# Global Curvature Analysis and Segmentation of Volumetric Data Sets Using Trivariate B-spline Functions

Octavian Soldea
octavian@cs.technion.ac.il

Gershon Elber
gershon@cs.technion.ac.il

Ehud Rivlin
ehudr@cs.technion.ac.il

Technion-Israel Institute of Technology, Computer Science Department, Haifa, Israel

## Abstract

*This paper presents a scheme to globally compute, bound, and analyze the Gaussian and mean curvatures of an entire volumetric data set, using a trivariate B-spline volumetric representation. The proposed scheme is not only precise and insensitive to aliasing, but also provides a method to* globally *segment the images into volumetric regions that contain convex or concave* (*elliptic*) *iso-surfaces, planar or cylindrical* (*parabolic*) *iso-surfaces, and volumetric regions with saddle-like* (*hyperbolic*) *iso-surfaces, regardless of the value of the iso-surface level. This scheme, which derives a new differential scalar field for a given scalar field, could easily be adapted to other differential properties.*

## 1 Introduction and Previous Work

The availability of a broad variety of volumetric images, in recent years, raises new problems and challenges for the scientific community. While 3D images are typically obtained from 3D laser scanners, volumetric images are commonly obtained from devices such as CT, MRI, and SEM. One common need in both image types is segmentation. Segmented images are employed in vision and image processing or understanding applications. Segmentation techniques of volumetric images are considered, for example, in [16].

One way of approaching the segmentation problem is by computing or estimating differential geometric properties of the analyzed objects [11]. Among the differential properties that are widely used, Gaussian curvature takes a center stage, being a fundamental prescription of an intrinsic surface property. Curvature computation is employed in many of the segmentation algorithms known in the literature and used in practice. Volumetric segmentation processes could exploit the fact that the curvature values of all iso-surfaces are intrinsic to the shape, thus invariant to rotation and translation, among other qualities.

The information provided by curvature analysis could be used in applications where volumetric data is found, from CT medical scans all the way to CT security scans. Such applications could be registration, metamorphosis, and recognition of primitive parts.

Curvature properties of volumetric images were investigated in [9]. The author of [9] presents methods for computing the curvature of piecewise linear two-manifold surfaces and three-manifold graphs of trivariate functions. [9] presents several examples of synthetic objects that were color-coded to follow curvature values. Moreover, several measurements of the errors of the approximation are presented using the root mean square between exact analytic and approximated values over each point of interest of several synthetic images.

Being able to view the boundaries of objects, in general, the human eye experiences difficulties in fully comprehending the geometry of a volumetric image. Ways to provide visual cues have been sought in rendering two-dimensional projections of the volume. One example is [13], in which the author employs volumetric visualization using strokes along lines of curvature. The strokes follow the principal directions, thus enhancing the perception of the image in an intuitive way. The computation of the strokes employs a line integral convolution.

In [10], the authors define a new type of transfer function for direct volume rendering. This new type of transfer function evaluates the principal curvatures at discrete points.

The transfer function associates with each pair of principal curvatures a color, and emphasizes the elliptic, parabolic (both planar and cylindric), and hyperbolic regions of the analyzed objects.

In [15], the authors describe a multi-dimensional transfer function technology that employs measurements of several differential characteristics. The differential characteristics are evaluated using a convolution of the input image with different filters. The authors show results that emphasize valleys and ridges, visualizing surface smoothing, and visualizing the uncertainty of iso-surfaces.

In [20, 19], the authors present a technique for estimating curvature values of iso-intensity surfaces from volumetric data sets. They refer to the principal curvatures $k_1$ and $k_2$ as the largest and second (principal) curvatures, where $|k_1| > |k_2|$. Moreover, the authors define four types of possible registration curves, considering the local minimum or maximum of the largest or next-to-largest curvature, along lines of curvature. The estimation of curvature properties requires the approximation of derivatives in the input images and in [20, 19] this is performed by applying a discrete local Gaussian filter over the volumetric image.

The authors in [23] use a global curvature analysis approach and present a technique for evaluating the curvature and the torsion of 3D vector fields. Several differential characteristics of the 3D vector fields are used for computing iso-surfaces which finally are employed towards the analysis and visualization of the input images. The importance of [23] could also be found in developing a general framework of formulas for computing curvature fields.

The estimation of curvature properties requires the approximation of second order derivatives, a difficult task when discrete, piecewise constant, voxel representations are employed. In order to estimate the second order derivative, any numerical algorithm must consider the tradeoff between the provided accuracy and the time consumed. For instance, there are algorithms, such as the central finite difference approximation to second order derivatives, (for example, see [4]) that can efficiently approximate the second order derivative and provide a certain degree of accuracy. When accuracy is crucial, methods such as Richardson extrapolation (see [4]) are to be used.

In this paper, we present a technique to globally and simultaneously compute the Gaussian curvature for (all iso-surfaces of) volumetric images. The presented scheme offers advantages in its improved accuracy in detecting boundaries of curvature-based segmented regions and in its ability to perform global curvature analysis that is insensitive to aliasing as well as ignorant of a specific iso-level. Given a volumetric data set $f(u, v, w)$, we are able to compute $K(u, v, w)$, a scalar field that represents the Gaussian curvature of the iso-surface at $(u, v, w)$. The introduced computational capability opens the way for a more precise and robust global curvature-based segmentation of volumetric data sets. All contemporary algorithms, including the above and to the best of our knowledge, compute the curvature properties in discrete locations only. In the presented new approach, we are able to compute curvature properties functions defined over each point of the volume.

This paper is organized as follows. In Section 2, we provide some necessary mathematical background. In Section 3, we describe the mechanism of evaluating the Gaussian and mean curvatures used in our presented approach. In Section 4, segmentation of the volume as well as curvature-based iso-surface extractions are considered using the introduced scheme. A few advantages over traditional voxel-based schemes are also presented. In Section 5, several examples of our algorithm applied to volumetric and 3D scanned images are portrayed. Finally, in Section 6, we conclude.

## 2 Background

Consider $f(u, v, w)$, a $C^{(2)}$ trivariate function. In this section, we briefly present the mathematical background necessary to compute the Gaussian curvature of an iso-surface $f(u, v, w) = f_0$. We use the main differential components as in [20, 19].

Given a bivariate function $g(u, v)$, denote by $g_u$ and $g_v$ the two partial derivatives of $g(u, v)$ in the $u$ and $v$ directions, respectively. Similarly, for any trivariate function $h(u, v, w)$, let $h_u$, $h_v$, and $h_w$ be the partial derivatives of $h(u, v, w)$ with respect to the $u$, $v$, and $w$ directions.

¿From the implicit function theorem, there exists a scalar function $S(u, v)$ that prescribes the parametric surface $\mathcal{S}(u, v) = (u, v, S(u, v))$, which is, locally, a parameterization of an iso-surface level $f(u, v, S(u, v)) = f_0$, $f_0$ constant. Then, by differentiating with respect to $u$, we have $f_u(u, v, S(u, v)) + f_w(u, v, S(u, v)) S_u = 0$ and further, one can deduce that $S_u = \Leftrightarrow \frac{f_u}{f_w}$, and $\mathcal{S}_u = \left(1, 0, \Leftrightarrow \frac{f_u}{f_w}\right)$.

Let $E, F,$ and $G$ and $L, M,$ and $N$ be the coefficients of the first and second fundamental forms, respectively, and $n = \mathcal{S}_u \times \mathcal{S}_v$ be the unnormalized normal to $\mathcal{S}$ at $(u, v)$. Then,

$$E = \langle \mathcal{S}_u, \mathcal{S}_u \rangle = 1 + S_u^2 = 1 + \frac{f_u^2}{f_w^2} = \frac{f_u^2 + f_w^2}{f_w^2} = \frac{\widetilde{E}}{f_w^2}. \quad (1)$$

Similarly, other differential components can be computed as well:

$$F = \langle \mathcal{S}_u, \mathcal{S}_v \rangle = \frac{f_u f_v}{f_w^2} = \frac{\widetilde{F}}{f_w^2}, \quad (2)$$

$$G = \langle \mathcal{S}_v, \mathcal{S}_v \rangle = \frac{f_v^2 + f_w^2}{f_w^2} = \frac{\widetilde{G}}{f_w^2}, \quad (3)$$

$$D = EG \Leftrightarrow F^2 = \frac{f_u^2 + f_v^2 + f_w^2}{f_w^2} = \frac{\widetilde{D}}{f_w^2}, \quad (4)$$

$$L = \langle \mathcal{S}_{uu}, \overline{n} \rangle$$

$$
\begin{aligned}
&= \frac{2f_u f_w f_{uw} \Leftrightarrow f_u^2 f_{ww} \Leftrightarrow f_w^2 f_{uu}}{\widetilde{D}^{\frac{1}{2}} f_w^2} \\
&= \frac{\widetilde{L}}{\widetilde{D}^{\frac{1}{2}} f_w^2}, && (5) \\
M &= \langle \mathcal{S}_{uv}, \overline{n} \rangle \\
&= \frac{f_u f_w f_{vw} + f_v f_w f_{uw} \Leftrightarrow f_u f_v f_{ww} \Leftrightarrow f_z^2 f_{uv}}{\widetilde{D}^{\frac{1}{2}} f_w^2} \\
&= \frac{\widetilde{M}}{\widetilde{D}^{\frac{1}{2}} f_w^2}, && (6)
\end{aligned}
$$

and

$$
\begin{aligned}
N &= \langle \mathcal{S}_{vv}, \overline{n} \rangle \\
&= \frac{2f_v f_w f_{vw} \Leftrightarrow f_v^2 f_{ww} \Leftrightarrow f_w^2 f_{vv}}{\widetilde{D}^{\frac{1}{2}} f_w^2} \\
&= \frac{\widetilde{N}}{\widetilde{D}^{\frac{1}{2}} f_w^2}, && (7)
\end{aligned}
$$

where $\overline{n}$ is the normalized surface normal. Then, the Gaussian curvature of an iso-surface of $f(u, v, w)$ equals

$$
K = \frac{LN \Leftrightarrow M^2}{D} = \frac{\widetilde{L}\widetilde{N} \Leftrightarrow \widetilde{M}^2}{D^2 f_w^6} = \frac{\widetilde{L}\widetilde{N} \Leftrightarrow \widetilde{M}^2}{\widetilde{D}^2 f_w^2} = \frac{\widetilde{K}}{\widetilde{D}^2 f_w^2}. \qquad (8)
$$

The values in Equations $(1)$ to $(8)$ are $w$⇔biased and sensitive to a vanishing $f_w$. Processing further, we have,

$$
\begin{aligned}
&K \\
&= \Big( \left(2f_u f_w f_{uw} \Leftrightarrow f_u^2 f_{ww} \Leftrightarrow f_w^2 f_{uu}\right) \\
&\quad \left(2f_v f_w f_{vw} \Leftrightarrow f_v^2 f_{ww} \Leftrightarrow f_w^2 f_{vv}\right) \\
&\quad \Leftrightarrow \left(f_u f_w f_{vw} + f_v f_w f_{uw} \Leftrightarrow f_u f_v f_{ww} \Leftrightarrow f_w^2 f_{uv}\right)^2 \Big) \\
&\quad / \left(\widetilde{D}^2 f_w^2\right) \\
&= \Big( \overline{4 f_u f_v f_w^2 f_{uw} f_{vw}} \Leftrightarrow \underline{2 f_u f_v^2 f_w f_{uw} f_{ww}} \\
&\quad \Leftrightarrow 2 f_u f_w^3 f_{uw} f_{vv} \Leftrightarrow \underline{\underline{2 f_u^2 f_v f_w f_{vw} f_{ww}}} + f_u^2 \widehat{f_v^2} f_{ww}^2 \\
&\quad + f_u^2 f_w^2 f_{vv} f_{ww} \Leftrightarrow 2 f_v f_w^3 f_{uu} f_{vw} + f_v^2 f_w^2 f_{uu} f_{ww} \\
&\quad + f_w^4 f_{uu} f_{vv} \Leftrightarrow f_u^2 f_w^2 f_{vw}^2 \Leftrightarrow f_v^2 f_w^2 f_{uw}^2 \Leftrightarrow f_u^2 \widehat{f_v^2} f_{ww}^2 \\
&\quad \Leftrightarrow f_w^4 f_{uv}^2 \Leftrightarrow \overline{2 f_u f_v f_w^2 f_{uw} f_{vw}} + \underline{\underline{2 f_u^2 f_v f_w f_{vw} f_{ww}}} \\
&\quad + 2 f_u f_w^3 f_{uv} f_{vw} + \underline{2 f_u f_v^2 f_w f_{uw} f_{ww}} \\
&\quad + 2 f_v f_w^3 f_{uv} f_{uw} \Leftrightarrow 2 f_u f_v f_w^2 f_{uv} f_{ww} \Big) / \left(\widetilde{D}^2 f_w^2\right) \\
&= \big(2 f_u f_v f_{uw} f_{vw} + 2 f_u f_w f_{uv} f_{vw} + 2 f_v f_w f_{vu} f_{uw} \\
&\quad \Leftrightarrow 2 f_u f_w f_{uw} f_{vv} \Leftrightarrow 2 f_v f_w f_{uu} f_{vw} \Leftrightarrow 2 f_u f_v f_{uv} f_{ww} \\
&\quad + f_w^2 f_{uu} f_{vv} + f_u^2 f_{vv} f_{ww} + f_v^2 f_{uu} f_{ww} \\
&\quad \Leftrightarrow f_u^2 f_{vw}^2 \Leftrightarrow f_v^2 f_{uw}^2 \Leftrightarrow f_w^2 f_{uv}^2 \big) / \widetilde{D}^2 \\
&= \frac{\widetilde{\widetilde{K}}}{\widetilde{D}^2}, && (9)
\end{aligned}
$$

having similar and cancelled out terms marked together. $K$ is now symmetric with respect to $u, v,$ or $w$. $K(u, v, w)$ is defined for the entire parametric domain of $f(u, v, w)$. Denote by $\overset{\Leftrightarrow}{\nabla} f = \left(\frac{\partial f}{\partial u}, \frac{\partial f}{\partial v}, \frac{\partial f}{\partial w}\right)$ the gradient of $f$, and assume $\overset{\Leftrightarrow}{\nabla} f$ is never zero. Then, given a $(u_0, v_0, w_0)$ location, the iso-surface of $f(u, v, w)$ at $(u_0, v_0, w_0)$ is well-defined. Having a well-defined iso-surface at $(u_0, v_0, w_0)$, $K(u_0, v_0, w_0)$ is also well-defined, as $\widetilde{D}^2 = \left\langle \overset{\Leftrightarrow}{\nabla} f, \overset{\Leftrightarrow}{\nabla} f \right\rangle$ never vanishes. It is important to note that $K$ is a rational, provided $f$ is.

In a similar way to the computation of $K$ in Equations $(8)$ and $(9)$, a formula for $H$, the mean curvature, can be derived as

$$
\begin{aligned}
&H \\
&= \big(2 f_u f_v f_{uv} + 2 f_u f_w f_{uw} + 2 f_v f_w f_{vw} \\
&\quad \Leftrightarrow \left(f_v^2 + f_w^2\right) f_{uu} \Leftrightarrow \left(f_u^2 + f_w^2\right) f_{vv} \\
&\quad \Leftrightarrow \left(f_u^2 + f_v^2\right) f_{ww} \big) / \left(2\widetilde{D}^{\frac{3}{2}}\right) \\
&= \frac{\widetilde{\widetilde{H}}}{2\widetilde{D}^{\frac{3}{2}}}. && (10)
\end{aligned}
$$

Note that $H^2$ is a rational function as well.

## 3 Curvatures of Iso-Surfaces of Trivariate B-spline Functions

Let $B_{i,k,\boldsymbol{\tau}}(t)$ be the $i$th B-spline blending function of degree $k$ defined over knot sequence $\boldsymbol{\tau}$ [2]. Then, consider function

$$
f(u, v, w) = \sum_{i=0}^{n_u} \sum_{j=0}^{n_v} \sum_{l=0}^{n_w} p_{i,j,l} \, B_{i,k_u,\boldsymbol{\tau}_u}(u) \, B_{j,k_v,\boldsymbol{\tau}_v}(v) \, B_{l,k_w,\boldsymbol{\tau}_w}(w), \quad (11)
$$

as a trivariate B-spline function, with $\chi = (n_u + 1)(n_v + 1)(n_w + 1)$ scalar coefficients $p_{i,j,l}$. Hereafter, we will simply employ $B_i(t)$ or $B_{i,k}(t)$ to denote $B_{i,k,\boldsymbol{\tau}}(t)$ whenever the degree and the knot vector can be inferred from the context. Given a regular, piecewise constant volumetric data set, one can treat it as a piecewise constant B-spline trivariate. Moreover, a piecewise trilinear B-spline trivariate will also interpolate that volumetric data set by simply using the voxels' data values as the $p_{i,j,l}$ coefficients of the trivariate. For higher order trivariate functions, the result is only an approximation when $p_{i,j,l}$ are the coefficients. Hence, in practice, two options are available. Either solve an interpolation problem, fitting $f(u, v, w)$ to the original piecewise constant data, or alternatively, provide a bound on the error of the approximation, when using the voxels' data values as the $p_{i,j,l}$ coefficients.

Consider $\tau_i \in \boldsymbol{\tau}_u$, a single interior knot in the $u$ direction such that $\tau_i < \tau_{i+1}$, and similarly for $\tau_j \in \boldsymbol{\tau}_v$, $\tau_j < \tau_{j+1}$

and $\tau_l \in \boldsymbol{\tau}_w,\ \tau_l < \tau_{l+1}$. Then, in each polynomial sub-domain $\mathcal{D}_{i,j,l} : [\tau_i, \tau_{i+1}) \times [\tau_j, \tau_{j+1}) \times [\tau_l, \tau_{l+1})$ of the parametric space of $f$, the Gaussian curvature, $K$, is a rational function in $p_{i,j,l}$ and in $(u, v, w)$, where $u \in [\tau_i, \tau_{i+1})$, $v \in [\tau_j, \tau_{j+1})$, $w \in [\tau_l, \tau_{l+1})$. In this section, we will describe how to efficiently compute the Gaussian curvature $K(u, v, w)$, in two steps. First, in Section 3.1, we evaluate the numerator and denominator of $K$ as a trivariate Bézier representation in each sub-domain, $\mathcal{D}_{i,j,l}$. Then, in a second step that is briefly described in Section 3.2, we merge the rational form of $K$ in all the polynomial sub-domains into a simple B-spline trivariate function representation, over the entire domain of $f(u, v, w)$.

## 3.1 Evaluation of $K$ and $H$ as Bézier Forms

We seek to define $K$ using the coefficients $p_{i,j,l}$ of $f(u, v, w)$ for a single sub-domain $\mathcal{D}_{i,j,l}$ and compute the numerator and denominator of the Gaussian curvature following Equation $(9)$. The expression for $K$ as a function of the coefficients $p_{i,j,l}$ was obtained with the aid of the Maple [3] symbolic manipulation program and is too long to provide as part of this work. One can find this symbolic code in [1]. We use a symbolic interpolation process to convert the result to a Bézier form. In a similar way to $K$, $H^2$ can be evaluated as well.

Let $\theta_{i,n}(t) = \begin{pmatrix} n \\ i \end{pmatrix} (1 \Leftrightarrow t)^{n-i} t^i$ be the Bernstein-Bézier basis function of degree $n$. Consider $f$ for one sub-domain $\mathcal{D}_{i,j,l}$ and let $\psi(u, v, w)$ be one of these differential components, defined over $\mathcal{D}_{i,j,l}$, as presented in Equations $(1) \Leftrightarrow (8)$. $f$ and $\psi$ are both polynomials.

Let $o_u, o_v$, and $o_w$ be the degrees of $\psi(u, v, w)$ in $u$, $v$, and $w$, respectively, following Table 1, and let

$$\begin{aligned} &\theta(u, v, w) \\ &= \sum_{i=0}^{o_u} \sum_{j=0}^{o_v} \sum_{l=0}^{o_w} q_{i,j,l} \theta_{i,o_u}(u)\, \theta_{j,o_v}(v)\, \theta_{l,o_w}(w), \end{aligned} \quad (12)$$

be a Bézier polynomial function of the same degree. We seek a Bézier representation for the polynomial function of $\psi$ and find it by the uniqueness of the polynomial representation and symbolic interpolation constraints at $\mathcal{O} = (o_u + 1)(o_v + 1)(o_w + 1)$ unique parameter values.

In brief, pre-evaluate $K\left(\frac{i}{o_u}, \frac{j}{o_v}, \frac{l}{o_w}\right)$ for all $i \in \{0..o_u\}$, $j \in \{0..o_v\}$, and $l \in \{0..o_w\}$, so that we get $\mathcal{O}$ equations in $p_{i,j,l}$ only. Then, and once given a specific $f$, the $p_{i,j,l}$ coefficients are substituted in. The value of $K$ at these $\mathcal{O}$ locations is used to derive the Bézier form, $(12)$, yielding the $q_{i,j,l}$ coefficients. The details of this process follow.

Assume that $f(u, v, w)$ is a piecewise polynomial function of degrees $k_u, k_v, k_w$. Table $1$ summarizes the degrees of the different differential terms leading to $K$, following

| Component | Degree in $u$ | Degree in $v$ | Degree in $w$ | Degree in $p_{i,j,l}$ coefficients |
|---|---|---|---|---|
| $f(u,v,w)$ | $k_u$ | $k_v$ | $k_w$ | 1 |
| $f_u(u,v,w)$ | $k_u - 1$ | $k_v$ | $k_w$ | 1 |
| $f_{uu}(u,v,w)$ | $k_u - 2$ | $k_v$ | $k_w$ | 1 |
| $f_{uv}(u,v,w)$ | $k_u - 1$ | $k_v - 1$ | $k_w$ | 1 |
| $\widetilde{D} = f_u^2 + f_v^2 + f_w^2$ | $2k_u$ | $2k_v$ | $2k_w$ | 2 |
| $\widetilde{E} = f_u^2 + f_w^2$ | $2k_u$ | $2k_v$ | $2k_w$ | 2 |
| $\widetilde{L} = 2f_u f_w f_{uw} - f_u^2 f_{ww} - f_w^2 f_{uu}$ | $3k_u - 2$ | $3k_v$ | $3k_w - 2$ | 3 |
| $\widetilde{K} = \widetilde{L}\widetilde{N} - \widetilde{M}^2$ | $6k_u - 2$ | $6k_v - 2$ | $6k_w - 4$ | 6 |
| $\widetilde{\widetilde{K}} = \widetilde{K}/f_w^2$ | $4k_u - 2$ | $4k_v - 2$ | $4k_w - 2$ | 4 |
| $\widetilde{\widetilde{H}}^2$ | $6k_u$ | $6k_v$ | $6k_w$ | 6 |

**Table 1. Degrees of differential components.**

Equations $(1) \Leftrightarrow (8)$. With the aid of Maple [3], we represent the differential components in Table $1$ as polynomial functions in $u, v, w$ and $p_{i,j,l}$. For example, $\widetilde{D}$, is a polynomial of degrees $2k_u, 2k_v, 2k_w$ in $(u, v, w)$, respectively. Further, because $\overset{\Leftrightarrow}{\nabla} f$ is a linear polynomial in $p_{i,j,l}$, $\widetilde{D} = \left\langle \overset{\Leftrightarrow}{\nabla} f, \overset{\Leftrightarrow}{\nabla} f \right\rangle$ is a quadratic function in $p_{i,j,l}$. While, in general, the degrees of the terms grow larger as we progress, $\widetilde{\widetilde{K}}$ is obtained from $\widetilde{K}$ by dividing by $f_w^2$. Thus the degrees of $\widetilde{\widetilde{K}}$ are smaller than those of $\widetilde{K}$ (see Equation $(9)$).

Consider a prescribed trivariate $f$ with known coefficients $p_{i,j,l}$, which could be substituted into $\psi(u, v, w)$. The problem of deriving $(12)$ could be mapped to a system of equations with $\mathcal{O}$ unknowns $(q_{i,j,l})$, and $\mathcal{O}$ constraints:

$$\begin{aligned} &\theta\left(\frac{i}{o_u}, \frac{j}{o_v}, \frac{l}{o_w}\right) \\ &= \psi\left(\tau_i + \frac{i(\tau_{i+1} \Leftrightarrow \tau_i)}{o_u},\right. \\ &\qquad \tau_j + \frac{j(\tau_{j+1} \Leftrightarrow \tau_j)}{o_v}, \\ &\qquad \left.\tau_l + \frac{l(\tau_{l+1} \Leftrightarrow \tau_l)}{o_w}\right), \end{aligned} \quad (13)$$

where $i = 0, \ldots, o_u$, $j = 0, \ldots, o_v$, $l = 0, \ldots, o_w$. Note $p_{i,j,l}$ are now prescribed. System $(13)$ is modelled as follows. Consider matrices

$$\boldsymbol{\Theta} \in M_{\mathcal{O} \times \mathcal{O}},\ \boldsymbol{Q} \in M_{\mathcal{O} \times 1},\ \text{and } \boldsymbol{\Psi} \in M_{\mathcal{O} \times 1},$$

where $M_{i \times j}$ denotes a matrix of size $i$ by $j$. Let

$$r = l\,(o_u + 1)\,(o_v + 1) + j\,(o_u + 1) + i.$$

Then, the $s'$th element of the $r'$th row, $\boldsymbol{\Theta}_{r,s} \in \boldsymbol{\Theta}$, equals

$$\boldsymbol{\Theta}_{r,s} = \theta_{\alpha,o_u}\left(\frac{i}{o_u}\right)\theta_{\beta,o_v}\left(\frac{j}{o_v}\right)\theta_{\gamma,o_w}\left(\frac{l}{o_w}\right), \tag{14}$$

where $s = \gamma\,(o_u + 1)\,(o_v + 1) + \beta\,(o_u + 1) + \alpha$. Similarly, let the $s'$th element of $\boldsymbol{Q}$ be $\boldsymbol{Q}_s = \boldsymbol{q}_{\alpha,\beta,\gamma}$ and the $r'$th element of $\boldsymbol{\Psi}$ be $\boldsymbol{\Psi}_r = \psi\left(\tau_i + \frac{i(\tau_{i+1} - \tau_i)}{o_u}, \tau_j + \frac{j(\tau_{j+1} - \tau_j)}{o_v}, \tau_l + \frac{l(\tau_{l+1} - \tau_l)}{o_w}\right)$. Then, Equation $(13)$ is equivalent to

$$\boldsymbol{\Theta}\,\boldsymbol{Q} = \boldsymbol{\Psi}. \tag{15}$$

$\boldsymbol{\Psi}_r$ is symbolically pre-evaluated once into $\mathcal{O}$ polynomial equations in $p_{i,j,l}$, as in the right side of Equation $(13)$. Given specific $f$, all the $\Psi_r$ functions are evaluated by substituting in the $p_{i,j,l}$ coefficients.

$\boldsymbol{\Theta}$ is also independent of the input and hence one can precisely pre-compute $\boldsymbol{\Theta}$ and $\boldsymbol{\Theta}^{-1}$ once. Further, $\boldsymbol{\Theta}$ is guaranteed to be non-singular since we evaluate $\theta\,(x, y, z)$ at three-dimensional independent parametric points of the form, $\left(\frac{i}{o_u}, \frac{j}{o_v}, \frac{l}{o_w}\right)$, points that are also known as node points or Greville abscissas [7]. Furthermore, the functions, $\theta_{i,o_u}(u)$, $\theta_{j,o_v}(v)$, and $\theta_{l,o_w}(w)$, form a basis for the polynomials of degree $o_u$, $o_v$, and $o_w$, in $u, v$, and $w$, respectively.

The diagonal elements of matrix $\boldsymbol{\Theta}$ are larger than any other element in the same column or row due to the fact that $\theta_{j,k}\left(\frac{j}{k}\right) \geq \theta_{j,k}(x)$ for any $x \in [0, 1]$ [2]. Hence, this chosen interpolation scheme provides more stability when the linear system of equations is solved, as

$$\boldsymbol{Q} = \boldsymbol{\Theta}^{-1}\boldsymbol{\Psi}. \tag{16}$$

Yet, direct inversion of $\boldsymbol{\Theta}$ is not trivial.

Let $y_j$ be a sequence of numerical values, $j \in \{0, 1, \ldots, r\}$. Then, the Vandermonde matrix is a matrix of the form

$$\boldsymbol{V}an\,(y_0, y_1, \ldots, y_r) = \begin{pmatrix} y_0^0 & y_0^1 & \cdots & y_0^r \\ y_1^0 & y_1^1 & \cdots & y_1^r \\ \vdots & \vdots & \vdots & \vdots \\ y_r^0 & y_r^1 & \cdots & y_r^r \end{pmatrix}.$$

There is a strong relationship between $\boldsymbol{\Theta}$ and Vandermonde matrices. It is known that Vandermonde matrices have large condition numbers and this fact implies that the computation of their inverse is numerically unstable. Moreover, multiplying matrices with increasing condition numbers would typically yield results with increased numerical errors. A good presentation of the numerical problems that appear when using matrices with large condition numbers can be found in [12]. If not enough, herein the size of matrices $\boldsymbol{\Theta}$ and $\boldsymbol{\Theta}^{-1}$ is very large, in the thousands. For example, in the case of $k = 3$, $\boldsymbol{\Theta}, \boldsymbol{\Theta}^{-1} \in M_{2197 \times 2197}$.

The matrix $\boldsymbol{\Theta}$ defined in $(14)$ can be decomposed into three lower dimensional matrices using the Kronecker product. Let $(\boldsymbol{\theta}^u)_{i,j} = \theta_{i,o_u}\left(\frac{j}{o_u}\right)$, $(\boldsymbol{\theta}^v)_{i,j} = \theta_{i,o_v}\left(\frac{j}{o_v}\right)$, and $(\boldsymbol{\theta}^w)_{i,j} = \theta_{i,o_w}\left(\frac{j}{o_w}\right)$. Then, we have

$$\boldsymbol{\Theta} = \boldsymbol{\theta}^u \otimes \boldsymbol{\theta}^v \otimes \boldsymbol{\theta}^w, \tag{17}$$

where $\otimes$ denotes the Kronecker product of matrices [8, 17]. Moreover, $\boldsymbol{\Theta}^{-1} = (\boldsymbol{\theta}^w)^{-1} \otimes (\boldsymbol{\theta}^v)^{-1} \otimes (\boldsymbol{\theta}^u)^{-1}$. The inversion of the matrices $\boldsymbol{\theta}^u, \boldsymbol{\theta}^v$, and $\boldsymbol{\theta}^w$ requires much less computational resources than the inversion of $\boldsymbol{\Theta}$ and more importantly, can be made far more precise. The Kronecker based dimensional reduction can also be interpreted as interpolations in successive dimensions, that is of curves, followed by surfaces, and then volumes [2].

### 3.1.1 System Dimensions Reducing

Recall that in Section 3.1 we solve the system defined in Equation $(15)$ using $(16)$, which assumes the computation of $\boldsymbol{\Theta}^{-1}$. There are several drawbacks in trying to solve directly such a large system. The first drawback is found in the time complexity required for the multiplications performed in $(16)$. In this section, we present a computation scheme that is more efficient as well as less memory requirements demanding than the ones used in $(15)$.

Let $\boldsymbol{P} \in M_{m,n}$, and let $\boldsymbol{P}\,(i, :)$ be the $i$⇔th row of $\boldsymbol{P}$. Similarly, let $\boldsymbol{P}\,(:, i)$ be the $i$⇔th column of $\boldsymbol{P}$.

Define the single column matrix $\overset{\Leftrightarrow}{\boldsymbol{P}} = \begin{pmatrix} \boldsymbol{P}\,(:, 1) \\ \boldsymbol{P}\,(:, 2) \\ \vdots \\ \boldsymbol{P}\,(:, n) \end{pmatrix} \in M_{mn,1}$. We use left arrow to define matrices in the opposite sense, that is if $\boldsymbol{P} \in M_{mn,1}$ then $\overset{\Leftrightarrow}{\boldsymbol{P}} \in M_{m,n}$ and so on.

In the following, we employ an additional property of the Kronecker product. Assume $\boldsymbol{P} \in M_{m,n}, \boldsymbol{U} \in M_{o,p}$, $\boldsymbol{A} \in M_{o,n}$, and $\boldsymbol{B} \in M_{p,m}$. Then,

$$(\boldsymbol{A} \otimes \boldsymbol{B})\,\overset{\Leftrightarrow}{\boldsymbol{P}} = \overset{\Leftrightarrow}{\boldsymbol{U}} \Longleftrightarrow \boldsymbol{B}P\boldsymbol{A}^T = \boldsymbol{U}, \tag{18}$$

see [8] for details. $(18)$ allows us to compute the vector $\boldsymbol{\Psi}$ as the solution of the system $(15)$ as follows. We can model $\boldsymbol{Q} = \overset{\Leftrightarrow}{\boldsymbol{q}}$ and $\boldsymbol{\Psi} = \overset{\Leftrightarrow}{\boldsymbol{\psi}}$. Then, $(15)$ is equivalent to $\boldsymbol{\Theta}\overset{\Leftrightarrow}{\boldsymbol{q}} = \overset{\Leftrightarrow}{\boldsymbol{\psi}}$ and by $(17)$ we have

$$\begin{aligned} \boldsymbol{\theta}^u \otimes \boldsymbol{\theta}^v \otimes \boldsymbol{\theta}^w \overset{\Leftrightarrow}{\boldsymbol{q}} &= \overset{\Leftrightarrow}{\boldsymbol{\psi}} \\ \Leftrightarrow \boldsymbol{\theta}^u \otimes (\boldsymbol{\theta}^v \otimes \boldsymbol{\theta}^w)\,\overset{\Leftrightarrow}{\boldsymbol{q}} &= \overset{\Leftrightarrow}{\boldsymbol{\psi}}. \end{aligned} \tag{19}$$

Moreover, using $(18)$, $(19)$ is equivalent to

$$\begin{aligned}(\boldsymbol{\theta}^v \otimes \boldsymbol{\theta}^w)\, \overleftrightarrow{\boldsymbol{q}}\, (\boldsymbol{\theta}^u)^T &= \overleftrightarrow{\boldsymbol{\psi}} \\ \Leftrightarrow \boldsymbol{\theta}^v \otimes \boldsymbol{\theta}^w \overleftrightarrow{\boldsymbol{q}} &= \overleftrightarrow{\boldsymbol{\psi}} \left( (\boldsymbol{\theta}^u)^T \right)^{-1}. \qquad (20)\end{aligned}$$

Denote the $i$⇔th column of $\overleftrightarrow{\boldsymbol{q}}$ and $\overleftrightarrow{\boldsymbol{\psi}} \left( (\boldsymbol{\theta}^u)^T \right)^{-1}$ by $\overleftrightarrow{\boldsymbol{q}}\,(:, i)$ and $\left( \overleftrightarrow{\boldsymbol{\psi}} \left( (\boldsymbol{\theta}^u)^T \right)^{-1} \right)(:, i)$, respectively. Then, we can once again split the matrix system of Equations $(20)$ using $(18)$. Thus, $(20)$ is equivalent to,

$$\begin{aligned}\boldsymbol{\theta}^v \left( \overleftrightarrow{\boldsymbol{q}}\,(:, i) \right) (\boldsymbol{\theta}^w)^T &= \overleftrightarrow{\left( \overleftrightarrow{\boldsymbol{\psi}} \left( (\boldsymbol{\theta}^u)^T \right)^{-1} \right)}(:, i) \\ \overleftrightarrow{\boldsymbol{q}}\,(:, i) &= (\boldsymbol{\theta}^v)^{-1} \overleftrightarrow{\left( \overleftrightarrow{\boldsymbol{\psi}} \left( (\boldsymbol{\theta}^u)^T \right)^{-1} \right)}(:, i) \\ &\quad \left( (\boldsymbol{\theta}^w)^T \right)^{-1}. \qquad (21)\end{aligned}$$

In the case of Gaussian curvature computation and $k = 3$, the decomposition $(21)$ allows us to solve $(17)$ as a set of $13^2$ systems of equations, each one requiring $O\left(13^2\right)$ products. In total we need $O\left(13^4\right)$ products, in contrast to $O\left(13^6\right)$ products that are required in $(16)$. We do not count the time required for inversion, that is done in the precomputation stages. Note that the complexity of solving one system means a multiplication of one of the matrices $(\boldsymbol{\theta}^u)^{-1}, (\boldsymbol{\theta}^v)^{-1}$, or $(\boldsymbol{\theta}^w)^{-1} \in M_{13\times 13}$ with a vector of length $13$.

Since the inverse matrices with which we now work have lower dimensions than the matrices used in $(16)$, the accuracy of computations is bound to be improved. In the case of Gaussian curvature computation for $k = 3$ and double precision, $K$ is computed with an accuracy of ten decimal digits in the mantissa using the solution presented in this section. Under the same conditions, one could only achieve an accuracy of two decimal digits for $\boldsymbol{\Theta}^{-1}$, when $(16)$ was applied directly.

### 3.2 Merging into a Single B-spline Form

Our aim now is to merge the $\chi$ Bézier trivariates from Section 3, each defined for a different domain $\mathcal{D}_{i,j,l}$, into one large B-spline trivariate function that is defined over the entire domain of $f\,(u, v, w)$. Consider the univariate case where several Bézier curve segments are merged into one B-spline curve with $C^{(0)}$ continuity. The motivation for the $C^{(0)}$ examples stems from considering the curvature continuity of cubic splines. The extension to the trivariate case is a simple generalization that takes place in each of its three axes, independently. Bézier curves can be merged into a single B-spline curve by multiplying each interior knot $k$ times, where $k$ is the degree of the curve, and copying the coefficients of the Bézier curve into the B-spline one's, successively.

## 4 Segmentation of Volumetric Data Sets Using $K$ and $H$

This section presents several segmentation mechanisms of volumetric data sets using the scalar Gaussian curvature function we have just computed. The segmentation using the mean curvature is similar to the Gaussian one. Furthermore, several issues of augmenting and speeding up the evaluation process are also discussed.

Consider the Gaussian curvature of an iso-surface represented by a trivariate B-spline function. Given a scalar B-spline trivariate function $f\,(u, v, w)$ (Equation $(11)$), we are able to *symbolically* compute Equation $(9)$, and represent the trivariate function $K\,(u, v, w)$ as a scalar B-spline trivariate function which globally represents the Gaussian curvature of any iso-surface of $f\,(u, v, w)$, for all possible locations, and hence, iso-levels. In other words, $K\,(u, v, w)$ is a rational form, provided $f$ is. If $f$ can be represented as a B-spline volumetric function, so can $K$. As an example, if $f$ is a triquadratic or a tricubic polynomial, the numerator of $K$ is a trivariate function of degrees six or ten, respectively, whereas its denominator has degrees eight or twelve, respectively, in each direction (see Table 1). With this approach, we are able to globally and simultaneously analyze all the regions in the entire volume, for which the iso-surfaces assume certain Gaussian curvature values. $K\,(u, v, w)$ could be prescribed as either a Bézier or a B-spline trivariate function, two forms that can yield bounds on the values that $K$ can assume at a certain arbitrary sub-domain $\mathcal{D}_{i,j,l}$ by simply examining the coefficients of the function at that sub-domain. Further, with the subdivision capability of these representations, one can easily construct a divide-and-conquer algorithm to robustly converge to locations with specific values of $K$. These properties allow one to segment volumes in regions of interest (characterized by certain Gaussian curvature values) directly and without the need for an exhaustive sampling search. Moreover, this search, being symbolic and global, is immune to aliasing, is precise to within machine precision, and is independent of iso-values.

One of the most difficult problems in volumetric image processing is handling the size of the data. As stated earlier, if $f$ is a triquadratic or a tricubic, the numerator of $K$ is a trivariate function of degree six or ten in $u, v,$ and $w$, respectively, whereas its denominator has degree eight or twelve. As a consequence, the two trivariates that represent the numerator and the denominator of $K\,(u, v, w)$ increase the needed data-size by a factor of $\left(\frac{7+9}{3}\right)^3$ for a tri-quadratic or $\left(\frac{11+13}{4}\right)^3$ for a tri-cubic, in each axis. For contemporary volumetric data sets, such an increase, of more than two orders of magnitudes, could be devastating. A remedy could be found in breaking the input volume into pieces, and examining $K\,(u, v, w)$ incrementally, in each polynomial subdomain instead of the entire domain of $f\,(u, v, w)$. In other words, we evaluate $K$ for each polynomial sub-domain, $\mathcal{D}_{i,j,l}$, as described in Section 3.1, segment, and immediately purge this $K$ for $\mathcal{D}_{i,j,l}$. No merging stage, as described in Section 3.2, is actually conducted for this segmentation application. At every point of time of the algorithm, only one $K(u, v, w)$ for one domain $\mathcal{D}_{i,j,l}$ is allocated.

We start now with a simple segmentation example, considering the solution for $K(u, v, w) = K_0$. This problem could be solved simply by applying the traditional marching cubes [18] algorithm to $K$. For example, if $K_0 = 0$, one is simultaneously extracting all the parabolic manifolds in the volume, regardless of their iso-values.

As part of the volumetric segmentation process, one can employ a geometric constraint solver for multivariate rational B-spline functions [6]. Specifically, the solver can seek the simultaneous solution of

$$\left\{ \begin{array}{ll} f(u,v,w) = f_0, & f_0 \text{ constant}, \\ K(u,v,w) = K_0, & K_0 \text{ constant}. \end{array} \right.$$

Equality as well as inequality constraints can be given to the solver. Hence, one can also solve for

$$\left\{ \begin{array}{ll} f(u,v,w) = f_0 & f_0 \text{ constant}, \\ |K(u,v,w)| \leq K_0, & K_0 \text{ positive constant}. \end{array} \right.$$

Assume we are interested in processing and segmenting several iso-levels, $f_0, f_1, \ldots, f_n$. For each polynomial sub-domain $\mathcal{D}_{i,j,l}$, we compute the gradient of $f(u, v, w)$ as a trivariate Bézier, $\overset{\Leftrightarrow}{\nabla} f_{i,j,l}(u, v, w) = \left( \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v}, \frac{\partial f}{\partial w} \right)$. We further process sub-domain $\mathcal{D}_{i,j,l}$ only if $\left\| \overset{\Leftrightarrow}{\nabla} f_{i,j,l}(u, v, w) \right\|$ presents magnitudes greater than a certain threshold in $\mathcal{D}_{i,j,l}$. This test is conducted by examining the magnitude of the control points of $\overset{\Leftrightarrow}{\nabla} f$ and allows us to process only sub-domains that contain information above a certain noise level. Again, note that the gradient does not depend on a certain iso-level value. For sub-domains that are found to contain a sufficiently large gradient, we simultaneously solve for $|K| \leq \epsilon$ and $f = f_i, i = 0, 1, \ldots, n$ using the above mentioned multivariate solver [6]. While solving for $K = 0$ is potentially simpler, as only the numerator of $K$ needs to be processed, this approach was found to be unstable and too noise sensitive when noisy data was provided. In order to solve for $|K| \leq \epsilon$, we have to intersect the solutions of $\widetilde{\widetilde{K}} \Leftrightarrow \epsilon \widetilde{D}^2 \leq 0$ with the ones of $\widetilde{\widetilde{K}} + \epsilon \widetilde{D}^2 \geq 0$ (recall Equation $(9)$), where $\epsilon > 0$ is some low positive constant. This scheme is demonstrated in Algorithm 1 for this example that seeks the parabolic regions in an iso-surface.

## 5 Examples of Segmenting Using $K$ and $H$

In all examples presented in this section, we compute the Gaussian and the mean curvatures over each sub-domain

**Algorithm 1**
**Input:**
*$f(u, v, w)$ : a trivariate volumetric data set;*
*$f_0$ : desired iso-level;*
*$\epsilon$ : level of Gaussian curvature below which, we assume it is a parabolic domain;*
*$n_0$ : gradient's noise level;*
**Output:**
*iso-surface $f_0$, with a prescribed curvature property;*
**Algorithm:**

```
SegmentVolume($f, f_0, \epsilon, n_0$) ;
Begin
    if ($\min(f) \leq f_0$ and $\max(f) \geq f_0$)
        if ($f$ is not a single polynomial)
            $f_a, f_b \Leftarrow$ subdivided $f$ in an interior knot
                                        along $u$, $v$, or $w$;
            SegmentVolume($f_a, f_0, \epsilon, n_0$) ;
            SegmentVolume($f_b, f_0, \epsilon, n_0$) ;
        else
            $\overset{\Leftrightarrow}{\nabla} f \Leftarrow$ gradient of $f$;
            if ($\exists\, u, v, w$ such that $\left\| \overset{\Leftrightarrow}{\nabla} f \right\| > n_0$)
                $K \Leftarrow$ Gaussian scalar field of $f$;
                solve for $\left\{ \begin{array}{l} f = f_0, \\ |K| \leq \epsilon, \end{array} \right.$ ;
                purge $K$;
            fi
            purge $\overset{\Leftrightarrow}{\nabla} f$;
        fi
    fi
End
```

$\mathcal{D}_{i,j,l}$, evaluate or compute solutions for curvature constraints with the multivariate solver, and then immediately purge the trivariate representing $K$ and $H$ over $\mathcal{D}_{i,j,l}$, as is shown in of Algorithm 1. As presented in Section 3.2, it is possible to compute the whole trivariate B-spline function $K$ for all the domain of $f$. The memory requirements for $K$ for a $40^3$ tricubic volumetric function is around $40^3 \times 8 \times (11^3 + 13^3)$ bytes $\approx 1.7$ gigabytes, where $40^3$ is the number of $\mathcal{D}_{i,j,l}$ sub-domains, $8$ bytes are assumed for each double precision number, and in each sub-domain there are $11^3 + 13^3$ coefficients in the rational Bézier representation of $K$. Hence, the explicit representation of $K$ and $H$ for the entire domain is expected to be rarely computed, using contemporary hardware.

We present a few examples of segmenting volumetric data sets using the proposed curvature computation scheme. The examples presented were constructed with the aid of the IRIT Solid Modeling system [5]. All the images were computed on a machine with four $2.4$ GHz Pentium $4$ processors with $4$ gigabytes of memory. In each image, red, green, and blue represent regions with convex or concave (elliptic) iso-surfaces, planar or cylindrical $(\text{parabolic})$ iso-surfaces, and volumetric regions with saddle-like $(\text{hyperbolic})$ iso-surfaces, respectively, having positive, zero, and negative Gaussian curvature values.

Figure 1 $(\text{a})$ is a synthetic volumetric image of a cylinder and six spheres where the Gaussian curvature of a certain pre-selected constant iso-level was computed using this new scheme. The input image is a trivariate B-spline function $f\,(u, v, w)$ and the presented algorithm computed $K\,(u, v, w)$ as a new scalar trivariate B-spline. We have extracted a pre-selected constant iso-level of $f\,(u, v, w)$ and evaluated the values of the Gaussian curvature $K\,(u, v, w)$ at the nodes of the iso-surface. The nodes were colored with red, green, and blue everywhere the Gaussian curvature is positive, zero, and negative, respectively. Gradual change from red to green and blue is employed as well, following the gradual change of $K$. The trivariate volumetric image has the size of $40 \times 40 \times 40$ coefficients. It takes $45$ minutes and $30$ minutes to compute the numerator and denominator of $K$, respectively.

Figure 1 $(b)$ represents a view of a volumetric model of an engine block where the Gaussian curvature of a certain pre-selected constant iso-level was computed using this new scheme. This trivariate volumetric image of an engine has the size of $256 \times 256 \times 110$ coefficients. It takes eleven and a half, and seven and a half hours to compute the numerator and denominator of $K$, respectively. Around $600$ megabytes of memory were required to compute and analyze the volume using the Gaussian curvature.

Figures 2 $(a)$ and $(b)$ portray two volumetric images of an iron protein molecule where the Gaussian and square of the mean curvatures of a certain pre-selected constant iso-level were computed using this new scheme. In Figure 2 $(b)$ the colors are computed using the sign of the mean curvature. The trivariate volumetric image has the size of $68 \times 68 \times 68$ coefficients. The computation of the numerator of $K$ requires one and a half hours while its denominator computation takes $16$ minutes. The computation of the numerator of $H$ necessitates $41$ minutes. The computation of the numerator of $H^2$ requires $4$ hours and $45$ minutes while its denominator computation takes $48$ minutes.

Our segmentation technique detects regions in which the enclosed iso-surfaces are elliptic, parabolic, or hyperbolic. Further, the parabolic manifold subdivides the volume into elliptic and hyperbolic region. The geometry of the molecules often provides specialists with valuable information about the behaviour of substances. Figures 3 shows the zero level set of $K\,(u, v, w) = 0,$ of the iron protein model shown in Figure 2. This ability to robustly derive the zero set of $K$, or any other differential form, is a direct consequence of our global rational representation of $K(u, v, w)$ for the given function $f(u, v, w)$, either as one B-spline form or as piecewise- Bézier form.

## 6 Conclusions

In this work, we have presented a scheme to globally derive curvature properties of volumetric data sets. The scheme is global, immune to aliasing and capable of detecting the curvature properties regardless of iso-level values. We map a given scalar field $f\,(u, v, w)$ to another differential scalar field such as $K\,(u, v, w)$ .

In the presented work, uniform knot sequences were employed throughout. Non uniform knot sequences could be employed, for example, if the volumetric data set is sampled unevenly. Nothing prevents the analysis and segmentation procedures presented here from using non uniform knot sequences, an option that can be easily added.

The presented algorithm computes a scalar field $K\,(u, v, w)$ , given scalar field $f\,(u, v, w)$ . Hence, any volumetric rendering scheme, such as splating and/or ray casting [14], could be used to render the volumetric field of $K\,(u, v, w)$ . We expect to experiment with such rendering approaches in the near future.

Although the problem has a continuous nature, like many other volumetric processing problems, the proposed solution is clearly parallelizable. We believe that employing concurrent or parallel variants of the algorithms, and/or implementing the curvature evaluation schemes on dedicated hardware, could greatly speed up this process.

## 7 Acknowledgments

The engine block's volumetric data set is part of the VolVis [22] public domain library and the iron protein model is part of the public data set of VTK [21]. The first author thanks Emil Dan Kohn for his technical advice related to various programming tasks in this present and Avraham Sidi for several observations focused on the Kronecker product.

This work was supported in part by the Fund for Promotion of Research at the Technion, Haifa, Israel, in part by the Bar-Nir Bergreen Software Technology Center of Excellence and in part by NSF grant $(IIS0218809)$ . All opinions, findings, conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of the sponsoring agencies.

## References

[1] *Gaussian Curvature of Volumetric Data Sets.* http://www.cs.technion.ac.il/~octavian/gaussian_curvature_vol.

[2] E. Cohen, R. F. Riesenfeld, and G. Elber. *Geometric Modeling with Splines: An Introduction.* A. K. Peters, July 2001.

[3] R. M. Corless. *Essential Maple 7. An Introduction for Scientific Programmers, second edition.* www.maplesoft.com, Spinger, ISBN: 0-387-95352-3, 2002.

[4] G. Dahlquist and Å. Björck. *Numerical Methods.* Prentice-Hall, Englewood Cliffs, New Jersey, 1974.

[5] G. Elber. *Irit Version 8.0.* http://www.cs.technion.ac.il/∼irit.

[6] G. Elber and M.-S. Kim. *Geometric Constraint Solver Using Multivariate Spline Functions.* Proceedings of the Sixth ACM Symposium on Solid Modeling and Computer Graphics, pp. 171-175, October 11-12 2001.

[7] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide*. Fourth Edition, Academic Press, 1996.

[8] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* Third Edition, The Johns Hopkins University Press, pp. 180-181, 1996.

[9] B. Hamann. *Visualization and Modeling Contours of Trivariate Functions, PhD Thesis*. Arizona State University, 1991.

[10] Jiří Hladůvka, Andreas König, and Eduard Gröller. *Curvature-Based Transfer Functions for Direct Volume Rendering*. Proceedings of Spring Conference on Computer Graphics and its Applications, SCCG 2000, pp. 58-65, 2000.

[11] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bumke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon, and R. B. Fischer. *An Experimental Comparison of Range Image Segmentation Algorithms.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 18, Number 7, pp. 673-689, July 1996.

[12] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, pp. 336, 338-339, 1985.

[13] V. Interrante. *Illustrating Surface Shape in Volume Data via Principal Direction-Driven 3D Line Integral Convolution*. Computer Graphics (Proceedings of SIGGRAPH 97) , pp. 109-116, 1997.

[14] A. Kaufman. *Volume Visualization* (*Tutorial*). IEEE Computer Society Press, Los Alamitos, CA., 1991.

[15] Gordon Kindlmann, Ross Whitaker, Tolga Tasdizen, and Torsten Möller. *Curvature-Based Transfer Functions for Direct Volume Rendering: Methods and Applications.* IEEE Visualization, accepted for publication, October 2003.

[16] S. Lakare, M. Wan, M. Sato, and A. Kaufman. *3D Digital Cleansing Using Segmentation Rays*. IEEE Visualization Annual Conference, pp. 37-44, 2000.

[17] Charles F. Van Loan. *The Ubiquitous Kronecker Product*. to appear in Journal of Computational and Applied Mathematics, 2003.

[18] W. E. Lorensen and H. E. Cline. *Marching cubes: A high resolution 3D surface construction algorithm.* Computer Graphics (Proceedings of SIGGRAPH 87) , Volume 21, Number 4, pp. 163-169, July 1987.

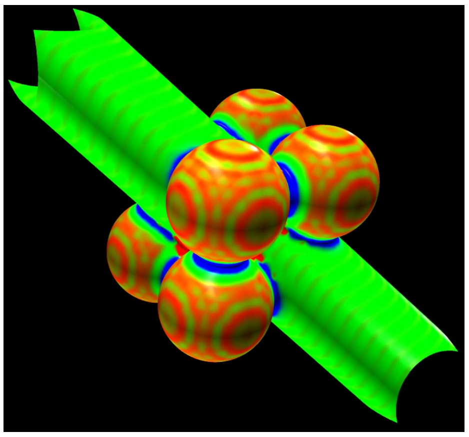
[19] J.-P. Thirion and A. Gourdon. *The 3D Marching Lines Algorithm and its Application to Crest Lines Extraction*. Rapports de Recherche, No. 1672, Programme 4, Robotique, Image et Vision, Unité de Recherche Inria-Rocquencourt, Inria, May 1992.

[20] J.-P. Thirion and A. Gourdon. *Intrinsic surface properties from surface triangulation*. Differential Characteristics of Isointensity Surfaces. Computer Vision and Image Understanding, Volume 61, Number 2, pp. 190-202, March 1995.

[21] Visualization Toolkit. http://www.vtk.org.

[22] Volume Visualization. http://www.volvis.org.

[23] T. Weinkauf and H. Theisel. *Curvature Measures of 3D Vector Fields and Their Applications.* V. Skala (editor): Journal of WSCG'2002, International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Plzen, Czech Republic, Number 10, pp. 507-514, February 2002.


(a)

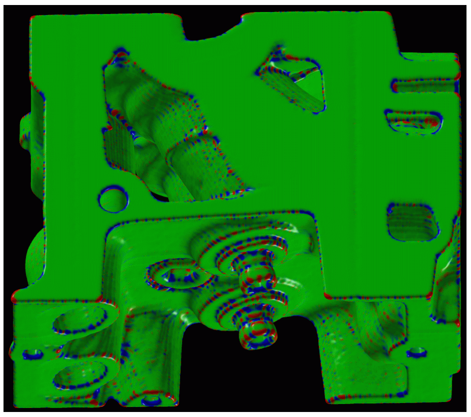
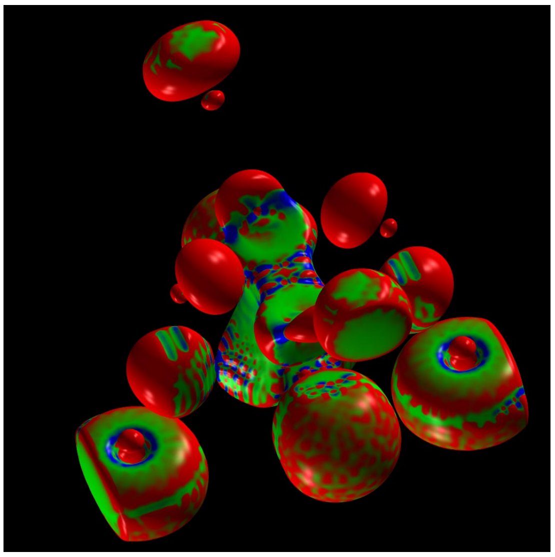

(b)

**Figure 1. In (a), a synthetic volumetric image of a cylinder and six spheres is represented, and in (b), a model of an engine block is shown. Red, green, and blue represent volumetric regions with elliptic, parabolic, and hyperbolic iso-surfaces, respectively, having positive, zero, and negative Gaussian curvature values.**


(a)

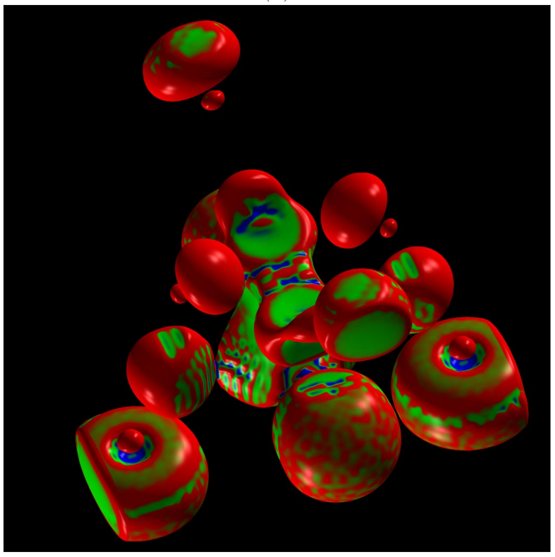

(b)

**Figure 2. In (a) and (b) two volumetric images of an iron protein molecule are presented. Red, green, and blue represent volumetric regions having positive, zero, and negative Gaussian and mean square curvature values respectively.**

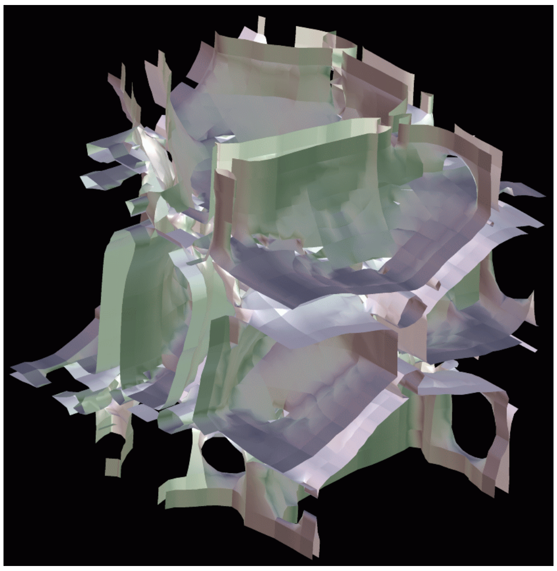


**Figure 3. (left column) An iso-surface at level zero of $K$ of the iron protein in Figure 2 (a). This iso surface globally prescribes the parabolic regions of all iso-surface, simultaneously.**