective). This type of projection forces objects to appear smaller as they get further away from the viewer along the *w* axis. If we visualize the frustum projection of a tesseract, we notice a second, smaller cube inside the larger cube that we would not normally observe in cross-section. This second cube is one of the eight bounding the tesseract, and it appears smaller because it lies further away along the *w* axis. We note

how the 2D rendering of the tesseract in Fig. 1 has two distinct vanishing points, which are associated with our 4D camera and the standard 3D camera, respectively. While outside the scope of this paper, it is worth reminding that frustum projections of certain 4D geometry may self-intersect [ZV00]. Continuum does not directly address such cases, since their implications on the user experience vary based on the type of application.

3.2. Transformations

Below we describe how standard transformations such as translation, rotation and scaling generalize to four dimensions, and which perceived effects these transformations create on the part of the viewer.

Translation. In cross-section, a 4D object moving along w can be differently sliced based on its intersection with the current hyperplane, until all of its vertices exit the hyperplane, making the object itself invisible to the eye. In Fig. 2, for instance, an imaginary weapon shoots a bullet in the shape of a hyperdodecahedron (also known as a "120-cell"), enforcing upon such bullet a sinusoidal movement along w. The effect perceived by the viewer is that of a morphing geometry that periodically appears on and disappears. In frustum projection, an object moving further away along w is perceived as smaller, while an object moving closer becomes larger. If the 4D viewing frustum and the standard 3D frustum do not share the same x, y, and z coordinates, the object can be perceived as increasingly warped the closer it gets to the 4D camera (examples of this effect can be seen later in Fig. 6 and Fig. 7).

Rotation. Rotations in four dimensions do not generalize to an additional rotation around the w axis. The number of *principal rotations*, in fact, is given by the number of distinct *pairs* of orthogonal axes. As shown in Fig. 3, the four axes generate **six** orthogonal planes, which we denote as x, y, z, t, u, and v for brevity (with t, u, and v indicating the xw, yw, and zw planes, respectively). Rotation around the first three can be referred to as the 3D rotation of a 4D object, and does not reveal any information about the higher geometry of the object. Meanwhile, rotating a tesseract around the t, u, v planes produces an interesting visual effect: in cross-section, the 3D projection of the tesseract seems to stretch in one direction; in frustum projection, the cubes composing the tesseract seem to "turn inside-out" (Fig. 3). While it is convenient to express rotations through the six variables described above, Continuum internally represents rotations as 4x4 matrices (and general transformations as 5x5 matrices) to avoid gimbal lock [APA04, SH11a].

Scale. When using the cross-section approach, we can imagine the extent of an object in the fourth dimension ("hyper-depth") as its resistance to translations along w. As shown in Fig. 4, the projection of larger scale objects *persists* (i.e. is visible) for a longer period of time as the viewer moves along w. In frustum projection, scaling a 4D object means its vertices move closer or further away from the viewer, creating a 4D analogue to standard depth. For instance, increasing the scale of a tesseract would make its "inner" cube (the one situated at $w = 1$) appear smaller and smaller to the 4D viewer.

3.3. Physics

Continuum implements a straight generalization of the laws observed in the 3D universe, assuming principles such as Newton's
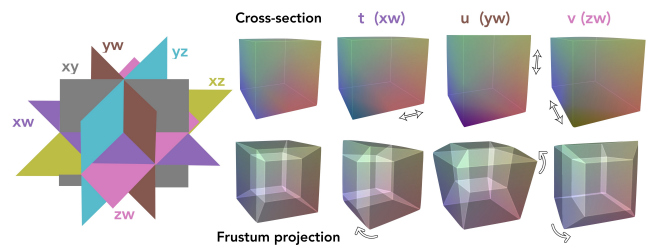

Figure 3: *4D rotations. Rotations in four dimensions do not just have an axis, but a plane around which everything else rotates. The four orthogonal axes define four mutually orthogonal 3D hyperplanes or **six** 2D orthogonal planes. For this reason, Continuum defines rotations in 4D space through a set of six angles (three variables—t, u, and v—more than a conventional 3D rendering engine). Standard 3D rotations exhibit the expected behavior, but rotations around the three new planes are perceived as mesh deformations. The figure above shows how rotating a tesseract seems to stretch the mesh in cross-section, and to turn the object inside-out when using frustum projection.*
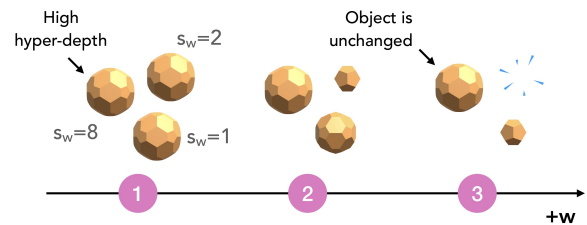

Figure 4: *Offsetting hyper-depth. Scaling an object along the fourth dimension changes the w coordinate of its vertices. This can influence the size and the geometry of its projection. The image above shows cross-sections of three identical—but differently scaled—4D objects as the viewer moves in the w direction. We note how projections of objects with higher hyper-depth are more persistent.*

universal gravitation still hold for higher dimensional objects. Similarly to 4D Toys [Bos20], we support a simple collision system using hypercubes and hyperspheres as colliders, enabling interaction between 4D objects such as the one shown in Fig. 5. While this is not the central topic of this paper, we want to note that higher dimensional physics may not necessarily generalize our existing understanding of physics in the 3D world. For instance, Newton's universal gravitation and Coulomb's electromagnetic laws follow from Gauss's law, which (assuming its validity in higher dimensions) would change from an inverse-square to an inverse-cube law due to the extra degree of freedom in the fourth dimension [McM09]. This change would obviously affect how object trajectories, spin, and energy are computed.

4. Conceiving a 4D Universe

In this section we go beyond the scope of visualizing simple 4D phenomena and ask ourselves which laws, constraints, and design criteria should broadly be kept in consideration when designing 4D applications. At the same time, we explore different content and authoring patterns to facilitate the editing of 4D scenes. A subset of the design considerations outlined here will be then operationalized in the use case presented in Section 5.
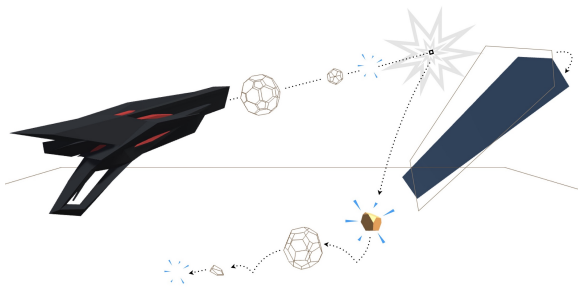
Figure 5: *4D Physics. In Continuum, physics on 4D objects are treated as straightforward generalizations of 3D physics. In the example above, the same gun from Fig. 2 shoots a bullet which exits the current cross-section and hits a hidden hyperplane. As result of the collision, the hyperplane slowly rotates around the t, u, and v planes and becomes partially visible, while the bullet falls to the ground due to gravity, rolling away until its w coordinate again falls outside of the cross-section.*
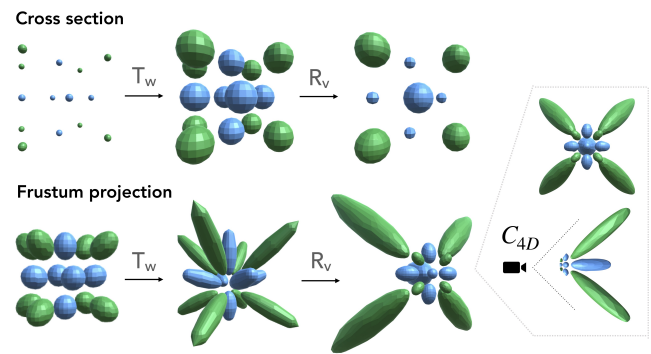

Figure 6: *Animating 4D primitives. This figure shows the perceived shape deformation associated with the translation of ten hyperspheres along w, followed by a rotation around v. This process demonstrates how simple 4D primitives can be combined and animated to form non-trivial, dynamic 3D projections. We note how, in frustum projection, meshes stretch along the direction of the 4D viewing frustum based on their w distance.*

4.1. Structure and Movement

Conceiving an n-dimensional application strictly defines the number of independent directions in which its space extends, but does not necessarily indicate how objects *move* within such a space. Depending on inherent geometrical structure and dimensions, different movement scenarios are in fact possible.

Euclidean hyperspace vs compact dimensions. If we consider a car riding on a roller coaster, or an ant crawling around a metal can, we immediately think of trajectories winding through a three-dimensional world. However, a closer look reveals that the roller coaster ride can be described with only one independent variable, and is effectively one-dimensional. Similarly, the ant will perceive its own movement on the surface of the can as two-dimensional, and would be completely unaware of living in a 3D world [McM08]. In relation to this concept of *compactification*, a four dimensional application does not necessarily imply the use of 4D content nor the ability to freely move in all dimensions. In this sense, we might consider adding spatial constraints within the "flat" Euclidean model presented in Section 3. We could imagine, for instance, the existence of 3D beings living in the different 3D subspaces of a tesseract, with these subspaces being connected through curved dimensions such as a wormhole or a 3D passage. Despite living in a higher dimensional space, such beings would only experience a 3D universe, and their movement would be constrained to a lower number of independent variables.

Ability to move in higher dimensions. Common experience suggests that humans cannot voluntarily move along a fourth dimension. String theory, however, suggests the existence of subatomic particles, called *gravitons*, that can move in all existing space (and time) dimensions. In computer graphics applications, it might be convenient to imagine the existence of certain macroscopic beings who are like gravitons, and therefore able to move across all dimensions. For instance, such a 4D being would be capable of escaping any 3D prison by moving along *w*, suggesting a scientifically-minded take on fictional concepts such as ghosts being able to pass through walls or disappear.

Spatial extent of higher dimensions. Astrophysical and cosmological observations, high-energy particle collider data, and experiments on Newton's law of universal gravitation [McM09] suggest that any extra dimension would have to be bounded in space to explain the limits of human perception. According to superstring theory, such extra dimensions would be sized in order of Plank length (about 10^{-33}cm)—which would be highly impractical in a concrete 4D application, as it would eliminate 4D perspective and cause abrupt hyper-plane transitions in cross-section. However, it might be worth considering a range of admissible *w* coordinates for facilitating the authoring process, for defining multiple gravity vectors, or for performance reasons. In cross-section, for instance, user interactions with visible 4D objects can happen only in a small window of *w* values, proportional to the hyper-depth of 4D content.

In the examples in this paper, we chose to model the universe as a four-dimensional, unbounded, Euclidean (non-compact) space. In such hyper-space, only certain macroscopic beings can move in the fourth dimension, along which we do not consider any gravitational force. While not compliant with physical evidence about the structure and extent of higher dimensions, this implementation is (almost) a straightforward generalization of 3D Euclidean space, and its practicality makes it convenient for studying the effects of 4D projections.

4.2. Content

After defining the structure of our hypothetical hyperuniverse, the next step is to try to identify the entities who might populate it, and how they could be constructed.

Procedural generation through mathematical functions. Similarly to how a tesseract can be created by extending a cube along the *w* axis, several 4-polytopes (also known as *polychora*) can be constructed from regular polyhedra [Cox73] using *lift* or *join* operations [Bri93]. Just as a 3D polyhedron is bounded by polygonal faces, so a polychoron is bounded by polyhedral cells, which need to meet under certain angle constraints in order to form a closed 3-surface [Bri09]. These perfect geometries are fundamental for educational purposes and can be used as 4D primitives, sometimes

delivering surprising visual effects like the one shown in Fig. 6 or like the crystal geometries in Miegakure [Bos]. However, 4D primitives tend to fall short of conveying the complexity and realism of the world we are used to living in.

Analogy-based design. 4D content can be designed by observing how certain concepts generalize from lower to higher dimensions. To start, we could try to imagine the nature of imaginary 4D humanoids by looking at how 2D and 3D beings behave, and then considering the objects and machinery they might build while trying to generalize devices such as wheels, gears, belts and pulleys [McM08]. For instance, we can imagine a 2D stool as having at least one leg—or more likely two legs—to avoid falling over. In the 3D world, a three-legged stool would balance perfectly, since any three non-linear points lie in a plane, while the presence of more than three legs could make the stool wobble if one leg were slightly longer than the others. Thinking along these lines, we can imagine a 4D stool as a cubinder with at least four legs, and placed next to one of the six cuboid-shaped sides of a 4D table. The same principle of balance would have us consider 4D humanoids to possess at least three legs—possibly four for stability and symmetry. Straightforward generalization is sometimes challenging and not always possible. For instance, in 3D we can connect objects together with a rope, which needs to be tied in a knot. This operation is not feasible in 2D, nor is it in 4D, where the knot would easily come undone due to the extra degree of freedom. In relation to the mathematical concept of *chirality*, humans in 3D space cannot superimpose their left hands onto their right hands. This operation is possible, however, through rotations in 4D [DL12]. Higher dimensional objects may also not have direct 3D analogies, and be conceivable only by hypothetical 4D beings. For instance, it would be possible for 4D beings to paint 3D pictures without the need for perspective, or, like humans can spot a cavity in a 2D surface, to directly observe the internal structure of 3D objects (theme known as "clairvoyance" [Lea03, Bra23]). While many other lower-to-higher dimensional analogies can be found in the literature [Abb84, Ruc14, Ruc02], this design methodology requires systematic thinking and is not easily scalable—making it difficult to apply beyond simple prototypes.

Data-driven shape synthesis. Another way of creating 4D content could stem from existing research on 3D *shape synthesis*, which aims to build generative models to support the creation of new, diverse, and realistic shapes along with associated part semantics and structure. Many approaches to synthesizing geometry involve 3D reconstruction, which uses lower-dimensional features, visual clues and prior information about geometry to generate 3D shapes. Examples include the estimation of 3D human joint positions and skin surfaces from 2D images [GK19], and the sketch-based generation of freeforms from sparse 2D strokes [LPL*17]. An alternate approach involves the synthesis of objects from a latent lower dimensional vector [Hin90]. This generally requires an initial dataset of 3D geometries encoded into a lower-dimensional embedding, which is then used to generate variations on such geometries [LXC*17, WWL*18, MGY*19, CFG*15]. While promising, these approaches still deal with simple and mostly convex geometry, represented as voxels (point clouds) or bounding boxes, and are still far from being generalizable. Applying such methods to 4D content generation has two primary limitations: 1) since we do not know how 4D objects might actually look and behave, it is hard to define semantically meaningful geometrical constraints,
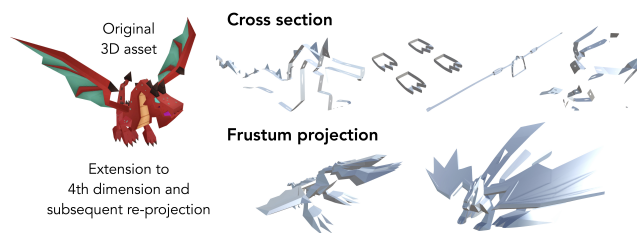

Figure 7: *Reprojecting existing 3D assets. As well as 4D polytopes, standard 3D meshes can also be handled in four dimensions. In the example above, a 3D asset is extended along w and then reprojected into 3D. We notice how 4D rototranslations slice the mesh into unexpected shapes in cross-section, whereas frustum projection shows the mesh stretching in the direction of the 4D camera.*

and 2) it is not currently feasible to build a curated dataset of 4D objects that would contain enough data samples and variation to allow for a machine learning approach.

Extension and 4D transformation of existing 3D assets. A final, more simplistic approach, consists of extending existing 3D objects into the fourth dimension, similarly to how we extend a cube into a tesseract. Specifically, we use lifting and tetrahedralization [Ber03] in the examples shown in this paper. While likely sacrificing the higher dimensional significance of such objects, this approach has the advantage of guaranteeing 4D object reprojections appear similar to their 3D counterparts when at rest. By applying simple 4D transformations, we can tune the 3D appearance (as shown in Fig. 7), all while retaining most of the semantics associated with a three-dimensional experience. This approach can be interpreted as a method to *use 4D to generate new 3D content*, more than as a way to create meaningful 4D content.

4.3. Authoring

After having defined the fundamental structure of our hypothetical hyperuniverse and the possible entities inhabiting it, the next challenge lies in identifying methods and design patterns to concretely author meaningful 4D experiences. While 3D editing tools such as Unity 3D, Unreal Engine, and Autodesk Maya have now become ubiquitous, there are currently no general-purpose tools dedicated to the creation of higher dimensional environments. Central design challenges include the difficulty of conceiving and manipulating such geometries, differing camera controls, the scarcity of available 4D content, and the integration of 4D content with existing 3D assets. In the following, we assume the usage of a standard 3D editor (e.g. Unity3D) with Continuum integration as authoring tool—implying that the user is able to translate, rotate, and scale objects in three dimensions as usual (e.g. through gizmos and mouse interaction, or manual numerical input). In this context, the *w* coordinate, 4D rotations, and scaling of a 4D object may be arbitrarily modified by the user. Doing so will generate the projection of a 4D object, which is then treated as a normal 3D asset within the authoring tool. From the user's perspective, this makes manipulating 4D objects similar to handling 3D meshes, with the additional possibility of regulating their geometry and visibility through a set of *4D parameters*.
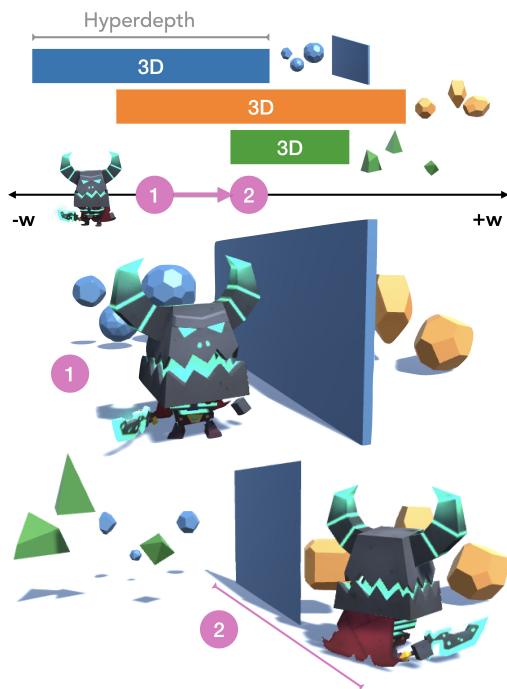
Figure 8: *Combining 3D worlds by offsetting their hyper-depth. One possible way to author a 4D environment consists of designing a set of 3D environments, extending them along the fourth dimension, and then placing them at different w coordinates in 4D space. A viewer moving along the fourth dimension would experience a smooth transition from one 3D world to the next. Objects from different 3D environments can be simultaneously visible, with possibly different persistence, based on the overlap of their hyper-depths. In the cross-section example above, a character cannot reach its target due to the presence of a wall belonging to the blue world (1). By applying a translation along w, the wall shrinks while exiting the cross-section, allowing the character to pass (2). At the same time, objects from the green world become visible.*

Layout. One design choice involves deciding how much *native* 4D content to include and how to position it in hyperspace. A possibility would be to consider authoring an environment where *all* content comes from intrinsically 4D geometries. Unfortunately, most available 4D content derives from abstract mathematical entities whose natural interactions in higher dimensions would be particularly hard to conceive of within a 3D editor. In this case, proper placement and behavior of such objects could be defined through scripting and custom mathematical functions, and then visually validated through the authoring tool—but the result would be a geometrical world probably too different from anything resembling the common human experience.

An alternative approach would be to let the user author 3D worlds as usual, and then combine them within the four-dimensional space according to some *placement* criteria. One simple example would involve defining a set of independent 3D worlds, and lending each of them a different coordinate along the *w* axis. If we assume such worlds have no hyper-depth, a viewer would be able to experience those worlds, one at a time, when moving at their respective *w* coordinate. However, if we extend the 3D worlds into 4D so as to arbitrarily define their hyper-depth, we can create a *transition*
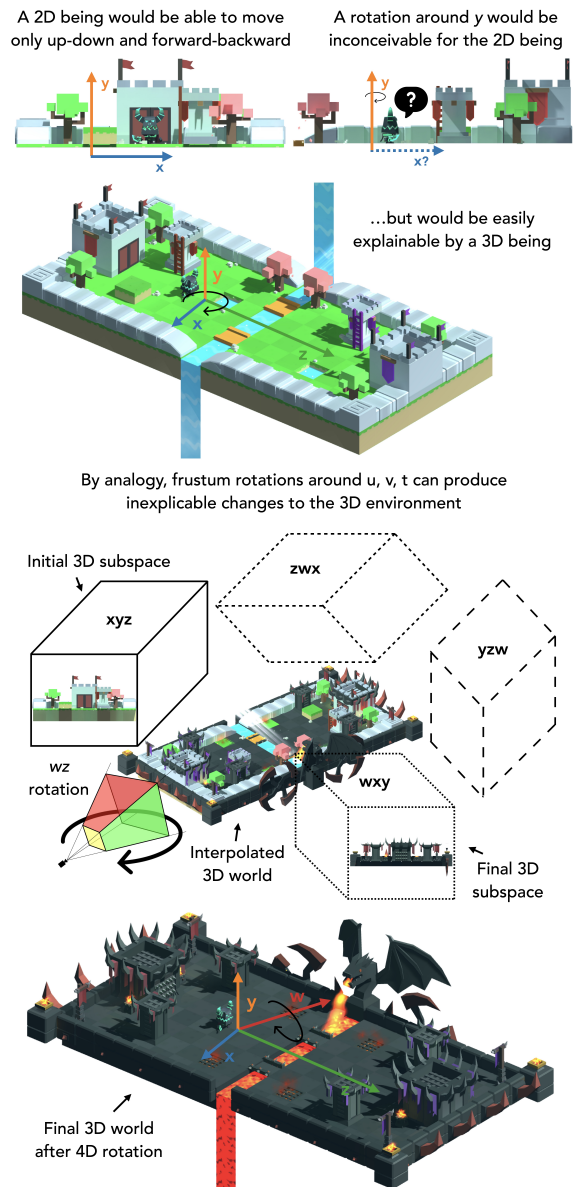

Figure 9: *Combining 3D worlds through orthogonal subspaces. A possible technique to adopt when authoring 4D environments consists of using 4D rotations to transition between different 3D worlds. To use an analogy, a two-dimensional being, unaware of the fact that it lives within a 3D world, would only be able to see two directions of motion, and could rotate only around the z axis, perpendicular to the xy plane it lies on. If this character could perform the action—to him inconceivable—of rotating around y, the world surrounding him would inexplicably change. This change, however, would make perfect sense from the perspective of someone able to perceive the higher dimension. Similarly, 4D rotations would enable the viewer to transition from one 3D world to another, with the effect of "warping" the surrounding environment.*

effect between such 3D environments, possibly overlapping their content and creating environmental effects such as doors opening or objects materializing (Fig. 8). In a similar fashion, 4D rota-
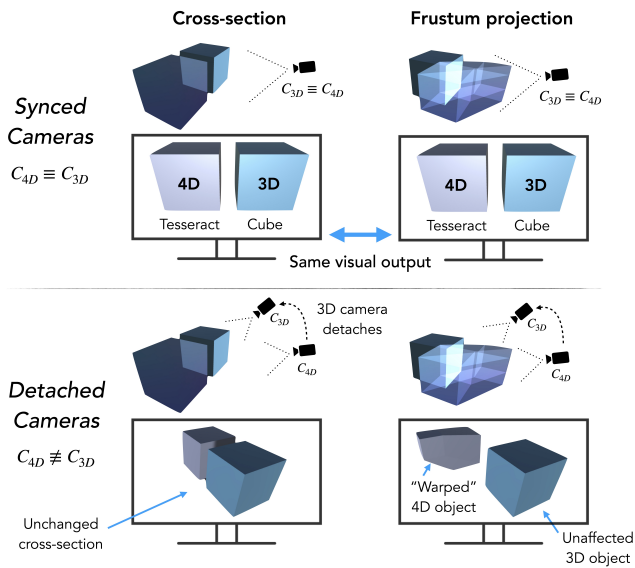
Figure 10: *Camera setup for mixed 3D/4D environments. In both cross-section and frustum projection, the 4D and standard 3D cameras can be synced as to share the same 3D rototranslation components, and treated as a single viewer. While both projection methods might lead to the same visual output on the screen, the internal 3D representation of the scene generated by frustum projection involves morphed geometries, so as to preserve the 4D perspective effect in the final screen rendering. This implies that, if the 3D camera is detached from the 4D viewer, 4D geometries will be perceived as "warped" and will behave differently from their 3D counterparts.*

tions can be used to transition between 3D worlds placed in different subspaces of our hypothetical hyperuniverse (Fig. 9). Such transitions would require, however, extending any existing 3D asset into four dimensions, and fully recomputing the entire scene geometry—operations that can significantly reduce the application performance, as later discussed in Section 6. Possible workarounds include renouncing some 4D transition effects (perhaps "simulating" them through a standard 3D engine), and simplifying the scene geometry. Miegakure [Bos], for instance, achieves smooth cross-section performance by limiting the number of visible objects and emphasizing the of use of textures on simple 4D primitives.

A third and final approach to authoring a 4D scene consists of designing a single, standard 3D world, and integrating a relatively small, but more carefully chosen, number of 4D entities. In this case, intrinsically 3D assets would be unaffected by 4D physics and projection parameter changes. One advantage of this approach is that it does not require us to extend 3D assets into 4D in order to place them in higher-dimensional space, and therefore drastically reduces the computation required to project them back into 3D. At the same time, it allows us to maintain standard 3D authoring practices, focusing on the definition of few selected 4D interactions, which are inherently more time-consuming to design.

Camera Setup. While in standard graphics pipelines we have the concept of a single rendering camera, which transforms a 3D world into the visuals we experience on our 2D screen, in our Continuum engine the designer has an additional viewer (or camera) at their disposal—opening up a set of new authoring possibilities. If

we directly generalize the existing 3D approach, we can imagine the existence of a single viewer, able to independently move and rotate in 4D space. If an application involves only 4D assets, movement within the environment can be performed by simply allowing the user to control the 4D camera, and leaving the 3D camera fixed. Problems arise only when the environment is comprised of both 3D and 4D assets. If we desire a straightforward generalization of a standard 3D viewer, we can define the 3D and 4D cameras as to share the 3D components of their transformations, with the 4D camera able to independently move along w and rotate around u, t, and v. We call this approach "synced cameras", as from the user perspective it is equivalent to control a single, unified 4D viewer. We note how in this case the camera transformation is applied twice to 4D objects (first when they are projected into 3D, and then again when projected to screen space), requiring a simple offset correction on such geometries. A different methodology involves instead considering the two cameras as independent. The typical use case here would be in applications involving limited 4D assets, where the user can freely move the 3D camera in the environment and the 4D camera is separately used to control the appearance of 4D geometries. This "detached cameras" approach can possibly hew to the same visual results of the "synced camera" approach when using the cross-section method, but produces completely different results when using frustum projection. In fact, the two detached cameras guarantee the existence of two separate vanishing points in the final screen rendering, enabling the perception of both depth and *hyper-depth*. Specifically, frustum projection morphs meshes with respect to the 4D viewer when passing from higher-dimensional geometry into 3D space, and the independent movement of the 3D camera can be leveraged to observe such *warped* geometries (Fig. 10)—a visual effect that could be considered physically unrealistic or artistically interesting depending on the use case. This approach is exemplified in the next section, where a new method for smoothly transitioning between cross-section and frustum projection is also presented.

5. Use Case: Across Dimensions

In the previous section we outlined a number of ways to conceive of a hypothetical 4D universe, discussing the potential but also the practicality of each. Here we settle on a *subset* of the methodologies presented in Section 4, and relatedly present the implementation of a prototype single-player videogame called Across Dimensions. We designed this game with the specific goals of 1) creating a simple, non-didactic, 4D-enabled application which is not too far from common experiences of 3D (as opposed to existing abstract geometry visualization tools), 2) identifying 4D interactions that are also meaningful from a 3D perspective, and 3) exploring how the cross-section and frustum projection methods can be integrated and organically controlled by the user. With these criteria in mind, we chose to define a relatively simple hyperuniverse *structure*, characterized by flat, Euclidean geometry (i.e. no compact dimensions). The user is able to freely move along the unbounded x and z dimensions, but is subject to gravity along the y axis and can move along w only within a certain range and under certain circumstances. So as to preserve familiarity with ordinary 3D videogames and to compensate for the lack of intrinsically 4D assets, the game's *content* consists of a mix of 4D and 3D objects—and 4D objects exist either as a composition of 4D primitives (Fig. 6) or as the 4D extension of an existing 3D asset (Fig. 14). In terms of *authoring*, we started modeling a 3D world using the Unity3D editor, and then

Figure 11: *Across Dimensions. Across Dimensions is a prototype videogame we developed as a use case for our Continuum engine, where the player can navigate a classic 3D world populated by 4-dimensional beings. The player has at his disposal an arsenal of tools (weapons) that enable him to interact with and manipulate 4D objects, which hide in the environment, and may suddenly appear to confront the player. In the figure above, the player uses 3D radar to estimate the w coordinate of 4D objects and spot enemies despite them being outside of the current cross-section.*

made careful additions of hand-picked 4D content using our Continuum integration, offsetting assets along the w axis based on the principle previously shown in Fig. 8. Overall, these design choices had the effect of giving the user the impression of playing a regular videogame at first, and then gradually revealing hidden higher-dimensional content and mechanics.

Gameplay. Across Dimensions is a first-person shooting game with puzzle and exploration elements, set in a fictional sci-fi world. While exploring this apparently 3D world, the player discovers and collects special crystals that fill up a dedicated energy bar, which can be drawn on in order to perform 4D actions such a moving along the w dimension, manipulating higher-dimensional objects, and modifying projection parameters and perspective. Most of these actions are performed through a set of sci-fi weapons with unique abilities. These are also used to fight 4D enemies that the user encounters along the way. In the current implementation, Across Dimensions is a laptop-based videogame available on Windows and MacOS, and uses standard mouse and keyboard controls as input.

Navigation and camera control. Basic navigation in 3D space is performed through the standard combination of directional keys (WASD or arrow keys) and mouse movement for camera rotation around the x, y, and z axes. When enough crystal energy is accumulated, the player presses two separate keys to smoothly move in the w direction, which has the effect of allowing him to encounter or leave behind certain 4D objects positioned at different w coordinates. Meanwhile 3D assets in the scene always remain visible. To help the player navigate a world where many objects are hidden in higher dimensions, we introduce the concept of holographic 3D radar (Fig. 11). Similarly to a normal radar display, the user's location is positioned at the center, with the radar plane defined in the x and z directions, and positioned at the same w coordinate as the player. 4D objects appear on the radar as 3D pins with an altitude proportional to their w offset with respect to the player, suggesting that objects showing on the radar plane are currently visible, while
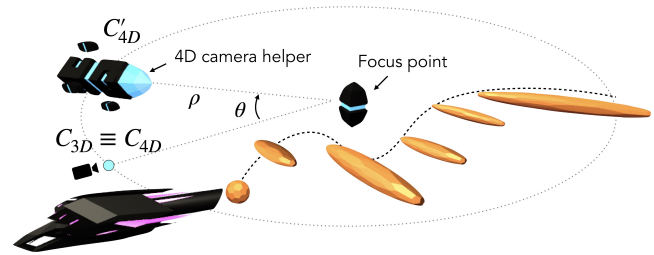

Figure 12: *Camera control. Across Dimensions supports smoothly transitioning from cross-section to frustum projection. When this happens, the player can independently control the 4D camera along an orbital trajectory with the help of in-game visual aids.*

objects above or below the radar plane require the player to move along the w direction in order to reach them.

A synced cross-section camera setup is applied by default, so that 4D objects moving along the fourth dimension appear to gradually enter and exit the current cross-section hyperplane. In this modality, the 4D camera simply follows the 3D camera, and is only in charge of changing the cross-section hyperplane. To make the game more challenging and explore novel camera setups, we decided to allow the player to unlock advanced control of the 4D camera. Using crystal energy, the player can press a key to switch from the cross-section camera to the frustum projection camera. In the instant in which this transition happens, the 4D camera is in sync with the 3D camera and—if certain criteria are met—no noticeable change is perceived. Specifically, we observed this effect takes place if the surrounding geometry does not have a 4D rotational component, and if there are no nearby 4D objects outside of the current cross-section (as they would otherwise be "revealed" by the frustum projection). When the user moves in 3D space, the 3D camera then detaches from the 4D camera, generating two distinct vanishing points for the z and w dimensions and consequently warping the 4D objects in the scene, as we previously explained in Fig. 10. The warped geometry generated by the frustum projection is a function of the initial position of the user with respect to the 4D object, whose 3D projection then remains unchanged when the user moves around it (since the 4D camera position remains fixed).

The next step involves enabling control of the position and orientation of the 4D camera, which would allow the user to dynamically morph the 4D environment around him after an initial switch to frustum projection. Unfortunately, this operation introduces four new positional and six new rotational variables into the game, and is not easy to conceive of, especially when already focused on the existing mechanics of the game. Therefore, we decided to limit 4D camera control to an orbital trajectory starting from the player's initial position, and having a fixed focus point around which the camera rotates. Since the focus point is automatically set to be few meters in front of the player during the switch to frustum projection, this leaves only one additional variable, the *orbit angle*, which the user can control via two dedicated keys. Since moving the 4D camera and shifting the w vanishing point are hard to imagine, two new objects appear in the scene when frustum projection is activated, dynamically reflecting the the focus point position and 4D camera position, respectively (Fig. 12). These two in-game indicators are also shown on the 3D radar display, allowing the player to control them even if they are not currently on screen.

When the player decides to switch back to cross-section or when the energy bar is depleted, the 4D camera position gradually rejoins the 3D camera through interpolation, while keeping its orientation locked on the focus point. This method allows for smoothly switching back to cross-section projection with synced cameras, easing the transition between these two completely different projection methods. While 4D perspective can be used to achieve unique mesh deformations and visual effects, we designed Across Dimensions as to limit the use of frustum projection to a set of carefully chosen scenarios, mostly due to its computational cost and parametric complexity—which require extra time to fine-tune the appearance of 4D geometries and test game mechanics. Cross-section, by contrast, is more straightforward and can be uniformly applied to many 4D objects at a time, enabling the designer to easily create "hide and reveal" [Bos] experiences.

Interacting with 4D elements. Movement itself represents an initial, basic way of interacting with 4D content. Moving along w while in cross-section can allow the player to pass through walls and reach a target situated behind them (as shown in Fig. 8). In other cases, the movement of the 4D camera can be used to turn certain objects into others, such as turning a barrier into a building that the user can enter to proceed in the game. When close to certain 4D objects and equipped with a relevant tool, the player can perform direct manipulation of geometry through 4D rotations, utilizing dedicated keys that are indicated on-screen. For example, the player might be stuck on a platform and unable to proceed beyond a steep dropoff. Through 4D manipulation, this problem could be solved by molding a bridge out of a 4D geometry situated at the edge of the cliff, and then placing it so as to create a path to the adjacent platform. Since it is inherently hard to estimate the results produced by 4D rotation, we apply the concept of "ghost geometries" [LRY*15,Kag16b] and visualize floating *previews* of the shapes that the 4D object could assume (Fig. 13). The player can also perform more complex actions through a set of weapons which he acquires throughout the game. One example is the weapon already introduced in Fig. 2 and Fig. 5, which shoots hyperdodecahedral bullets that fluctuate along the w axis. These bullets periodically become invisible as they exit the current cross-section, but their 4D movement can be continuously tracked using the 3D radar display. Each bullet has enough power to move simple 4D objects on contact, allowing one to remove obstacles and hit targets that are currently invisible (e.g. enemies farther away in the w direction. When used in combination with frustum projection, the shape and trajectory of bullets can be dynamically altered by moving the 4D camera, thus making it possible to hit targets around corners or to expand the bullet's collider in a specific direction. A second weapon has the effect of generating a radial blast that brings all nearby 4D objects to the same w coordinate as the player, revealing hidden objects or enemies. Similarly, another weapon produces an electrical field that modifies the rendering material of nearby 4D objects and shows their wireframe, allowing the user to easily identify them when their presence is disguised among similar-looking 3D assets.

Fighting enemies. Enemies in Across Dimensions are either combinations of 4D primitives (Fig. 6) or are generated by extending existing 3D assets (Fig. 7), an operation which guarantees content that is visually coherent with its 3D surroundings. Since extending 3D assets does not guarantee the generation of meaningful limbs or skeletal structures in 4D, we decided to "animate" enemies simply

Figure 13: *4D manipulation. To facilitate the manipulation of higher dimensional objects, we preview the effects of 4D transformations on their geometry. In the figure above, the player applies a 4D rotation to an object in order to build a bridge, creating a path to an adjacent platform.*

Figure 14: *Animating enemies. After extending existing 3D assets into the 4th dimension, we can animate their 3D reprojections by applying simple 4D transformations. In the cross-section example above, an object morphs from a static tree into a jellyfish, and finally into a gun just by having 4D rotations applied to it.*

by applying to their geometry pre-defined 4D rototranslations. This simple method allows for the generation of unconventional visual effects, such as the morphing of a tree into a jellyfish and then into a weapon (Fig. 14). In Across Dimensions, enemies hide at different coordinates along the w axis, and can independently move in all four dimensions, freely entering the current cross-section to attack the player or exiting this cross-section to flee. Certain enemies remain partially visible, but hide their weak spots outside of the visible spectrum, so that the player must follow his bullets on the 3D radar after they have exited the cross-section. Sometimes the player can even loot crystals hidden inside the 4D geometry of an enemy by forcing that enemy to perform particular movements or by controlling the 4D camera.

6. Discussion

Here we provide additional considerations that emerged during the Across Dimensions design process, and discuss the challenges and possible benefits of integrating 4D features with existing 3D applications.

Authoring Hybrid 3D / 4D Worlds. The integration of 4D mechanics into Across Dimensions undoubtedly introduced new, creative forms of interaction. Designing the game proved however challenging, particularly with respect to content authoring. The decision to start by defining a basic 3D environment, and then later add 4D content definitely helped with spatial orientation and facilitated storytelling. This decision was made due to our desire to
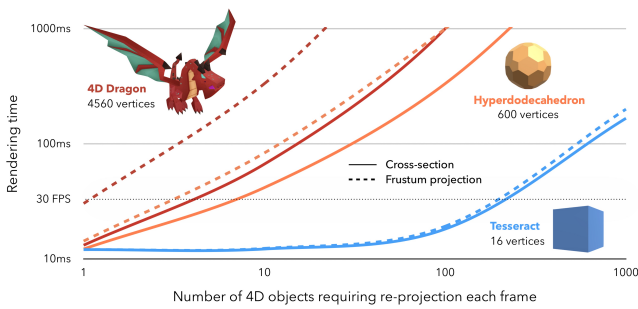
Figure 15: *Rendering performance. After initial projection, static 4D assets have the same performance footprint as 3D assets. However, animating such objects requires continuously recomputing their projected 3D meshes. The chart above shows how applying a continuous 4D rotation to different on-screen objects affects rendering performance in both cross-section and frustum projection. Performance tests have been measured on a 2018 MacBook Pro at full resolution.*

not create a completely abstract game, and also to the limited availability of 4D content. Even when using simple geometries, it is difficult to deploy too many 4D objects in a scene due to an inherent behavioral complexity, which requires that the designer take more time to carefully integrate these objects in the game's storyline. The effects of 4D transformations and camera projections are in fact hard to predict, often leading to a trial-and-error approach to content authoring. In this sense, it may be worth exploring recommender systems for placing and manipulating 4D content, so as to help the designer preview and apply a set of automatically generated 4D behaviors. Another factor to consider is the trade-off between integrating native 4D elements and approximating some of their behaviors through more conventional techniques such as 3D mesh deformations, which could provide similar visuals with a more deterministic behavior. Along these lines, we recommend a further study to explore which mechanics can be achieved uniquely through 4D interactions.

Performance Limitations. Thanks to our Continuum implementation, static 4D objects in the scene are treated as normal 3D assets after they are projected for the first time. However, when objects currently on-screen undergo a 4D transformation or the 4D camera is moved, the 3D projections of 4D objects need to be recomputed. As shown in Fig. 15, mesh deformations take a big toll on overall rendering performance, especially when using frustum projection and 4D objects with higher numbers of vertices, and more sophisticated GPU implementations [CFHH09] should be considered. In particular, entering and exiting cross-section hyperplanes and rotating the 4D camera while using frustum projection are two of the most computationally expensive operations, since they require recomputing meshes dynamically. To circumvent these limitations and maintain a reasonable (>30 FPS) frame rate, we designed Across Dimensions as to prioritize the use of 4D primitives, and to make the user interact with no more than a couple complex 4D objects at a time.

Parametric Explosion and Camera Control. While standard videogames and applications generally do not require more than a few forms of input to move the camera or interact with 3D objects, a 4D application can add up to twenty rototranslational variables

atop already existing ones. This parametric explosion makes it particularly cumbersome to create fast-paced applications that allow full control of both 3D and 4D cameras and free manipulation of 4D objects. Therefore, we believe it makes sense to settle on allowing the user to control only a subset of these parameters, as we did by limiting the 4D camera movement to an orbital trajectory, or by constraining the axes for 4D manipulation. Instead of seeing this as a limitation, we believe choosing different subsets of parameters could lead to the creation of more unique applications with compelling mechanics. In particular, it would be interesting to continue exploring the possible behaviors of the 4D camera, harnessing what is sometimes unpredictable visual potential. Due to the computationally expensive footprint, trade-offs such as alternating frustum projection with the much simpler cross-section should be explored. Even more importantly, dedicated visual aids should be provided to users, such as the 3D radar display and the 4D manipulation previews in Across Dimensions.

Making Higher Dimensions More Accessible. Despite science theories actually predict the existence of higher dimensions [McM08], conceiving of and visualizing higher dimensional entities has historically been a niche research topic, in which interest has rarely stemmed from potential usefulness—except for a few applications to crystallography and particle physics [Cox73]. Visualizing higher dimensions has been seen as an educational tool to better understanding our own 3D world, but its underlying technical complexity, level of abstraction, and fragmented resources make this topic very difficult to approach. For this reason we aimed at presenting a different take on 4D visualization, as to *make higher dimensions more accessible* to our community. Specifically, we tried to extend common 3D knowledge and present a self-contained paper with tangible examples, encompassing all major design considerations required to conceive a higher-dimensional world. While our use case focuses on a videogame application, wherein game designers introduce unexpected scenarios and generate artistic content through 4D concepts, we hope the tools provided in this work can lead our readers to out-of-the-box thinking and inspire new creative applications to multiple domains.

7. Conclusion

In this work we built upon extant literature on higher dimensional visualization, taking a different path through the discipline: instead of focusing on the rendering and manipulation of 4D mathematical objects in a sandbox, we attempted to conceive of a much wider 4D universe—changing focus from the observation of a single geometry for educational purposes to a *user-centric* exploration of a more complex high-dimensional narrative. In particular, after observing the effects of reducing the dimensionality of 4D geometries, we studied the trade-offs related to defining a higher-dimensional universe in terms of geometrical structure, content generation, and authoring strategies—comparing the use of both cross-section and frustum projection approaches. Finally, we showcased how multiple 4D and 3D objects can coexist and interact with each other in the same application, proposing a videogame as a concrete use case for how a four-dimensional engine might be integrated into existing rendering engines such as Unity3D.

References

[Abb84] ABBOT E.: *Flatland: A Romance of Many Dimensions*. Seeley & Co., 1884. 2, 6

[Agu06] AGUILERA J. C.: Virtual reality and the unfolding of higher dimensions. In *Stereoscopic Displays and Virtual Reality Systems XIII* (2006), vol. 6055, International Society for Optics and Photonics, p. 60551V. 2

[APA04] AGUILERA A., PÉREZ-AGUILA R.: General n-dimensional rotations. *WSCG Short Communications* (2004). 4

[Arn72] ARNOLD P.: A proposal for a study of the mechanisms of perception of, and formation of internal representations of, the spatial fourth dimension. *Accomplishment Summary 71*, 72 (1972), 223–235. 2

[AWC*09] AMBINDER M. S., WANG R. F., CROWELL J. A., FRANCIS G. K., BRINKMANN P.: Human four-dimensional spatial intuition in virtual reality. *Psychonomic bulletin & review 16*, 5 (2009), 818–823. 2

[B*14] BANCHOFF T., ET AL.: Salvador dalí and the fourth dimension. *Proceedings of the Bridges* (2014). 2

[Ban90a] BANCHOFF T. F.: *Beyond the third dimension*. Scientific American Library New York, 1990. 2

[Ban90b] BANCHOFF T. F.: From flatland to hypergraphics: Interacting with higher dimensions. *Interdisciplinary Science Reviews 15*, 4 (1990), 364–372. 2

[Ban92] BANKS D.: Interactive manipulation and display of surfaces in four dimensions. In *Proceedings of the 1992 symposium on Interactive 3D graphics* (1992), pp. 197–207. 2

[Ban02] BANCHOFF T. F.: Computer graphics in mathematical research, from icm 1978 to icm 2002: A personal reflection. In *Mathematical Software*. World Scientific, 2002, pp. 180–189. 1

[Ben91] BENEDICKT M.: Cyberspace: first steps. *The MIT Press* (1991). 2

[Ber03] BERN M.: Adaptive mesh generation. In *Error estimation and adaptive discretization methods in computational fluid dynamics*. Springer, 2003, pp. 1–46. 6

[Bos] BOSCH M. T.: Miegakure [hide & reveal]. https://miegakure.com. Accessed: 2020-12-01. 3, 6, 8, 10

[Bos20] BOSCH M. T.: N-dimensional rigid body dynamics. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 55–1. 2, 3, 4

[BPRS98] BAJAJ C. L., PASCUCCI V., RABBIOLO G., SCHIKORE D. R.: Hypervolume visualization: a challenge in simplicity. In *IEEE Symposium on Volume Visualization (Cat. No. 989EX300)* (1998), IEEE, pp. 95–102. 2

[Bra23] BRAGDON C. F.: *A Primer of Higher Space: The Fourth Dimension, to which is Added, Man the Square, a Higher Space Parable*. AA Knopf, 1923. 6

[Bri87] BRISSON H. E.: *Hypergraphics: Visualizing Relationships in Art, Science, and Technology*, vol. 24. Westview Press, 1987. 2

[Bri93] BRISSON E.: Representing geometric structures in d dimensions: Topology and order. *Discrete & Computational Geometry 9*, 4 (1993), 387–426. 5

[Bri09] BRIGGS T. S.: Exploring hyperspace with the geometric product, 2009. 5

[BS78] BANCHOFF T., STRAUSS C.: The hypercube: projections and slicing. *International Film Bureau, Chicago, Ill* (1978). 2

[BS07] BALSYS R. J., SUFFERN K. G.: Point based rendering of implicit 4-dimensional surfaces. In *Computer Graphics, Imaging and Visualisation (CGIV 2007)* (2007), IEEE, pp. 31–40. 2

[BWC04] BHANIRAMKA P., WENGER R., CRAWFIS R.: Isosurface construction in any dimension using convex hulls. *IEEE Transactions on Visualization and Computer Graphics 10*, 2 (2004), 130–141. 2

[CD18] CAVALLO M., DEMIRALP Ç.: A visual interaction framework for dimensionality reduction based data exploration. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (2018), pp. 1–13. 1

[CFG*15] CHANG A. X., FUNKHOUSER T., GUIBAS L., HANRAHAN P., HUANG Q., LI Z., SAVARESE S., SAVVA M., SONG S., SU H., ET AL.: Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015). 6

[CFHH09] CHU A., FU C.-W., HANSON A., HENG P.-A.: Gl4d: A gpu-based architecture for interactive 4d visualization. *IEEE transactions on visualization and computer graphics 15*, 6 (2009), 1587–1594. 2, 3, 11

[CH94] CROSS R. A., HANSON A. J.: Virtual reality performance for virtual geometry. In *Proceedings Visualization'94* (1994), IEEE, pp. 156–163. 2

[Cox73] COXETER H. S. M.: *Regular polytopes*. Courier Corporation, 1973. 5, 11

[DL12] DALVIT E., LUMINATI D.: How to see a cube moving into its mirror image. *The Mathematical Gazette 96*, 537 (2012), 471–479. 6

[Duf96] DUFF M. J.: M theory (the theory formerly known as strings). *International Journal of Modern Physics A 11*, 32 (1996), 5623–5641. 2

[Egg99] EGGINTON W.: On dante, hyperspheres, and the curvature of the medieval cosmos. *Journal of the History of Ideas 60*, 2 (1999), 195–216. 1

[GK19] GULER R. A., KOKKINOS I.: Holopose: Holistic 3d human reconstruction in-the-wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 10884–10894. 6

[HC93] HANSON A. J., CROSS R. A.: Interactive visualization methods for four dimensions. In *Proceedings Visualization'93* (1993), IEEE, pp. 196–203. 1, 2

[Hei41] HEINLEIN R.: And he built a crooked house. *Astounding Science Fiction* (1941), 2010–11. 2

[Hen13] HENDERSON L. D.: The fourth dimension and non-euclidean geometry in modern art (revised edition). *Leonardo Book Series, London and New York, MIT Press* (2013). 1, 2

[HH92a] HANSON A. J., HENG P. A.: Four-dimensional views of 3d scalar fields. In *Proceedings of the 3rd conference on Visualization'92* (1992), IEEE Computer Society Press, pp. 84–91. 2

[HH92b] HANSON A. J., HENG P. A.: Illuminating the fourth dimension. *IEEE Computer Graphics and Applications*, 4 (1992), 54–58. 2

[HIM99] HANSON A. J., ISHKOV K. I., MA J. H.: Meshview: Visualizing the fourth dimension. *Overview of the MeshView 4D geometry viewer* (1999). 2

[Hin90] HINTON G. E.: Connectionist learning procedures. In *Machine learning*. Elsevier, 1990, pp. 555–610. 6

[HS94] HAUSMANN B., SEIDEL H.-P.: Visualization of regular polytopes in three and four dimensions. In *Computer Graphics Forum* (1994), vol. 13, Wiley Online Library, pp. 305–316. 2

[HZ91] HOFFMANN C. M., ZHOU J.: Some techniques for visualizing surfaces in four-dimensional space. *Computer-Aided Design 23*, 1 (1991), 83–91. 2

[HZ05] HANSON A. J., ZHANG H.: Multimodal exploration of the fourth dimension. In *VIS 05. IEEE Visualization, 2005.* (2005), IEEE, pp. 263–270. 2

[Jam09] JAMROZ D.: Multidimensional labyrinth–multidimensional virtual reality. In *Man-Machine Interactions*. Springer, 2009, pp. 445–450. 2

[Jea31] JEANS J.: The mysterious universe. *The British Journal of Radiology 4*, 43 (1931), 351–354. 2

[Kag16a] KAGEYAMA A.: Keyboard-based control of four-dimensional rotations. *Journal of Visualization 19*, 2 (2016), 319–326. 2

[Kag16b] KAGEYAMA A.: A visualization method of four-dimensional polytopes by oval display of parallel hyperplane slices. *Journal of Visualization 19*, 3 (2016), 417–422. 2, 10

[Kak16] KAKU M.: *Hyperspace: A scientific odyssey through parallel universes, time warps, and the tenth dimension*. Oxford University Press, 2016. 2

[Lea03] LEADBEATER C. W.: *Clairvoyance*. Library of Alexandria, 1903. 6

[LPL*17] LI C., PAN H., LIU Y., TONG X., SHEFFER A., WANG W.: Bendsketch: modeling freeform surfaces through 2d sketching. *ACM Transactions on Graphics (TOG) 36*, 4 (2017), 1–14. 6

[LRY*15] LI N., REA D. J., YOUNG J. E., SHARLIN E., SOUSA M. C.: And he built a crooked camera: a mobile visualization tool to view four-dimensional geometric objects. *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications* (2015), 1–5. 2, 10

[LXC*17] LI J., XU K., CHAUDHURI S., YUMER E., ZHANG H., GUIBAS L.: Grass: Generative recursive autoencoders for shape structures. *ACM Transactions on Graphics (TOG) 36*, 4 (2017), 1–14. 6

[McM08] MCMULLEN C.: *The Visual Guide to Extra Dimensions. Volume 1: Visualizing the Fourth Dimension, Higher-Dimensional Polytopes, and Curved Hypersurfaces*. CreateSpace Independent Publishing Platform, 2008. 1, 3, 5, 6, 11

[McM09] MCMULLEN C.: *The Visual Guide to Extra Dimensions. Volume 2: The Physics of the Fourth and Higher Dimensions, and Current and Upcoming Experiments to Detect Extra Dimensions*. CreateSpace Independent Publishing Platform, 2009. 4, 5

[MGY*19] MO K., GUERRERO P., YI L., SU H., WONKA P., MITRA N., GUIBAS L. J.: Structurenet: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575* (2019). 6

[Mil98] MILLER A. I.: Albert einstein's special theory of relativity. *" Includes a translation by AI Miller of A. Einstein's" On the Electrodynamics of Moving Bodies", XX, 446 pp. 74 figs.. Springer-Verlag New York"* (1998), 74. 2

[MNSH17] MIWA T., NAKAI Y., SAWADA H., HASHIMOTO S.: 4-d space visualization with hidden hypersurface removal. In *2017 International Symposium on Micro-NanoMechatronics and Human Science (MHS)* (2017), IEEE, pp. 1–5. 2

[MOI*19] MATSUMOTO K., OGAWA N., INOU H., KAJI S., ISHII Y., HIROSE M.: Polyvision: 4d space manipulation through multiple projections. *SIGGRAPH Asia 2019 Emerging Technologies* (2019), 36–37. 2

[MSH13a] MIWA T., SAKAI Y., HASHIMOTO S.: Four-dimensional viewing direction control by principal vanishing points operation and its application to four-dimensional fly-through experience. In *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration* (2013), pp. 95–104. 3

[MSH13b] MIWA T., SAKAI Y., HASHIMOTO S.: Interactive four-dimensional space exploration using viewing direction control based on principal vanishing points operation. In *Australian Computer-Human Interaction Conference* (2013), Springer, pp. 21–46. 3

[MSH15] MIWA T., SAKAI Y., HASHIMOTO S.: 4-d spatial perception established through hypercube recognition tasks using interactive visualization system with 3-d screen. In *Proceedings of the ACM SIGGRAPH Symposium on Applied Perception* (2015), pp. 75–82. 2

[Nah01] NAHIN P. J.: *Time machines: Time travel in physics, metaphysics, and science fiction*. Springer Science & Business Media, 2001. 2

[Nol67] NOLL A. M.: A computer technique for displaying n-dimensional hyperobjects. *Communications of the ACM 10*, 8 (1967), 469–473. 2

[Nov98] NOVAK M.: Transarchitectures and hypersurfaces - operations of transmodernity. *Architectural Design*, 133 (1998), 84–93. 2

[OBL13] OHORI K. A., BOGUSLAWSKI P., LEDOUX H.: Representing the dual of objects in a four-dimensional gis. In *Developments in multidimensional spatial data models*. Springer, 2013, pp. 17–31. 2

[OLS17a] OHORI K. A., LEDOUX H., STOTER J.: Modeling and manipulating spacetime objects in a true 4d model. *Journal of Spatial Information Science 2017*, 14 (2017), 61–93. 2

[OLS17b] OHORI K. A., LEDOUX H., STOTER J.: Visualising higher-dimensional space-time and space-scale objects as projections to r3. *PeerJ Computer Science 3* (2017), e123. 2

[OSY19] OGMEN H., SHIBATA K., YAZDANBAKHSH A.: Perception, cognition, and action in hyperspaces: Implications on brain plasticity, learning, and cognition. *Frontiers in Psychology 10* (2019). 2

[Pas06] PASSAGES W.: Unravelling the mysteries of the universe's hidden dimensions, 2006. 2

[Pau84] PAUL D.: The eleventh dimension. *Science Digest* (1984). 2

[Pet10] PETKOV V.: Space, time, and spacetime. *Fundamental Theories of Physics. Springer-Verlag, Berlin* (2010). 2

[RHR92] ROBBIN T., HENDERSON L. D., RUCKER R. V.: *Fourfield; Computers, Art and the Fourth Dimension*. Bulfinch Press, 1992. 2

[Ruc02] RUCKER R.: *Spaceland: A novel of the fourth dimension*. Macmillan, 2002. 6

[Ruc12] RUCKER R.: *Geometry, relativity and the fourth dimension*. Courier Corporation, 2012. 2

[Ruc14] RUCKER R.: *The fourth dimension: Toward a geometry of higher reality*. Courier Corporation, 2014. 2, 6

[SH07] SAKAI Y., HASHIMOTO S.: Four-dimensional space-time visualization for understanding three-dimensional motion. *The Journal of the Institute of Image Electronics Engineers of Japan 36*, 4 (2007), 371–381. 2

[SH11a] SAKAI Y., HASHIMOTO S.: Four-dimensional geometric element definitions and interferences via five-dimensional homogeneous processing. *Journal of visualization 14*, 2 (2011), 129–139. 4

[SH11b] SAKAI Y., HASHIMOTO S.: Four-dimensional mathematical data visualization via embodied four-dimensional space display system. *Forma 26*, 1 (2011), 11–18. 2

[Ste78] STEWART B.: *The unseen universe: or, physical speculations on a future state*. Macmillan, 1878. 2

[VF03] VON FOERSTER H.: Cybernetics of epistemology. In *Understanding Understanding*. Springer, 2003, pp. 229–246. 2

[Wer00] WERTHEIM M.: *The pearly gates of cyberspace: A history of space from Dante to the Internet*. WW Norton & company, 2000. 2

[Wil93] WILLINK A.: *The World of the Unseen: An Essay on the Relation of Higher Space to Things Eternal*. Macmillan, 1893. 1

[WLG97] WEGENKITTL R., LOFFELMANN H., GROLLER E.: Visualizing the behaviour of higher dimensional dynamical systems. In *Proceedings. Visualization'97 (Cat. No. 97CB36155)* (1997), IEEE, pp. 119–125. 2

[WWL*18] WU Z., WANG X., LIN D., LISCHINSKI D., COHEN-OR D., HUANG H.: Structure-aware generative network for 3d-shape modeling. *arXiv preprint arXiv:1808.03981* (2018). 6

[WYF*13] WANG W. M., YAN X. Q., FU C.-W., HANSON A. J., HENG P. A.: Interactive exploration of 4d geometry with volumetric halos. In *The 21th Pacific Conference on Computer Graphics and Applications-Short Papers* (2013), Citeseer, pp. 1–6. 2

[Yan15] YAN X.: *New directions in four-dimensional mathematical visualization*. PhD thesis, Nanyang Technological University, Singapore, 2015. 1

[YFH12] YAN X., FU C.-W., HANSON A. J.: Multitouching the fourth dimension. *Computer 45*, 9 (2012), 80–88. 2

[You20] YOUNGBLOOD G.: *Expanded cinema*. Fordham University Press, 2020. 2

[ZH07] ZHANG H., HANSON A.: Shadow-driven 4d haptic visualization. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1688–1695. 2

[Zöl82] ZÖLLNER J. K. F.: *Transcendental Physics: An Account of Experimental Investigations from the Scientific Treatises of Johann Carl Friedrich Zöllner*. WH Harrison, 1882. 2

[ZV00] ZACHARIÁŠ S., VELICHOVA D.: Projection from 4d to 3d. *Journal for Geometry and Graphics 4*, 1 (2000), 55–69. 4

[ZWR14] ZHANG H., WENG J., RUAN G.: Visualizing 2-dimensional manifolds with curve handles in 4d. *IEEE transactions on visualization and computer graphics 20*, 12 (2014), 2575–2584. 2