EUROGRAPHICS 2004 / M.-P. Cani and M. Slater
(Guest Editors)

Volume 23 (2004), Number 3

VOTS: VOlume doTS as a Point-Based Representation of Volumetric Data

Sören Grimm†, Stefan Bruckner†, Armin Kanitsar† and Eduard Gröller†

Vienna University of Technology, Austria

Abstract

We present Volume dots (Vots), a new primitive for volumetric data modelling, processing, and rendering. Vots are a point-based representation of volumetric data. An individual Vot is specified by the coefficients of a Taylor series expansion, i.e. the function value and higher order derivatives at a specific point. A Vot does not only represent a single sample point, it represents the underlying function within a region. With the Vots representation we have a more intuitive and high-level description of the volume data. This allows direct analytical examination and manipulation of volumetric datasets. Vots enable the representation of the underlying scalar function with specified precision. User-centric importance sampling is also possible, i.e., unimportant volume parts are still present but represented with just very few Vots. As proof of concept, we show Maximum Intensity Projection based on Vots.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Graphics Data Structures and Data Types

1. Introduction

Volumetric data processing is commonly sample-based. Volumetric models consist of a huge number of samples. Each sample contains only information about the data at one specific position. In order to reconstruct a continuous function, neighborhood connectivity is necessary. Volumetric data is often given on a rectilinear grid. The main advantage hereby is, that the positions of the data samples are stored implicitly and that it allows efficient spatial addressing of the data. The rigid shape of such grids becomes more and more a limitation factor as the data sets are constantly increasing in size due to more advanced acquisition devices. Furthermore, the fraction of non relevant volumetric regions in relation to areas of interest is steadily increasing. Non relevant volumetric regions, such as empty space, should not be represented explicitly. Moreover, this explicit representation needs costly storage resources and introduces additional complexity in processing algorithms. For example, extensive research in volume rendering is devoted to efficiently skipping empty space. Furthermore the discrete nature of grid representations makes it difficult to apply analytic methods to analyze the data. The maximum along a ray, for example, is often estimated by sampling techniques. A more intuitive solution would be to analytically compute the maximum directly. This would be more exact and more efficient.

In this paper we address these issues and propose a new primitive for volumetric data modelling, processing, and rendering: Volume dots (Vots). Vots are a functional representation of sample-based volumetric data. A Vot comprises the coefficients of a Taylor series expansion, which describes the underlying data of a given region. This approach converts a discrete representation into an implicit representation, and therefore allows to exploit the advantages of analytically processing the data. Vots are a more intuitive and high-level description of the data. They enable the application of focus and context strategies to visualize data and allow to represent regions with different levels of detail. Vots also allow to leverage resources where they are needed, because they can be placed at any position.

The Vot representation opens new ways of data processing, it does not intend to replace the conventional representations. Vots are not well suited for very complex datasets with a high level of variation among the samples, in which every sample is of equal importance. However, Vots work

† {grimm | bruckner | kanitsar | groeller}@cg.tuwien.ac.at

well for volumetric data in which only parts of the volume are of great importance.

This paper is structured as follows: Section 2 surveys related work. Section 3 presents the general Vots data structure. Section 4 describes the Vot generation. Section 5 shows Maximum Intensity Projection based on Vots as an example application. In Section 6 results are presented and discussed. Finally in Section 7 we conclude our work and present ideas for future work.

2. Related Work

Recently more and more research is focused on point-based primitives for representation, modeling, processing, and rendering. The main reason for this is the increasing amount of data due to more advanced acquisition devices. Common representations reach their limits in the sense of performance and usability, therefore new ways of data representations have to be exploited. Our research is also focused on such a point-based representation and is mainly inspired by the following work:

Levoy [LW85] first proposed points as a rendering primitive in the mid-eighties. Following this idea, Pfister et al. [PZvBG00] discussed Surfels as a powerful paradigm to efficiently render complex geometric objects at interactive frame rates. Unlike classical surface discretizations, i.e., triangles or quadrilateral meshes, surfels are point primitives without explicit connectivity. Welsh et al. [WM03] present an algorithm that uses wavelets to convert regular sampled point data to an irregular point hierarchy without reducing the precision of the data. Alexa et al. [ABCO*01] propose a definition of a smooth manifold surface from a set of points close to the original surface. It is based on local maps from differential geometry, approximated by the method of moving least squares. Carr et al. [CBC*01] propose to use Radial Basis Functions (RBF) to reconstruct surfaces from point-cloud data. Surfaces are defined implicitly as the zero set of a RBF. Hopf et al. [HE03] propose a hierarchial splatting algorithm to visualize very large scattered point data at interactive frame-rates. Expensive re-sampling of the data is hereby avoided. Qu et al. [QKSK03] propose a rendering primitive called O-Buffers. It is a flexible structure that stores the positions of arbitrarily distributed samples relative to a regular grid. Rössl et al. [RZNS03] present a new approach to reconstruct non-discrete models from gridded volume samples. As a model, they use quadratic, trivariate super splines on a uniform tetrahedral partition. Csebfalvi et al. [CSK03] propose a volume-rendering technique based on Monte Carlo integration. A point cloud of random samples is generated using a normalized continuous reconstruction of the volume as a probability density function. This point cloud is then projected onto the image plane. Lu et al. [LME*02] present a framework for an interactive direct volume illustration system that simulates traditional stipple drawing. Xie at al. [XWH*03] address the problem of surface reconstruction of highly noisy point clouds. They fit at each sample point a quadric field which are then blended together to produce a pseudo-signed distance field. Turk et al. [TO02] introduce new techniques for modelling with interpolating implicit surfaces. A 3D implicit function is created using a variational scattered data interpolation approach. The result surface is described by an iso-surface of this function. Ohtake et al. [OBA*03] investigate a shape representation, the multi-level partition of unity implicit surface, that allows to construct surface models from sets of points. There are also approaches which propose hybrid solutions. For example Wilson et al. [WMC02] propose to mix conventional hardware assisted texture-based volume rendering with point-based rendering.

We extend and integrate these ideas and present a new primitive for volumetric data modelling, processing, and rendering, comprising an efficient compact representation of the underlying volumetric data.

3. Vots Data-Structure

Many volumetric datasets can be seen as a volumetric scalar function $f : \mathcal{U} \subseteq \Re^3 \rightarrow \Re$, where $\mathcal{U}$ is the domain of f. With Vots we propose a piece-wise representation $\tilde{f}_i : V_i \subseteq \mathcal{U} \rightarrow \Re$ of the volumetric scalar function f. V_i with $\bigcup_{i=1..N} V_i \subseteq \mathcal{U}$ partitions the underlying space $\mathcal{U}$. Unimportant areas of $\mathcal{U}$, e.g., background areas are omitted. The scalar function over set V_i is represented by an individual Vot $\mathcal{V}_i$. Analogous to a Taylor series expansion all the relevant information for local function reconstruction is concentrated at a specific point P_i within a Vot $\mathcal{V}_i$. Relevant information includes the function value and higher order derivatives, such as gradient and Hessian matrix, at this point P_i. Vots are thus a set of points $\{P_1, P_2, ..., P_N\}$ in $\Re^3$. Each point $P_i = (P_i{}^x, P_i{}^y, P_i{}^z)$ constitutes a Taylor expansion point and locally represents the scalar volume function via the Taylor series expansion.

In general, the Taylor series expansion is defined as:

$$f(P+\Delta P) = \sum_{|\alpha|\leq N} \frac{1}{\alpha!} \partial^\alpha f(P) \Delta P^\alpha + R$$

$$R = \sum_{|\alpha|=N+1} \frac{1}{\alpha!} \partial^\alpha f(P+\theta\Delta P) \Delta P^\alpha,\ \theta \in [0,1]$$

with:

$$\begin{array}{lcl} \alpha & \in & \{x,y,z\}^* \\ \partial^\alpha & = & \frac{\partial^{\alpha_x}}{\partial x^{\alpha_x}} \frac{\partial^{\alpha_y}}{\partial y^{\alpha_y}} \frac{\partial^{\alpha_z}}{\partial z^{\alpha_z}} \\ \alpha! & = & \alpha_x! \alpha_y! \alpha_z! \\ P^\alpha & = & \underbrace{P^x \dots P^x}_{\alpha_x} \underbrace{P^y \dots P^y}_{\alpha_y} \underbrace{P^z \dots P^z}_{\alpha_z} \end{array}$$

Hereby α_μ denotes the number of occurrences of character μ in string α. For Vots, only terms of the Taylor series expansion up to a specific degree N are taken into account. We

approximate the Taylor series expansion by:

$$f(P+\Delta P) \approx \tilde{f}(P+\Delta P) = \sum_{|\alpha|\leq N} \frac{1}{\alpha!} \partial^{\alpha} \tilde{f}(P) \Delta P^{\alpha} \qquad (1)$$

Due to increasing storage demands and computational complexity with higher degrees, a degree of two or three is a good choice from a practical point of view. The derivatives up to degree three are given as:

$$\begin{aligned}
\nabla \tilde{f}(P) &= \begin{pmatrix} \tilde{f}_x \\ \tilde{f}_y \\ \tilde{f}_z \end{pmatrix} \\
H_{\tilde{f}}(P) &= \begin{pmatrix} \tilde{f}_{xx} & \tilde{f}_{xy} & \tilde{f}_{xz} \\ \tilde{f}_{yx} & \tilde{f}_{yy} & \tilde{f}_{yz} \\ \tilde{f}_{zx} & \tilde{f}_{zy} & \tilde{f}_{zz} \end{pmatrix} \\
T_{\tilde{f}}(P) &= \begin{pmatrix} \tilde{f}_{xxx} & \tilde{f}_{xyx} & \tilde{f}_{xzx} \\ \tilde{f}_{yxx} & \tilde{f}_{yyx} & \tilde{f}_{yzx} \\ \tilde{f}_{zxx} & \tilde{f}_{zyx} & \tilde{f}_{zzx} \end{pmatrix}, \\
&\quad \begin{pmatrix} \tilde{f}_{xxy} & \tilde{f}_{xyy} & \tilde{f}_{xzy} \\ \tilde{f}_{yxy} & \tilde{f}_{yyy} & \tilde{f}_{yzy} \\ \tilde{f}_{zxy} & \tilde{f}_{zyy} & \tilde{f}_{zzy} \end{pmatrix}, \\
&\quad \begin{pmatrix} \tilde{f}_{xxz} & \tilde{f}_{xyz} & \tilde{f}_{xzz} \\ \tilde{f}_{yxz} & \tilde{f}_{yyz} & \tilde{f}_{yzz} \\ \tilde{f}_{zxz} & \tilde{f}_{zyz} & \tilde{f}_{zzz} \end{pmatrix}
\end{aligned}$$

Hereby ∇ denotes the gradient, H denotes the Hessian matrix, and T the tensor of third partial derivatives. The Hessian matrix, as well as the tensor of the third derivatives are symmetric. This property can be exploited for storage optimization.

Since each Vot $\mathcal{V}_i$ represents a certain neighborhood of the volumetric scalar function f, we also define a validity area V_i. This area V_i can be of arbitrary shape. From a practical point of view convex shapes such as spheres, ellipsoids, boxes, k-dops, etc. are advantageous. Each of these validity areas V_i is defined in such a way that for a given error ε (which might be zero) it holds:

$$\forall (P_i + \Delta P_i) \subseteq V_i : |f(P_i+\Delta P_i) - \tilde{f}(P_i+\Delta P_i)| < \varepsilon. \qquad (2)$$

A basic Vot $\mathcal{V}_i$ consists of:

- Position P_i.
- $(\partial^{\alpha} \tilde{f}(P_i))_{|\alpha|\leq N}$
- Validity area V_i for a given ε.

Vot properties can be extended to include attributes such as time-step, importance, etc. Vots represent the underlying volume data with data centric importance. Important data areas (i.e., large function variation) are represented with many small Vots (small validity areas V_i). Homogenous areas are represented with just a few large Vots. In addition the user may locally vary the approximation error ε, allowing a user-centric importance sampling. Focus areas are represented with $\varepsilon = 0$. The context areas might have large errors, which could, for example, increase with distance to the focus area. The Vots data structure concentrates the information where data needs it and the user wants it.

4. Vot Generation

Before presenting a general method for Vot construction we first show, for illustration purposes, how to directly obtain a Vot representation for a 2x2x2 cell of a rectilinear grid. As reference reconstruction we assume trilinear interpolation within the cell.

4.1. Vot Generation For A Cell

A cell is given as eight pairs: $(P_{ijk}, f_{P_{ijk}})_{i,j,k\in\{0,1\}}$. Hereby P_{ijk} denotes a position at one of the corners of the cell, and $f_{P_{ijk}}$ the corresponding function value. Furthermore we assume a trilinear reconstruction filter. The expansion point needed for the Taylor expansion is defined as the center of the cell by:

$$P = \frac{1}{8} \sum P_{ijk}$$

The terms $\tilde{f}(P)$, $\nabla \tilde{f}(P)$, $H_{\tilde{f}}(P)$, and $T_{\tilde{f}}(P)$ for the Taylor series expansion up to degree three can directly be specified by:

$$\begin{aligned}
\tilde{f}(P) &= \tfrac{1}{8} \textstyle\sum f_{P_{ijk}} \\
\nabla \tilde{f}(P) &= \tfrac{1}{4} \begin{pmatrix} \sum_{j,k} f_{P_{1jk}} - \sum_{j,k} f_{P_{0jk}} \\ \sum_{i,k} f_{P_{i1k}} - \sum_{i,k} f_{P_{i0k}} \\ \sum_{i,j} f_{P_{ij1}} - \sum_{i,j} f_{P_{ij0}} \end{pmatrix} \\
H_{\tilde{f}}(P) &= \tfrac{1}{2} \begin{pmatrix} 0 & \tilde{f}_{xy} & \tilde{f}_{xz} \\ \tilde{f}_{xy} & 0 & \tilde{f}_{yz} \\ \tilde{f}_{xz} & \tilde{f}_{yz} & 0 \end{pmatrix} \\
T_{\tilde{f}}(P) &= \tilde{f}_{xyz}
\end{aligned} \qquad (3)$$

where

$$\begin{aligned}
\tilde{f}_{xy} = &\sum_{ijk\in\{000,001,110,111\}} f_{P_{ijk}} - \\
&\sum_{ijk\in\{010,011,100,101\}} f_{P_{ijk}} \\
\tilde{f}_{xz} = &\sum_{ijk\in\{000,010,101,111\}} f_{P_{ijk}} - \\
&\sum_{ijk\in\{001,011,100,110\}} f_{P_{ijk}} \\
\tilde{f}_{yz} = &\sum_{ijk\in\{000,011,100,111\}} f_{P_{ijk}} - \\
&\sum_{ijk\in\{001,010,101,110\}} f_{P_{ijk}}
\end{aligned}$$

and

$$\begin{aligned}
\tilde{f}_{xyz} = &\sum_{ijk\in\{001,010,100,111\}} f_{P_{ijk}} - \\
&\sum_{ijk\in\{000,011,101,110\}} f_{P_{ijk}}
\end{aligned}$$

Every data value within the cell can be reconstructed by evaluating Equation 1. The Vots representation of a cell requires altogether eight values. $\tilde{f}(P) \rightsquigarrow 1$, $\nabla \tilde{f}(P) \rightsquigarrow 3$, $H_{\tilde{f}}(P) \rightsquigarrow 3$, and $T_{\tilde{f}}(P) \rightsquigarrow 1$. All the remaining values are either redundant due the symmetry of the Hessian matrix or are zero. In terms of storage requirement a Vot representation is equal to a cell representation. However, in a regular grid, grid points are reused for (eight) neighboring cells. A straightforward

conversion of a regular grid into a Vot structure would therefore increase the storage requirements by a factor of eight. A closer look reveals that only every other cell (in each of the three spatial directions) must be represented by a Vot. The function in cells which do not contain a Vot can be exactly reconstructed from the neighboring Vots. Such an approach would not change the storage requirements. With the Vots representation we have a more intuitive specification of the underlying function.

4.2. General Vot Generation

One of the major reasons for the introduction of Vots is to be able to leverage resources where they are needed. Homogeneous or non important regions should be represented just by few resources. On the other hand, inhomogeneous and important regions should be represented by an adequate amount of resources. The amount is defined by the desired accuracy. Vots provide this feature, as they can be placed at any arbitrary position.

In the following we present an approach to generate a Vot for a given set of m scattered data points $Q_j \in \Re^3$ with data values f_{Q_j}. We assume that a Vot uses the Taylor series expansion up to degree $N = 3$, as shown in Equation 1. This can be extended to arbitrary degrees straightforwardly. To generate the Vots, we use an approach which is similar to the linear regression approach for normal vector estimation used in [NCKG00]. To be able to apply this approach we define the mean square error of the fitting process as:

$$E(\cdots) = \sum_{j=1}^{m} (\tilde{f}(Q_j) - f_{Q_j})^2 \tag{4}$$

Hereby f_{Q_j} denotes the function values given at the m scattered points, $\tilde{f}(Q_j)$ denotes the function value reconstructed by the approximated Taylor series $\tilde{f}$ at Q_j, and E is the sum of the squared differences between the original values and the reconstructed values. As Taylor series expansion point we choose the center of gravity:

$$P = \frac{1}{m} \sum Q_j$$

The unknown variables of E are:

$$\tilde{f}, \tilde{f}_i, \tilde{f}_{ij}, \tilde{f}_{ijk}$$

where $i, j, k \in \{x, y, z\}$ and $\tilde{f}_i, \tilde{f}_{ij}, \tilde{f}_{ijk}$ denote the partial derivatives $\partial^i \tilde{f}, \partial^{ij} \tilde{f}, \partial^{ijk} \tilde{f}$. The number of unknowns can be reduced due to symmetry of the Hessian matrix H and the the tensor T of the third derivatives to 20 unknowns. Furthermore we define:

$$\Delta Q_j = Q_j - P = ({Q_j}^x - P^x, {Q_j}^y - P^y, {Q_j}^z - P^z)$$

The minimum of the error function E is determined by taking the partial derivatives with respect to the unknowns and setting these partial derivatives to zero. To achieve this, $\tilde{f}(Q_j)$ is substituted according to Equation (1) with:

$$\sum_{|\alpha| \le 3} \frac{1}{\alpha!} \partial^\alpha \tilde{f}(P) \Delta {Q_j}^\alpha$$

The derivatives are given as:

$$\begin{aligned}
\frac{\partial E}{\partial \tilde{f}} &= 2 \sum_{j=1}^{m} \left(\sum_{|\alpha| \le 3} \tfrac{1}{\alpha!} \partial^\alpha \tilde{f}(P) \Delta {Q_j}^\alpha - f_{Q_j} \right) \\
\frac{\partial E}{\partial \tilde{f}_x} &= 2 \sum_{j=1}^{m} \left(\sum_{|\alpha| \le 3} \tfrac{1}{\alpha!} \partial^\alpha \tilde{f}(P) \Delta {Q_j}^\alpha - f_{Q_j} \right) \Delta {Q_j}^x \\
&\vdots \\
\frac{\partial E}{\partial \tilde{f}_{xy}} &= 2 \sum_{j=1}^{m} \left(\sum_{|\alpha| \le 3} \tfrac{1}{\alpha!} \partial^\alpha \tilde{f}(P) \Delta {Q_j}^\alpha - f_{Q_j} \right) \Delta {Q_j}^x \Delta {Q_j}^y \\
&\vdots \\
\frac{\partial E}{\partial \tilde{f}_{xxy}} &= 2 \sum_{j=1}^{m} \left(\sum_{|\alpha| \le 3} \tfrac{1}{\alpha!} \partial^\alpha \tilde{f}(P) \Delta {Q_j}^\alpha - f_{Q_j} \right) (\Delta {Q_j}^x)^2 \Delta {Q_j}^y \\
&\vdots
\end{aligned} \tag{5}$$

Setting the derivatives to zero, leads to the following system of linear equations:

$$M \begin{pmatrix} \tilde{f} \\ \tilde{f}_x \\ \vdots \\ \tilde{f}_{xy} \\ \vdots \\ \tilde{f}_{xxy} \\ \vdots \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{m} f_{Q_j} \\ \sum_{j=1}^{m} f_{Q_j} \Delta {Q_j}^x \\ \vdots \\ \sum_{j=1}^{m} f_{Q_j} \Delta {Q_j}^x \Delta {Q_j}^y \\ \vdots \\ \sum_{j=1}^{m} f_{Q_j} (\Delta {Q_j}^x)^2 \Delta {Q_j}^y \\ \vdots \end{pmatrix} \tag{6}$$

where M is the 20×20 matrix resulting from the following sum of vector direct products:

$$M = \sum_{j=1}^{m} \begin{pmatrix} 1 \\ \Delta {Q_j}^x \\ \Delta {Q_j}^y \\ \Delta {Q_j}^z \\ (\Delta {Q_j}^x \Delta {Q_j}^x)/2 \\ \Delta {Q_j}^x \Delta {Q_j}^y \\ \Delta {Q_j}^x \Delta {Q_j}^z \\ (\Delta {Q_j}^y \Delta {Q_j}^y)/2 \\ \Delta {Q_j}^y \Delta {Q_j}^z \\ (\Delta {Q_j}^z \Delta {Q_j}^z)/2 \\ (\Delta {Q_j}^x \Delta {Q_j}^x \Delta {Q_j}^x)/6 \\ 3 \cdot (\Delta {Q_j}^x \Delta {Q_j}^x \Delta {Q_j}^y)/6 \\ 3 \cdot (\Delta {Q_j}^x \Delta {Q_j}^x \Delta {Q_j}^z)/6 \\ 3 \cdot (\Delta {Q_j}^x \Delta {Q_j}^y \Delta {Q_j}^y)/6 \\ 3 \cdot (\Delta {Q_j}^z \Delta {Q_j}^x \Delta {Q_j}^z)/6 \\ \Delta {Q_j}^x \Delta {Q_j}^y \Delta {Q_j}^z \\ (\Delta {Q_j}^y \Delta {Q_j}^y \Delta {Q_j}^y)/6 \\ 3 \cdot (\Delta {Q_j}^y \Delta {Q_j}^y \Delta {Q_j}^z)/6 \\ 3 \cdot (\Delta {Q_j}^y \Delta {Q_j}^z \Delta {Q_j}^z)/6 \\ (\Delta {Q_j}^z \Delta {Q_j}^z \Delta {Q_j}^z)/6 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ \Delta {Q_j}^x \\ \Delta {Q_j}^y \\ \Delta {Q_j}^z \\ (\Delta {Q_j}^x \Delta {Q_j}^x)/2 \\ \Delta {Q_j}^x \Delta {Q_j}^y \\ \Delta {Q_j}^x \Delta {Q_j}^z \\ (\Delta {Q_j}^y \Delta {Q_j}^y)/2 \\ \Delta {Q_j}^y \Delta {Q_j}^z \\ (\Delta {Q_j}^z \Delta {Q_j}^z)/2 \\ (\Delta {Q_j}^x \Delta {Q_j}^x \Delta {Q_j}^x)/6 \\ 3 \cdot (\Delta {Q_j}^x \Delta {Q_j}^x \Delta {Q_j}^y)/6 \\ 3 \cdot (\Delta {Q_j}^x \Delta {Q_j}^x \Delta {Q_j}^z)/6 \\ 3 \cdot (\Delta {Q_j}^x \Delta {Q_j}^y \Delta {Q_j}^y)/6 \\ 3 \cdot (\Delta {Q_j}^z \Delta {Q_j}^x \Delta {Q_j}^z)/6 \\ \Delta {Q_j}^x \Delta {Q_j}^y \Delta {Q_j}^z \\ (\Delta {Q_j}^y \Delta {Q_j}^y \Delta {Q_j}^y)/6 \\ 3 \cdot (\Delta {Q_j}^y \Delta {Q_j}^y \Delta {Q_j}^z)/6 \\ 3 \cdot (\Delta {Q_j}^y \Delta {Q_j}^z \Delta {Q_j}^z)/6 \\ (\Delta {Q_j}^z \Delta {Q_j}^z \Delta {Q_j}^z)/6 \end{pmatrix}^T \tag{7}$$

The inversion of matrix M produces the solution

for the unknown variables. The error is calculated by $\varepsilon = max_j|\tilde{f}(Q_j) - f_{Q_j}|$. The mechanism described allows the construction of a Vot for a given input set of points $Q_j, (j = 1, ..., m)$.

4.3. Vot-Space

Vots allow an importance-based representation of volume data. To achieve this they abandon the implicit connectivity information of regular grids. The most basic question the data structure has to answer is: Given an arbitrary point P find the corresponding Vot $\mathcal{V}_i$ $(P \in V_i)$ so that the function value at position P can be determined. Depending on the shape of the validity area V_i efficient indexing structures from computational geometry, such as range trees, interval trees, octrees and bounding volume hierarchies can be used to accelerate this search. Application dependent, indexing structures might also help to quickly address, for example, all the Vots whose gradient is within a certain magnitude or direction range.

A Vot-Space $(\mathcal{V}_j, \mathcal{I})$ comprises a set of Vots $\mathcal{V}_j$ and a set of indexing structures $\mathcal{I}$. $\mathcal{I}$ contains at least I^*, which is an unsorted list of all Vots. It is used to address each Vot. For some application this simple indexing structure is sufficient, see Section 5. It is application specific on which kind of indexing structure a Vot-Space depends on.

5. Application

To give a proof of concept of our new data structure, we show an application of Vots. We present Maximum Intensity Projection of a Vot-Space. Maximum Intensity Projection [MGK99] is a technique that displays the maximum scalar value seen through each image pixel. By depicting the maximum data value, high intensity structures contained in the data are captured. A straight-forward method for calculating Maximum Intensity Projection is to perform ray casting and search for the maximum sample value along each ray. We use this visualization method to illustrate the advantages of Vots. Instead of using sampling, we analytically determine the maximum of a viewing ray within the validity area of a Vot.

5.1. Vots Generation

As input we assume a rectilinear grid. It is given as a set of pairs $G = \{(P_j, f_{P_j}),\ j = \{1, ..., N\}\}$ where $P_j = ({P_j}^x, {P_j}^y, {P_j}^z)$ defines a position within the grid and f_{P_j} the corresponding function value $f(P_j)$. The task is to find a small number of Vots $\mathcal{V}_j$ which covers completely the underlying volumetric data. Reconstruction should be bound by an error ε, as given in Equation 2.

We use a growing approach. As growing criteria we define the function $\mathcal{F}(T)$ as follows:

$$\mathcal{F}(T) = max_j|\tilde{f}(Q_j) - f_{Q_j}|, Q_j \in T \qquad (8)$$

Figure 1: *Vot density distribution of lobster dataset: 114 665 Vots generated from 445 568 input samples.*

where $T \subseteq G$. $\mathcal{F}$ implements Equation 6, it solves the linear equation system and returns the maximal error. Furthermore, for simplicity we assume box-shaped validity areas of the Vots. As possible start positions of Vots we choose the positions given by G. The validity area of each Vot is initially cell-sized, covering eight adjacent grid positions. At this point, we iteratively start the growing process of each Vot by increasing one of its validity area dimensions either in the positive or negative direction. In every step we test the error computed by $\mathcal{F}$. The current input set T is defined by all grid positions which lie within the increased validity area. According to the outcome of $\mathcal{F}$ and the given ε-bound we either keep the increased validity area as new area or we keep the old validity area. A Vot grows until error $\mathcal{F}(T)$ is larger than ε. According to Equation 8 the error is just tested at positions Q_j.

The result of this process is a set of the largest Vots for every grid position. The next step is to find a minimal subset of the set of Vots which completely cover the underlying volumetric data. To achieve this we assign to each Vot $\mathcal{V}_j$ a cover weight $\mathcal{W}_j$. The weights initially correspond in size to the validity areas V_j. The Vots are sorted according to their weights $\mathcal{W}_j$ in decreasing order. We take the Vot with the largest $\mathcal{W}$ and assign it to the subset of the minimal Vots. We determine the number of grid points this Vot would cover in the grid. Only those grid positions are counted, which are not covered by other Vots in the current small subset. We adapt the weights $\mathcal{W}_j$ of all the remaining Vots, according to the number of grid positions they could cover, which are not already covered by other Vots out of the current small subset. Then the Vots are re-sorted and the process starts all

over. Once the complete grid is covered the process stops and we have found a small subset of Vots which defines our Vot-Space with a given indexing structure I^*.

Figure 1 and Figure 2 show the Vot distribution of typical data sets. Dark areas correspond to high Vot density and bright areas to low Vot density.

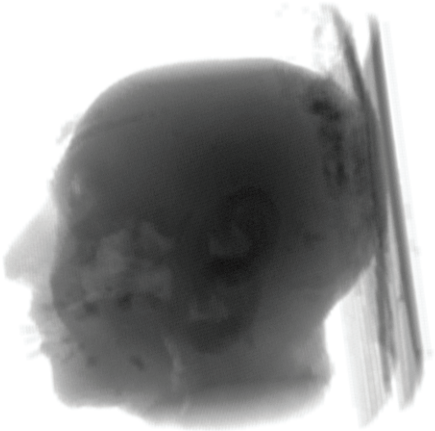

Figure 2: *Vot density distribution of UNC head dataset: 570 690 Vots generated from 1 746 360 input samples.*

5.2. Maximum Intensity Projection of a Vot-Space

For Maximum Intensity Projection the following question arises: Given a Vot $\mathcal{V}$ and a viewing direction, how do we compute the maximum along this ray. To answer this, we look again at the Taylor series expansion of one Vot, as given in Equation 1. For simplicity reasons we choose $N = 2$. From this follows that the Taylor series is given as:

$$\tilde{f}(P+\Delta P) = \tilde{f}(P) + \nabla\tilde{f}(P)\Delta P + \tfrac{1}{2}\Delta P\, H_{\tilde{f}}(P)\, \Delta P \qquad (9)$$

A ray r is given as:

$$r(t) = S + tD$$

To determine the extreme the ray equation is set into Equation 9 and the first derivative is taken. Furthermore we define $\Delta P_S := S - P$ and obtain:

$$\partial^1 \tilde{f}(S+tD) = a_0 + a_1 t$$

with

$$\begin{aligned} a_0 = \;& \tilde{f}_x D^x + \tilde{f}_y D^y + \tilde{f}_z D^z + \\ & \tilde{f}_{xx} D^x \Delta P_S{}^x + \tilde{f}_{yy} D^y \Delta P_S{}^y + \tilde{f}_{zz} D^z \Delta P_S{}^z + \\ & \tilde{f}_{xy}(D^x \Delta P_S{}^y + D^y \Delta P_S{}^x) + \tilde{f}_{xz}(D^x \Delta P_S{}^z + D^z \Delta P_S{}^x) + \\ & \tilde{f}_{yz}(D^y \Delta P_S{}^z + D^z \Delta P_S{}^y) \\ a_1 = \;& \tilde{f}_{xx}(D^x)^2 + \tilde{f}_{yy}(D^y)^2 + \tilde{f}_{zz}(D^z)^2 + \\ & 2\tilde{f}_{xy} D^x D^y + 2\tilde{f}_{xz} D^x D^z + 2\tilde{f}_{yz} D^y D^z \end{aligned}$$

The position of the extreme is then determined by setting the first derivative to zero. The resulting equation is solved with respect to t:

$$t = \frac{-a_0}{a_1}$$

To determine wether the extreme is a maximum or minimum, we set $S + tD$ into Equation 9, take the second derivative and obtain:

$$\begin{aligned} \partial^2 \tilde{f}(S+tD) = \;& \tilde{f}_{xx}(D^x)^2 + \tilde{f}_{yy}(D^y)^2 + \tilde{f}_{zz}(D^z)^2 + \\ & 2\tilde{f}_{xy} D^x D^y + 2\tilde{f}_{xz} D^x D^z + 2\tilde{f}_{yz} D^y D^z \end{aligned}$$

The sign of the second derivative determines if the extreme is a maximum or a minimum. Knowing these two derivatives it is straightforward to determine the maximum along a ray within the validity area of a Vot. There are two cases:

1. A maximum within the validity area is found.
2. No maximum within the validity is found, the maximum along the ray occurs at one of the intersection points with the validity area box.

With this method, the maximum scalar value of an arbitrary ray passing through a Vot can be determined. The algorithm now works as follows: The unsorted list of Vots is traversed. For each Vot, its validity region is represented by a polygonal model. For every ray that intersects the Vot, the intersection point S is calculated and the maximum along the ray is computed and stored in an image. There is one image for each visible face of the polygonal model. In our case, the validity area is always box-shaped, therefore at most three faces are visible. Texture mapping is used to transform the images according to the validity area geometry. The images are textured onto the corresponding faces of the model, as illustrated in Figure 3. We utilize the graphics hardware's

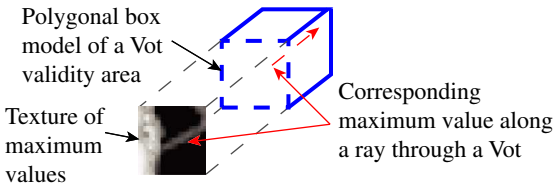

Figure 3: *Maximum Intensity Projection of one Vot.*

capability to perform maximum blending. With our prototype application we rendered two example datasets shown in Figure 4 and Figure 5.

6. Discussion and Results

This work is at an initial stage and still has many issues to address. The brute-force algorithm, of Section 5.1, for converting a rectilinear data set into a Vot-Space representation has high computational complexity. Even for small data-sets, for example of size 128^3, it needs several hours of computation time on a commodity PC. This is mainly due to the applied growing strategy. The larger a validity area V_i becomes,

Figure 4: *Maximum Intensity Projection of Lobster: 114 665 Vots.*

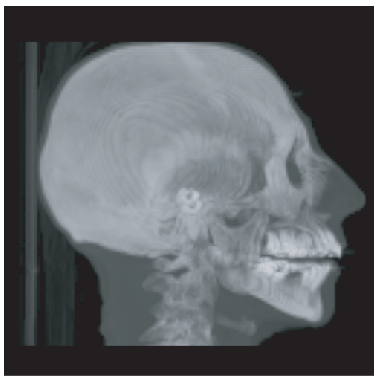

Figure 5: *Maximum Intensity Projection of UNC head: 570 690 Vots.*

Error ε:	0.4096	4.096	40.96	409.6
(a) ⇝ # Vots:	570 690	538 919	333 650	35 079
(b) ⇝ # Vots:	114 665	114 665	112 604	60 353

Table 1: *Number of Vots for different error bounds ε. The error ε defines the maximum allowed deviation with respect to the original samples. Reconstructing data samples in between the original samples can lead to larger errors. The data range interval is [0,4095]. (a) UNC head (126x126x110): 1 746 360 samples. (b) Lobster (118x118x32): 445 568 samples.*

the more number of fitting operations have to be performed. Currently we grow Vots from every grid-position and later on a huge number of redundant Vots are discarded. It is obvious that for example centers of homogeneous regions are better choices to place Vots than inhomogeneous regions. We believe that applying a more sophisticated seeding strategy will considerably reduce the Vots generation effort.

Another issue which needs to be addressed is the size of Vots and the number of Vots needed. In Figure 1 and Figure 2 a quite high number of Vots is used. Such high numbers are only of limited practicability. A Vot with a Taylor series expansion of up to degree three defines a rather rigid underlying volume function with limited shape possibility. Medical data often contains high frequencies and noise. To adhere to the defined error bound the Vots are thus rather limited in size. To obtain bigger Vots or a smaller number of Vots one has to allow a higher error ε. However, a higher error ε leads to discontinuities. Table 1 shows the resulting number of Vots for different error bounds ε. Discontinuity at the borders can be handled through blending of adjacent Vots or of Vots with overlapping validity areas V_i. Overlapping is not an issue in the presented MIP application, as the maximum along a ray remains the same even if it is determined multiple times due to the overlapping. In our current implementation the error function of Equation 8 is evaluated only at sample positions Q_j. To ensure the error bound also in between a finer sampling of this error function is recommended. The overall fitting process remains the same.

Finally a rendering approach of a Vot-Space is needed, which performs at least in the same range as conventional volume rendering approaches. In general it should be possible to apply image- and object-order direct volume rendering. We plan to investigate both. For an image order approach one could send a ray through the Vot-Space, *jumping* from Vot to Vot in the correct visible order while resampling each individual Vot along the ray. The *jumping* corresponds to empty space skipping. For object order rendering one could apply a technique similar to splatting. However, an appropriate interpolation kernel has to be developed as a Vot contains the condensed information of a region instead of one sample position.

7. Conclusion and Future Work

We propose a novel primitive for volumetric data modelling, processing, and rendering. As we move the data representation from a discrete to an implicit representation, a new paradigm is presented. The new function oriented paradigm is a more intuitive and constructive representation of the data. The volumetric data is divided into regions to achieve a more expressive representation. Some Vots represent larger regions than others, but all Vots represent the data in the same way. The size of a Vot can be adjusted by modifying the allowed error bound ε. This allows user-centric importance sampling. Unimportant regions are represented by just a few Vots, while important regions are represented with many Vots. One Vot contains all the information about the volumetric data within a region, thus no explicit connectivity between Vots is necessary for reconstruction. Furthermore, the Vots representation allows to process the data analytically as shown with Maximum Intensity Projection. In general, Vots open a wide range of new data examination approaches. In the future we will continue to explore the new possibilities

and approaches that the new paradigm of Vots introduces. One challenge is to construct Vots from other types of data structures, such as point clouds, unstructured grids, curvilinear grids, etc. Different strategies must be developed to obtain an accurate conversion. Vots provide a starting point for new types of volume processing, new ways of rendering, even exploring new types of visualization based on the condensed information that a Vot contains. It would also be interesting to map Vots onto graphics hardware and analyze their capabilities from that point of view.

A related area we will investigate is a point cloud representation of volume data. Moving Least Square (MLS) will be used to dynamically fit a Vot to an arbitrary re-sampling position. In this case the fitting process of Section 4.2 will be slightly modified as each point is weighted by the distance to the re-sampling position. We believe similar operations as done for MLS fitting of surfaces can also be done for MLS fitting of volumetric functions, such as point cloud thinning, importance-based sorting of the point cloud, etc.

8. Acknowledgements

We would like to thank László Neumann for many helpful discussions. We also would like to thank the Computer Graphics Laboratory, ETH Zürich for the initial fruitful discussions we had at a joined workshop. The work presented in this publication has been funded by the ADAPT project (FFF-804544). ADAPT is supported by *Tiani Medgraph*, Vienna (http://www.tiani.com), and the *Forschungsförderungsfonds für die gewerbliche Wirtschaft*, Austria. See http://www.cg.tuwien.ac.at/research/vis/adapt for further information on this project.

References

[ABCO*01] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Point set surfaces. In *IEEE Visualization 2001* (2001), pp. 21–28.

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3D objects with radial basis functions. In *SIGGRAPH 2001* (2001), ACM, (Ed.), pp. 67–76.

[CSK03] CSEBFALVI B., SZIRMAY-KALOS L.: Monte carlo volume rendering. In *IEEE Visualization 2003* (2003), pp. 449–456.

[HE03] HOPF M., ERTL T.: Hierarchical splatting of scattered data. In *IEEE Visualization 2003* (2003), pp. 433–440.

[LME*02] LU A., MORRIS C. J., EBERT D., RHEINGANS P., HANSEN C.: Non-photorealistic volume rendering using stippling techniques. In *IEEE Visualization 2002* (2002), pp. 211–218.

[LW85] LEVOY M., WHITTED T.: *The Use of Points as Display Primitives*. Tech. Rep. 85-022, The University of North Carolina at Chapel Hill, Department of Computer Science, 1985.

[MGK99] MROZ L., GRÖLLER E., KÖNIG A.: Real-time maximum intensity projection. In *Data Visualization '99*, Eurographics. 1999, pp. 135–144.

[NCKG00] NEUMANN L., CSÉBFALVI B., KÖNIG A., GRÖLLER E.: Gradient estimation in volume data using 4D linear regression. In *Computer Graphics Forum (Eurographics 2000)* (2000), pp. 351–358.

[OBA*03] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H. P.: Multi-level partition of unity implicits. In *Transactions on Graphics* (2003), ACM, (Ed.), pp. 463–470.

[PZvBG00] PFISTER H., ZWICKER M., VAN BAAR J., GROSS M.: Surfels: Surface elements as rendering primitives. In *SIGGRAPH 2000* (2000), ACM, (Ed.), pp. 335–342.

[QKSK03] QU H., KAUFMAN A., SHAO R., KUMAR A.: A framework for sample-based rendering with O-buffers. In *IEEE Visualization 2003* (2003), pp. 441–448.

[RZNS03] RÖSSL C., ZEILFELDER F., NÜRNBERGER G., SEIDEL H.-P.: Visualization of volume data with quadratic super splines. In *IEEE Visualization 2003* (2003), pp. 393–400.

[TO02] TURK G., O'BRIEN J. F.: Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics 21*, 4 (2002), 855–873.

[WM03] WELSH T., MUELLER K.: A frequency-sensitive point hierarchy for images and volumes. In *IEEE Visualization 2003* (2003), pp. 425–432.

[WMC02] WILSON B., MA K.-L., CORMICK P. S. M.: A hardware-assited hybrid rendering technique for interactive volume visualization. In *IEEE Symposium on Volume Visualization and Graphics 2002* (2002), pp. 123–130.

[XWH*03] XIE H., WANG J., HUA J., QIN H., KAUFMAN A.: Piecewiese C^1 continous surface reconstruction of noisy point clouds via local implicit quadric regression. In *IEEE Visualization 2003* (2003), pp. 91–98.