# Classification for Fourier Volume Rendering

Zoltán Nagy, Gero Müller, Reinhard Klein
Institute for Computer Science II
Bonn University
Römerstraße 164, D-53117 Bonn, Germany
$\{$zoltan$|$gero$|$rk$\}$@cs.uni-bonn.de

## Abstract

*In the last decade, Fourier Volume Rendering (FVR) has obtained considerable attention due to its $\mathcal{O}(N^2 logN)$ rendering complexity, where $\mathcal{O}(N^3)$ is the volume size. Although ordinary volume rendering has $\mathcal{O}(N^3)$ rendering complexity, it is still preferred over FVR for the main reason, that FVR offers bad localization of spacial structures. As a consequence, it was assumed, that it is hardly possible to apply 1D transfer functions, which arbitrarily modify voxel values not only in dependence of the position, but also the voxel value. We show that this assumption is not true for threshold operators. Based on the theory of Fourier series, we derive a FVR method, which is capable of integrating all sample points greater (or alternatively, lower) than an iso-value $\tau$ during rendering, where $\tau$ can be modified interactively during the rendering session. We compare our method with other approaches and we show examples on well-known datasets to illustrate the quality of the renderings.*

## 1. Introduction

Volume rendering is a technique for visualizing a three-dimensional scalar or vector field. Existing volume rendering methods can be roughly classified into two main categories, depending on their rendering time complexity: (i) spacial domain methods and (ii) frequency domain methods. The first class of methods is nowadays the method of choice, since arbitrary lighting models can be simulated and material properties can be modelled freely. Since every voxel of the dataset has to be visited at least once, the main disadvantage of these methods is their high $\mathcal{O}(N^3)$ rendering complexity, where $\mathcal{O}(N^3)$ is the size of the dataset. The second class of algorithms work in Fourier space. The situation is opposite here: using the Fourier Projection-Slice theorem [13], rendering can be accomplished fast, with $\mathcal{O}(N^2 logN)$ complexity. This advantage is gained by restricting the lighting model to have low albedo, no interreflection, and no self-occlusion; this kind of optical model is called *X-ray-like*, since the final projections of the investigated datasets look like as if screened by X-rays.

A further problem with Fourier rendering models is its weakness in locating spacial structures. That is, once the preprocessing step is finished, the user is hardly able to apply classification techniques - like transfer functions - on the dataset during rendering, not to mention interactively. The reason is, that for a given function $f(x)$ (here: the input dataset) and a transfer function $g(x)$ the integral $\int g(f(x))dx$ can not be decomposed resp. factorized in general into a pre-transformed variant $\mathcal{F}\{f\}$ and a fast modifiable $\mathcal{F}\{g\}$, where $\mathcal{F}\{\cdot\}$ denotes the Fourier transform.

In this work we show, that latter problem is solvable for a special class of transfer functions, namely

$$g_\tau(u) = \left\{ \begin{array}{ll} u & if\ u \geq \tau \\ 0 & otherwise \end{array} \right\}, \qquad (1)$$

i.e., $g_\tau(u)$ is a threshold function with iso-value $\tau$. This particular class of functions occurs e.g. in medical applications, where soft tissue with low opacity has to be separated from bone with high opacity in X-ray-like renderings. Our method allows for manipulating the $\tau$-value interactively during rendering.

Our paper is organized as follows. In section 2, we review previous work. In section 3 mathematical foundations for our actual method are derived. The resulting algorithm is discussed in section 4, which is further analyzed in section 5. Sections 6 and 7 are dedicated to discussing the results. Throughout the text, we use the notation in figure 1.

## 2. Previous Work

### 2.1. Fourier Volume Rendering

FVR was first proposed by Levoy [12] and Malzbender [14]. Levoys paper contains a rich variety of shading

| | |
|---|---|
| $B_i(x), B_i^\tau(x)$ | Basis function without/with dependence on $\tau$ |
| $\delta(t)$ | Dirac delta function |
| $\delta_{l,m}$ | Kronecker symbol, $\begin{cases} 1 & if\ l\ =\ m \\ 0 & otherwise \end{cases}$ |
| $\mathcal{F}_n\{\}, \mathcal{F}_n^{-1}\{\}$ | nD forward/backward Fourier transform |
| $H(x)$ | Heaviside step function, $\begin{cases} 1 & if\ x\ \geq 0 \\ 0 & otherwise \end{cases}$ |
| $g_\tau(u)$ | Transfer function |
| $g \star f$ | Convolution of $g$ with $f$ |
| $N_{xyz}$ | Normal at surface/voxel position $(x, y, z)$ |
| $R$ | Residual energy |
| $\tau \in [0, 1]$ | iso-value |
| $u_{xyz} \in [0, 1]$ | scalar/opacity value at voxel position $(x, y, z)$ |
| $W$ | Energy in the fourier expansion spectrum |
| $W(k)$ | Energy in the fourier coefficient $k$ |
| $\omega = (\theta, \phi)$ | Solid angle |
| $Y_{lm}(\omega)$ | Spherical harmonics |
| $z, \overline{z}$ | Variable $z$ and its complex conjugate |

**Figure 1. Notation.**

models for FVR: X-rays with depth cueing, X-rays with directional shading and both simultaneously. Depth cueing has a general applicability for FVR, since it can be integrated into most FVR frameworks and it substitutes occlusion effects in an acceptable way. In his conclusions, Levoy discusses three strategies for applying segmentation operators on the dataset, which are important in our context. The operator must be expressible as one of the following forms:

1. **linear combination of basis functions:** the operator has the form

$$g(x) = \sum_{i=1}^{n} a_i B_i(x), \qquad (2)$$

   where an application of the Fourier-transform on the dataset $f(x)$ yields

$$\mathcal{F}\{g(f(x))\} = \sum_{i=1}^{n} a_i \mathcal{F}\{B_i(f(x))\}, \qquad (3)$$

   Thus $n$ pretransformed volumes have to be initiated to perform segmentation.

2. **convolution:** Given a convolution kernel $g(x)$ and a volume $f(x)$, the well-known Fourier convolution formula states, that

$$\mathcal{F}\{g(x) \star f(x)\} = \mathcal{F}\{g(x)\} \cdot \mathcal{F}\{f(x)\} \qquad (4)$$

   This means, that during the slicing phase, the single samples in $\mathcal{F}\{f(x)\}$ are weighted width samples in $\mathcal{F}\{g(x)\}$ at the same sampling position.

3. **multiplication by a function of position in the spacial domain:** Analogously, a multiplication of $g(x)$ with $f(x)$ corresponds to a convolution in the Fourier domain:

$$\mathcal{F}\{g(x) \cdot f(x)\} = \mathcal{F}\{g(x)\} \star \mathcal{F}\{f(x)\} \qquad (5)$$

   For these type of operators, Levoy discusses, how one can get rid of the expensive convolution by smartly choosing the resampling filter for the slice extraction phase.

Our method has some resemblance with the first of the three types of segmentation operators mentioned above, but there is a fundamental obstacle, which makes an direct application for our problem impossible. In equation (1), we assume function, which is actually bivariate, since it is not only depending on the argument $u$, but also on the threshold value $\tau$. Application of the FFT would lead to

$$\mathcal{F}\{g_\tau(f(x))\} = \sum_{i=1}^{n} a_i \mathcal{F}\{B_i^\tau(f(x))\}, \qquad (6)$$

i.e., the parameter $\tau$ can not be pulled out of the argument of the Fourier transform and hence all datasets running over $i$ have to be retransformed for a change of $\tau$. The main achievement of this paper is to resolve this conflict and to present a smart factorization, where the dependency of the argument of $\mathcal{F}\{\}$ from $\tau$ can be removed in order to achieve interactive manipulation of $\tau$.

Malzbender's paper, in contrary, focusses on signal-theoretical aspects, like derivation of FVR from the Projection-Slice theorem, the choice of resampling filters and antialiasing. The latter two issues are in particular important for the quality of the renderings, as larger resampling filters with higher accuracy have to be traded for rendering speed.

Gross et al.[5] proposed two types of Wavelet-based volume rendering methods: (i) Wavelet Space Ray-Tracing and (ii) Wavelet Based Volume Splatting (WS). By combining FVR and Wavelets, WS can circumvent expensive interpolation filters used in pure FVR. Westenberg and Roerdink [19] extended the idea of WS to Fourier-Wavelet Volume Rendering (FWVR). The benefit of using these methods over FVR are their capability of allowing local level-of-detail.

Totsuka and Levoy [18] introduced shading for FVR, which was later improved in quality by Entezari et al. [3]. The method of Entezari et al. has great similarities with our algorithm. While they derive a method for interactively shading the volume using SHRMs, our contribution is tailored to the classification model described above. Although there is an analogy between their method and ours, our derivation is significantly different and does not require lighting calculations with spherical harmonics expansions.

Kaneda et al. [7] developed a method for fast volume rendering using adjustable color maps. Although they proposed a spacial-domain method, their core has resemblance with our approach and is therefore discussed in subsection 5.2.

## 2.2. Transfer Functions

Transfer functions (TFs) form a classification technique for replacing voxel values locally in the dataset according to some user-defined and/or dataset specific criterion. We now review common TFs used so far for ordinary volume rendering.

In its most simple form, a 1D TF defines the replacement of a scalar value by another one. Due to its limited applicability, the importance of higher dimensional TFs was soon recognized. Levoy [11] used 2D TFs, with the opacity value in the first and the gradient magnitude in the second dimension. Multidimensional TFs [6, 8, 10, 20] use additionally high-order derivatives and curvature of the scalar function representing the data.

The process of finding good transfer functions can be structured in a variety of ways. Automatic methods are more restricted to the coloring of the single RGB channels, as proposed by Muraki et al. [16], who use a radial basis function network in combination with independent component analysis (ICA) to accomplish the aforementioned task. Semi-automatic resp. user-driven methods assist the user in finding TFs by giving valuable hints during exploration. Examples are (i) the Contour Spectrum [2], which visually summarizes the space of iso-surfaces in terms of some metrics, (ii) the Design Gallery [15], which creates an interface to the space of possible transfer functions, (iii) the technique of König and Göller [9], where thumbnail renderings give information about the spaces of data values, colors and opacity, and (iv) the ISpace [17] technique, where classification is reduced to interactive clipping in ICA space.

The advantage of above methods is that they are easy to map onto the graphics accelerator. Unfortunately, all of these methods work in the spacial domain, i.e., none of the methods could be integrated into the FVR framework so far in order to make interactive exploration possible.

## 3. Factorization of the transfer function

In this section, we derive a basic rendering equation suitable for use in FVR.

First, assume we have given a voxel position (x,y,z), the respective scalar value $u_{xyz}$ there and the transfer function in equation (1). The new voxel value after applying the transfer function is

$$g_\tau(u_{xyz}) = \left\{ \begin{array}{ll} u_{xyz} & if\ u_{xyz} \geq \tau \\ 0 & otherwise \end{array} \right\} = u_{xyz} H(u_{xyz} - \tau). \tag{7}$$

One objective of this chapter is to rewrite the Heaviside function in the preceding equation in a manner, that it can be approximated by Fourier series. We are then in the position to factorize the step function into components, which only depend on $u_{xyz}$ and $\tau$. When applying the FFT, the factor containing $u_{xyz}$ can be pretransformed and scaled by the factor containing $\tau$.

First, we concentrate ourselves on the step function $H(u_{xyz} - \tau)$. Assuming $\tau \in [0, 1]$, we can write $H$ as a convolution with the Dirac-function:

$$H(u_{xyz} - \tau) = \int_0^1 \delta(s - \tau) H(u_{xyz} - s) ds \tag{8}$$

We now want to approximate the two factors in the integral in terms of Fourier series expansions. To apply these representations later on, we take care of the integral to run over the whole period $[0, 2\pi]$:

$$H(u_{xyz} - \tau) = \int_0^{2\pi} \delta(\frac{s}{2\pi} - \tau) H(u_{xyz} - \frac{s}{2\pi}) ds \tag{9}$$

Next, we develop the Heaviside and the Dirac-delta function into Fourier expansions with the general representations

$$H(u_{xyz} - \frac{s}{2\pi}) = \sum_{k=-\infty}^{\infty} c_k(u_{xyz}) e^{iks} \tag{10}$$

and

$$\delta(\frac{s}{2\pi} - \tau) = \sum_{l=-\infty}^{\infty} d_l(\tau) e^{ils} \tag{11}$$

(Please note, that Fourier series expansion assumes periodic functions with period $[0, 2\pi]$. This issue is picked up is section 5). When plugging equations (10) and (11) into equation (9), we obtain due to the orthogonality relation of the Fourier basis functions $e^{iks}$ and $e^{ils}$

$$\begin{aligned} H(u_{xyz} - \tau) &= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} c_k d_l \int_0^{2\pi} e^{is(k+l)} ds \\ &= 2\pi \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} c_k d_l \delta_{k,l} \\ &= 2\pi \sum_{k=-\infty}^{\infty} c_k d_k \end{aligned} \tag{12}$$

The coefficients $c_k$ and $d_k$ can be found canonically by integrating the involving functions against the Fourier basis functions over the range $[0, 2\pi]$:

$$c_k(u_{xyz}) = \frac{1}{2\pi} \int_0^{2\pi} H(u_{xyz} - \frac{t}{2\pi}) e^{ikt} dt \tag{13}$$

and

$$d_k(\tau) = \frac{1}{2\pi}\int_0^{2\pi} \delta(\frac{t}{2\pi} - \tau)e^{ikt}dt \tag{14}$$

Latter equation can easy be evaluated using the substitution $s := \frac{t}{2\pi}$ and the definition of the Dirac-delta function:

$$\begin{aligned} d_k(\tau) &= \frac{1}{2\pi}\int_0^{2\pi} \delta(\frac{t}{2\pi} - \tau)e^{ikt}dt \\ &= \int_0^1 \delta(s-\tau)e^{i2\pi ks}ds \\ &= e^{i2\pi k\tau} \end{aligned} \tag{15}$$

For $c_k$, a case differentiation is required. For $k = 0$ and the same substitution $s := \frac{t}{2\pi}$,

$$\begin{aligned} c_0(u_{xyz}) &= \frac{1}{2\pi}\int_0^{2\pi} H(u_{xyz} - \frac{t}{2\pi})dt \\ &= \int_0^1 H(u_{xyz} - s)ds = u_{xyz} \end{aligned} \tag{16}$$

since $0 \leq u_{xyz} \leq 1$. For $k \neq 0$ we have analogously

$$\begin{aligned} c_k(u_{xyz}) &= \frac{1}{2\pi}\int_0^{2\pi} H(u_{xyz} - \frac{t}{2\pi})e^{ikt}dt \\ &= \int_0^1 H(u_{xyz} - s)e^{2\pi iks}ds \end{aligned} \tag{17}$$

A further substitution $r := u_{xyz} - s$ removes the dependency of the integrand from $u_{xyz}$:

$$\begin{aligned} c_k(u_{xyz}) &= -\int_{u_{xyz}}^{u_{xyz}-1} H(r)e^{2\pi ik(u_{xyz}-r)}dr \\ &= e^{2\pi iku_{xyz}}\int_{u_{xyz}-1}^{u_{xyz}} H(r)e^{-2\pi ikr}dr \\ &= e^{2\pi iku_{xyz}}\int_0^{u_{xyz}} e^{-2\pi ikr}dr \end{aligned} \tag{18}$$

The latter step is valid, since $-1 \leq u_{xyz} - 1 \leq 0$ according to our assumption. Finally,

$$c_k(u_{xyz}) = \begin{cases} u_{xyz} & if\ k = 0 \\ \frac{i(1-e^{2\pi iku_{xyz}})}{2\pi k} & otherwise \end{cases} \tag{19}$$

Now that we have closed forms for $c_k(u_{xyz})$ and $d_k(\tau)$, we can expand equation (7) as follows:

$$\begin{aligned} g_\tau(u_{xyz}) &= u_{xyz}H(u_{xyz} - \tau) \\ &= 2\pi u_{xyz}\sum_{k=-\infty}^{\infty} c_k(u_{xyz})d_k(\tau)\ (eq.\ (19), (15)) \end{aligned}$$

At this point, we can apply the 3D Fourier transform in an efficient way, since the variables $u_{xyz}$ and $\tau$ are separated in the arguments of $c_k$ and $d_k$ and the Fourier transform runs only over arguments with the scalar value $u_{xyz}$ involved:

$$\begin{aligned} \mathcal{F}_3\{g_\tau(u_{xyz})\} &= 2\pi\sum_{k=-\infty}^{\infty} d_k(\tau) \\ &\quad \cdot\ \mathcal{F}_3\{u_{xyz}c_k(u_{xyz})\} \end{aligned} \tag{20}$$

For practical issues, it is understood that only a finite number of summands can be used. In the appendix, we show, that the energy contained in the summand decays quadratically with growing $|k|$, so a quite precise approximation of the equation can be found by using only a few summands involving $c_k(u_{xyz})$ and $d_k(\tau)$.

## 4. Algorithm

From equation (20), the final algorithm can be concluded. We assume, that $D(x, y, z)$ is the input dataset and $-t \leq k \leq t$ for some small $t$.

### 4.1. Preprocessing

**P1** Initialize scalar datasets $V_k$ by setting

$$V_k(x, y, z) := D(x, y, z)c_k(D(x, y, z))$$

**P2** Apply the 3D forward FFT on each $V_k(x, y, z)$, resulting in $\widehat{V}_k(x, y, z)$.

### 4.2. Rendering

During runtime, the following steps are executed for a new viewing direction $\overrightarrow{V}$:

**R1** Sample a slice $S_k(x', y')$ through the origin of the transformed dataset $\widehat{V}_k(x, y, z)$, perpendicular to $\overrightarrow{V}$.

**R2** Weight each sample point $S_k(x', y')$ by $2\pi d_k(\tau)$.

**R3** Calculate a 2D image $I(x', y')$ by pixelwise accumulating $S_k(x', y')$.

**R4** $\mathcal{F}_2^{-1}\{I(x', y')\}$ is the desired result for the given view $\overrightarrow{V}$.

## 5. Analysis of the principal equation

### 5.1. Robustness aspects

During our derivations in the previous section, we have tacitly assumed, that the expansion of the Fourier coefficients takes place over periodic functions with period $2\pi$. This is awkward with respect to the Heaviside function

$H()$ in equation (18), since we cannot actually realize a step-function, but only a window-function with a only a finite range being unequal to zero. Since only half of the range $[0, 2\pi]$ is one, and zero, otherwise, we have repetition artifacts, when $u_{xyz}$ is shifted. We overcome this burden by pre-scaling the scalar values $u_{xyz}$ and $\tau$ to the domain $[0, \frac{1}{2}]$. Since we already work with float precision, we do not loose accuracy by applying this scaling.

It is further important to note, that complex summands appear on the right-hand side of equation (20), although on the left-hand side a real function is presumed. To guarantee a real-valued result, the index $k$ in the approximation of the sum in equation (20) should run from $-t$ to $t$, since summands for $k$ and $-k$ sum up to real-valued result (we only consider $k \neq 0$ here):

$$c_k d_k + c_{-k} d_{-k} = \frac{sin(2\pi k\tau + 2\pi k u_{xyz}) - sin(2\pi k u_{xyz})}{\pi k}$$

## 5.2. Comparison with other methods

An interesting point in our equation is its similarity to the method of Entezari et al. [3]. The authors deduce an equation for diffuse lighting $E(x, y, z)$ of the volume at the position $(x, y, z)$ using spherical harmonics $Y_{lm}$:

$$E(x, y, z) = \sum_{l} \sum_{m} \sqrt{\frac{4\pi}{2l+1}} Y_{lm}(\omega_L) Y_{lm}(\omega_{N_{xyz}}), \tag{18}$$

where $\omega_L$ is the direction of the *directional* light source and $\omega_{N_{xyz}}$ gives the direction of the normal vector at the voxel position $(x, y, z)$. In a similar fashion, the 3D FFT operator $\mathcal{F}_3\{\}$ can be applied here, where the argument of $\mathcal{F}_3\{\}$ only depends on $\omega_{N_{xyz}}$. While the basic concept of our approach using the factorization resembles that of the aforementioned approach, our main point is to tailor the base equation to classification rather than illumination.

Kaneda et al.[7] presented a ray-casting approach which allows for fast modification of color maps. The method works in two phases. First, during preprocessing, some basis images are calculated for a particular perspective, having a color map-independent representation of the view. In the second phase, the basis images are composed, where each image is weighted by Fourier moments. These weights can be derived - and thus, modified quickly- by decomposing the color maps into Fourier coefficients during rendering. The underlying ray-casting integral containing the color map function therefore can be factored out of the integral and calculated separately, as in our method. Note that the entire approach is a physical domain, rather than a frequency domain one.

We further investigated the decomposition of the transfer function in equation (7) by SVD (singular value decomposition). Assuming $|G|$ (e.g., $|G| = 256$ for 8Bit accuracy) distinct values in the representation of the scalars $u_{xyz}$ and $\tau$, we can represent the transfer function (7) in form of a matrix $A$, where

$$A_{ij} = \frac{i}{|G|} H(\frac{i-j}{|G|}). \tag{18}$$

Using SVD, this matrix can be decomposed into three matrices $U$, $V$ and a diagonal matrix $D$, where $A = UDV^T$. $A$ then can be expanded into a sum of outer products:

$$A = \sum_{k=1}^{|G|} D_{kk} U_{ik} V_{jk} \tag{18}$$

Since we assume our scalar values $\tau$ and $u_{xyz}$ to be quantized, we can define functions

$$\begin{aligned} f_\tau(k) &= U_{(|G|\tau),k} \; and \\ f_{u_{xyz}}(k) &= V_{(|G|u_{xyz}),k} \end{aligned}$$

Similar to our approach, this yields

$$\mathcal{F}_3\{g_\tau(u_{xyz})\} = \sum_{k=1}^{|G|} D_{kk} f_\tau(k) \mathcal{F}_3\{f_{u_{xyz}}(k)\} \tag{16}$$

Equality also generally holds when the $k$ sums up to $|G|$, but one might expect acceptable results for for the first few summands.

We have tested this approach and found the results comparable with the Fourier series approach. According to our experience, the Fourier based approximation allows for somewhat sharper discrimination between those voxel values to be preserved and those to be discarded from the rendition. Consequently, no unwanted portions of the volume will be rendered, whereas the desired ones will appear in the rendition. Therefore, we used the Fourier-based approach to generate our results.

## 6. Results

We have tested our algorithm on a Windows XP PC with a 3.06 GHz Intel PIV processor and 1GByte RAM with a variety of datasets, amongst other things with the Visible Female [1], the engine and the foot (obtained from `www.volren.org`). In order to reduce memory requirements, we have downsampled the datasets to voxel sizes $128^3$, $128^2 \cdot 55$ and $128^3$, respectively. Our software solution of the renderer is based on the FFTW [4] library, since it exploits the hardware environment in an optimal way. Our experiments shown in fig. 2 were carried out with 9 Fourier series coefficients. Preprocessing required about 35 seconds for any of the volumes, whereas rendering times were about

3fps for all of the datasets, for arbitrary views and for arbitrary window sizes, independently, if $\tau$ was altered or not.

In rows (1) and (2) of figure 2, we show an exploration process of the Visible Female using our method for the front and the side views, respectively. As the iso-value $\tau$ increases, soft matter, like skin and muscles disappear and the user can isolate skull and upper spinal court. This complies well to our expectations, since opacity values for bones and for the skull are distributed more in the higher part of the domain of iso-values.

These qualitative aspects can be confirmed on the foot dataset (fig. 2, row (3)), where an increasing of $\tau$ removes intensities around the bones of the toes.

We further investigated the well-known engine dataset (fig. 2, row (4)). As expected, for high iso-values $\tau$, only the valves and one side of the outer part of the engine remain after classification.

As it is evident from the examples, the overall energy in the projections decreases (i.e., the renderings appear darker), as the iso-value $\tau$ grows. This correct behavior of the algorithm is due to the tendency, that more samples are excluded from integration during projection, as $\tau$ is increased. For datasets with uniform probability distribution of the opacity values, this effect is continuously perceptible, when the threshold increases.

It is simple to deduce similar TFs using our technique. If we like to have a TF which replaces scalar values by zero above a certain threshold $\tau$, only the respective coefficients $c_k$ have to be altered, which can be done during runtime. We merely have to change the step function in equation (7) in order to obtain the desired coefficients.

Currently, our method suffers from somewhat high memory requirements, in the same fashion as the method of Entezari et al. We can further lower memory demands by using 16 Bit instead of 32 Bit precision for the representation of values in the Fourier domain, as suggested by [18] and [13]. However, FFTW currently does not support 16 Bit floats to our knowledge.

## 7. Conclusions and Future Work

We have introduced a special class of transfer functions for Fourier Volume Rendering, which allows interactive integration of samples values greater than a given threshold $\tau$ during projection. This opens fundamentally new perspectives in FVR classification, since earlier approaches were strongly restricted to low-level segmentation operators, as expressible by linear combination of some basis functions, convolutions or multiplications. Our method, in this sense, is a non-linear operator, and allows the interactive classification not only in dependence of position, but also of the voxel value itself. For transfer functions not matching the described ones, however, classification still has to be deferred to the preprocessing stage and interactive manipulation is not possible.

Our method is of particular interest for medics and physicist who work with order-independent projections of the input dataset and who like to classify structures according to the given scheme interactively. As for every FVR technique, our method guarantees the usual $\mathcal{O}(N^2 logN)$ rendering time complexity, which is in particular attractive with respect to rendering speed in future, when dataset sizes $N^3$ increase. Concerning hardware prerequisites, the proposed technique is useful on platforms were hardware acceleration is only weakly or not present (like PDAs), since the entire algorithm runs on the CPU.

Our FVR approach leaves scope for further development, especially for other TF types. We are convinced, that it will be possible to compose more sophisticated -even though not universal- TFs by our method in future. In all we strongly believe to have found a new directive in FVR, which makes exploration of datasets in Fourier space with its computational advantages a more viable and attractive alternative to spacial domain methods.

## A. Appendix

### A.1. Approximation accuracy

To estimate the error we make by using only a few summands in equation (20), we investigate the energy contained in the single coefficients $c_k$ and $d_k$ as follows. Let $W(k)$ be the energy of summand $k$. We obtain by defining $a := 2\pi k u_{xyz}$ and $b := 2\pi k\tau$ for $k > 0$:

$$\begin{aligned}
W(k) &:= (u_{xyz}c_k d_k)\overline{(u_{xyz}c_k d_k)} \\
&= u_{xyz}^2(c_k\overline{c_k})(d_k\overline{d_k}) \\
&= u_{xyz}^2(c_k\overline{c_k})(e^{ib}\overline{e^{ib}}) = u_{xyz}^2(c_k\overline{c_k}) \\
&= u_{xyz}^2\left(\frac{i(1-e^{ia})}{2k\pi}\overline{\frac{i(1-e^{ia})}{2k\pi}}\right) \\
&= u_{xyz}^2\frac{1}{4\pi^2k^2}((1-e^{ai})\overline{(1-e^{ai})}) \\
&= u_{xyz}^2\frac{1}{2\pi^2k^2}(1-\cos(a)) \\
&\leq \frac{1}{\pi^2k^2} \in \mathcal{O}(\frac{1}{k^2}) \qquad (11)
\end{aligned}$$

latter inequality holds, since $0 \leq u_{xyz} \leq 1$. Thus the energy of the summands decays quadratically with increasing $k$. Since $\sum_{k=1}^{\infty}\frac{1}{k^2} = \frac{\pi^2}{6}$, we can estimate a *worst-case upper bound* for the residual energy of our entire expansion by calculating

$$\begin{aligned} R &:= u_{xyz}^2 + 2(\sum_{k=1}^{\infty}\frac{1}{k^2})\frac{1}{\pi^2} - (u_{xyz}^2 + 2(\sum_{k=1}^{t}\frac{1}{k^2})\frac{1}{\pi^2}) \\ &= \frac{1}{3} - \frac{2}{\pi^2}\sum_{k=1}^{t}\frac{1}{k^2} \end{aligned} \qquad (11)$$

where $t > 0$ defines the number of coefficients. For instance, for $t = 6$, we have a *worst-case* error of about $9.3\%$.

## References

[1] National library of medicine. THE VISIBLE HUMAN PROJECT ®. `http://www.nlm.nih.gov/research/visible/visible_human.html`.

[2] C. Bajaj, V. Pascucci, and D. Schikore. The contour spectrum. *IEEE Visualization*, pages 167–173, 1997.

[3] A. Entezari, R. Scoggins, T. Möller, and R. Machiraju. Shading for fourier volume rendering. *IEEE VolVis*, pages 131–138, 2002.

[4] M. Frigo and J. S.G. Fftw: An adaptive software architecture for fft. *ICASSP conference proceedings*, 3:1381–1384, 1998. `http://www.fftw.org`.

[5] M. Gross, L. Lippert, R. Dittrich, and S. Häring. Two methods for wavelet-based volume rendering. *Computers & Graphics*, 21(3):237–252, 1997.

[6] J. Hladůvka, A. König, and E. Gröller. Curvature-based transfer functions for direct volume rendering. *Spring Conference on Computer Graphics*, 16:58–65, 2000.

[7] K. Kaneda, Y. Dobashi, K. Yamamoto, and H. Yamashita. Fast volume rendering with adjustable color maps. In *Proceedings of the 1996 symposium on Volume visualization (VolVis 96)*, pages 7–15. IEEE Press, 1996.

[8] G. Kindlmann and J. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of the 1998 IEEE symposium on Volume visualization (VolVis 98)*, pages 79–86. ACM Press, 1998.

[9] A. König and E. Gröller. Mastering transfer function specification by using volumepro technology. *Spring Conference on Computer Graphics*, 17:279–286, 2001.

[10] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE TVCG*, 8(3):270–285, 2002.

[11] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics & Apllications*, 8(5):29–37, 1988.

[12] M. Levoy. Volume rendering using the fourier projection-slice theorem. *Graphics Interface '92*, pages 61–69, 1992.

[13] B. Lichtenbelt. Fourier volume rendering. *Technical Report `HPL-95-75`. Hewlett Packard Laboratories*, pages 1–69, November 1995. `http://www.hpl.hp.com/techreports/95/HPL-95-73.html`.

[14] T. Malzbender. Fourier volume rendering. *ACM Transactions on Graphics*, 12(3):233–250, July 1993.

[15] J. Marks, B. Andalman, P. Beardsley, and H. Pfister. Design galleries: A general approach to setting parameters for computer graphics and animation. *ACM Computer Graphics (SIGGRAPH 97')*, pages 389–400, August 1997.

[16] S. Muraki, T. Nakai, Y. Kita, and K. Tsuda. An attempt for coloring multichannel mr imaging data. *IEEE TVCG*, 7(3):265–274, July-September 2001.

[17] I. Takanashi, E. Lum, K.-L. Ma, and S. Muraki. Ispace: Interactive volume data classification techniques using independent component analysis. $10^{th}$ *Pacific Graphics Conference on Computer Graphics and Applications (PG'02)*, pages 366–374, 2002.

[18] T. Totsuka and M. Levoy. Frequency domain volume rendering. *Computer Graphics Proceedings, Annual Conference Series*, pages 271–278, 1993.

[19] M. Westenberg and J. Roerdink. Frequency domain volume rendering by the wavelet x-ray transform. *IEEE Transctions On Image Processing*, 9(7):1249–1261, 2000.

[20] S. Yoshinabu, C.-F. Westin, and A. Bhalerao. Tissue classification based on 3d local intensity structures for volume rendering. *IEEE TVCG*, 6(2):160–179, 2000.

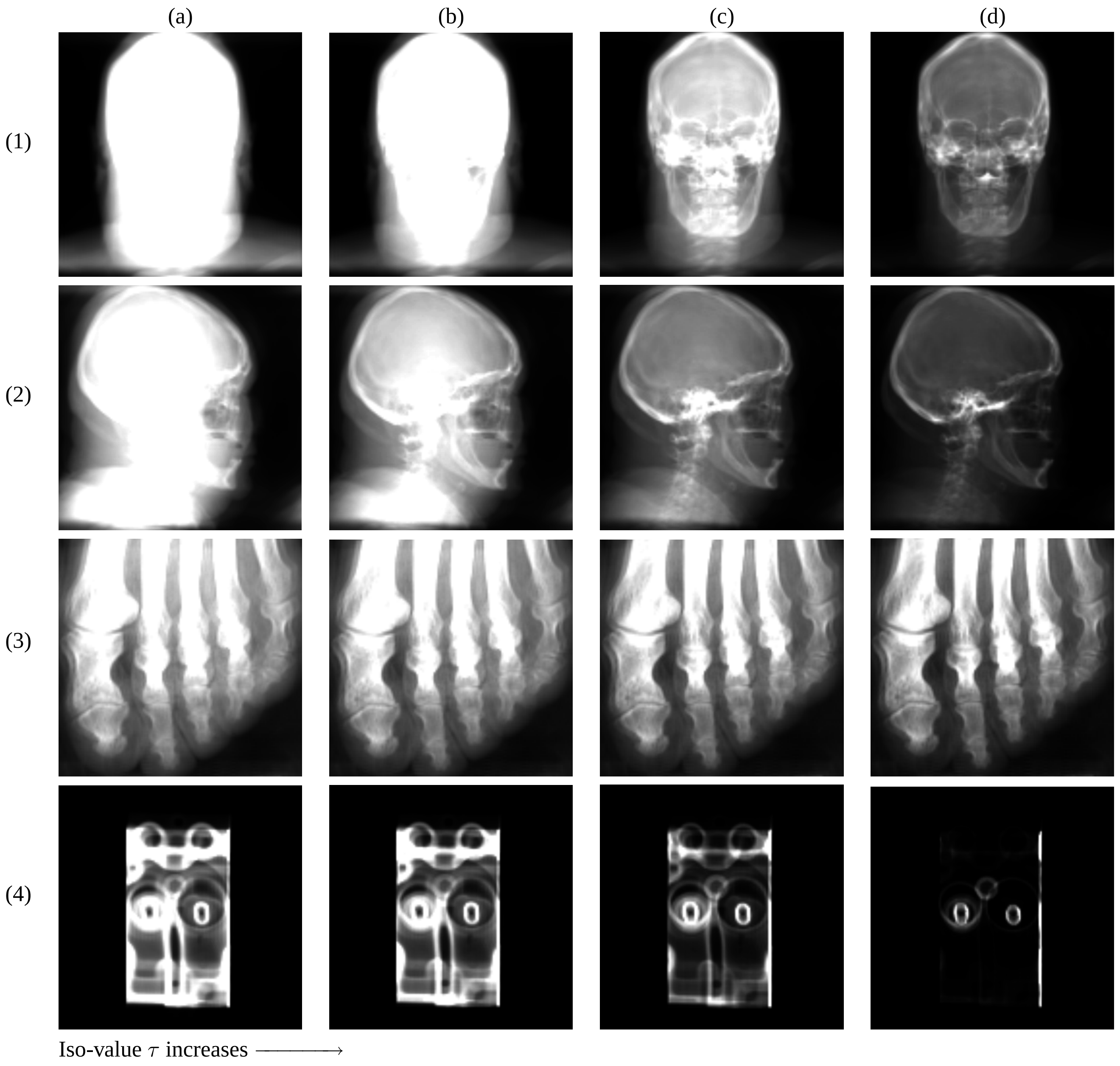

**Figure 2. *Rendering results*. Rows (1) (front view) and (2) (side view) show a classification process, exemplified on the VISIBLE FEMALE dataset. The images were generated for iso-values $\tau = \frac{10}{255}, \frac{30}{255}, \frac{50}{255}, \frac{70}{255}$. As can be seen from the examples, matter of high density like bones are properly separated from those of lower ones, like soft matter, as skin, etc. Row (3) shows the foot dataset, for iso-values $\tau = \frac{10}{255}, \frac{20}{255}, \frac{30}{255}, \frac{40}{255}$. The removal of soft matter is here especially remarkable in the upper half of the images. Row (4): Engine dataset for $\tau = \frac{40}{255}, \frac{80}{255}, \frac{120}{255}, \frac{160}{255}$. For $\tau = 160/255$, only the valves and the side part of the engine remain visible.**