# 4D Cubism: Modeling, Animation and Fabrication of Artistic Shapes

**Quentin Corker-Marin**
Bournemouth University
Glassworks, Ltd.

**Alexander Pasko**
Skolkovo Institute of Science and Technology
Bournemouth University

**Valery Adzhiev**
Bournemouth University

The paper deals with creating artistic shapes in a cubist style. We propose adding cubist features to (or cubification of) time-variant sculptural shapes. A new concept of a 4D cubist camera is introduced for multiple projections from 4D space-time to 3D space and 3D printing for stop-motion animation is applied.

This paper describes an original approach to creating time-variant artistic shapes in a particular fine arts tradition, namely related to the cubism movement. Dynamic sculptures created with help of computer-aided design and animation tools and related to a proper artistic style are surprisingly rare. There are well-established methods and tools allowing for modeling, rendering and animation of an almost infinite variety of shapes in the context of computer animation and the VFX industry as well as manufacturing and physical simulation. More specifically, some powerful design methods and tools for sculpturing static shapes have been available.[1] However, the creation of dynamic or kinetic art shapes is mostly concerned with traditional engineering and manufacturing practices.[2-3] Some recent artistic projects implemented using modern animation and fabrication tools did start to appear.[4-5]

Séquin[6] has formulated the requirements for "a dream CAD system" with a focus on designing and producing sculptural forms by artists. He stressed that it is the specific aesthetic criteria that are paramount in the context of the final evaluation of CAD tools for sculpting. Introducing the dynamic dimension adds even more specifics. The traditional production process related to computer animation is normally a collective multi-phased effort. An artist creating a dynamic sculpture will normally work as an individual, and as we will see later, modeling and animation phases are very much overlapped and even integrated.

There are several distinctive features in cubist art. The first step in almost any cubist art work is shape simplification including its approximation by relatively large planar facets or with a limited set of solid primitives such as cylinders, spheres and cones (see Figure 1a). Local shape distortions can also be made, such as moving or rotating salient features (see Figure 1b). Some works display full or partial blending between two or several given shapes. Some cubist sculptures represent articulated motion of human bodies with local distortions used to underline the motion (see Figure 1c).

a b c

Figure 1. Cubist sculptures by Umberto Boccioni: (a) Development of a Bottle in Space, 1913; (b) Antigraceful, 1913; (c) Unique Forms of Continuity in Space, 1913 (Source: Metropolitan Museum of Art, used with permission).

Cubist painters moved away from traditional methods of projecting a 3D scene onto a 2D canvas, instead experimenting with using multiple viewpoints in a single composition. This is reflected in the notion of a cubist camera.[7] We propose to look at the theme of taking an n dimensional (*n*D) object or scene and reducing it to an (*n-1*)D representation through the new concept of a 4D cubist camera, which, by analogy with the 3D cubist camera, is projecting two or more instances of a time-variant shape from 4D space-time to 3D space at the given time moment with possible preliminary blending between these instances.

In a nutshell, we propose both a theoretical framework for time-variant cubistic sculptural shapes as well as a technological pipeline embracing all the main phases of their production cycle. We have been developing algorithms and software tools that allow artists to realize every distinctive feature of the cubist style and then to render and animate them. Physical 3D printed shapes in the dynamic context are especially worthy of research as 3D printing is becoming an omnipresent technology across the creative industries including art, design and stop-motion animations.

## 4D CUBIST MODELING FRAMEWORK

In this section we introduce a general 4D modeling framework allowing for creating time-variant 3D shape model of a cubist sculpture. We assume that the given initial shape can be either modeled using 3D modeling tools or designed in physical form and then scanned.

### Shape Faceting

We consider a facet as a solid piece obtained by the set-theoretic (or Boolean) intersection between some solid primitive (solid ball or tetrahedron) with the initial shape turned into a solid. The aim of the faceting step is to subdivide the given shape model into facets, provide an interface to control the faceting and apply unique transformations to each of the facets later for producing cubist effects of the shape distortion. A set of solid primitives is introduced in accordance with the octree data described below.

Given an initial polygonal mesh object $M$, a subset $V$ of the points in $M$ are selected (where $V = \{v_1, v_2, \ldots, v_n\}$ and $v_i \epsilon M$ for $v_i$ in $V$ ) and passed to the octree generator. The generator takes these points and outputs a set of all the octree leaf nodes $N$, where $N = \{N_1, N_2, \ldots, N_m\}$. By manipulating the maximum depth of the octree and the maximum points per node, parts of the mesh with high point density are subdivided more than other lower resolution parts. This results in a non-uniform grid that can then be used to transform the original geometry with a higher degree of feature sensitivity than a uniform grid (see Figure 2).

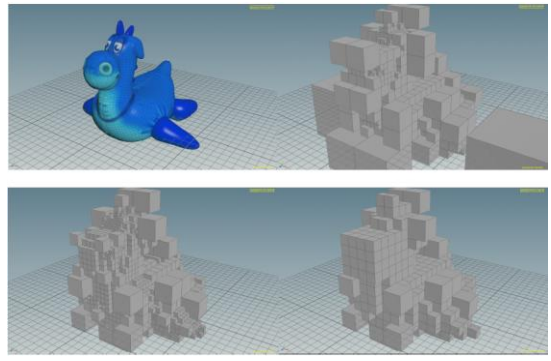

Figure 2. Building octrees for the given geometry. Top Left: Geometry passed into Octree node. Top Right: Result of depth limit at 10 and max points at 30. Bottom Left: Result of depth limit at 10 and max points at 10. Bottom Right: Result of depth limit at 5 and max points at 10.

It is illustrated in Figure 2 that, by changing the parameters of the octree node, feature sensitivity can be achieved to a greater or lesser degree depending on what the artist intends to do. This is one example of a dynamic octree produced by animating parameters. In addition, a dynamic octree can be built at the user's request at each time step (frame) for an animated shape.

## Faceting with Local Time-Variant Distortions

Local time-variant distortions can be made such as moving or rotating salient features of the initial shape represented by facets as illustrated in Figure 3. The steps of the faceting procedure are as follows:

1. building an octree for the initial shape (see Figures 3a and 3b);
2. introducing solid primitives in the octree nodes (see Figure 3c);
3. intersection of solid primitives with the initial shape producing facets (see figure 4);
4. rotating and scaling facets for local distortions (see Figure 3d).

Data is attached to each octree leaf node that represents how it will manipulate the original shape. The structure of each leaf node is $N_i = \{p_i, w_i, T_i, s_i\}$ for $N_i$ in $N$, where $p_i$ represents the node's position in the scene, which is its center with 3-dimensional coordinates, $w_i$ holds the node's width (one side length of the cube that represents the node), $T_i$ contains the node's set of transformations, which describe how the node will manipulate the facet locally. This is a user-defined set that can include scales, translations and rotations. Finally, $s_i$ is the node's associated primitive solid, which can be any 3D shape such as sphere, cube or tetrahedron (see Figure 3c) selected randomly or interactively by the artist to generate the facet by intersection of the primitive solid with the initial shape.

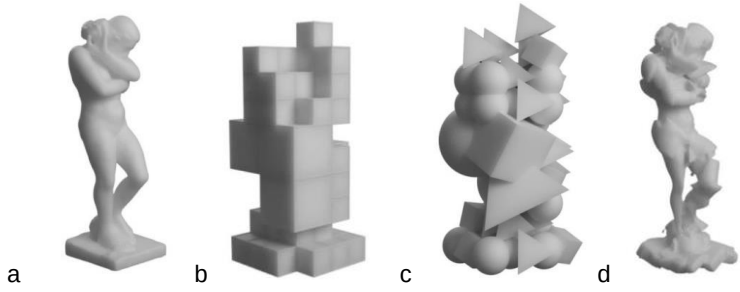


Figure 3. Steps of faceting the given sculpture: a) initial shape (scanned model "Eve", Three D Scans repository, http://threedscans.com); b) corresponding octree; c) associated primitive solids; d) rotated and scaled facets combined with manipulated initial mesh.

At the next step we involve a hybrid representation of the shape based on the initial mesh converted to Signed Distance Field (SDF, see sidebar). Once the octree data has been set, the nodes are iterated through and for each node $N_i$, its associated primitive solid $s_i$, is found and moved to the position of the leaf node $p_i$. The set-theoretic intersection is then applied between the solid and the original shape (both represented by SDFs) to generate a facet. This new facet resulting from intersection is then transformed using $T_i$. The transformation of each facet can be random or interactively controlled by the artist. These new facets are then merged together with the orig-

inal sculpture to produce the final sculpture (see Figure 3d). Note that a set of time-varying facets can be produced with a dynamic octree for each animation time step.

## 4D Cubist Camera with Space-Time Blending

Here we introduce the concept of a 4D cubist camera as a projection method from 4D space-time to 3D space. This is analogous to a 3D cubist camera, which projects a 3D scene from several directions onto a single 2D image place. Transparency can then be assigned to each projection image or a blend is made between them.

The 4D cubist camera projects two or more instances of a time-variant shape from 4D space-time to 3D space at the given time moment with possible preliminary blending between these instances. We employ space-time blending (see sidebar) for combining (or compressing) several shape instances (frames) into a single shape. Note that in this process animation is considered as modeling in 4D with projections to 3D.

Another aim of using space-time blending was to generate motion trails to follow the facets produced by the octree method described earlier. This was to produce effects similar to those seen in the sculptures of the Italian futurist Umberto Boccioni, an artist heavily influenced by cubism (see Figure 1c).

To achieve this, instead of blending between two separate objects, the blend is performed between non-adjacent shape instances (frames) *t* and *t-n*. By careful tweaking of the blend parameters, a satisfactory motion trail is produced that follows an animated object with a seamless transition between the object and its trail. This allows for the effects of the surrounding frames of an animation to influence the current frame being viewed, producing a cubist-type projection of 4D space-time onto 3D space.

# IMPLEMENTATION AND RESULTS

This section describes the main phases of a practical production pipeline allowing for creation of modelled, rendered, and animated cubist artefacts. Technically, it is an interactive, node-based system implemented on a modern computer animation development platform such as Houdini developed by Side Effects (https://www.sidefx.com/). It allows the user to create their own custom nodes in a number of ways including the Python scripting language and the Houdini C++ SDK. The techniques used in the implementation rely heavily on the use of sparse volumes (discretized SDFs) to calculate intersections, unions, and to manipulate volumes. Houdini utilizes the OpenVDB library (http://www.openvdb.org/) developed by DreamWorks. A supplementary video is available at https://www.youtube.com/watch?v=Vc9tYpRjlRo which includes actual animation sequences and images of artefacts.

## Octree Generation, Faceting and local Distortions

The octree implementation is encapsulated in a single Python node. The node was assigned parameters to allow the artist to control the generation of the octree. This allowed the artist to control the size and position of the tree, how it is drawn in the viewport and the depth and point limit (illustrated in Figure 2). The octree outputs a set of points that represent the positions of each of the leaf nodes. Each point has an additional attribute that represents the width of the node. The interface for the octree node allows the artist to output boxes in the place of points, making visualization more straightforward while the artist sets the desired parameters.

Once the octree has been generated, the original mesh is faceted and transformed (see Figure 4). To do this, the correct attributes must be set for the points produced by the octree node. These are the associated primitive solid and transformation values. This is implemented in several Houdini “wrangle” nodes that add the additional attributes “rotation” and “primitive solid” to the points produced by the octree node. While the primitive solid remains constant throughout a sequence, the transformation values can be time driven to produce more interesting dynamic sculptures. The artist can manipulate the rotation limit and speed sliders to alter the period and

amplitude of the rotations. One could even edit the code in the wrangle node directly to produce their own custom rotations.

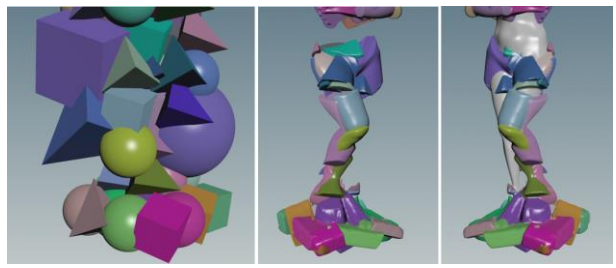

Figure 4. Primitive solids and corresponding faceting with local distortions: solid primitives assigned to octree nodes (left), facets obtained by intersection and rotated by the artist with assigned colors corresponding to the primitives (middle), facets combined with the original sculpture (right).

The faceting happens inside a loop that iterates through each point as shown in the pseudocode:

```
1  new_shape = empty_shape
2  for each node in N:
3     solid = node.solid
4     solid.moveTo( node.position )
5     new_facet = solid ∩ original_shape
6       for t in node.transforms:
7             new_facet.transform( t )
8  new_shape = new_shape U new_facet
```

Firstly, the primitive solid is copied to the point and converted into a VDB object (lines 3-4). Then the intersection of the primitive solid VDB and a VDB of the original mesh is calculated producing a new facet (line 5), the intersection result is then transformed (lines 6-7) and combined with the output of the loop for each of the other points (line 8).

This method relies heavily on converting each piece to and from a VDB because the use of this library produces much cleaner and more stable set-theoretic results than Boolean operations between polygonal meshes. However, to produce high quality results, the volume resolution must be high and the conversion is slow. Figure 4 shows zooms to the legs area of the sculpture in Figure 3 with primitive solids and corresponding rotated facets. Note that for the “Eve” model (see Figure 3a) the process of converting meshes to VDBs at final resolution has taken 13.1 sec.

## 4D Cubist Camera Implementation

The 4D cubist camera projection with space-time blend was implemented as a single node that takes in two shapes and allows the user to blend between them. The process involves three bounded blends in space-time, two of them to produce smoothed half cylinders using blended subtraction and the third blends between these two half cylinders. Blended subtraction is used to generate the half cylinders as this operation results in a much smoother final blend.

The user interface allows the artist to control the positions and blend parameters for each of the three blends individually and the global resolution of the volumes used to perform the blend. The blend point is the position in time that the blend is centered around. The blend radius controls the distance in the time dimension that the blend is performed across.

## Combined Faceting and Space-Time Blending in Motion

Note that the presented faceting and space-time blending phases can follow each other. All the processing phases are illustrated in Figure 5 starting from a pure cube and its octree generation, assigning solid primitives to octree nodes and faceting, followed by space-time blending of two time instances of the same cube and faceting the result with a dynamic octree, because the blend shape changes at every time step.

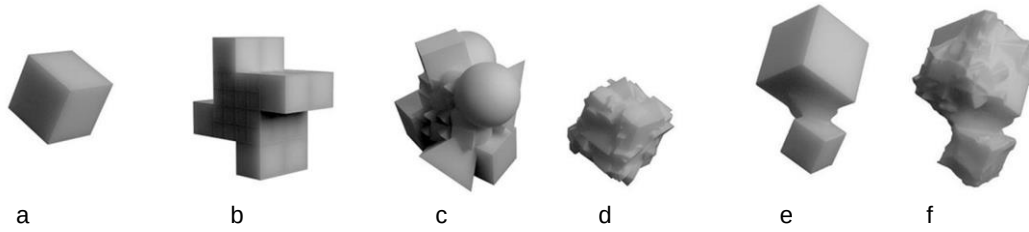


Figure 5. Combination of faceting and space-time blend in motion. a): initial moving, scaling and rotating cube; b): octree for a rotated cube; c): solid primitives in the octree nodes; d): faceting with local transformations; e): space-time blend trail between two cube instances; f): faceted space-time blend.

Similar processing steps applied to an articulated human figure are depicted in Figure 6: dynamic faceting of an articulated figure (see Figure 6 left); projection of three instances of the figure taken at different time moments and combined with two consecutive space-time blending operations (see Figure 6 middle) and the same blended shape with dynamic faceting (see Figure 6 right).

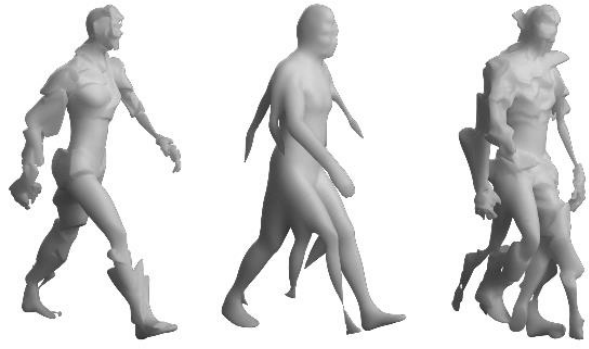

Figure 6. Human figure with articulated motion, faceting and space-time blend. left: faceted walking figure; middle: space-time blend between three shape instances; right: faceted blended figure.

A breakdown of the time it takes to create a single frame of modeling-animation (see Figure 6) using a computer with a 4-core 2.5 GHz Intel Core i7:

- single frame total time: 1m 7.8s
- time spent doing space-time blending for that frame: 37.1s
- time spent converting to VDBs for space time blending: 25s
- time spent faceting for that frame: 30.7s
- time spent converting to VDBs for faceting: 19.2s

## Artistic Rendering

We have implemented and applied the 3D cubist camera (see sidebar) rendering to animated cubified sculptures. The examples in Figure 7 show the result of combining the cubist camera rendering with dynamic faceting of an animated mesh without (left image) and with artistic effects applied in compositing (right image).

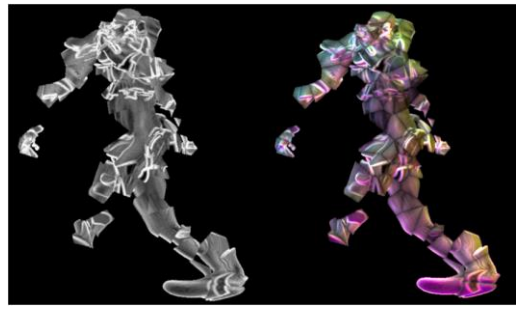

Figure 7. Cubist camera rendering of faceted walking human figure: without (left) and with (right) artistic effects.

Here, the colored image is produced by the application of artistic effects obtained by compositing together different passes

produced by the cubist renderer. It is done by multiplying together the color pass, normal pass, facet pass and facet id pass. The color pass shows the mesh saliency (curvature) where areas of high saliency on the mesh are colored white and low saliency regions are black, highlighting areas of detail on the mesh.

Notice how in Figure 7 the cubist camera allows the bottom of the feet to be visible alongside the stomach area. The cubist camera also allows the back leg, normally further away and smaller in a traditional perspective projection, to appear larger in the final image than the closer, front leg.

## 3D FABRICATION

One of the possible outcomes of the pipeline processing can be 3D printed objects. Figure 3 and Figure 8 (left) illustrate cubification of a static sculpture starting from scanned data and finishing with printing on an Ultimaker 2 3D printer (http://www.ultimaker.com).

Figure 8. 3D printed artefacts: the cubified sculpture (left); phases of the cubification with dynamic faceting suitable for stop-motion animation (right).

Figure 8 (right) illustrates another branch of our pipeline, where a scanned face is converted to a mesh and then to an SDF and undergoes set-theoretic union with a Function Representation (FRep, see sidebar) mug model. Then, several phases of faceting and local distortions with a dynamic octree are applied to the right part of the obtained shape. Such 3D printed animation "frames" are suitable for stop-motion animation currently re-emerging in the film industry. The simplest way of producing such an animation is placing several instances of a time-varying 3D printed shape in a pre-film animation device such as a zoetrope.

## DISCUSSION AND FUTURE WORK

This paper describes an original approach to creating dynamic sculptures in the fine arts tradition of cubism thus extending a traditional suite of rather technical, lacking art specifics, methods available to artists seeking novel artistic forms. The step-by-step creative process is logical, semantically simple and cohesive, and should be easily reproducible.

This project has served as inspiration for some original theoretical notions. Robbin in his book[8] draws the reader's attention to parallels between cubism and 4D space, *n*D space and non-Euclidian geometries. Our 4D Cubist Camera integrating modeling, rendering and animation in the context of multidimensional conceptual space rather radically extends the concept of "expressive non-realistic modeling".[9] We also propose to use not just basic motion depicting techniques thus crafting "motion sculptures",[10] but to exploit a rich suite of artist-controlled methods to apply a cubist non-photorealistic style to animated and fabricated artefacts.

There is a potential for further development, both in advancing the underlying algorithms and their implementation. In particular, a more sophisticated technique for combining the space-time blending with the faceting is to be developed. Better continuity between different frames resulting in animations without flickering fractures can also be achieved. There is also the potential for adding complex material attributes to the blends. Another idea for further development of the animation branch is to introduce cubist motion, where smooth animation curves are approximated by broken polylines with some additional distortions.

# SIDEBAR: TECHNICAL BACKGROUND

Here we briefly characterize the main mathematical models we have employed in our algorithm development.

## Hybrid Representation

The Function Representation (FRep) and function-based modeling are well-researched topics,[11] where objects are implicitly defined with real functions (or scalar fields), and for the given shape (point set) M its defining continuous function is $f(x) \geq 0$, if $x \in M$, and $f(x) < 0$, if $x \notin M$.

Note that the function changes its sign at the shape boundary. There are many developed operations taking single or multiple shapes and producing a new shape in 3D or higher dimensional space. The defining function can be generated for input point clouds with an interpolating surface, shapes with a curve skeleton, mesh surfaces (see details below) or voxel arrays. This makes FRep essentially a hybrid representation of volumetric objects as not only an object surface, but also its internal points which are uniformly defined by a function.[12]

## Signed Distance Fields

For a given closed polygonal mesh representing a boundary of some volume object, the most natural way to obtain a defining function is to introduce a Signed Distance Field (SDF). The absolute value of the SDF at any given point is the minimal distance from this point to the mesh surface. The SDF has zero value at any point on the mesh and changes its sign when crossing the mesh from outside to inside.[13] We prefer an SDF representation of the polygonal mesh rather than the mesh itself for our purposes, because set operations are very stable with SDF and do not result in holes, cracks and self-intersections as it happens with meshes.

## Space-time Blending

Space-time blending is a method for metamorphosis between FRep solids. The idea of performing bounded blending in 4-dimensional space-time was suggested by Galina Pasko and colleagues.[14] To perform the blend over time, two FRep solids, defined in $n$-dimensions are converted into $(n+1)$-dimensional space-time infinite cylinders. Each cylinder is converted into a half-cylinder by performing a subtraction operation with a half-space in space-time to leave a time gap between them for the shape transition. Then a bounded blend is applied to half-cylinders across time. An intermediate blended shape for the specific time moment is obtained by taking a cross-section of the blended shape orthogonal to the time axis.

## Cubist Camera

Our rendering system was built with techniques based on the notion of a 3D cubist camera. The brief summary here is based on the work by Sami Arpa and colleagues[15] and involves a two pass rendering system from a spherical camera. First, the scene is rendered from a spherical camera to produce a main image and a saliency image. These two images are used to generate a fracturing pattern that is then applied to the camera to produce spatial imprecision and perspective correction. It can rely on Voronoi fracturing pattern. The fractured camera is then used for a second pass of rendering to produce the final cubist images which then go through a process of applying artistic effects.

# REFERENCES


1. C. H. Séquin, “2-Manifold Sculptures”, *Bridges Conf. Proc*., 2015, pp. 17-26.
2. F. Popper, *Origins and Development of Kinetic Art*, New York Graphic Society/Studio Vista, 1968.
3. N. Dodgson, “Engineering Art and Telling Tales: Anish Kapoor at the Royal Academy”, *Interdisciplinary Science Reviews*, vol. 41, no. 4, 2017, pp. 281-296.
4. V. Adzhiev, P. Comninos, and A. Pasko, “Augmented Sculpture: Computer Ghosts of Physical Artifacts”, *Leonardo*, vol. 36, no. 3, 2003, pp. 211-219.
5. S. Hessels, “Lenticular Waterwheels: Simultaneous Kinetic and Embedded Animation”, *SIGGRAPH 2017 Art Papers*, *Leonardo*, vol. 50, no. 4, 2017, pp. 394-399.
6. C. H. Séquin, “CAD tools for Aesthetic Engineering”, *Computer-Aided Design*, vol. 37, no. 7, 2005, pp. 737-750.
7. A. Glassner, “Digital Cubism”, *IEEE Computer Graphics and Applications*, vol. 24, no. 3, 2004, pp. 82-90, and vol. 24, no. 4, 2004, pp. 84-95.
8. T. Robbin, *Shadows of Reality, the Fourth Dimension in Relativity, Cubism and Modern Thought*, Yale University Press, New Haven and London, 2006, 139p.
9. R. Gal et al., “3D Collage: Expressive Non-realistic Modeling**”,** *5th international symposium on Non-photorealistic animation and rendering, Proc*. ACM, 2007, pp. 7-14.
10. R. H. Kazi, “ChronoFab: Fabricating Motion”, *CHI Conference on Human Factors in Computing Systems Proc.* ACM, 2016, pp. 908-918.
11. A. Pasko et al., “Function Representation in Geometric Modelling: Concepts, Implementation and Applications”, *The Visual Computer*, vol. 11, no. 8, 1995, pp. 429-446.
12. D. Kravtsov et al., “Embedded Implicit Stand-ins for Animated Meshes: a Case of Hybrid Modelling”, *Computer Graphics Forum*, vol. 29, no. 1, 2010, pp. 128-140.
13. D. Koschier, C. Deul, and J. Bender, “Hierarchical hp-Adaptive Signed Distance Fields**”**, *ACM Siggraph / Eurographics Symposium on Computer Animation Proc.*, 2016, *pp.* 189-198.
14. G. Pasko, A. Pasko, and T. Kunii, “Space-time Blending”, *Computer Animation and Virtual Worlds*, vol. 15, no. 2, 2004, pp. 109-121.
15. S. Arpa et. al., “Perceptual 3D Rendering Based on Principles of Analytical Cubism”, *Computers & Graphics*, vol. 36, no. 8, 2012, pp. 991-1004.


# ABOUT THE AUTHORS


**Quentin Corker-Marin** is a junior 3D artist at Glassworks Ltd., London. In the end of 2017 he graduated from the National Centre for Computer Animation (NCCA), Bournemouth University, UK with a first class honours degree in computer visualisation and animation. He was awarded the second place in the ACM Student Research Competition at SIGGRAPH 2017. His interests are in procedural systems and digital effects. Contact him at quentinjcm@hotmail.com.

**Alexander Pasko** is a professor at the Center for Design, Manufacturing and Materials, Skolkovo Institute of Science and Technology, Russia, and at NCCA, Bournemouth University, UK. He has a PhD from Moscow Engineering Physics Institute - The National Research Nuclear University (MEPhI). His research interests include solid and volume modelling, Computer-Aided Design, digital fabrication, and computer art. Contact him at apasko@bournemouth.ac.uk.

**Valery Adzhiev** is a principal academic at NCCA, Bournemouth University. He has a PhD in Computer Science from Moscow Engineering Physics Institute - The National Research Nuclear University (MEPhI). His research interests include computer animation, geometric modelling, programming languages, computer fabrication, and computer art. Contact him at vadzhiev@bournemouth.ac.uk.