Hybrid Function Representation for Heterogeneous Objects

A. Tereshin 1, A. Pasko 2, O. Fryazinov 1, V. Adzhiev 1

1 National Centre for Computer Animation, Bournemouth University, United Kingdom
2 Skolkovo Institute of Science and Technology, Russia

Abstract
Heterogeneous object modelling is an emerging area where geometric shapes are considered in concert with their internal physically-based attributes. This paper describes a novel theoretical and practical framework for modelling volumetric heterogeneous objects on the basis of a novel unifying functionally-based hybrid representation called HFRep. This new representation allows for obtaining a continuous smooth distance field in Euclidean space and preserves the advantages of the conventional representations based on scalar fields of different kinds without their drawbacks. We systematically describe the mathematical and algorithmic basics of HFRep. The steps of the basic algorithm are presented in detail for both geometry and attributes. To solve some problematic issues, we have suggested several practical solutions, including a new algorithm for solving the eikonal equation on hierarchical grids. Finally, we show the practicality of the approach by modelling several representative heterogeneous objects, including those of a time-variant nature.

Keywords: hybrid representation, distance fields, eikonal solver, function representation, heterogeneous objects, volumetric modelling, time-variant objects.

1. Introduction
Heterogeneous volumetric object modelling is a rapidly developing field and has a variety of different applications. Volume modelling is concerned with computer representation of object surface geometry as well as its interior. Homogeneous volume modelling, better known as solid modelling, deals with volume interior uniformly filled by a single material. Heterogeneous object is a volumetric object with interior structure where different physically-based attributes are defined, e.g. spatial different material compositions, micro-structures, colour, density, etc. [KT07, LFSL20]. This type of objects is widely used in applications where the presence of the interior structures is an important part of the model. Additive manufacturing, physical simulation and visual effects are examples of such applications.

The most widely used representations for defining heterogeneous objects are boundary representation, distance-based representation, function representation and voxels. Boundary representation (BRep) [LFAB14] maintains its prevailing role due to its numerous well-known advantages. It works well in solid modelling for objects consisting of a set of polygonal surface patches stitched together to envelope the uniform and homogeneous structure of its material. However, BRep is not inherently natural for dealing with heterogeneous objects, especially in the context of additive manufacturing and 3D printing [LEM*17], where volume-based multi-material properties are paramount as well as in physical simulation where the exact representation rather than an approximate one can be important [NMK*06].

On the contrary, volumetric representations in the form of voxels [WYJ*11] are more natural for defining such heterogeneous objects as they are based on volumetric grids. Voxels represent an object as a set of cubic cells at which the geometry along with the object attributes are defined. However, this representation essentially approximates both the geometry model and the material distribution in interior of the object as their definition is limited by the resolution of the voxel grid.

On the other hand, function-based, and more specifically, distance-based representations are able to represent the object and its interior structure in both continuous and discrete forms [JBS06]. They are exact, embrace a wide range of geometric shapes and naturally define many physically-based attributes. There are a lot of well-established operations for these representations. Most of them provide distances to the object surface. However, distance functions (DFs) are not essentially continuous, they can have medial discontinuities and are not necessarily smooth. This potentially results in non-watertight surfaces, and artefacts, such as creases, after applying some operations, for instance, blending and metamorphosis, which are important for many applications. Undesired artefacts (stresses, creases, etc.) can also appear as the result of defining distance-based attribute functions.

We consider function-based and distance-based representations as a promising conceptual and practical scheme to deal with het-
erogeneous objects, especially in the context of a number of topical application areas concerned with exact volume-based geometric modelling, animation, simulation and fabrication. However, the existing representational schemes of that type appear in many variations and the field as a whole exhibits a rather fragmented and not properly formalised suite of methods. There is an obvious need for a properly substantiated and unifying theoretical and practical framework. This challenge can be considered in the context of the emergence of new representational paradigms suitable for the modelling of applications, such as modelling of material structures, that was outlined and substantiated in [RRSS16].

In this work we propose a novel function-based representational scheme. We introduce a mathematical framework called hybrid function representation for defining a heterogeneous volumetric object with its attributes in continuous and discrete forms. It is based on hybridisation of several DF-based representations that unifies their advantages and compensates for their drawbacks. This representational scheme aims at dealing with heterogeneous objects with some specific time-variant properties important in physical simulations related to both geometry and attributes. The idea was initially tested in a short paper [TAFP19] where the scheme unifying the function representation (FRep) and the signed distance functions (SDFs) had been sketchily outlined.

The contributions of our work can be formulated as follows:

- We provide a thorough survey of the relevant representations, aiming at their classification and identifying their advantages and drawbacks. We formalise the notions of the adaptively sampled distance fields (ADFs) and interior distance fields (IDFs).
- On the basis of an analysis of the well-established FRep and DF-based representations, namely SDFs, ADFs and IDFs, we formulate the requirements for a novel unifying hybrid representation called HFRep.
- We propose a mathematically substantiated theoretical description of the HFRep with an emphasis on defining functions for HFRep objects’ geometry and attributes.
- We describe a basic algorithm for generating HFRep objects in terms of their geometry and attributes, and develop its main steps in a detailed step-by-step manner.
- We identify the problematic issues associated with several steps of the basic algorithm and propose several practical solutions. In particular, we present a novel hierarchical fast iterative method for solving the eikonal equation on hierarchical grids in 2D. The developed algorithm was used for generating HFRep based on FRep and ADF.

2. Related works

There is a huge body of works dealing with different aspects of representational schemes for volumetric heterogeneous objects. In this section we first concentrate on those works that deal with representations for geometric shapes. Then, we consider some existing hybrid representations. The basic methods for defining attributes in interior of volumetric objects are also reviewed.

2.1. Geometry representations

An overall geometry can be represented by boundary surfaces or by any other solid representational scheme including procedurally defined scalar fields.

Boundary representation (BRep) remains the most popular representation. It can be described by a polygonal or other surface model. Polygonal models can be represented as nested polygonal meshes bounding the regions with different material density values [LFAB14]. This representation scheme has the following problems: the possible presence of holes or gaps in a mesh, normals can be flipped, triangles in the mesh can be intersecting or overlapping with each other, polygonal shells can be noisy. Among many polygon-based approaches applicable to heterogeneous objects we pay attention to the diffusion surfaces introduced in [TSN10]. This approach deals with 3D surfaces with colours defined on both sides, such that the interior colours in the volume are obtained by diffusing colors from nearby surfaces. It was used for modelling objects with rotational symmetry. It is efficient to compute, but cross-sections of the mesh obtained with further triangulation could suffer from discretisation artefacts.

Another way to represent the overall object geometry is constructive solid geometry (CSG) [Req90]. Originally all solids were homogeneous, but later primitives could carry on some information that can be interpreted as a material index [Bow95]. The operations on attributes corresponding to set-theoretic operations were provided.

The most widely used method for defining heterogeneous objects is the voxel representation [WYJ+12, BKW18]. The object is subdivided into multiple cubic cells with defined geometric and attribute parts in them. However, geometric and attribute properties are essentially approximated according to the voxel grid resolution.

In the context of this work we pay a special attention to defining a heterogeneous object geometry using different types of scalar fields. The most common schemes of that type are already mentioned FRep [PASS95], SDFs [JBS06], ADFs [FPRJ00] as well as IDFs [BS06] represented by functions that we call interior distance functions (IDFs). We will discuss them in more details in the next section.

Another widely used approach for obtaining a continuous distance based definition of the object is to compute the solution of the optimal mass transportation [SRGB14]. This method assumes the numerical solution of a partial differential equation (PDE) dedicated to the Monge-Kantorovich optimisation problem which can be quite time-consuming. In [YYW12], volumetric objects with multiple internal regions were suggested to define the object-space multiphase implicit functions. These functions preserve sharp features of the object and in some cases provide better results than SDFs.

Distance-based objects can also be defined using the level-set method [GFO18] which provides an implicit representation of a moving front. The main advantage of this method is that it can handle various topological changes of the object thus implementing the dynamic implicit surfaces. The evolution of the front is controlled by the solution of the level-set equation. The obtained function is transformed into a signed distance function using the solu-
2.2. Hybrid representations

The main feature of any hybrid representation is that it unifies advantages of several representations and compensate for their drawbacks. In [PASS01], the concept of hypervolumes was introduced as an extension of the general object model [KBDH99] that unifies the advantages of FRep and hybrid volumes. Hypervolume describes a heterogeneous object as n-dimensional point-set with defined attributes, operations and relations over them. Another hybrid approach called hybrid surface representation was introduced in [LSD04]. It is based on BRep and an implicit surface representation and was used for heterogeneous volumetric modelling and sculpting.

There are approaches when an entire object can be split into disjoint or adjacent components sharing their boundaries. The space partitions can be defined by additional boundary surfaces or scalar fields. In the most general case, these partitions are represented by mixed-dimensional cells combined into a cell complex. The combination of a cellular representation and a functionally based constructive representation was proposed in [AKK*02]. This model makes it possible to represent dimensionally non-homogeneous elements and their cellular representations. The authors showed that attributes may reflect not only material, but any volumetric distribution such as density or temperature.

There are some works dedicated to the construction of hybrid representations based on SDFs, ADFs and IDFs. In [TP11], the authors have introduced hybridisation of meshfree, RBF-based, DF-based and collocating techniques for solving engineering analysis problems. The proposed technique enables exact treatment of some reinitialisation equation. Level-set methods have been used in many applications, such as shape optimisation, computational fluid dynamics, trajectory planning, image processing and others [GFO18].

Multi-material heterogeneous volumetric objects [NK*18] consist of three elements: object geometry, object components (e.g. domains, partitions or cells sharing their boundaries) and material distribution. Material distributions can be defined using material indexes, piecewise polynomials or continuous scalar fields that provide a resolution independent distribution.

To define material in the interior of the object, a spatial partitioning of the object in several spatial regions should be made. Perhaps, the most widely used approach is a voxel representation of the object [WYJ11]. In [HL09], Hiller and Lipson suggested to use a voxel data structure as a material building-block for layered manufacturing. In another work [DTD15], a bitmap voxel-based method that uses multi-material high-resolution additive manufacturing (AM) was introduced. The material properties are combined in local material compositions that are further fetched in a AM system. In [BK*18] a multi-material voxel-printing method using a high-resolution dithering technique was introduced. The material in the voxelised object is defined using spatial indexing.

Material distribution in interior of the volumetric object can also be defined using DF-based approaches. In [BSS04], DFs were used for parameterisation of the space by distances from the material features either exactly or approximately, taking into account that the defining attribute function should be at least C1 continuous to avoid creases and stresses in it. In [FPS15], an IDF based method for defining gradient materials was introduced. IDFs are represented as an approximate Euclidean shortest path and are used for interpolation between sources. In [SG17], the authors considered the decomposition of the geometry using the existing class of material distance-based functions that set up a material variation in heterogeneous objects using the medial axes transform.

3. Distance function-based representations

In this section we provide some mathematical background and outline in a formalised manner four functionally-based representations that will be used to devise the hybrid function representation to be introduced in Section 4. We describe in necessary detail the mathematical basics of those representations and propose the formal definitions for two of them, namely ADF and IDF. The advantages and drawbacks of the representations are also systematically outlined.

3.1. Mathematical background and notations

Let us introduce the mathematical definitions which will be used hereinafter. First, we introduce the definition of a metric space and a distance function that follows [HDE14]:

**Definition 3.1** Let X be a non-empty point set in a Euclidean vector space \( \mathbb{R}^n \) and let function \( d : X \times X \to \mathbb{R} \) be such that for points \( p, q, r \in X \subset \mathbb{R}^n \) the following conditions are satisfied:

- \( d(p, q) \geq 0 \)
- \( d(p, q) = 0 \) if and only if \( p = q \)
- \( d(p, q) = d(q, p) \)
- \( d(p, q) + d(q, r) \leq d(p, r) \). Then the function \( d(\cdot, \cdot) \) is called a metric or a distance function on set \( X \) and the pair \( (X, d) \) is called a metric space.
In this work we are focusing on distance-based representations for defining volumetric objects. Let us introduce a more instrumental notion for the distance function that satisfies definition 3.1 and that we will use subsequently in the next sections, as follows [Sch97]:

**Definition 3.2** Let X be a point set in a Euclidean vector space $\mathbb{R}^n$ and let $\langle \cdot, \cdot \rangle$ be an inner product defined in $\mathbb{R}^n$. Then the Euclidean norm of the point $p \in X$ is defined as $||p|| = \sqrt{\langle p, p \rangle}$. If $q \in X$ is another point, then the distance between these two points is defined as a function:

$$F_{DF}(p, q) = ||p - q|| = \sqrt{\langle p, q \rangle}$$  \hspace{1cm} (1)

In this work we deal with functionally defined objects that are specified as closed point subsets $G \subseteq X$. As we are dealing with the objects defined by the functions, a point membership classification is used to distinguish between exterior, boundary and interior of the object. Therefore, let us introduce a formal definition of the boundary $\partial G$ of the subset $G$ as follows:

**Definition 3.3** Let $G$ be a subset of the defined metric space $(X, d)$. The boundary $\partial G$ of this subset $G$ is defined as $\partial G = G_1 \cap G_2$, where $G_1$ is a closed set in $X$, and interior of $G$ is $G_{int} = \bigcup \{G_U : G_U \subseteq G\}$, where $G_U$ is an open set in $G$.

There are two important properties of the functions that we rely on in the next sections: continuity and smoothness. The continuity of the function is defined as follows [DZ10]:

**Definition 3.4** Let $X$ be an open subset of $\mathbb{R}^n$. Let $C(X)$ be the space of continuous functions $X \rightarrow \mathbb{R}^n$. Let $\mathbb{N}^n$ be the set of all tuples $\alpha = (\alpha_1, ..., \alpha_n) \in \mathbb{N}^n$. Then $|\alpha|$ is the order of $\alpha$ and $\partial^{\alpha}$ is the partial derivative. For an integer $k \geq 1$

$$C^k(X) := \{ f \in C^{k-1}(X) : \partial^\alpha f \in C(X), \forall \alpha, |\alpha| = k \}$$  \hspace{1cm} (2)

where

$$|\alpha| = \sum_{i=1}^{n} \alpha_i, \quad \partial^\alpha = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} ... \partial x_n^{\alpha_n}}$$  \hspace{1cm} (3)

In this work we discuss functions that are either at least $C^0$ or $C^1$ continuous. A function $f$ is said to be of class $C^0$ if it is continuous on $X \subseteq \mathbb{R}^n$. A function $f$ is said to be of class $C^1$ if it is differentiable and continuous on $X \subseteq \mathbb{R}^n$.

Formally, smoothness of the function follows from the previous definition and can be defined as in [Sch97]:

**Definition 3.5** A function $f : X \rightarrow \mathbb{R}^n$ is called smooth if it is $n$-times differentiable, i.e. if it belongs to a specific class of functions that can be defined as $C^n(X)$ for which $f^{(n)}$ exists and it is continuous, particularly if it satisfies $C^\infty(X) = \bigcap_{k=1}^{\infty} C^k(X)$.

### 3.2. Function representation

Let us introduce the definition of FRep [PASS95]:

**Definition 3.6** Let the geometric shape of the object $O_{FRep}$ be defined as a closed point subset $G$ of $n$-dimensional point set $X$ in Euclidean space $\mathbb{R}^n$ with $p = (x_1, ..., x_n) \in \mathbb{R}^n$ using a real-valued defining function $F_{FRep}(p)$. Then function representation is defined as

$$O_{FRep} := F_{FRep}(p) \geq 0$$  \hspace{1cm} (4)

The FRep function (see Fig. 1, (a)) provides the information about point membership:

$$\begin{cases}
F_{FRep}(p) < 0 & p \in X \setminus G \\
F_{FRep}(p) = 0 & p \in \partial G \\
F_{FRep}(p) > 0 & p \in G_{int}
\end{cases}$$  \hspace{1cm} (5)
Note, that in this work we will consistently follow this convention for all types of distance functions, meaning that they are positive inside and negative outside with respect to the zero-level set. The major requirement for $F_{FRep}(p)$ is to be at least $C^0$ continuous.

FRep is a high-level and uniform representation of multidimensional geometric objects. The subject of particular interest is 4D objects with fourth coordinate specified as time. FRep generalizes implicit surface modelling and extends a CSG approach. FRep has a closure property as operations applied to the FRep defining functions produce continuous resulting FRep functions. The FRep object can be defined as a primitive (e.g. sphere, torus, cylinder, octahedron etc.) or as a complex object that is defined in the form of a constructive tree. In this case, primitives are stored in the leaves of the tree and operations are stored in its nodes.

There exist many well-developed operations, e.g. set-theoretic operations, metamorphosis, blending and bounded blending, offsetting, bijective mapping and others [PASS95]. FRep covers traditional solids [Kar99], scalar fields, heterogeneous objects including both static and time dependent volumes [I01].

Fig. 1 (a) shows the FRep field obtained using 14 set-theoretic operations applied to triangles and rectangles to construct the ‘bat’. In Fig. 2 we present a constructive tree that describes how a FRep object in the form of a ’snow flake’, that was converted to SDF, was created using union $\cup$ and intersection $\cap$ set-theoretic operations.

In general case, the FRep field is not distance-based as interior and exterior field isolines do not precisely follow the object shape in terms of its zero-level boundary. The advantages and drawbacks of the representation can be found in table 1, in the first column.

### 3.3. Signed distance function

Let us introduce the definition of SDF that relies on definitions 3.1, 3.2 and 3.3:

**Definition 3.7** Let $(X, d)$ be a metric space. Let the geometric shape $G$ of the object $O_{SDF}$ be specified in $(X, d)$ as a point subset $G \subseteq X$. Then a signed distance function $F_{SDF}(p)$ is defined as:

$$F_{SDF}(p) = \begin{cases} 
  d(p, \partial G) & \text{if } p \in G \\
  -d(p, \partial G) & \text{otherwise}
\end{cases} \tag{6}$$

where $d(p, \partial G) \equiv F_{DF}(p, \partial G)$. Then the SDF representation is defined as follows:

$$O_{SDF} := F_{SDF}(p) \geq 0 \tag{7}$$

The SDF function is at least $C^0$ continuous as it can be not differentiable at some points of Euclidean space $\mathbb{R}^n$ and it has gradient discontinuities on the object’s medial axes. The SDF representation provides the information about point membership in the same manner as FRep.

There are a lot of operations defined for SDFs. One of them is offsetting [BVG19]. A lot of operations for SDFs were defined in [PT92]. These are surface interpolation, multiple-object averaging, spatially-weighted interpolation, texturing, blending, set-theoretic operation and metamorphosis. Morphological operations such as erosion and dilation [You83] can be also applied to SDFs.

SDF can be used for a material definition in heterogeneous objects [BST04], additive manufacturing [BDN16], collision detection problems, particle simulations [KKL15] and others.

Fig. 1, (b) shows the SDF field generated for the ‘bat’ object. As it can be seen, the isolines are spaced equidistantly and follow the shape of the object. The advantages and drawbacks of SDF can be seen in table 1, second column.

### 3.4. Adaptively sampled distance function

Adaptively sampled distance function (ADF) [FPRJ00] is a distance function that is computed on hierarchical grids, e.g. tree-like data structures. ADF satisfies all the requirements of definitions 3.1, 3.2 and 3.7. To our knowledge, there is no well-established formal definition of ADF in the literature. There are several works where ADF is interpreted in a different way compared to [FPRJ00]. For example, in [WYJ11] ADF was defined using T-meshes with different interpolation operation for restoring the field, in [KDB16] it was suggested to use a hierarchical hp-adaptation for constructing ADF, in [TF18] it was suggested to construct ADF using estimation of the principal curvatures of the input surface. In this work we introduce a formal definition of ADF. Let us first give the definition of the hierarchical tree structure:

**Definition 3.8** Let a set of nodes and edges $(Q, E)$ be an undirected connected graph $T$ that contains no loops and starts at some particular node of $T$. Then such a graph $T$ is defined as a tree.

Let space $\mathbb{R}^n$ be subdivided according to the local details using some k-ary tree $T := (Q, E)$ with nodes $q \in Q$. Each node $q$ is defined as an n-dimensional cell. According to the SDF definition 3.7 we need to compute the distance to the boundary $\partial G$ of the geometric subset $G$. Taking these preliminaries into account, let us formulate the ADF definition in the constructive manner:

**Definition 3.9** Let the geometric shape $G \subseteq X$ of the object $O_{ADF}$ be defined in a metric space $(X, d)$. Let $(X, d)$ be subdivided into nodes $q \in Q$ with corner vertices $p_i$ according to the level of detail using k-ary tree $T := (Q, E)$. Let the boundary $\partial G$ be subdivided with the maximum tree depth, while $X \setminus G$ and $G_{\partial}$ be subdivided with some minimum tree depth. Let the corner vertices of Figure 3: An ADF field generated for the FRep object ‘heart’. The distance field is restored using bilinear interpolation. a) The constructed quadtree for the ‘heart’ object. b) The computed unsigned distance field.
in the slice corresponds to the ‘source’ point. c) the SDF slice of the same model with computed interior and exterior distances. Colour changing reflects how the distances are changing from interior to exterior of the object.

Figure 4: The IDF field computed for the ‘Stanford Bunny’ 3D mesh using the method described in [RLF09] and an SDF slice to show the difference in nature of these fields. a) the IDF field computed on the boundary of the mesh. Black isolines show how the field is changing according to the shape of the object; b) the interior slice of the mesh with computed IDF s. The yellow point p,

the boundary nodes q be defined as $p_b$. Then the distance function between these points is $d(p, p_b) ≡ FDF(p, p_b)$. Thereafter, the ADF distance function $F_{ADF}(p)$ on the tree $T$ is restored at each node $q$ using some interpolation function $F_I(p)$ and is defined as follows:

$$F_{ADF}(p) = \begin{cases} (F_I \circ F_{DF})(p) & \text{if } p \in G \\ -(F_I \circ F_{DF})(p) & \text{otherwise} \end{cases} \quad (8)$$

The ADF representation is defined in the form of an inequation:

$$O_{ADF} := F_{ADF}(p) \geq 0 \quad (9)$$

The ADF field generated as it was described in [FPRJ00] has $C^1$ discontinuities where the cells of different size appear and it has $C^1$ discontinuities caused by the bilinear/trilinear interpolation that was used for restoring a DF at each cell. The generated ADF field for the FRep ‘heart’ object can be seen in Fig. 3. The distance field was restored using bilinear interpolation, therefore, the field is non-smooth and has $C^1$ discontinuities (some isolines are disconnected).

The ADF representation provides the information about point membership in the same manner as FRep. The subset $X$ can be subdivided using one of the types of k-ary trees: quadtrees or octrees. ADF can be used for an efficient interactive real-time modelling, e.g. sculpting, of the heterogeneous objects as the tree data structure provides fast access to object’s geometry and its specified attributes. ADF is also suitable for solving surface restoration problems [DCL*08, TF18]. It supports the same operations as SDF.

ADF are especially suitable for dynamic simulations [KDB16], for example, morphing between shapes, as a hierarchical data structure can efficiently be rebuilt at each animation frame [FPRJ00].

The advantages and drawbacks of ADF can be seen in table 1, third column.

3.5. Interior distance function

Interior distance function (IDF) is not a well-established notion yet as in literature there is neither a general approach for generating DFs of this rather broad nature nor one unique name for them. In this work we suggest to use this notion for a representation with a defined function obtained as follows: the distance function is computed on the boundary of the object as a shortest path between boundary points and then the generated distances are smoothly interpolated in the object’s interior. Let us introduce the definition of IDF that relies on the definitions specified in subsection 3.1:

**Definition 3.10** Let the geometric shape $G \subseteq X$ of the object $O_{IDF}$ be defined in a metric space $(X, d)$. Let points $p_b$ belong to $\partial G$, and let points $p_{in}$ belong to $G_{in}$. Let a distance function $d(p_i, p_b) \equiv F_{DF}(p_i, p_b) = ||p_i - p_b||_G$ between any boundary points $p_b$ and $p_{in}$ on a curved domain $\partial G$ be recovered. Thereafter, by constructing an interpolation function $F_I(F_{DF}(p_b, p_{in}): p_{in})$ that is at least $C^1$ continuous, boundary distances are extended to interior of the object $O_{IDF}$. Therefore, the IDF function can be defined as:

$$F_{IDF}(p_{in}) = F_I(F_{DF}(p_b, p_{in}), p_{in}) \quad (10)$$

where $0 \leq i, j < N$, $N$ is the number of boundary points, $0 \leq k < M$, $M$ is the number of interior points. The IDF representation is defined in the form of an inequation:

$$O_{IDF} := F_{IDF}(p) \geq 0 \quad (11)$$

Note, that the boundary of the object is treated as the source for generating interior distances.

Technically, IDF is usually obtained by solving a partial differential equation (PDE) or applying some numerical method, e.g., graph approaches [LRL11] or Markov chains [CL06]. Among PDE-based methods the following methods can be considered as representative: geodesic distances obtained as the solution of heat equation [CWW13], diffusion maps combined with smooth barycentric interpolation of the distances in interior of the object [RLF09], the
optimal mass transport [SRGB14] and some others. IDF is usually used in the tasks related to shape analysis [RLF09], geometry restoration [PS12], morphing and less commonly for an attribute definition in the interior of the object [FSP15].

In Fig. 4 we show how the approach described in [RLF09] can be applied to the polygonal mesh of the 'Stanford Bunny'. The distances are computed on the boundary, as it can be seen in the part (a), using the diffusion maps, and then propagated in interior of the object, as it can be seen in the part (b), using the barycentric interpolation. A simple procedure of IDF isolines generation is the computing of distances from the fixed 'source' point \( p \) to other points of the mesh. If we compare two pictures shown in Fig. 4 (b) and (c), we can see that the distance fields obtained in interior of the bunny are completely different. The IDF field (b) is smooth and continuous while the SDF field (c) is not smooth and has some sharp features in interior of the object.

The advantages and drawbacks of IDF can be found in table 1, last column.

### 3.6. Heterogeneous objects

In the previous subsections we have discussed how geometric shape of objects can be defined using distance-based methods. In this subsection we discuss how attributes can be considered in concert with the geometric shape of the object to represent the heterogeneous object. Let us first introduce a general definition of the heterogeneous object.

#### Definition 3.11

Let the object \( O_{H} \) be defined as a two component tuple: geometric shape \( G \subseteq X \) in the form of a multidimensional point-set geometry and attributes \( A_{i} \) corresponding to the physical properties of the object \( O_{H} \). Then such object \( O_{H} \) is a heterogeneous object defined as:

\[
O_{H} := (G,A_{1},...,A_{n}),
\]

where \( n \in \mathbb{N} \) is the number of attributes.

Attribute distributions specified in heterogeneous objects \( O_{H} \) can be uniform or non-uniform. For instance, the simple example of the uniform distribution can be a homogeneously coloured object. As to non-uniformity, it can be presented as porous structures or microstructures with non-linear varying density.

In this work we will apply the hypervolume model [PASS01] to define heterogeneous objects \( O_{H} \), using FRep or any other distance function-based representation. A hypervolume object is defined as follows:

#### Definition 3.12

Let the geometric shape \( G \) of \( O_{H} \) be defined by a real-valued function \( F_{G}(p) \), \( p \in \mathbb{R}^{d} \) that is at least \( C^{0} \) continuous and let attributes be defined by any \( F_{A_{i}}(p) \). Then heterogeneous object \( O_{H} \) is defined as:

\[
O_{H} := (G,A_{1},...,A_{n}) : (F_{G}(p),F_{A_{1}}(p),...,F_{A_{n}}(p)),
\]

where \( n \in \mathbb{N} \) is the number of attributes.

In general case, attribute functions \( F_{A_{i}}(p) \) are not necessarily continuous. However, as it was shown in [BST04], better control of the attributes on the surface and in the interior of the distance-based objects can be achieved when the attribute defining functions are parameterised by the distances. The main requirement for the distance function is to be at least \( C^{1} \) continuous. This requirement prevents the appearing of stress concentrations, creases and other singularities in modelled attribute distributions.

There are several interesting examples discussed in [BST04]. In particular, the distance-based smooth and differentiable attribute functions were applied to represent a parabolic distribution of the grained refractive index in Y-shaped solid of the waveguide. In this case, it is important that the distribution of the index of refraction is uniform and smooth. Another example is to use such distance-based attribute functions for modelling different types of materials, e.g., silicon carbide (SiC). It is important to note that the approach introduced in [BST04] was not applied to such attributes as textures, colours and similar attributes.
4. Hybrid function representation

In this section we introduce and systematically describe a general approach for defining heterogeneous volumetric objects using a hybrid function representation (HFRep). First, we list the requirements for HFRep, then outline its mathematical basics, and finally describe its properties with respect to four basic DF-based representations.

4.1. Problem statement

Let us give the exact problem statement. Our goal is to propose a hybrid function representation (HFRep) that is suitable for defining volumetric heterogeneous objects. We assume that the geometric shape \( G \) of the given object is defined by FRep, and its defining function is known. To devise the HFRep embracing advantages and circumventing disadvantages of FRep, SDF, ADF, IDF, it is essential to obtain a real-valued defining function in an \( n \)-dimensional Euclidean space with the following properties:

1. The HFRep function should provide sufficiently accurate distance approximation in Euclidean space \( \mathbb{R}^n \) without \( C^0 \) and \( C^1 \) discontinuities.
2. The HFRep function should be at least \( C^0 \) continuous with possibility to enforce it to be at least \( C^1 \) continuous.
3. The HFRep function should satisfy the point membership test: it should be positive in interior of the geometric shape \( G \), take exact zero values only at the object boundary \( \partial G \) and it should be negative in exterior of the geometric shape \( \mathbb{X} \setminus G \);
4. The HFRep should be a multidimensional object representation; in particular, dealing with 4D objects is of paramount importance to cover time-variant models with the fourth 'time' coordinate;
5. The HFRep representation should be suitable for the heterogeneous object modelling allowing for defining attribute functions related to the geometry;
6. The HFRep attribute functions should depend on evaluation point \( p \in G \) and be parameterised by distance values of the obtained HFRep geometry function.

The fulfilment of these conditions guarantees that the generated object will be watertight and such operations as blending and metamorphosis will not suffer from creases. Overall, the defining HFRep function that is considered in concert with attribute functions parameterised by distances will be suitable for dealing with multi-material aspects of heterogeneous objects including time-variant ones.

4.2. Definition of the hybrid function representation

First, we provide a mathematical definition for the geometric aspects of HFRep. Then we add the part related to attributes. The
geometric shape $G$ of an HFRep object $O_{HFRep}$ is defined as follows:

**Definition 4.1** Let the geometric shape $G \subseteq X$ of the object $O_{HFRep}$ be defined in a metric space $(X, d)$. Given at least $C^0$ or $C^1$ continuous FRep function $F_{HFRep}(p)$, the distance to the object boundary $\partial G$ is defined as $(F_I \circ F_{DF})(p)$, where $F_I(\cdot)$ is at least $C^1$ continuous interpolation function and $d(\cdot, \cdot) \equiv F_{DF}(\cdot, \cdot)$ is a distance-based function, in particular SDF, ADF or IDF. Then the HFRep function is defined as follows:

$$F_{HFRep}(p) = (F_{sign} \circ F_{HFRep})(p) \cdot (F_I \circ F_{DF})(p) \quad (14)$$

where $F_{sign}(\cdot)$ is an at least $C^1$ continuous function that provides a sign for the computed function $(F_I \circ d)(p)$ and satisfies the FRep point membership test, equation (5). Finally, the HFRep representation is defined as:

$$O_{HFRep} := F_{HFRep}(p) \geq 0 \quad (15)$$

The continuity of the HFRep function $F_{HFRep}(p)$ depends on the continuity of the FRep function $F_{HFRep}(p)$. In the case when we are dealing only with geometric shapes, it is sufficient to have $C^1$ continuity for the HFRep function. Otherwise, in case of heterogeneous object modelling, the HFRep function should belong to the class of functions that are at least $C^1$ continuous. We will give details on how to control the continuity of the HFRep function later in this section.

**Figure 7:** (a) ‘Star’ object and its FRep field; (b) the HFRep ‘star’ object generated on the basis of the FRep object. The boundary of the FRep object (a) was extracted and then used for computing boundary distances. The obtained distances were interpolated in interior of the HFRep ‘star’ object using barycentric interpolation and mean-value coordinates. The isolines and colour show how the field changes from the source point $p_s$ towards the object boundary;

| Inherited from FRep | Inherited from SDF | Inherited from ADF | Inherited from IDF |
|---------------------|--------------------|--------------------|-------------------|
| • The continuity of the HFRep function depends on the continuity of the FRep function. | • HFRep provides at least $C^0$ continuous distance function. | • HFRep provides at least $C^0$ continuous distance function. | • HFRep provides at least $C^0$ continuous distance function. |
| • The HFRep object is watertight. | • the HFRep function is Lipschitz continuous and Frechet differentiable everywhere. | • the HFRep object is watertight. | • the HFRep object is watertight. |
| • HFRep represents multidimensional objects, in particular 4D objects with the fourth coordinate specified as time. | • the HFRep function satisfies the solution of the eikonal equation; | • the HFRep object can be efficiently discretised and rendered. | • the HFRep object can be efficiently discretised and rendered. |

**Table 2:** Properties of the hybrid function representation that depend on the combination of FRep with one of the distance fields.

Now let us show that $F_{HFRep}(p)$ continuity is either $C^0$ or $C^1$. By applying a smoothing interpolation function $F_I(\cdot)$ that is at least $C^1$ continuous to the discrete unsigned distance field (UDF) obtained using $F_{DF}(p, \partial G) \in C^0$, we enforce the property $(F_I \circ F_{DF})(p) \in C^1$. The composition of functions $(F_{sign} \circ F_{HFRep})(p)$ is at least $C^0$ or $C^1$ continuous, depending on the continuity of $F_{HFRep}(p)$. The theorem about the continuity of the composition of two continuous functions was proven in [Bomb07]. Therefore the continuity of the HFRep function is defined as: $C_{HFRep} = \min(C_{F_{sign} \circ F_{HFRep}}, C_{F_I \circ F_{DF}})$, where $m = 0$ or $m = 1$, $k = 1$, i.e. the minimum class of continuity between two function compositions.

Now on the basis of **definition 3.12**, we can formulate the definition of the heterogeneous HFRep object $O_{H_{HFRep}}$ as follows:

**Definition 4.2** Let the geometric shape $G$ of $O_{H_{HFRep}}$ be defined by at least $C^1$ continuous $F_G(p) = F_{HFRep}(p)$ distance-based function. Let the attribute $A_i$ be defined as a real-valued function $F_{A_i}(F_{HFRep}(p), p)$. Then the HFRep heterogeneous object $O_{H_{HFRep}}$ is defined as:

$$O_{H_{HFRep}} := \begin{cases} F_G(p) := F_{HFRep} \geq 0 & i = 0, \ldots, n \in \mathbb{N} \end{cases} \quad (16)$$

where $n$ is a number of attributes.

The properties of the introduced hybrid function representation are outlined in table 2. It shows be-pair combinations of FRep with one of the distance fields, namely SDF, ADF or IDF.

Fig. 5 shows a metamorphosis between two oscillating 4D geometric shapes (‘heart’, initially the FRep object then converted to HFRep, ‘cube’, initially the BRep object then converted to SDF) using the space-time blending (STB) method [TAPA20]. The result is a non-distance functionally defined watertight object that is continuous and smooth.

In Fig. 6 (d), we demonstrate the restored distance field computed on the hierarchical grid obtained for the initial FRep object defined as a ‘treble clef’, Fig. 6 (a). There is neither $C^2$ nor $C^3$ discontinuities in the field as it can be seen in Fig. 6, (c) or (d). All the isolines are smooth and continuous.

In Fig. 7, (b) we show a simple example of interior distances computed for the FRep ‘star’ object, Fig. 7 (a), that was constructed using seven set-theoretic operations. First, the boundary of the FRep object was extracted for computing boundary distances.
Then, the interior of the obtained convex contour was triangulated. Finally, the boundary distances were propagated in interior of the shape as it is described in [RLF09]. The black isolines show that the obtained field is at least $C^1$ continuous as they are smoothly changing in the object interior.

5. Basic algorithm for generating HFRep

Let us outline in a step-by-step manner the algorithmic solution on generating the HFRep functions. The basic algorithm covers all paired combinations of FRep with DF representations, namely SDF, ADF and IDF, and allows to generate both a geometric shape and attributes. Some steps of the basic algorithm will be slightly different depending on the particular type of the DF paired with FRep. All examples in this and the following sections were computed on a laptop with a 2.6 GHz Intel Skylake 6700 processor and 16 Gb of RAM.

Let us start from the algorithm for generating a geometric shape of the object $OH_{HFRep}$. Fig. 8 demonstrates the generated function field for each step of the basic algorithm.

5.1. Algorithm for HFRep geometry generation

1. According to the definition 4.1, we start the construction of an HFRep object $OH_{HFRep}$ from defining the FRep function $F_{HFRep}(p)$ for its geometric shape $G$. The FRep function $F_{HFRep}(p)$ can be defined analytically, with function evaluating algorithm or using a point cloud for which it is possible to obtain a real-valued at least $C^0$ continuous $F_{HFRep}(p)$. It could also be a complex FRep object that is obtained in the form of a constructive tree.

At this step we can also enforce HFRep function $F_{HFRep}(p)$ to be at least $C^1$ continuous as its continuity depends on the continuity of $F_{HFRep}(p)$. We have to examine the obtained $F_{HFRep}(p)$ for continuity and differentiability. The most practically used FRep set-theoretic operations in the form of the following R-function system are [PASS95]:

$$f_o(f_1(p), f_2(p)) = f_1 + f_2 - \sqrt{f_1^2 + f_2^2}$$

(17)

These functions have $C^1$ discontinuity in points where both arguments are equal to zero. Accordingly, the resulting function will only be $C^0$ continuous. If we need to obtain at least $C^1$ continuous resulting function, we can apply another R-function system that is at least $C^{n-1}$ continuous [Rva82]:

$$f_o(f_1(p), f_2(p)) = \begin{cases} f_1, & \forall f_1 \leq 0, f_2 \geq 0; \\ f_2, & \forall f_2 \leq 0, f_1 \geq 0; \\ (-1)^{n+1}(f_1 + f_2)^{1/n}, & \forall f_1 < 0, f_2 < 0; \end{cases}$$

(18)

where $f_1(p)$ and $f_2(p)$ are FRep functions.

2. The values of the function $F_{HFRep}(p)$ are used as an input for computing distance functions $F_{DF}(p, \partial G)$ that should satisfy one of the definitions 3.7, 3.9 or 3.10. At this step we obtain an unsigned distance function that is defined as:

$$F_{DF}(p) = d(p, \partial G), \quad \forall p \in X$$

(19)

Fig. 8, (b) shows the unsigned distance field that was obtained on the basis of a typical SDF generation algorithm [LL92]. It took 0.018 seconds to generate the UDF field.
If the distances are computed using ADF, first, we need to subdivide the space using a hierarchical data structure, e.g., quadtree, Fig. 6, (b) and during its construction we also need to compute basis functions, basis vertices and extraction operators for the hierarchical splines. Then we need to compute the distances at the corner vertices of each cell. Finally, we restore distances in interior of each cell using at least $C^1$ continuous spline-based interpolation to obtain a smooth and continuous distance field, e.g., shown in Fig. 6, (c). It took 9 seconds to generate the UDF field using our unoptimised implementation.

Specifically for IDFs, the function $F_{IDF}(p)$ is defined according to equation (10). Distances are computed on the boundary of the object $O_{FRep}$ and then interpolated in its interior. In Fig. 7 (a), we can see the field of the FRep-defined ‘star’ object that was used for generating HFRep IDF-based field that is shown in Fig. 7, (b). It took 16 seconds to generate the IDF field of this ‘star’ object.

In Fig. 9, (a) we show a possible extrapolation scheme that can be used to obtain distances in exterior of the object and make an IDF-based field signed at the last step of this algorithm.

To do this, we need to use the boundary distances (Fig. 9, (a), dark blue circles) and an appropriate at least $C^1$ continuous extrapolation operation, that will be used for obtaining distances outside the object (9, Fig.7(a), red circles).

3. The distance field obtained at the previous step is unsigned and discrete as it was computed on the finite point subset $X \subset \mathbb{R}^n$. To enforce the continuity and smoothness of the computed field, we need to apply some at least $C^1$ continuous interpolation function $(FI \circ FDF)(p)$ to the generated unsigned field, e.g. spline-based:

$$F_{smDF}(p) = (FI \circ FDF)(p) \quad (20)$$

We also need to apply a smoothing operation to an IDF field if at the previous step an extrapolation operation was applied.

Otherwise, IDFs are smooth as smoothness is their inherent property. An important requirement for the interpolation function $FI(\cdot)$ is to avoid introducing extra zeros in the distance field generated using function $FDF(p, \partial G)$. It took 0.013 seconds to compute a smoothed UDF field for the ‘robot’ object.

4. Finally, as the distance field obtained after previous steps is unsigned, we need to restore the field sign to distinguish between exterior $X \setminus \partial G$, boundary $\partial G$ and interior $G_{int}$ of the object $O_{HFRep}$. We suggest to use some at least $C^1$ continuous step-function $F_{0}(F_{HRRep}(p))$ with the scope $[-1, 1]$, that depends on the values of the defining FRep function $F_{HRRep}(p)$ and approximates its well-defined behaviour ($-1$ in exterior of the object, 0 on the boundary of the object and +1 in interior of the object). Therefore, the resulting HFRep function $F_{HRRep}(p)$ is defined according to definition 4.1 as follows:

$$F_{HRRep}(p) = (F_{0} \circ F_{HRRep})(p) \cdot F_{smDF}(p) \quad (21)$$

The HFRep field generated by this function can be seen in Fig. 8, (c). It took 0.037 seconds to generate the HFRep field of the ‘robot’ object. After a geometric shape of the HFRep object $O_{HFRep}$ was generated, we can apply different operations to it provided that they are realised by functions which are at least $C^0$ continuous. The HFRep object is also compatible with other distance-based objects. However, to preserve the distance properties for the object obtained after applying multiple operations, we might need to apply the steps of this algorithm again to this object.

There is a limited number of operations that preserve the distance property for the geometric shape $G$ of the object obtained after their application. These operations are rigid (Euclidean) transformations: rotations, translations, reflections or their combination. Another distance preserving operations [PT92] are affine translations, offsetting, linear surface interpolation, surface blurring and compression, set-theoretic operations in the form of $\min(f_1(p), f_2(p))$ or $\max(f_1(p), f_2(p))$ [Rva82]. In cases of other operations [RMD11] (e.g., scaling, blending, space-time blending, twisting, tapering and sweeping, set-theoretic operations in the form of R-functions [PASS95]) after their application, we have to apply the basic algorithm to the obtained object to restore the distance property.

To make the HFRep representation continuous on the whole domain of the Euclidean space $\mathbb{R}^n$, we suggest to apply some at least $C^1$ continuous extrapolation operation to the generated field of the object. To explain this idea in more details let us consider the following example shown in Fig. 9, (b). In this figure we have two blue objects defined on their own pink grids and spaced from each other, so their defining grids are not overlapping. If we want to work with them, e.g. by applying some operation, we need somehow to define the distances in the points of interest of the green grid. One can extrapolate and average the distances between two pink grids and avoid full reinitialisation of the distances for both objects.

5.2. Algorithm for HFRep attribute definition

To set up the attributes in interior of the HFRep object $O_{HFRep}$ we assume that we have obtained a $C^1$ continuous distance function for a geometric shape. Now we can deal with the attributes that are parameterised by the distances as it was required by definition 4.2. Object attributes could be of different nature and there is no single
1. Depending on the nature of the attributes and how they are distributed in the interior of the object $O_{HFRep}$, there are two possible types of object partitioning: single and multiple partitions. At this step we need to subdivide an object $O_{HFRep}$ according to the chosen partitioning scheme.

2. Then we specify and evaluate an attribute function $F_{HFRep}(p, p)$ for each partition to set up the attributes at the points $p \in G$. These functions depend on the evaluation coordinate and are parameterised by the computed distance using $F_{HFRep}(p)$ values.

3. In case when we have a multiple partitioned object with several specified attributes, we can obtain a single attribute function for all subsets $A_{i}$ by applying some interpolation, e.g. transfinite interpolation [RSST01] or space-time transfinite interpolation [SFA*15].

The more detailed discussion how to deal with attributes will be provided in section 7.

6. Algorithmic solutions for HFRep geometric shape generation

In this section we provide a detailed description of several particular steps of the basic algorithm outlined in the previous section.

We consider a variety of combinations of FRep and SDF, ADF or IDF representations and propose a number of original solutions for solving problematic issues. The first step of the algorithm has been already discussed. In the next sections we discuss steps 2 - 4 (see Figs. 8, (b) and (c)).

6.1. Step 2: Generation of the unsigned distance field

In this subsection we describe the solutions for generating UDfs. We show how some existing techniques can be used in this context and also introduce a novel method for the ADF generation.

**SDF generation.** To compute an approximate UDF, the most widely used class of methods is the distance transform (DT) [JBS06]. DTs are efficiently generated on regular grids. In this work we suggest using the vector DT in which the vector components are propagated across the uniform grid. It provides a sufficiently accurate distance approximation. We follow the typical vector DT algorithm described in [LL92] for 2D case and [Dan80] for 3D case.

A definitive way to obtain an accurate DF for the object is to numerically solve the eikonal equation or the level-set PDEs [GM19]. The numerical solution of PDE is quite time-consuming unless it is a multi-threading implementation of the method. The accuracy of the field is also highly dependent on the method. One of the robust methods for solving the eikonal equation is the fast iterative method (FIM) [JW08]. It numerically solves a nonlinear Hamilton-Jacobi PDE defined on a Cartesian grid with a scalar speed function:

$$H(p, \nabla \phi) = |\nabla \phi(p)|^2 - \frac{1}{f^2(p)} = 0, \quad \forall p \in X \subset \mathbb{R}^n \quad (22)$$

where $X$ is a domain in $\mathbb{R}^n$, $\Gamma$ is the boundary condition, $\phi(p)$ is a travel time of the distance from the source to the grid point $p$, $f(p)$ is a positive speed function and $H(p, \nabla \phi)$ is the Hamiltonian. The computed numerical solution is an unsigned distance on a uniform grid.

**ADF generation.** To generate UDF on the hierarchical grid we briefly outline an original adaptation of the FIM method for solving the eikonal equation that also utilises PHT-splines [WYJ*11] capability of the accurate geometry restoration. Our algorithm partly relies on the algorithm introduced in [JW08] to inherit its advantages such as independent computation of each node and a simple data structure (an active list $L$ or a doubly linked list) for handling node updates. A detailed description of the hierarchical FIM (HFIM) algorithm will be presented elsewhere.

The algorithm consists of two parts: (1) initialisation of the grid and (2) iterative updates of the numerical solution of the eikonal equation. First, we subdivide the space using quadtree/octree according to the values of the FRep field. We need to subdivide the exterior and interior of the FRep object with a small tree depth and its boundary with the maximum tree depth. While executing the hierarchical subdivision of the space, we also need to compute the basis functions for the PHT-splines [WYJ*11] and reconstruct the PHT-spline surface that will be used for restoring distances in interior of each cell node.

1. We need to define the FRep object $O_{FRep}$ using at least $C^0$ continuous FRep function $F_{FRep}(p)$. In the Fig. 6, (a) we show the defined FRep object that we will further subdivide in the next step.

2. After we have defined $O_{FRep}$ using $F_{FRep}(p)$, we need to subdivide space using some tree-structucture according to the
following rule: the exterior \( \mathcal{X} \backslash G \) of \( \Omega_{\text{FRep}} \) should be subdivided with lowest tree depth, the boundary \( \partial G \) of \( \Omega_{\text{FRep}} \) should be subdivided with highest tree depth and the interior \( G_{\infty} \) of \( \Omega_{\text{FRep}} \) should be subdivided according to the user needs. In the Fig. 6, (b) we show the subdivided FRep object.

3. During the subdivision process of the Euclidean space, we have to compute PHT-spline basis functions at each cell of the constructing T-mesh including Bezier extraction operators and new basis indices. The obtained basis functions, operators and tensor products will be further used for the restoration of the distance field.

4. After we finished the subdivision of the Euclidean space according to steps two and three, we need to traverse the tree and set to zero those vertices of the cells that store the FRep values approximately equal to zero. The rest of the vertices are set to a relatively huge value. Thereafter, vertices that are equal to zero and the corresponding nodes are stored in the active list.

5. The iterative computation of the solution of the eikonal equation on the hierarchical grid follows the logic of the FIM algorithm [JW08], but all steps are executed taking into account a hierarchical nature of the grid. The eikonal equation is iteratively solved using the first order upwind Godunov discretisation scheme that is modified for computations on the grid with irregular steps. The computed solution is stored and updated in all nodes that share the same vertices. The iterative computation is finished when the active list is empty.

6. After obtaining the solution of the eikonal equation at each corner vertex of each cell of the hierarchical grid, we can restore the distance field using the already constructed PHT-spline surface.

As we have stated in subsection 3.4, the ADF field has \( C^0 \) discontinuities that arise after the hierarchical subdivision where cells of different size appear. In Fig. 10, (a), the discontinuities in the white isolines are located in the red circles. \( C^1 \) discontinuities are introduced by the bilinear/trilinear interpolation that is used for the field restoration in interior of each cell (see Fig. 10, a). As it can be seen in Fig. 10, (b) the field generated by our method with PHT-spline restoration of the field successfully solves these drawbacks. All the isolines are continuous and smooth.

The accuracy of the restored field can be enhanced if an error based refinement operation for the hierarchical is implemented. This will allow to obtain a more accurate PHT-spline approximation of the computed field as well as a more accurate solution of the eikonal equation on the refined grid. Our implementation of the proposed HFIM method is also not well-optimised yet, and the algorithm should be further developed for parallel computations.

**IDF generation.** IDs are usually computed using the solution of some PDE equations or, alternatively, some graph-based approach. We suggest using the approach described in [RLF09].

The generation of IDs is based on propagation of the distances computed on the boundary of the mesh in its interior. We will use Fig. 11 with the generated IDF field to explain how this method works. We start from triangulating an input geometric shape \( G \) of the FRep object \( \Omega_{\text{FRep}} \) to generate the boundary surface \( \partial G \) for further computations. The method, described in [RLF09] was applied to tetrahedralised meshes and consists of two parts.

First, we embed surface vertices \( p_i \) in some m-dimensional \( \mathbb{R}^m \) space using a map \( p_i \rightarrow p_i^\ast \in \mathbb{R}^m \). This map was suggested to compute using diffusion maps introduced in [CL06]. It can be obtained by computing an eigendecomposition \( \{\lambda_k, \phi_k\}_{k=1}^n \) of a discrete Laplace-Beltrami operator of the mesh. In Fig. 11(a) we show the diffusion map obtained for the ‘heart’ object. The diffusion distances are computed as a Euclidean distance using obtained eigenvalues and eigenvectors [dGGV08].

After the diffusion distances were computed on the surface of the mesh as it can be seen in Fig. 11 (b), they are extended to the in-

**Figure 11:** HFRep based on hybridisation of FRep and IDF generated for FRep ‘heart’ object using the method from [RLF09]. a) the diffusion map computed on the surface of the object that is used for restoring distances at the shape boundary; b) the distances obtained at the boundary of the object shape that are shown as black isolines; c) is the tetrahedral slice of the mesh with isolines corresponding to the interior distances. The yellow point \( p_s \) corresponds to the ‘source’ point defined in the object interior.
terior of the mesh using barycentric interpolation. If point \( q \in G_{\text{int}} \), then the barycentric representation of it is \( q \mapsto q' = \sum \omega_i(q) v_i \), where \( \omega_i(\cdot) \) are barycentric coordinates (e.g. mean-value coordinates in 2D [HF06] or in 3D [JSW05]). Finally, the distance in interior of the mesh can be obtained using computed diffusion distances \( F_{\text{DF}}(p, p_j) \) and barycentric interpolation.

In Fig. 11 (c) we show a slice of the ‘heart’ object. The IDF field was computed between fixed ‘source’ point and other points of the mesh. One can see that the interior field is continuous, smoothly changing and following the boundary of the object. In our implementation it took 0.7 seconds to compute the tetrahedralised FRep ‘heart’ object. To compute the diffusion map it took 0.4 seconds, and the computation of the IDF field took 115 seconds.

At this step we can apply an extrapolation operation (e.g., using a wavenumber based extrapolation [TK00]) to the obtained IDF field to propagate the distances to exterior of the object using the already computed boundary distances. This operation will allow us to make the IDF field signed at the last step of the basic algorithm.

One of the limitations of the chosen method for the IDF generation is that it was specifically developed for tetrahedral meshes [RLF09]. As a consequence, to make it compatible with other already discussed hybrid schemes, the generated tetrahedral mesh should be converted to a voxel representation and the computed IDF field interpolated in voxel centres. This method is also relatively slow to compute.

6.2. Step 3: Smoothing an obtained UDF

The resulting distance function \( F_{\text{DF}}(p) \) for SDF, ADF or IDF is unsigned and satisfies the equation (19). Having obtained UDF, we need to smooth the generated discrete field. To enforce an at least \( C^1 \) continuity and essential smoothness for the obtained field, we need to use some at least \( C^1 \) continuous interpolation function, e.g. B-splines or bicubic/tricubic splines [Kno00].

6.3. Step 4: Distinguishing between interior, boundary and exterior of the object

At the previous step we had obtained a smooth and continuous unsigned distance function \( F_{\text{USDF}}(p) \) defined by equation (20) that we used to compute UDF. Now, at the fourth step of the basic algorithm, we need to define the sign of UDF. To restore the sign we suggest to use a smooth step-function that depends on the values of the FRep function \( F_{\text{FRep}}(p) \), defined at the first step of the basic algorithm. The step-function \( F_\sigma(F_{\text{FRep}}(p)) \) should satisfy the following requirements:

1. it is approximately equal \(-1\) when it corresponds to the exterior of the FRep object, \( F_{\text{FRep}}(p) < 0 \);
2. it should be approximately equal to 0 on the boundary of the FRep object, \( F_{\text{FRep}}(p) = 0 \);
3. it should be approximately equal to 1 inside the FRep object, \( F_{\text{FRep}}(p) > 0 \);
4. it should be at least \( C^1 \) continuous everywhere in a Euclidean space \( \mathbb{R}^d \);
5. it should barely modify the values of UDF.

We have identified two classes of functions which satisfy these requirements. These are sigmoid functions and spline functions, particularly cubic splines with Hermite end conditions, to estimate the slopes [Kno00]. In this work we use the hyperbolic tangent sigmoid function [VMR88] (see Fig. 12). By controlling slope parameter \( s_j \), it is possible to get nearly step-function behaviour around zero:

\[
F_{\text{SIG}}(x) = \frac{r}{1 + \exp(-2x/s_j)} - \frac{r}{2}, \quad \forall x \in \mathbb{R} \tag{23}
\]

where \( r \) controls the range of the \( F_\sigma(x) \) along y-axes. We need to set
Figure 14: The illustration of the HFRep heterogeneous object based on the FRep and SDF representations with incorporated microstructure. a) the rendered HFRep ‘sphere’ object using the sphere-tracing method (left) and its isolines (right); b) the rendered HFRep ‘heart’ object using sphere-tracing method (left) and its isolines (right);

parameter \( r = 2 \) to make the function (23) be defined in the interval \((-1, 1)\) along the \( y \)-axes.

The continuity of the HFRep function can be visualised as it is shown in Fig. 13. In Fig. 13 (a), we show half of the ‘star’ object that was generated with \( s_l = 10^{-5} \) to follow the step-function shape as close as possible. In Fig. 13 (b), we show half of the ‘star’ object that was generated with \( s_l = 0.1 \) to smooth the isolines shape. We can see that the \( C^1 \) continuity of the generated distance field is preserved and the obtained geometric shape of the object is watertight.

7. Dealing with attributes in HFRep framework

In this section we show how we practically work with HFRep heterogeneous objects in terms of their attributes. In section 5 we have outlined the basic algorithm for generating HFRep attribute functions. However, there is no universal approach for dealing with HFRep object attributes because of their widely various nature. In this section we show how the proposed framework works for some representative attributes, namely, microstructures, colour and material attributes. We show the microstructures (Fig. 14), two heterogeneous objects with defined procedural marble material parameterised by the SDF distance (Fig. 16), an HFRep 2D ‘H’ object with three different parameterisations of the procedural wood function by the IDF distance (Fig. 17), a heterogeneous model of the COVID-19 virus cell (Fig. 18) and two models of metamorphosis dealing with a dynamic (time-variant) smooth transition from one HFRep object to another (Figs. 15 and 19).

7.1. Microstructures

In Fig. 14 we demonstrate how microstructures in interior of the \( O_{HFRep} \) object are implemented. The microstructures were defined as incorporated infinite slabs in interior of the ‘sphere’ and ‘heart’ objects using set-theoretic operations (18). The infinite slabs were defined according to [PFV'11] as follows:

\[
S(p) = \sin(\nu \odot p + \phi) + l; \quad (24)
\]

where \( S(p) \geq 0 \) is a vector function, with components defined as a set of slabs orthogonal to either \( X \) or \( Y \) or \( Z \)-axes, \( \nu \) is a frequency vector, with components defined as the distance between parallel slabs along one of the axes, \( p \) is a point \( p \in X \), \( \phi \) is a phase vector, with components defined as the position of slabs on one of the axes with respect to the origin and \( l, -1 < l < 1 \) is a threshold vector that together with frequency parameters controls the thickness of each slab. Then the basic algorithm was applied to the obtained function to compute the HFRep objects with microstructures. In our implementation it took 14 seconds to generate the FRep field for the object with microstructures shown in Fig. 14(b). The computation of the distance transform took 16 seconds and the smoothing of the computed unsigned distance took 3 seconds.

Implementation was done using C++ and OpenGL. The HFRep geometric shape was computed as a scalar field which was stored in a 3D texture. Then it was passed into a fragment shader for assigning a single colour attribute and rendered using the sphere-tracing method.

7.2. Procedural textures

We can specify attributes as simple procedural textures. Fig. 15 shows two heterogeneous HFRep objects \( O_{HFRep} \) with coloured wooden textures that were obtained using a procedural function.
f_{\text{wood}}(p)$. This function is constructed using hash table $htab(p)$ allowing for random sampling of the position values $p$ multiplied by the frequency $v$. The procedural function for the wood can be defined as follows:

$$g(p) = htab(p \cdot v) \cdot c;$$

$$f_{\text{wood}}(p) = g(p) - \text{int}(g(p));$$

where $c > 1$ is a constant, $g(p)$ is a noise function, $\text{int}(g(p))$ is an integer part of the function $g(p)$ output value. To parameterise $f_{\text{wood}}(p)$ by the distance, we assign the distance values to the frequency parameter $v$.

Then a simple segmentation of the geometric shape of the objects was done (see Fig. 15,(1)). We split the shape into four regions and assign colours using the obtained HFRep distance function $F_{\text{HFRep}}(p)$ and procedural function $f_{\text{wood}}(p)$ that defines the texture of the wood. The generated objects were used as inputs for 2D heterogeneous metamorphosis on the basis of the space-time blending (STB) method to handle geometry transformation and the space-time transfinite interpolation (STTI) to handle colour transformation [TAF*20]. This example was implemented using C++ and OpenCV.

In Fig. 17 we show three textured ‘H’ HFRep objects. The textures for these objects were generated using three different IDF based parameterisations of the procedural function for the wood. In Fig. 16 we show two 3D HFRep objects ‘heart’, (a), and ‘torus’, (b), with procedurally defined marble material with different parameterisation by the HFRep distance.

### 7.3. Voxel based attributes

In this subsection we discuss how HFRep objects with voxel-based attributes can be defined. Two following examples were implemented in SideFX Houdini using the OpenVDB library.

In Fig. 18 we show a 3D model of the COVID-19 cell that was obtained using 207 set-theoretic operations. In Fig. 18, (b), we can see the interior structure of the COVID-19 cell [MG20]. The central part representing the RNA and N-protein was defined using SDF that was further combined with the HFRep spherical shell of the cell. The M-protein was also defined as a combination of the SDF arc and two HFRep spheres. The rest of the elements were defined using HFRep. Each element is mono-coloured and colours are assigned per-voxel.

Fig. 19 demonstrates a 3D metamorphosis between two heterogeneous objects that are a combination of the HFRep and SDF defined objects [TAPA20]. This example served as one of tests for the ‘4D Cubism’ project [CMPA18]. The input and target objects are two SDF cubes spaced from each other. These input shapes were segmented using an octree data structure to make it possible a local faceting and distortions. Two colours were assigned to them per-voxel. Then different HFRep and SDF ‘cubist’ features were assigned to selected areas of the two basic SDF cubes, which were coloured per-voxel as randomly chosen colours from the specified range. Then we apply the same combination of methods as we have discussed before for the 2D metamorphosis. The generated colour and geometric shape transformations happen simultaneously and interconnectedly. In Fig. 19, (4) (sliced), we show how the interior of the object is transformed during the 3D metamorphosis process.

### 8. Conclusions

In this work we have introduced a theoretical and practical framework for modelling volumetric heterogeneous objects on the basis of a novel unifying functionally-based hybrid representation called HFRep. First, we have identified four conventional representational schemes related to scalar fields of different kinds, namely FRep, SDF, ADF and IDF, suggested a formalisation of those approaches and described their advantages and drawbacks. This has allowed us to formulate the requirements for a unifying hybrid representation. The defining functions in the core of HFRep are continuous and have a distance property everywhere in a Euclidean space.
They also have several other useful properties. We have defined the mathematical basics of the representation and developed an algorithmic procedure allowing to generate HFRep objects in terms of their geometry and attributes.

To make our approach practical, we have provided a detailed description of the main steps of the algorithm and identified some problematic issues associated with them. This has required employing a number of techniques of different nature, separately and in combination. Some of these techniques were already described in literature, others had to be improved or developed. In particular, a new FIM algorithm for solving the eikonal equation on hierarchical grids has been developed.

While the boundary representation will remain the main and prevailing instrument for geometric modelling, we believe that the functionally-based representations generalising a well-established implicit modelling approach, are becoming more important in the context of some modern applications. Hopefully, HFRep that embraces advantages and circumvents drawbacks of PRep, SDF, ADF, IDF will find its applications.

Future work will be concerned with developing operations over HFRep objects in the context of different applications, especially related to physical simulation, additive manufacturing and visual effects. In technical terms, we aim to develop an HFRep field extrapolation procedure for IDFs and beyond the computing domain. One of the interesting directions will be the introduction of the attribute definition in the interior of the volumetric object using the FIM method for 3D hierarchical grids.

References

[ABGA04] Allegre R., Barbier A., Galin E., Akkouche S.: A hybrid shape representation for free-form modelling. In Proceedings of the Shape Modeling International 2004 (2004), SMI ’04, IEEE Computer Society, pp. 7–18.

[BDN16] Barclay J., Dhokia V., Nasreih A.: Additive manufacturing simulation using signed distance fields. In Sustainable Design and Manufacturing 2016 (2016), Setchi R., Howlett R. J., Liu Y., Theobald P., (Eds.), Springer International Publishing, pp. 435–444.

[BKW18] Bader C., Kohl D., Weaver J. C., Sharma S., Hosny A., Costa J., Oxman N.: Making data matter: Voxel printing for the digital fabrication of data across scales and domains. Science Advances 4, 5 (2018), 2, 3.

[SMC04] Schmitt B.: Cellular-functional modeling of heterogeneous objects. In Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications (2002), SMA ’02, ACM, pp. 192–203.

Figure 19: Metamorphosis between multiple coloured objects using the STB and STTI techniques. Colours of the initial objects are procedurally defined per voxel. Supplementary videos: figure19.m4v and figure19_slice.m4v.
A. Tereshin & A. Pasko & O. Fryazinov & V. Adzhiev / Hybrid Function Representation for Heterogeneous Objects

[1] Delfour M., Zolesio J.-P.: Shapes and Geometries: Metrics, 1105
Analysis, Differential Calculus, and Optimization, Society for Industrial and Applied Mathematics, 2010. 4
[2] Fprisken S. F., Perry R. N., Rockwood A. P., Jones T. R.: Adaptively sampled distance fields: A general representation of 1106
shape for computer graphics. In Proceedings of the 27th Annual Con-
ference on Computer Graphics and Interactive Techniques (2000), SIG-
[3] Graf H. O., ACM Press/Addison-Wesley Publishing Co., pp. 249–254.
[4] [FS15] Fryazinov O., Sanchez M., Pasko A.: Shape conforming
volumetric interpolation with interior distances. Comput. Graph. 46, C
(2015), 149–155. 3, 7
[5] [GFO18] Gibou F., Fedkiw R., Osher S.: A review of level-set meth-
ods and some recent applications. Journal of Computational Physics 355.
[6] [HL09] Hiller J., Lipson H.: Analysis and design of digital materials
for physical 3d voxel printing. Rapid Prototyping 15 (2009), 137–149. 3
[7] [IBS06] Jones M. W., Baerentzen J. A., Sramek M.: 3d distance
fields: a survey of techniques and applications. IEEE Transactions on
Visualization and Computer Graphics 12, 4 (2006), 581–599. 1, 2, 12
[8] [JSW05] Ju T., Schaefer S., Warren J.: Mean value coordinates for
closed triangular meshes. ACM Trans. Graph. 24, 3 (2005), 561–566. 14
[9] [JW08] Jeong W., Whittaker R.: A fast iterative method for eikonal
equations. SIAM Journal on Scientific Computing 30, 5 (2008), 2512–2536.
[10] [Kar99] Karczmarczuk J.: Geometric modelling in functional style.
In Proc. of the III Latin-American Workshop on Functional Program-
ing. CLAPF’99 (1999), pp. 8–9. 5
[11] [KBH99] Kumar V., Burns D., Dutta D., Hoffmann C.: A
framework for object modeling. Computer-Aided Design 31, 9 (1999).
[12] [Kap16] Koschier D., Deul C., Bender J.: Hierarchical hp-
adaptive signed distance fields. In Proceedings of the ACM SIG-
GRAPH/Eurographics Symposium on Computer Animation (2016), SCA
[13] [KKL15] Kim J.-H., Kim C.-H., Lee J.: A hybrid sdf for the detailed
representation of liquid–solid mixed surfaces. Computer Animation and
[14] [KLD06] Kues M., Löffler S., Schreck B.: Procedural function-based
modelling of volumetric microstructures. Graphical Models 73, 5 (2011), 165 – 181. 15
[15] [LJ08] Lipton Y., Raman S., Liu M.: Computing the inner dis-
tance of volumetric models for articulated shape description with a vis-
sibility graph. IEEE Transactions on Pattern Analysis and Machine Intel-
ligence 33, 12 (2011), 2538–2544. 6
[16] [LS04] Luehr Kim, Sukhatme G. S., Desbrun M.: A haptic-
rendering technique based on hybrid surface representation. IEEE Com-
puter Graphics and Applications 24, 2 (2004), 66–75. 3
[17] [LH21] Luehr Kim, Sukhatme G. S., Desbrun M.: A haptic-
rendering technique based on hybrid surface representation. IEEE Com-
puter Graphics and Applications 24, 2 (2004), 66–75. 3
[18] [LF10] Lipman Y., Rastegari R. M., Funkhouser T. A.: Bihar-
morphic distance. ACM Trans. Graph. 29, 7 (2010), 27:1–27:11. 3
[19] [LR11] Liu Y.-S., Ramani K., Liu M.: Computing the inner dis-
tance of volumetric models for articulated shape description with a vis-
sibility graph. IEEE Transactions on Pattern Analysis and Machine Intel-
ligence 33, 12 (2011), 2538–2544. 6
[20] [LSAS04] Luehr Kim, Sukhatme G. S., Desbrun M.: A haptic-
rendering technique based on hybrid surface representation. IEEE Com-
puter Graphics and Applications 24, 2 (2004), 66–75. 3
[21] [MG20] Luehr Kim, Sukhatme G. S., Desbrun M.: A haptic-
rendering technique based on hybrid surface representation. IEEE Com-
puter Graphics and Applications 24, 2 (2004), 66–75. 3
[22] [NK*18] Ngo T. D., Kashiwano G., Miyazato G., Nguyen K. T., Hui D.: Additive manufacturing (3d printing): A review of materials,
methods, applications and challenges. Composites Part B: Engineering
143 (2018), 172 – 196. 3
[23] [NM*06] Nealen A., Müller M., Keiser R., Boxerman E., Carlsson M.: Physically based deformable models in computer graphs-
ics. Computer Graphics Forum 25, 4 (2006), 809–836. 1
[24] [PASS95] Pasko A., Adzhiev V., Souren A., Savchenko V.: Func-
tion representation in geometric modeling: concepts, implementation and applications. The Visual Computer 11, 9 (1995), 429–446. 4, 5, 10, 11
[25] [PASS01] Pasko A., Adzhiev V., Schmitt B., Schlack C.: Con-
structive hypervolume modeling. Graphical Models 63, 6 (2001), 413 –
442. 3, 7
[26] [PFV*17] Pasko A., Fryazinov O., Vilbrandt T., Fayolle P.-A., Adzhiev V.: Procedural function-based modeling of volumetric microstructures. Graphical Models 73, 5 (2011), 165 – 181. 15
[27] [PS12] Patane G., Spagnuolo M.: Smi 2012: Full local approxima-
tion of scalar functions on 3d shapes and volumetric data. Comput. Graph.
36, 5 (2012), 387–397. 1
[28] [PTF21] Payne B. A., Toga A. W.: Distance field manipulation of sur-
face models. IEEE Transactions on Visualization and Computer Graphics
12, 1 (1992), 65–71. 5, 11
[29] [Reg80] Requicha A. G.: Representations for rigid solids: Theory,
methods, and systems. ACM Comput. Surv. 12, 4 (1980), 437–464. 2
[30] [RLF90] Rastegari R. M., Lipman Y., Funkhouser T.: Interior
distance using barycentric coordinates. In Proceedings of the Sym-
posium on Geometry Processing (2009), SGP’09, Eurographics Associa-
tion, pp. 1279–1288. 6, 7, 10, 13, 14
[31] [RMD11] Reiner T., Meck G., Dachsacher C.: Smi 2011: Full
paper: Interactive modeling of implicit surfaces using a direct visual-
ization approach with signed distance functions. Comput. Graph. 35, 3
(2011), 596–603. 11
[32] [RRS16] Regli W., Rossignac J., Shapiro V., Srinivasan V.: The
new frontiers in computational modeling of material structures.
Computer-Aided Design 77 (2016), 73 – 85. 2
[33] [RST01] Ravach V. L., Shiouko T. I., Shapiro V., Tsukanov I.: Trans-
finite interpolation over implicitly defined sets. Computer Aided Geometric Design 18, 3 (2001), 195 – 220. 12
[34] [Rva82] Ravach V. L.: Theory of R-functions and Some Applications.
Naukovaya Dumka, 1982. (rus). 10, 11
[35] [Sch97] Schechter E.: Chapter 25 - fréchet derivatives. In Handbook of
Analysis and Its Foundations, Schechter E., (Ed.). Academic Press.
San Diego, 1997, pp. 661 – 687. 4
[36] [SFA*15] Sanchez M., Fryazinov O., Adzhiev V., Cominos P., Pasko A.: Space-time transfinite interpolation of volumetric material
properties. IEEE Transactions on Visualization and Computer Graphics
21, 2 (2015), 278–288. 12
[37] [SG17] Sharma G. K., Gurumoorthy B.: A hybrid approach
to define and represent material distribution in heterogeneous objects.
Computer-Aided Design and Applications 14, 1 (2017), 70–82. 3
[SRGB14] Solomon J., Rustamov R., Guibas L., Butscher A.: Earth mover’s distances on discrete surfaces. ACM Trans. Graph. 33, 4 (2014), 67:1–67:12. 2, 7

[Sul15] Sullivan A.: Hybrid adaptively sampled distance fields, 2015. Patent number: 9122270, United States, US patent. 3

[TAF∗20] Tereshin A., Adzhiev V., Fryazinov O., Marrington-Reeve F., Pasko A.: Automatically controlled morphing of 2d shapes with textures. SIAM Journal on Imaging Sciences 13, 1 (2020), 78–107. 16

[TAFP19] Tereshin A., Adzhiev V., Fryazinov O., Pasko A.: Hybrid Function Representation with Distance Properties. In Eurographics 2019 - Short Papers (2019), Cignoni P., Miguel E., (Eds.), The Eurographics Association, pp. 17–20. 2

[TAPA20] Tereshin A., Anderson E., Pasko A., Adzhiev V.: Space-time blending for heterogeneous objects. In Eurographics 2020 - Short Papers (2020), Wilkie A., Banterle F., (Eds.), The Eurographics Association. 9, 16

[TF18] Tang Y., Feng J.: Multi-scale surface reconstruction based on a curvature-adaptive signed distance field. Computers Graphics 70 (2018), 28 – 38. CAD/Graphics 2017. 5, 6

[TK00] Tam C. K., Kurbatskii K. A.: A wavenumber based extrapolation and interpolation method for use in conjunction with high-order finite difference schemes. J. Comput. Phys. 157, 2 (2000), 588–617. 14

[TP11] Tsukanov I., Posireddy S. R.: Hybrid method of engineering analysis: Combining meshfree method with distance fields and collocation technique. Journal of Computing and Information Science in Engineering 11 (2011), 031001:1–031001:9. 3

[TSN10] Takayama K., Sorkine O., Nealen A., Igarashi T.: Volumetric modeling with diffusion surfaces. In ACM SIGGRAPH Asia 2010 Papers (2010), no. Article 180 in SIGGRAPH ASIA ’10, Association for Computing Machinery. 2

[VMR+88] Vogl T. P., Mangis J. K., Rigler A. K., Zink W. T., Alkon D. L.: Accelerating the convergence of the back-propagation method. Biological Cybernetics 59, 4 (1988), 257–263. 14

[WYJ+11] Wang J., Yang Z., Jin L., Deng J., Chen F.: Parallel and adaptive surface reconstruction based on implicit phi-splines. Computer Aided Geometric Design 28, 8 (2011), 463 – 474. Solid and Physical Modeling 2010. 1, 2, 3, 5, 12

[You83] Young I. T.: Image analysis and mathematical morphology, by J. Serra. Academic Press, London, 1982, xviii + 610 p. $90.00. Cytometry 4, 2 (1983), 184–185. 5

[YYW12] Yuan Z., Yu Y., Wang W.: Object-space multiphase implicit functions. ACM Trans. Graph. 31, 4 (2012). 2

[101] Đurić R., Czanner S., Inoue H.: Growth animation of human organs. The Journal of Visualization and Computer Animation 12, 5 (2001), 287–295. 5