VOIDGA:
A View-Approximation Oriented Image Database Generation Approach

Jonas Lukasczyk* TU Kaiserslautern
Eric Kinner† TU Kaiserslautern
James Ahrens‡ Los Alamos National Laboratory
Heike Leitte§ TU Kaiserslautern
Christoph Garth¶ TU Kaiserslautern

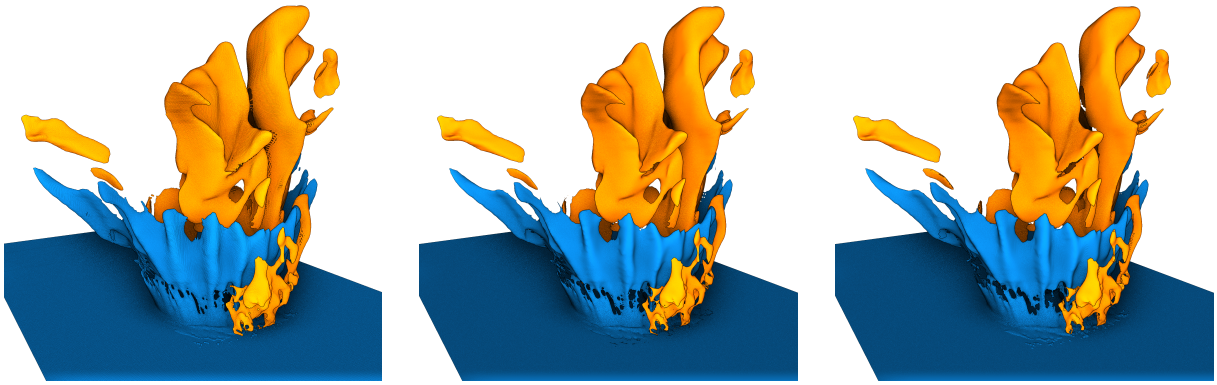

Figure 1: Comparison between the generated views (left and right) and the ground truth (middle) for the asteroid impact dataset yA31 at cycle time 29945. The orange and blue surfaces are contours of the temperature ($0.2\,eV$) and water density field ($0.002\,g/cm^3$), respectively. The views have been approximated using only 24 depth images with a resolution of 512^2 pixels, whereas the original triangulation has around 3 million triangles. Our approach first computes the position of each depth image pixel in world coordinates, and then renders the resulting geometry either via a point cloud (left) or a surface triangulation (right). The used depth images have been chosen with VOIDGA to bound the maximum approximation error for the current view.

ABSTRACT

In this work, we propose a novel view-approximation oriented image database generation approach (VOIDGA) that enables the adequate generation of arbitrary views. Our approach utilizes Depth Image Based Rendering (DIBR) techniques to derive novel views based on a set of depth images. In contrast to approaches that store a huge amount of images to cover a wide range of possible view directions, VOIDGA identifies and stores only those images that significantly contribute to the overall view-approximation quality while bounding the resulting approximation error. This further reduces the size of image databases and the number of images that need to be processed by DIBR algorithms. We demonstrate VOIDGA on several challenging real-world examples, and compare our approximations against ground truth renderings using two image-based metrics.

Keywords: Image Database, Depth Image Based Rendering, Geometry Reconstruction, View Approximation

1 INTRODUCTION

The increasing size and complexity of datasets make it necessary to reduce the amount of stored information while still supporting effective data exploration through interactive visual interfaces. Especially in the case of extreme scale simulations, it is often impossible to interactively render—or even store—an acceptable set of simulation states for *post hoc* analysis due to bandwidth and I/O constraints. To address this issue, Ahrens et al. [1] proposed the ParaView Cinema concept as an image-based approach for the *post hoc* exploration of simulation output. In their approach, an image database is created *in situ* consisting of color and depth images of simulation elements taken from various camera positions. Such databases are several orders of magnitude smaller than the simulation data they are derived from, and they enable the real-time exploration of extreme scale simulations by querying and compositing images from the database. Rudimentary image database viewers facilitate basic camera movement by simply snapping to the closest available camera position for a requested viewpoint [2, 23, 32]. As a principled limitation, these viewers cannot visualize viewpoints that had not been foreseen and specified during database generation. However, depth image based rendering (DIBR) algorithms enable the approximation of novel views based on existing database elements, which supports unconstrained camera interaction for visual exploration (Fig. 1). Such algorithms, however, introduce approximation errors that depend on the quality of the used DIBR technique and the input depth images. It is also not clear which and how many images are needed to adequately approximate a wide range of novel views.

In this paper, we address these issues by taking the first steps towards leveraging the information stored in an image database to its full potential. Specifically, we propose a novel view-approximation oriented image database generation approach (VOIDGA) that determines a minimal set of input depth images that enable the approximation of new views within a certain error bound. The core concept of VOIDGA is to identify and store images that significantly contribute to the overall view-approximation quality, while at the same time discarding images that can already be adequately approximated. This yields much smaller image databases than the ones produced by current state-of-the-art implementations which uniformly sample

*e-mail: jl@jluk.de
†e-mail: e_kinner15@cs.uni-kl.de
‡e-mail: ahrens@lanl.gov
§e-mail: leitte@cs.uni-kl.de
¶e-mail: garth@cs.uni-kl.de

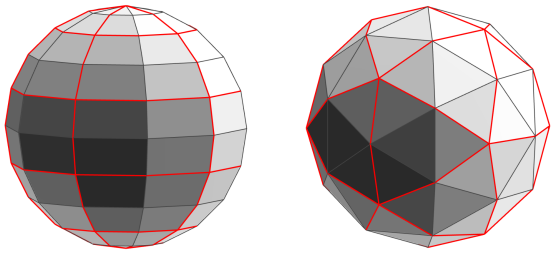

Figure 2: Sampling grids for the Cinema database generation: a latitude-longitude parameterized sphere (left), and a refined icosphere (right). The first subdivision is shown in red, the second subdivision in gray. The icosphere vertices are uniformly spread, while the latitude-longitude grid oversamples poles, and undersamples the equator.

images on a spherical grid (Fig. 2). This also results in a reduced set of images that need to be processed by DIBR algorithms while still guaranteeing a minimum approximation quality.

We demonstrate the effectiveness of VOIDGA on several real-world examples of varying complexity, including strongly jagged surfaces and line geometry which represent worst-case scenarios for depth image based geometry approximation algorithms. To assess the quality of the resulting approximations, we deploy two image-based metrics to measure the perceived and actual shape distortion between ground truth images and the approximations. Furthermore, our approach can act as an optional addition to existing image databases, and can be employed independently from other database dimensions; such as time or parameter values. To summarize, the contributions of this work are:

- A survey of literature on IBR methods suited for scientific image database element approximation.
- A detailed description of a DIBR implementation for the approximation of arbitrary views for Cinema databases.
- A novel view-approximation oriented image database generation approach (VOIDGA) that derives a minimal set of images to approximate arbitrary views within a certain error bound.
- The demonstration that this approach allows fluid camera interaction with acceptable errors, which are examined qualitatively and quantitatively using two image-based metrics over several databases created from real-world use cases.

2 Related Work

Image Based Rendering (IBR) methods and their integrated geometry approximation algorithms have been extensively studied in the context of remote rendering [3, 6, 17], image-based meshing [7, 9, 27, 31, 43, 44], 3D video processing [26, 38], and many more. In remote rendering, they significantly reduce server and bandwidth load by enabling clients to extrapolate new views based on already transmitted images without additional requests to the server [6]. As soon as the camera from the client diverges too much, the server generates and sends new depth images to the client. This is especially useful if the visualizations require computational or data intensive procedures. In 3D video processing, they allow to *post hoc* create stereoscopic images based on video-plus-depth footage [38]. They are also used to mesh objects based on multiple photographs, which enables photorealistic texture mapping [34, 39], the digital archiving of cultural heritage [44], and the complete reconstruction of indoor as well as outdoor environments [27, 43].

Shum and Kang [35] point out that all these methods require either implicit [4,5,9,21,42,44], explicit [6,27,30,31], or no [18,25] geometry information to create novel views based on feature registration, geometry approximation, or plenoptic functions, respectively. Implicit geometry approximation algorithms interpolate between images by detecting and tracking features—such as the optical flow [4, 21, 42] or SIFT [9]—which creates visually appealing transitions between different views. However, the interpolated images do not necessarily have to coincide with reality, and often exhibit ghosting and warping artifacts [37]. IBR algorithms that use no geometry information interpret large, dense sets of images as two-dimensional slices of the four-dimensional light field function [18, 25]. All images are used to approximate the light field which is subsequently sampled to generate novel views. This produces high quality results as long as the light field approximation is good enough, but this requires a huge amount of images (1k+) and even compressed representations [18] do not scale for non-static scenes.

On the other side of the IBR spectrum are algorithms that explicitly derive the implied geometry of depicted objects based on depth images. These images can be obtained from sensors [27, 43], estimators [10, 19, 20, 22], or directly from the rendering pipeline [1]. As it is straight-forward to generate Cinema databases that contain precise depth images of 3D rendered objects—such as iso-surfaces, streamlines, and particles—we focus in this paper on Depth Image Based-Rendering (DIBR) techniques. This does not mean, however, that the other approaches do not have merit or are inapplicable for Cinema databases. In the following, we present an overview across the development of DIBR techniques, which is also the basis of our exemplary implementation. Like other DIBR methods, our implementation is based on the fact that each pixel of the depth image corresponds to a 3D point on the depicted surface (Fig. 3a and b). These points can be computed by inverting the projection that was used to generate the depth image [6, 27, 30, 31], which yields a set of independent points in 3D space. A simple way to render the resulting locations is to represent them as a point cloud, called splatting [28, 31, 36, 46] (Fig. 3b). However, this creates gaps between points; especially when the depth image has a low resolution. The gaps can be filled by increasing the point size (which leads to a strong divergence from the original surface), or by increasing the point number (which requires high-res depth images). Another way to solve this problem is to continuously fill these gaps. For this it is necessary to link neighboring points of the depth image by creating a surface patch between them, i.e., derive a triangulation based on the depth image. As a first step, one can create two triangles between four neighboring pixels to create a piecewise linear approximation of the surface between the points (Fig. 3c). This fills all gaps, but also creates surface patches between pixels with very different depth values (Fig. 3c). This is known in the DIBR literature as the depth discontinuity [26, 38, 45]. A trivial solution to this problem is to use a distance threshold to discard distorted triangles (Fig. 3d). Unfortunately, there exists no threshold value that will always produce the best results as this value strongly depends on the smoothness of the depicted object. Moreover, removing such triangles creates gaps again that must be either filled by a variant of splatting [26], or by incorporating the implied geometry from multiple depth images [5, 6, 27] (Fig. 4).

Our algorithm follows the previously described outline, but is optimized for the view approximation of Cinema databases as these pose additional challenges. Most prominently, it is not possible to derive new image artifacts once the database has been build, since this would require rerunning the simulation. Therefore, it is necessary to determine *in situ* various database parameters (e.g., the image resolution and sampling rate) that will later enable the approximation of views within a certain error threshold. To assess the quality of the approximated views, we use two image metrics: the Depth Difference (DD) [15], and the Multi-Scale Structural Similarity Metric (MS-SSIM) [41]. An advantage of these image-based metrics is that they fit well in the context of Cinema databases, and that they are independent of the actual data representation. Euclidean-based mesh distortion metrics—including the Hausdorff Distance,

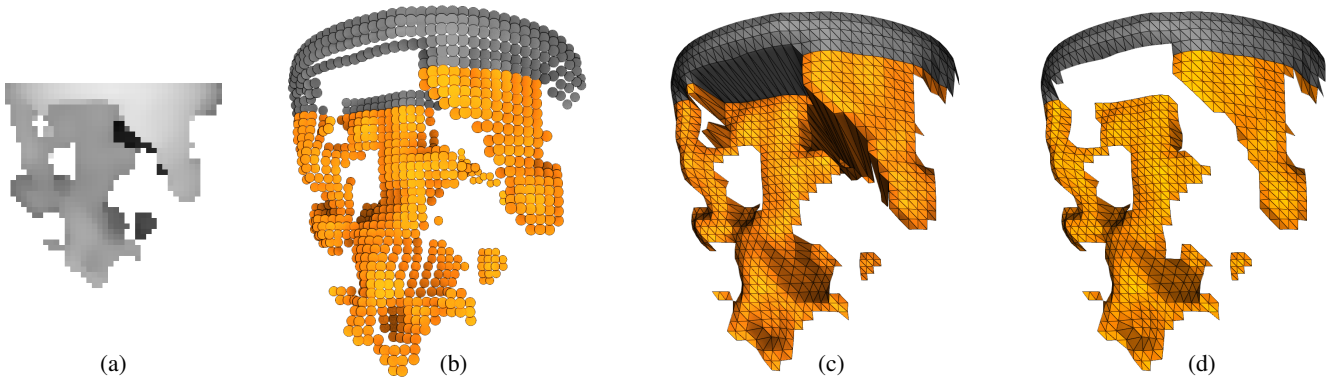

Figure 3: Illustration of the forward mapping of a single 40x40 depth image (a) for the viscous finger dataset. The resulting points can either be directly visualized by splatting (b), or by approximating the surface between the points ((c) and (d)). Splatting (b) creates gaps that need to be filled either by increasing the point number (i.e., image resolution) or the point size. The surface approximation creates piecewise-linear patches between vertices (c), but a distance threshold is needed to discard distorted triangles (d).

the Root Mean Square Error, the Mesh Structural Distortion Measure 2, and the Mean Opinion Scores—explicitly compare two different representations, i.e., the triangle meshes. These require either that the meshes have the same number of vertices, or that a point-correspondence has to be established beforehand [14]. In the case of extreme scale simulations—which is the primary application of Cinema databases—they are expensive to compute and very restrictive.

3 Method

In the following we describe our DIBR implementation that approximates the depicted surface geometry, the error metrics that are used to evaluate the resulting approximation quality, and our novel view approximation-oriented image database generation approach.

3.1 Depth Image Based Rendering Implementation

In this section we describe our DIBR implementation that enables the generation of novel views based on image database elements. The core idea of this approach is to deform for each individual depth image a uniform grid to approximate its depicted geometry, discard distorted parts of the approximation, and finally compose the approximations from multiple depth images into a single image. Fig. 5 shows the outline of our DIBR algorithm which is completely implemented in the standard OpenGL rendering pipeline. Note, however, that this just serves as an exemplary implementation and that VOIDGA can use any DIBR method.

3.1.1 Stage 1: Initialization

First, we initialize a Framebuffer Object (FBO) cache P that will be used in stage 2 to store the 3D world positions of the depth image pixels. We also create a uniform grid that will be deformed for each cached texture in stage 3. Each vertex of the grid corresponds to a single pixel. Thus, the number of vertices corresponds to the image resolution, and the xy-coordinate of each vertex is the 2D-index of its associated pixel. To create piecewise-linear surface patches between the vertices, we create two triangles for each group of four pixels that are adjacent to each other. Even for full HD resolutions, this yields a number of triangles that can easily be handled by standard graphics hardware. The topology of the resulting mesh stays the same throughout the entire rendering pipeline, and its geometry will only be updated on the GPU during a vertex shader call at stage 3.

3.1.2 Stage 2: Caching

To avoid recomputing the 3D world positions of all depth image pixels each time the camera is changed, we cache the positions of each depth image D_i in a new position texture P_i. In contrast to the depth image that stores at each pixel the depth value, the position texture stores at each pixel the 3D world position of its corresponding depth image pixel. If the depth image was generated by an orthographic camera then the position of each depth image pixel can be calculated by

$$p(u,v,d) = \vec{c_p} + u \cdot c_w \cdot (\vec{c_d} \times \vec{c_u}) + v \cdot c_h \cdot \vec{c_u} + d \cdot \vec{c_d} \qquad (1)$$

where $\vec{c_p}$, c_w, and c_h are the camera position, width, and height in world coordinates; $\vec{c_u}$ and $\vec{c_d}$ are the normalized camera up and direction vector; $u,v \in [-0.5, 0.5]$ are the coordinates of the pixel in image space; and d is the depth value of the pixel in world coordinates. We use an orthographic over a perspective camera model to avoid projective distortions [15]. Since each pixel can be processed individually, we compute all positions of D_i in one pixel shader pass that stores the locations in a new framebuffer P_i, which is then inserted into the texture cache P. In our experiments, this process takes roughly $0.02s$ for each 1024^2 depth image while not utilizing parallelism.

3.1.3 Stage 3: Forward Warps

In this stage, we reproject the locations stored in the texture cache P using the current camera setting. First, we clear color and depth channels of the framebuffer F, and then process each texture in the cache individually to incorporate its depicted surface into the overall view approximation.

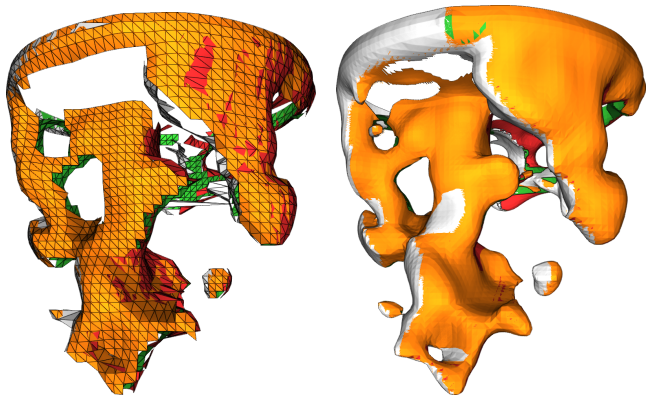

Figure 4: Composited approximations of the viscous finger dataset using four depth images with a resolution of either 40^2 (left) or 1024^2 (right). Colors encode the depth image that generates its corresponding surface patch. Depth images that depict the same part of a surface generate similar patches which causes z-fighting.

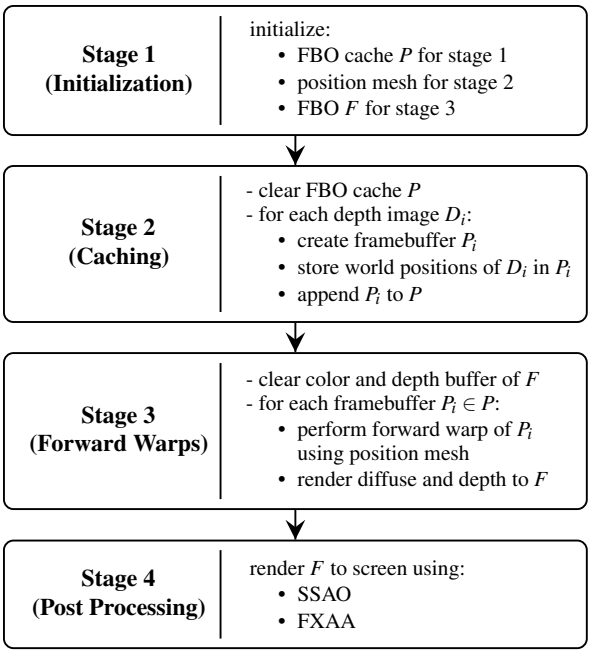

Figure 5: Outline of our DIBR pipeline. First, we initialize the data structures that are used in later stages. We then build a texture cache that stores the 3D world positions of the depth image pixels. Upon camera rotation, we process the cache and reproject positions using the new camera settings to write color and depth outputs in an intermediate framebuffer. Finally, we post-process the framebuffer using SSAO and FXAA.

Next, for each texture P_i we perform an optimistic forward mapping of the uniform grid vertices. Thus, in the vertex shader, each vertex looks up its corresponding 3D location in P_i and subsequently uses the projection of the current view to compute its new location in image space. This deforms the uniform grid to align with the positions depicted by the original depth image D_i. Subsequently, we have two options on how to render the grid: either we use splatting by rendering the vertices as points (Fig. 3b), or we render the triangles connecting the vertices (Fig. 3c). As we demonstrate in Sect. 4, both methods have their relative advantages. Splatting requires a certain point size (which might be relatively large) to fill gaps and give the illusion of looking at a surface. The triangulation, on the other hand, suffers from depth discontinuity problems, i.e., it also creates surface patches between vertices which are far apart. To compensate, we introduce the distance threshold to discard distorted triangles (Fig. 3d). Unfortunately, there exists no distance threshold that always produces the best results in every scenario. This parameter depends strongly on the depth image resolution and the smoothness of the depicted surface. As we decrease the threshold, we remove more and more distorted triangles and thus create less surface area, but the remaining approximated surfaces have a higher probability of belonging to the originally depicted surface.

If an image database only consists of depth images then we simply render a user-controlled color value for the resulting geometry. If the database also stores for each depth image a data texture that associates a scalar value with every depth image pixel—such as a temperature measurement at the depicted surface—then we also pass this texture to the shader. The data values are then rendered according to a user-specified color map (Fig. 8). In the case of splatting, we render the resulting color for the entire point (Fig. 8a). For the triangulation, we linearly interpolate the scalar values between triangle vertices (Fig. 8c). Note, in our current implementation we do not compute any lighting in this stage, although it would be possible. Instead, our deferred renderer computes the lighting in stage 4.

Finally, we write the color and depth value to framebuffer F, where each render call updates—but never clears—the current depth buffer. This way, the deformed grids get drawn with correct occlusion, and only a constant triangle mesh is supplied for each render call. The only thing that changes during each pass is the used position texture P_i. Since we use the conventional z-buffering algorithm, we can take advantage of hardware acceleration. Fig. 4 illustrates which depth image contributes which part of the resulting surface approximation. Images that depict the same part of the surface generate patches that overlap and thus cause z-fighting. In our case, this is advantageous as this indicates that the approximations agree on the shape of the depicted object. Another advantage of the modular design of our composition scheme is that it is possible to process any subset of the cache. This set could consist of all available depth images, or an intelligently chosen subset of neighboring camera locations. On average hardware, our current implementation processes up to 200 Full-HD depth images at interactive framerates ($> 30\,Hz$).

3.1.4 Stage 4: Post Processing

As we do not compute any lighting at stage 3, we use a customized Screen Space Ambient Occlusion (SSAO) shader to compute the global and local lighting. The global lighting greatly enhances spatial perception by darkening regions that are next to areas which are closer to the camera. For the local lighting, the SSAO shader approximates the normal at a fragment and subsequently computes its interaction with a point light source. The approximated normals are also used to create a silhouette effect by emphasizing hard edges. Furthermore, it is possible to adjust SSAO parameters such as the radius of the depth sampling as well as the emphasis of shadows and edges. The initial settings of these parameters can be used for any scene, but users can perform a fine tuning to highlight specific aspects of a model. Finally, the Fast Approximate Anti-Aliasing (FXAA) shader smoothes jagged edges.

3.2 Quality Metrics

To assess the significance of individual depth images during the use of VOIDGA (Sect. 3.3), and to validate our results (Sect. 4), we compare the approximated views to ground truth renderings using the Multi-Scale Structural Similarity Metric (MS-SSIM) [41] that estimates the perceived image similarity, and the Average Depth Difference (DD) [15] that measures the actual shape distortion.

The DD tries to capture the volumetric difference between two objects and is computed in two stages. First, it computes the Average Depth Difference (ADD) between two depth images by computing the actual depth value difference per pixel. Hence, it is assumed that the depth images have the same size, and that their values are in the range $[0,1]$. Similar to Cinema databases, this is done for multiple camera angles that are positioned on vertices of a grid that encapsulates the dataset and aim towards the object center. The DD is then given as the average of all computed ADDs.

The MS-SSIM is modeled after the assumption that the human visual system is highly adapted for extracting structural information from 3D projections. Thus, a measure of the structural similarity between images can provide a good estimate of the perceived image quality [40,41]. In contrast to the original SSIM, the MS-SSIM iteratively downsamples the input images to determine the luminance and contrast variations for varying resolutions. This allows to evaluate the structural similarity between images more independently from the actual image sizes. Similar to the ADD metric, we compute the MS-SSIM for multiple camera positions and build the average to evaluate the structural similarity across the entire approximation. However, the ADD computes an error value from 0 (identical) to 1 (complete opposite), whereas the MS-SSIM computes a score from 0 (not similar) to 1 (identical).

3.3 VOIDGA

We propose a novel view-approximation oriented image database generation approach (VOIDGA): a greedy algorithm that iteratively refines a sampling grid and then only stores potential database elements (also called artifacts) that significantly contribute to the overall approximation quality. To this end, VOIDGA consists of three phases: the database backbone generation, the database refinement, and the database downsampling. To run VOIDGA completely automatically, users have to specify the maximum number of database artifacts, the initial (and thus maximum) artifact resolution, and the initial error tolerances. In the following, we demonstrate VOIDGA using our DIBR implementation, and the ADD and MS-SSIM metrics, but it is possible to use VOIDGA with any method or metric.

3.3.1 Backbone Generation

First, we normalize the dataset geometry according to the dimensions of the unit-cube with center at the origin, and then select a sampling grid structure. A common way to generate Cinema databases is to uniformly sample along a latitude-longitude parameterized sphere that encapsulates the dataset, where the cameras are positioned at the grid vertices and aim towards the center (Fig. 2). For image database viewers that simply snap to the next available artifact, this creates intuitive transitions as it seems like the camera rotates along the lat-lon axes. However, this grid causes an oversampling at the poles, and an undersampling at the equator (Fig. 2 left). Since our DIBR method is not restricted to the actual artifact locations, we propose to use an icosahedron as a sampling primitive. In contrast to a lat-lon grid, each icosahedron refinement uniformly creates new sampling positions that equally cover possible view angles (Fig. 2 right). As these positions are eventually added to the database, we effectively improve the approximation quality in each step.

To generate the database backbone (a small set of database artifacts that are the basis for the view approximation), we sample images on the 12 vertices of the unrefined icosahedron (intersection of red lines in Fig. 2, right). For each vertex of the grid, we generate a depth image with the initial resolution using an orthographic camera where the width and height are set to the icosahedron diameter. Thus, each artifact encapsulates the complete dataset, and the 12 locations already provide a good view angle coverage. It is also possible to add model-specific view angles—such as interior locations—to the backbone if such important angles are known. The next phase uses the resulting artifacts as a basis for the view approximation.

3.3.2 Database Refinement

In this phase, VOIDGA iteratively refines the sampling grid and only adds artifacts that significantly contribute to the global approximation quality. Hence, it is necessary to derive two depth images for a position: the depth image D of the ground truth geometry, and the depth image $\hat{D}$ of the current view approximation using all available database artifacts. Note, the depth image $\hat{D}$ depends on the used DIBR algorithm (e.g., triangulation or splatting) and its respective parameters (e.g., the distance threshold and point size). In the case of our implementation, we initialize the DIBR pipeline by building the position texture cache using the database backbone.

To automatically tune the initial settings of the DIBR parameters, we derive both D and $\hat{D}$ at the positions where the approximation error can be assumed to be maximal, i.e., at the triangle centers of the current icosahedron refinement. Next, VOIDGA finds a local approximation error minimum by iteratively increasing the DIBR parameters. To do this, it is only necessary to derive the new depth images $\hat{D}$, and to compare them to the cached images D by computing the ADD and the MS-SSIM for each resulting pair. The ADD can be directly computed using the depth images, but the MS-SSIM compares color images. Since our DIBR pipeline is modular, we can simply feed the depth images individually into stage 4 of our rendering system to generate images that are equally shaded. As soon as the error increases, we stop the automatic tuning and communicate the current error and DIBR settings to the user. Although VOIDGA can run fully automatically, this gives users the option to directly compare the current approximation against the ground truth; either by directly contrasting the pairs, or by free camera movement as long as the ground truth can be rendered at interactive framerates. Moreover, users can adjust the database constraints and the error thresholds, which is especially useful if proper initial settings are unknown.

After the automatic tuning, VOIDGA refines the icosahedron grid which yields a set of potential database artifact locations at the new vertex positions. VOIDGA then derives for each position both depth images and computes the error metrics. Instead of storing all depth images D immediately in the database, VOIDGA discards all pairs that satisfy the error thresholds, and then only adds the ground truth image to the database that has the worst approximation quality. Next, it recomputes the depth images $\hat{D}$ at the remaining positions with the current database, and then repeats the previously described process. After all current positions have been processed, VOIDGA again refines the icosahedron, performs the parameter tuning, and selects important samples. This process is repeated until either the maximum number of artifacts is reached, or all candidates satisfy the error thresholds. This scheme reduces the number of stored artifacts, while asserting a minimal approximation quality at the missing sampling locations. In our experiments, sequentially executing this process took roughly one minute. Note, however, deriving new depth images and their scores is embarrassingly parallel.

3.3.3 Database Downsampling

Finally, VOIDGA communicates to the user the impact of the artifact resolutions on the overall approximation quality and the used disk space. Obviously, a lower artifact resolution results in worse approximations, but the benefit of a significant disk space gain might be worth a slightly worse approximation quality. Note, for this stage it is not necessary to actually recompute the depth images D and $\hat{D}$, instead they can be directly downsampled from the high-res images.

4 Results

In the following, we demonstrate the effectiveness of VOIDGA in conjunction with our DIBR implementation on several real-world examples of varying complexity. To validate the quality of the generated views, we examine the approximation error qualitatively and quantitatively based on two image similarity metrics.

4.1 Error Plots

We quantitatively evaluate the approximations generated by the proposed algorithm using the similarity metrics described in Sect. 3.2. Specifically, for a given resolution and number of database elements that were either generated uniformly (U) or via VOIDGA (V), we generate a total of $1,000$ random viewing directions over the unit sphere, derive both the ground truth and the approximated view, and subsequently compute the difference between them based on the ADD and MS-SSIM metrics. Fig. 6 depicts the distributions of both metrics grouped first by dataset, then by image resolution, and finally by the different sampling methods. We proceed to demonstrate and discuss these results in detail for each of the three datasets.

4.2 Viscous Fingering

We first demonstrate our approach on datasets that exhibit large smooth surface areas. Specifically, we generated image databases for the simulation ensemble that was provided for the 2016 Scientific Visualization Contest [11]. These simulations model the process of viscous fingering: an instability phenomenon that occurs in porous media at the interface between two fluids of distinct viscosity. In the specific case of the contest dataset, the simulations model a cylinder that is filled with pure water and that contains an infinite salt supply

Figure 6: Approximation errors resulting from our DIBR approach measured via the Multi-Scale Structural Similarity Metric (MS-SSIM; top set), and the Average Depth Difference (ADD; bottom set) for $1,000$ random viewing positions on the unit-sphere. For each metric, we compute the error that results from splatting (cool colors, top row) and triangulation (warm colors, bottom row) grouped first by dataset, then by image resolution, and finally by the selection method of database elements. Legends indicate how many images were used as the basis for the view approximation, e.g., the 42 images that are uniformly sampled on a refined icosahedron are shown in red. The gray set of values indicates the errors achieved by database elements that were chosen by VOIDGA to adequately approximate the depicted model. The violin plots illustrate for each case the histogram of errors over the random positions. *Top set:* MS-SSIM metric (higher values are better; 1.0 denoting identical images). It can be observed that as the resolution and number of base images increase, the approximation converges to the ground truth, with acceptable values (> 0.95) for all three datasets already reached for 42 images with a resolution of 256^2. Except for the streamline dataset, the top-heavy histograms indicate that the distribution is skewed strongly towards higher similarity with only few outliers. The VOIDGA approach selects more images than strictly necessary, but can be employed *in situ* without prior knowledge of the values underlying these diagrams. *Bottom set:* ADD metric in logarithmic scale (lower is better, 0.0 denoting perfect reproduction). Again, errors are low even for few images or comparatively low resolution.

at its top. As soon as the salt mixes with the water, the resulting solutions sink down to the bottom as they have a higher density than the surrounding water. Meanwhile, the solutions form structures with increased salt concentration value; called viscous fingers. We follow the approach of Lukasczyk et al. [24] to identify viscous fingers as isosurfaces of the salt concentration density field. Fig. 5 shows the ground truth isosurface geometry and the view approximations, where the viscous fingers and the salt supply are colored bright orange and dark gray, respectively. Quantitative results for this dataset are shown in Fig. 6, left column. For smooth surfaces as the ones found in this case study, triangulation outperforms the splatting technique. Not only does it exhibit a lower approximation error, but also achieves a higher frame rate. Splatting causes a warp of the original surface—i.e., creates an artificial width of the surfaces based on the point size—which causes the generated views to score lower on the image metrics. As shown in Fig. 6, VOIDGA uses fewer images (23) than the complete second icosahedron refinement (42), yet achieves similar error scores. Artifacts that depict the top of the salt supply are discarded by VOIDGA as the smooth surface of the supply can already be approximated by the database backbone.

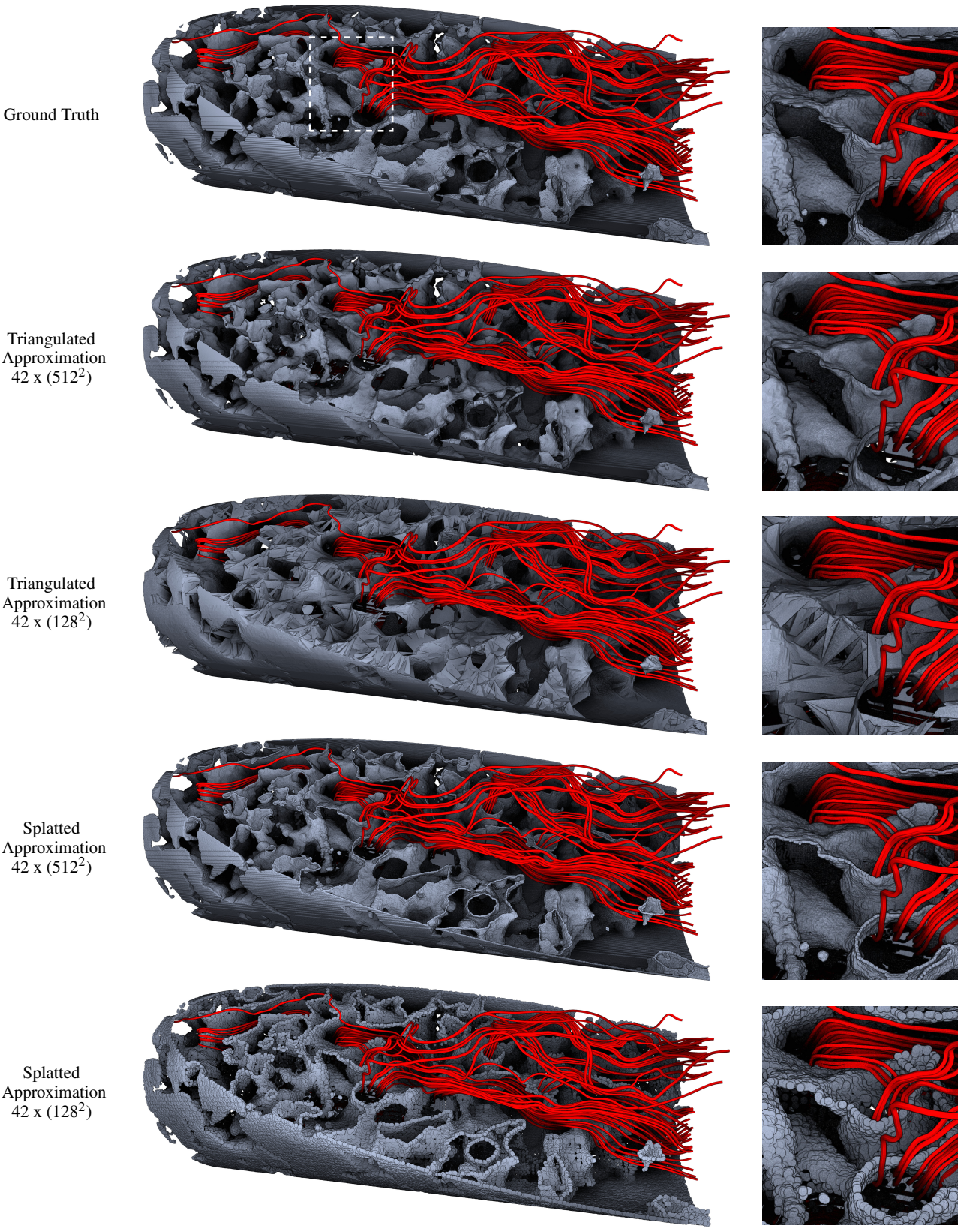

Figure 7: Generated views for the ground water dataset using different rendering modes and parameters. The images show the path of water (red streamlines) through a karst limestone ground sample (gray) that was taken in south Florida. The dataset was provided by the Texas Advanced Computing Center (TACC) and the Florida International University. All views have been generated using 42 depth images with a resolution of either 512^2 or 128^2 pixels. As the approximations show a view angle that was not covered by the used depth images, the approximation error is largest in the cavities where no geometric information is available. Nevertheless, the outer structure is accurately reconstructed even for the relatively low number and resolution of the depth images.

4.3 Asteroid Impact

A second dataset exhibiting large and relatively smooth isocontours is part of a threat assessment study of asteroid ocean impacts [33] that was made publicly available for the 2018 scientific visualization contest [12]. The dataset consists of several extreme scale simulations that model different impact scenarios for varying impact angles, asteroid sizes, and heights of potential airbursts. We generated isosurfaces for the temperature and water density field for impact scenario *yA*31, i.e., no airburst event, an asteroid diameter of 250 meters, and an entry angle of 45 degrees. Fig. 1 depicts the temperature and water density isosurfaces for level 0.2 eV and 0.002 g/cm^3 in orange and blue, respectively. The ground truth geometry consists of roughly three million triangles, while the generated view was derived based on only 12 depth images with a resolution of 512^2 pixels for each contour, i.e., 24 depth images with an uncompressed total size of $24MB$. The images were chosen using VOIDGA to ensure approximation error bounds of 0.001 ADD and 0.97 MS-SSIM for the current view. The large surfaces are accurately approximated, while the base of the water vapor exhibits some approximation errors. Triangulations do not create adequate surface patches of small features due to the low pixel density of the used depth images. Splatting, on the other hand, will render a point at the location of a small feature as long as it is depicted by at least a single depth image pixel. However, the splatted surface suffers from gaps, and the point borders induce a rough surfaces appearance. Increasing the point size to fill these gaps results in an artificial surface warp that negatively impacts the overall approximation quality.

4.4 Groundwater

In this case study, we demonstrate that our DIBR algorithm can also approximate very complex surfaces with an acceptable error, and that our approach enables the composition of approximated and explicitly stored geometries. To this end, we generated a Cinema database for a karst limestone ground sample that was taken in south Florida. The ground sample was provided by the Texas Advanced Computing Center (TACC) and the Florida International University as a triangulated surface consisting of roughly 8 million triangles (gray surface of Fig. 7 top). Domain experts involved in this research are primarily interested in the propagation of ground water through the stone cavities (red streamlines of Fig. 7). This dataset is challenging for depth image based geometry approximation since the complex structure of the cavities occlude most of the interior geometry. To compensate, it is necessary to sample depth images on a dense grid. In the following, we show that even for the low number of samples chosen by VOIDGA it is possible to adequately reconstruct the outer shell of the stone. However, to demonstrate the effects of undersampling, we use only 42 depth images with a resolution of either 512^2 or 128^2 pixels that are sampled on a once subdivided icosahedron (Fig. 2b).

The rows 2-3 and 4-5 of Fig. 7 show approximated views that are generated via surface triangulation or splatting, respectively. The lighting of the complete scene is performed in the post processing shader. Screen space ambient occlusion greatly enhances the perception of the stone porosity and the spatial arrangement of the streamlines. With the sparse sampling the outer structure of the stone is still accurately reconstructed, while deep inside the cavities it results in missing geometry. Moreover, fine details of the structure are only visible if the resolution of the depth images is high enough. Since the proposed triangulation algorithm requires at least three neighboring depth pixels that are below the distance threshold to create a surface patch, the resulting approximations ignore one pixel wide surface depictions. Splatting preserves these features, as each depth image pixel is still represented by a single point. However, the size of these points must be large enough so that their overlap fills the gaps between them. Large points also give the impression that surfaces have non-zero width, which is not true for triangulated surfaces. Yet, this greatly improves the 3D perception and emphasizes hard edges such as cavity openings. Nevertheless, rendering the point cloud is more expensive than rendering the triangulation. The triangulation also enables the more accurate estimation of surface normals, since splatting renders each point as a flat disc that faces the camera. This causes depth discontinuities at the disc boundaries.

To emphasize the impact of the artifact resolution, Fig. 7 also shows the geometry approximations that result from using images with only 128^2 pixels. These depth images do not have a sufficient resolution to represent small cavities, as neighboring pixels are too far apart in world space, while the drastically varying surface between the pixels is not depicted. However, even at this resolution, prominent features such as the big cavities are clearly identifiable. Triangulation requires a fairly high distance threshold to coincide with the rough shape of the original surface, which results in numerous distorted triangles. Splatting also requires a large point size to fill gaps. Because the triangulation connects neighboring pixels with similar depth values, splatting produces much better results for low resolution images as each pixel is still mapped to 3D space independent of its neighbors at the price of warping the resulting surfaces. This is also reflected in the error metrics (Fig. 6, middle). Especially the MS-SSIM is very sensitive to the surface warps. Furthermore, VOIDGA uses far fewer artifacts (82) than the maximum refinement level (162) while achieving similar approximation errors. Since the cylindrical stone sample has a relatively smooth backside, VOIDGA primarily stores images that depict the front.

An advantage of the modular design of the demonstrated DIBR algorithm is that the approximated geometry can be rendered together with other, non-approximated geometries. For instance, the red streamlines of Fig. 7 are explicitly stored geometries that are correctly composed with the approximated geometry. Based on this principle, extremely large simulation elements can be approximated by depth images, while specific features of smaller size can be stored explicitly.

4.5 Jet Streamlines

Sparse line geometry is another challenge for depth image based approximation techniques due to the strong depth variations of neighboring pixels. In the following, we demonstrate the quality of our view approximation for streamlines computed from a CFD *Jet* simulation. It models the injection of a jet into a medium at rest and the friction-based formation of vortical structures.

As expected, the approximations cause large errors for small image resolutions (Fig. 6, right column). Intuitively, the image resolutions are far too low to distinguish between individual streamlines. This is especially negative for the triangulation as neighboring streamlines are falsely connected via surface patches (Fig. 8, right). Although splatting can still produce convincing results even for low resolutions, the necessary large point size bloats the streamlines (Fig. 8, left) which has a dramatic impact on the error metrics.

For this dataset, we used VOIDGA to generate a database with a focus on high quality depth approximations rather than image similarity, effectively de-emphasizing color reproduction. Therefore, we enforced a strict ADD threshold (0.0004) and a relaxed MS-SSIM threshold (0.8). This bias towards depth image quality can also be observed in the respective error plots. Note that the VOIDGA database (72 images) and the maximum refinement level (162 images) achieve similar errors.

The colors of the streamlines encode their lifetime and are mapped *post hoc*. This requires to store for each depth image an additional floating point image that records at each pixel the lifetime of the depicted part of the streamlines. The ability to apply a color map *post hoc* on the approximated geometries demonstrates that our approach can be easily combined with the existing practice of image databases.

Splatting Ground Truth Triangulation

Figure 8: Comparison between the generated views (left and right) and the ground truth (middle) for the jet dataset. To emphasize potential visual errors, the views have been approximated by using only the database backbone (12 depth images) with a resolution of either 256^2 or 512^2 pixels for the splatting or triangulation technique, respectively. The color map was applied *post hoc* and encodes the lifetime of individual streamlines. For sparse line geometry, the triangulated approximation exhibits large artifacts as it connects neighboring streamlines; falsely considering them to be a single surface. Splatting generates comprehensible results, but the points have to be relative large to create the impression of looking at continuous surface geometry. This bloats the streamlines, and therefore the resulting images score badly for the used metrics.

5 Limitations

The scope of this paper is to demonstrate the application of DIBR techniques in the context of Cinema databases, with the aim of understanding the value of these techniques for *in situ* and *post hoc* visualization. This does not only include the DIBR approximation of views, but also the smart generation of databases that guarantee a maximum approximation error. Advancing from this core concept towards a practical system requires several extensions.

First, as our entire pipeline is based on DIBR techniques, it is necessary to derive and store depth images. Thus, our approach does currently not support the approximation of volume renderings and transparent geometry. In this case, one needs to adapt other techniques such as image warping [16] or volumetric depth images [8].

Our geometry approximation methods (triangulation and splatting) are fairly rudimentary DIBR approaches. Although they require a minimal overhead and already produce acceptable results, more advanced DIBR methods are expected to produce higher quality approximations. Such techniques can easily be integrated into VOIDGA due to its modular design. Moreover, both of our DIBR implementations require parameters (the distance threshold and the point size) whose values have a significant impact on the resulting approximation quality. VOIDGA is capable of automatically finding suitable initial parameters, but the current tuning procedure can get stuck in local extrema. To solve this problem, we will integrate more advanced optimization techniques such as simulated annealing [13].

Naturally, the effectiveness of VOIDGA depends strongly on the used comparison metrics. While we have selected metrics that represent geometry representation (ADD) and image similarity (MS-SSIM), both are not without drawbacks. The most significant problem is their strong dependence on the background to foreground ratio. In effect, a larger background will result in better similarity scores which is not ideal for real-world settings. Moreover, the ADD is computed for normalized depth values, and therefore depends on the precision of the depth buffer. The MS-SSIM, on the other hand, requires input parameters that can currently only be chosen heuristically [41]. Furthermore, both metrics evaluate the overall image quality, and thus neglect small but potentially important features.

6 Conclusion and Future Work

We presented a novel view-approximation oriented image database generation approach (VOIDGA) that determines and stores a minimal set of images for the generation of arbitrary views while bounding the maximum approximation error. As demonstrated on several challenging real-world examples, VOIDGA can reduce image database sizes and the number of images that need to be processed by DIBR methods. VOIDGA can also ensure that a disk space budget is used to its full potential, which stands to be useful for *in situ* visualization, but also for sharing visualization results where bandwidth usage is of importance.

We examined the resulting approximation error qualitatively and quantitatively via two image-based comparison metrics: the ADD and MS-SSIM. Based on our results, we found that even a relatively low number of database elements (~ 42) at medium resolution (512^2) can already produce high quality approximations. Moreover, smooth surfaces can be well approximated by triangulations, whereas extremely jagged surfaces, sparse line-geometry, and low-resolution depth images are best approximated by splatting.

Towards adapting VOIDGA for production use, many improvements appear possible. As described in Sect. 5, we plan to integrate more advanced DIBR methods and other error metrics to further improve the resulting approximation quality. This is possible due to VOIDGA's modular design. We also plan to investigate view-dependent resolutions and feature-based camera locations. For example, depictions of smooth surfaces could be stored at low resolution, whereas detailed surface variations and important features are depicted by high-res images. A combination with the work of Nouanesengsy et al. [29] appears fruitful.

Acknowledgments

This research was funded by the German Research Foundation (DFG) within the International Research Training Group IRTG 2057 "Physical Modeling for Virtual Manufacturing Systems and Processes". This work was also funded by a DOE ASCR program award, Drs. Lucy Nowell and Laura Biven, program managers.

REFERENCES

[1] J. Ahrens, S. Jourdain, P. O'Leary, J. Patchett, D. H. Rogers, and M. Petersen. An image-based approach to extreme scale in situ visualization and analysis. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 424–434. IEEE Press, 2014.

[2] G. Aldrich, J. Lukasczyk, J. D. Hyman, G. Srinivasan, H. Viswanathan, C. Garth, H. Leitte, J. Ahrens, and B. Hamann. A query-based framework for searching, sorting, and exploring data ensembles. *Computing in Science Engineering*, In Review.

[3] W. Bethel, B. Tierney, J. Lee, D. Gunter, and S. Lau. Using high-speed wans and network data caches to enable remote and distributed visualization. In *Supercomputing, ACM/IEEE 2000 Conference*, pp. 28–28. IEEE, 2000.

[4] K. Chen and D. A. Lorenz. Image sequence interpolation based on optical flow, segmentation, and optimal control. *IEEE Transactions on Image Processing*, 21(3):1020–1030, 2012.

[5] S. E. Chen and L. Williams. View interpolation for image synthesis. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pp. 279–288. ACM, New York, NY, USA, 1993.

[6] J. Cui, Z. Ma, and V. Popescu. Animated depth images for interactive remote visualization of time-varying data sets. *IEEE transactions on visualization and computer graphics*, 20(11):1474–1489, 2014.

[7] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 11–20. ACM, 1996.

[8] O. Fernandes, S. Frey, F. Sadlo, and T. Ertl. Space-time volumetric depth images for in-situ visualization. In *Large Data Analysis and Visualization (LDAV), 2014 IEEE 4th Symposium on*, pp. 59–65. IEEE, 2014.

[9] T. Gurdan, M. R. Oswald, D. Gurdan, and D. Cremers. Spatial and temporal interpolation of multi-view image sequences. In *German Conference on Pattern Recognition*, pp. 305–316. Springer, 2014.

[10] C. Hane, L. Ladicky, and M. Pollefeys. Direction matters: Depth estimation with a surface normal classifier. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 381–389, 2015.

[11] IEEE VIS. Scientific Visualization Contest 2016. `http://www.uni-kl.de/sciviscontest/`.

[12] IEEE VIS. Scientific Visualization Contest 2018. `http://sciviscontest2018.org/`.

[13] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[14] M. Krivokuća, B. Wuensche, W. Abdulla, and G. Lavoué. Investigating the rate0distortion performance of a wavelet0based mesh compression algorithm by perceptual and geometric distortion metrics. *WSCG'2012*, p. 10, 2012.

[15] M. Krivokuća, B. C. Wünsche, and W. Abdulla. A new error metric for geometric shape distortion using depth values from orthographic projections. In *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, pp. 388–393. ACM, 2012.

[16] P. Lacroute and M. Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pp. 451–458. ACM, New York, NY, USA, 1994.

[17] H. G. Lalgudi, M. W. Marcellin, A. Bilgin, H. Oh, and M. S. Nadar. View compensated compression of volume rendered images for remote visualization. *IEEE Transactions on Image Processing*, 18(7):1501–1511, 2009.

[18] M. Levoy and P. Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pp. 31–42. ACM, New York, NY, USA, 1996.

[19] B. Li, C. Shen, Y. Dai, A. van den Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1119–1127, 2015.

[20] J. Li, R. Klein, and A. Yao. A two-streamed network for estimating fine-scaled depth maps from single rgb images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3372–3380, 2017.

[21] C. Lipski. *Virtual video camera: a system for free viewpoint video of arbitrary dynamic scenes*. PhD thesis, 2013.

[22] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE transactions on pattern analysis and machine intelligence*, 38(10):2024–2039, 2016.

[23] J. Lukasczyk, G. Aldrich, M. Steptoe, G. Favelier, C. Gueunet, J. Tierny, R. Maciejewski, B. Hamann, and H. Leitte. Viscous Fingering: A Topological Visual Analytic Approach. In *Applied Mechanics and Materials*, vol. 869, pp. 9–19. Trans Tech Publ, 2017.

[24] J. Lukasczyk, R. Maciejewski, G. Weber, C. Garth, and H. Leitte. Nested Tracking Graphs. In *Computer Graphics Forum (Special Issue, Proceedings Eurographics/IEEE Symposium on Visualization)*, vol. 36, 2017.

[25] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 39–46. ACM, 1995.

[26] K. Mueller, A. Smolic, K. Dix, P. Merkle, P. Kauff, and T. Wiegand. View synthesis for advanced 3d video systems. *EURASIP Journal on image and video processing*, 2008(1):438148, 2009.

[27] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pp. 127–136. IEEE, 2011.

[28] Z. Ni, D. Tian, S. Bhagavathy, J. Llach, and B. S. Manjunath. Improving the quality of depth image based rendering for 3d video systems. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pp. 513–516. IEEE, 2009.

[29] B. Nouanesengsy, J. Woodring, J. Patchett, K. Myers, and J. Ahrens. ADR visualization: A Generalized Framework for ranking Large-Scale Scientific Data using Analysis-Driven Refinement. In *IEEE 4th Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 43–50. IEEE, 2014.

[30] J. Ogniewski. High-quality real-time depth-image-based-rendering. In *Proceedings of SIGRAD 2017, August 17-18, 2017 Norrköping, Sweden*, number 143, pp. 1–8. Linköping University Electronic Press, 2017.

[31] M. M. Oliveira. Image-based modeling and rendering techniques: A survey. *RITA*, 9(2):37–66, 2002.

[32] Open-Source Cinema Viewers. CVLIB. http://cinemaviewer.org/, 2018.

[33] J. Patchett, G. Gisler, B. Nouanesengsy, D. H. Rogers, G. Abram, F. Samsel, K. Tsai, and T. Turton. Visualization and analysis of threats from asteroid ocean impacts. *Los Alamos National Laboratory*, 2016.

[34] E. Reinhard, W. Heidrich, P. Debevec, S. Pattanaik, G. Ward, and K. Myszkowski. *High dynamic range imaging: acquisition, display, and image-based lighting*. Morgan Kaufmann, 2010.

[35] H. Shum and S. B. Kang. Review of image-based rendering techniques. In *Visual Communications and Image Processing 2000*, vol. 4067, pp. 2–14. International Society for Optics and Photonics, 2000.

[36] A. Smolic, K. Muller, K. Dix, P. Merkle, P. Kauff, and T. Wiegand. Intermediate view interpolation based on multiview video plus depth for advanced 3d video systems. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 2448–2451. IEEE, 2008.

[37] T. Stich, C. Linz, C. Wallraven, D. Cunningham, and M. Magnor. Perception-motivated interpolation of image sequences. *ACM Transactions on Applied Perception (TAP)*, 8(2):11, 2011.

[38] W. Sun, L. Xu, O. C. Au, S. H. Chui, and C. W. Kwok. An overview of free view-point depth-image-based rendering (dibr). In *APSIPA Annual Summit and Conference*, pp. 1023–1030, 2010.

[39] J. Unger, A. Gardner, P. Larsson, and F. Banterle. Capturing reality for computer graphics applications. In *SIGGRAPH Asia 2015 Courses*, p. 7. ACM, 2015.

[40] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE*

Transactions on Image Processing, 13(4):600–612, April 2004.

[41] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, vol. 2, pp. 1398–1402. Ieee, 2003.

[42] M. Werlberger, T. Pock, M. Unger, and H. Bischof. Optical flow guided tv-l1 video interpolation and restoration. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 273–286. Springer, 2011.

[43] H. Xu and B. Chen. Stylized rendering of 3d scanned real world environments. In *Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, pp. 25–34. ACM, 2004.

[44] H. M. Yilmaz, M. Yakar, S. A. Gulec, and O. N. Dulgerler. Importance of digital close-range photogrammetry in documentation of cultural heritage. *Journal of Cultural Heritage*, 8(4):428–433, 2007.

[45] S. Zinger, L. Do, and P. H. N. de With. Free-viewpoint depth image based rendering. *Journal of visual communication and image representation*, 21(5-6):533–541, 2010.

[46] M. Zwicker, H. Pfister, J. van Baar, and M. Gross. Surface splatting. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pp. 371–378. ACM, New York, NY, USA, 2001.