ISBN: 9978-989-8533-66-1 © 2017

DOWNSAMPLING METHODS FOR MEDICAL DATASETS

Jesús Díaz-García, Pere Brunet, Isabel Navazo and Pere-Pau Vázquez
ViRVIG Group - Universitat Politècnica de Catalunya, C/ Jordi Girona, 1-3, Barcelona, Spain

ABSTRACT

Volume visualization software usually has to deal with datasets that are larger than the GPUs may hold. This is especially true in one of the most popular application scenarios: medical visualization. Typically, medical datasets are available for different personnel, but only radiologists have high-end systems that are able to cope with large data. For the rest of physicians, usually low-end systems are only available. As a result, most volume rendering packages downsample the data prior to uploading to the GPU. The most common approach consists in performing iterative subsampling along the longest axis, until the model fits inside the GPU memory. This causes important information loss that affects the final rendering. Some cleverer techniques may be developed to preserve the volumetric information. In this paper we explore the quality of different downsampling methods and present a new approach that produces smooth lower-resolution representations, yet still preserves small features that are prone to disappear with other approaches.

KEYWORDS

Medical visualization, volume rendering, downsampling, Gaussian filtering

1. INTRODUCTION

Medical datasets are continuously increasing in size. Although larger models may be available for certain research purposes, in the common clinical practice the models are usually of up to 512 × 512 × 2000 voxels. These resolutions exceed the capabilities of conventional GPUs, the ones usually found in the medical doctors' desktop PCs. As a result, commercial solutions typically reduce the data by downsampling the dataset iteratively until it fits the available target specifications. The data loss reduces the visualization quality and this is not commonly compensated with other actions that might alleviate its effects.

Medical datasets are commonly stored as scalar fields, where each scalar represents the properties from the human anatomy (such as density values in Computational Tomography, CT). Downsampling scalar data implies an inevitable loss of information that provokes the loss of fine details and the modification of the original topology. Furthermore, since these scalar values are then used by means of Transfer Functions (TF, functions that map density values to opacity and color), downsampling a set of density values (e.g. by averaging) leads to inconsistencies when the transfer function is applied on coarser representations.

The downsampling process appears in many rendering techniques, such as the approaches that render large datasets by building a multiresolution structure (Boada et al., 2001). In these cases, intermediate nodes are also generated by subsampling the high resolution dataset. Again, using a simple downsampling scheme will produce important artifacts, such as the disappearance of small features. Some attempts have been developed to preserve the data, but they are commonly quite costly in terms of processing or data storage (Younesy et al., 2006, Sicat et al., 2014), or not suitable for medical datasets, such as when downsampling color data (Kraus and Brüger, 2008).

In this paper we analyze several downsampling methods and present a new approach that preserves small features for medical models, while keeping a low computation cost. Our contributions are twofold:

- An analysis of the effect of the most popular downsampling methods for medical datasets.
- A new low-cost downsampling filter tailored to preserve small features that can be combined with other methods such as multiresolution or bricking approaches.

The rest of the paper is organized as follows: Section 2 presents the previous work, Section 3 provides visual comparisons of previous techniques, and analyzes the factors that influence the quality of the final downsampled model. Section 4 presents our feature-preserving technique and discuss its advantages and shortcomings. Finally, Section 5 concludes this work by discussing the results and presenting some lines of future research.

2. PREVIOUS WORK

Medical imaging is challenged by the continuous increase in size of the datasets produced by capture devices (such as CT scanners). Even though GPUs also evolve, their horsepower lies behind the size of the data. This is especially true for physicians that receive the data from radiologists, since their workplaces are also commonly shared with other physicians, and, not being their main task the visualization of such data, they are not usually equipped with powerful GPUs. Thus, the data must be reduced in order to make it fit into their commodity GPUs and enable interactive inspection. Note that, this problem appears frequently in the clinical practice, since the datasets being captured continuously increase in size, and the GPUs cannot keep track with the growing size of the datasets.

As a result, a common approach is to compress the data and render from it. Since we want a solution that is applicable to modest PCs, this is often not a possibility since it implies large computation costs, and modern GPUs to perform the rendering. The interested reader can refer to the recent State of the Art by Balsa et al. (Balsa et al, 2014) for an introduction on compression for volume models.

Wang et al. (Wang et al., 2011) concentrate on the preservation of features in data reduction. This is done by assigning importance to the voxels according to the current transfer function. However, this requires an analysis of the dataset, and it is transfer function (TF) dependent. Therefore, if the TF changes, this computed data becomes invalid and has to be recomputed.

Younesy et al. (Younesy et al., 2006) improve the quality of the renderings in a multiresolution scheme by storing extra information for each node. Their initial analysis states that the ideal condition would be to store local histograms per coarse voxel, but in their implementation, these are substituted by an average density and its standard deviation. This multiplies the size of the dataset by three or four (depending on the precision of both values), and thus is impractical again for our case.

Sicat et al. (Sicat et al., 2014) present a technique for the storage of large datasets in a compressed way. They improve the consistent visualization of multiresolution volume datasets in comparison with the method of Younesy et al. by means of sparse *pdf volumes.* Again, this method does not suit our needs because it requires a higher amount of precomputation to generate their sparse structure, and uses important GPU resources for the decompression and rendering.

Díaz-García et al., (Díaz-García et al., 2016) perform a good work at *improving* the compressed dataset by applying what they call *adaptive transfer functions,* an algorithm that consists in using a small auxiliary structure that allows computing fitted TFs for the inner nodes in a multiresolution dataset. Their method succeeds at improving the final quality, but it is somewhat orthogonal to the analysis performed here. In any case, their approach can be combined with the filtering we present here, that can be used on the computation of the initial internal nodes dataset.

Kraus and Bürger (Kraus and Bürger, 2008) improve the reduction of the models, but using RGBA data. Since the medical models store the information as density values, and render them through the Transfer Functions, using colors has several shortcomings. First, using direct colors would multiply the model sizes by 4. Second, as already said, the interpolation in color space may produce larger artifacts than in density space, and therefore this plain interpolation is not suitable for medical models.

Kraus and Ertl. (Kraus and Ertl, 2001) also presented a method (named *topology-guided downsampling*) for downsampling volumetric datasets that tries to preserve the topology of the models. However, when applied to medical datasets, the results are limited, since, although the small features are preserved, the appearance of the subsampled models presents strong staircase artifacts and distort the position of the features (as shown later in this paper).

There is another family of methods tailored to improve the quality of the volumetric datasets by reducing noise. These are more suitable for datasets with high amount of noise and low resolution, such as the ones from ultrasound imaging. In general, such techniques are more focused on noise reduction, but sometimes they also combine the noise reduction with downsampling. Kwon et al. (Kwon et al., 2016), for instance, filter and denoise ultrasound images using a fast bilateral filter. As we will see in the next section, bilateral filter usually produces undesirable results for downsampling medical datasets such as the ones obtained from CTs. Another technique, more focused on noise reduction, is the method by Wang et al. (Wang et al., 2012). In our analysis, using noise reduction previous to subsampling does not improve the quality of the downsampled volumes.

There is also related bibliography about feature preserving downsampling methods for 2D images. Although these methods have not been designed specifically for volumetric datasets, their idea could be easily adapted to such case. For instance, Kopf et al. (Kopf et al. 2013) implement a content-adaptive downscaling method that uses a bilateral combination of two Gaussian kernels defined over space and color, calculated by means of an iterative process that can take long to finish. Öztireli and Gross (Öztireli and Gross 2015), on the other hand, formulate image downscaling as an optimization problem that maximizes a perceptual image quality metric based on the difference between the input and the output images. Also, more recently, Weber et al. (Weber et al. 2016) present an algorithm based on convolutional filters where input pixels contribute more to the output image the more their color deviates from their local neighborhood. They use a guidance image based on a preliminary downscaled image to be able to define pixel contributions, and a parameter λ that has to be manually adjusted, depending on the image, to provide the best results.

Besides these improved approaches, other typical filtering methods, such as averaging or Gaussian filtering will also be analyzed in the following section together with some of the aforementioned algorithms.

3. DOWNSAMPLING METHODS FOR MEDICAL DATASETS

We have analyzed several downsampling methods with volumetric datasets that range from the typical pure subsampling (e.g. taking one of the samples of the original dataset) to more elaborate approaches such as the topology-guided downsampling by Kraus and Ertl (Kraus and Ertl, 2001). The most important problem with most of the techniques is the lack of preservation of fine details. The analyzed techniques are:

- Subsampling
- Averaging
- Gaussian filtering
- Bilateral filtering
- Topology-guided downsampling (Kraus and Ertl, 2001)

For completeness, we have also experimented with other possibilities such as combining any of the previous downsampling methods with an extra previous noise reduction stage. Unfortunately, no significant gains are obtained. So, for the sake of clarity, we only comment the tested methods that exhibit noteworthy behavior and analyze the reasons behind their results. The results shown in the Figures of this paper have been obtained by ray-casting the volume models with appropriate transfer functions (TF). The density of samples along rays has been set to 1 sample per voxel of the original model, more than enough for the downsampled models. It should be observed that we are not performing an intrinsic analysis of these downsampling techniques, but an analysis of the resulting visual quality when they are used in the context of TF's and ray-casting volume rendering.

Subsampling refers to the method that, when downsampling a volume, for each subvolume of 8 voxels (2x2x2), chooses as the representative for the downsampled subvolume just one of the original values. In our case, we select one of the corners, although we did experiments with other samples with equivalent results.

The **averaging** filter simply consists in creating a new value by averaging all the values from the original volume. In our case, for one-level downsampling, this would consist in averaging the 8 values (2x2x2 subvolumes) of the upper level.

Besides those two simple methods, more elaborated techniques include Gaussian filtering and bilateral filtering. Both techniques have been extensively used in image processing algorithms. The core feature of Gaussian filtering is the noise reduction, at the cost of blurring edges. On the contrary, the bilateral filter is intended to preserve edges. In the following, we introduce both filters using the notation by Paris et al. (2007).

Gaussian filtering is a typical operation in image processing that consists in applying a low-pass filter over the image samples. Given an image *I*, the filtered version is computed as a weighted average of the neighboring pixels. So pixel *p* will have a value that depends on the pixels around a neighborhood *q* of the original pixel. Typically, this is denoted as:

$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in\mathcal{S}} G_{\sigma}(\|\mathbf{p}-\mathbf{q}\|)\, I_{\mathbf{q}},$$

where S is the kernel support and G is a two-dimensional filter kernel expressed as:

$$G_\sigma(x) = \frac{1}{2\pi\,\sigma^2}\ \exp\left(-\frac{x^2}{2\,\sigma^2}\right)$$

In our case, to downsample volumetric models, we apply this transformation in 3D, so the neighborhood, instead of consisting of a rectangular region, corresponds to a cube.

The **bilateral filter,** similarly to the Gaussian filter, is a weighted average of a neighborhood of pixels. The main difference is that the bilateral filter, besides using a spatial Gaussian kernel, takes into account the variation in the values of the pixels to preserve the edges. The rationale behind this is that two pixels should be weighted similarly, not only if their positions are similar, but also if their intensities are comparable too. This results in a formulation such as:

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q}\in\mathcal{S}} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_r}(I_{\mathbf{p}} - I_{\mathbf{q}})\, I_{\mathbf{q}}$$

where W is a normalization kernel defined as:

$$W_{\mathbf{p}} = \sum_{\mathbf{q}\in\mathcal{S}} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_r}(I_{\mathbf{p}} - I_{\mathbf{q}})$$

The bilateral filter appeared in several independent publications (Aurich and Weule, 1995, Smith and Brady, 1997, Tomasi and Manduchi, 1998), and has since then further improved, especially for speed (Paris and Durand, 2009).

Topology-guided downsampling is a method by Kraus and Ertl (Kraus and Ertl, 2001) that better preserves the topology of the scalar field by preferably selecting the values of critical points. To do this, every voxel in the original dataset is classified as a regular, a saddle, or an extremum point. Then, during dowsampling, the most representative sample in the neighborhood of each voxel is taken.

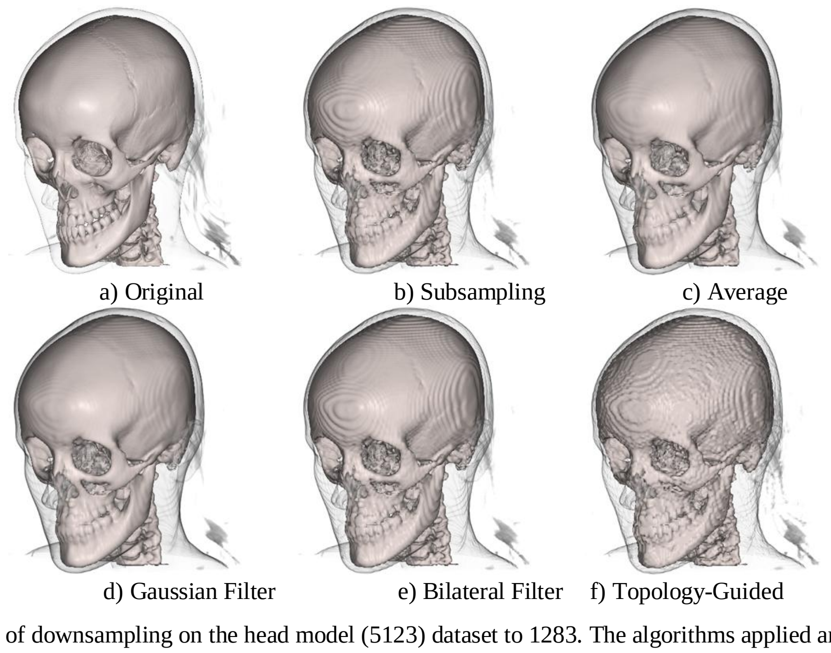
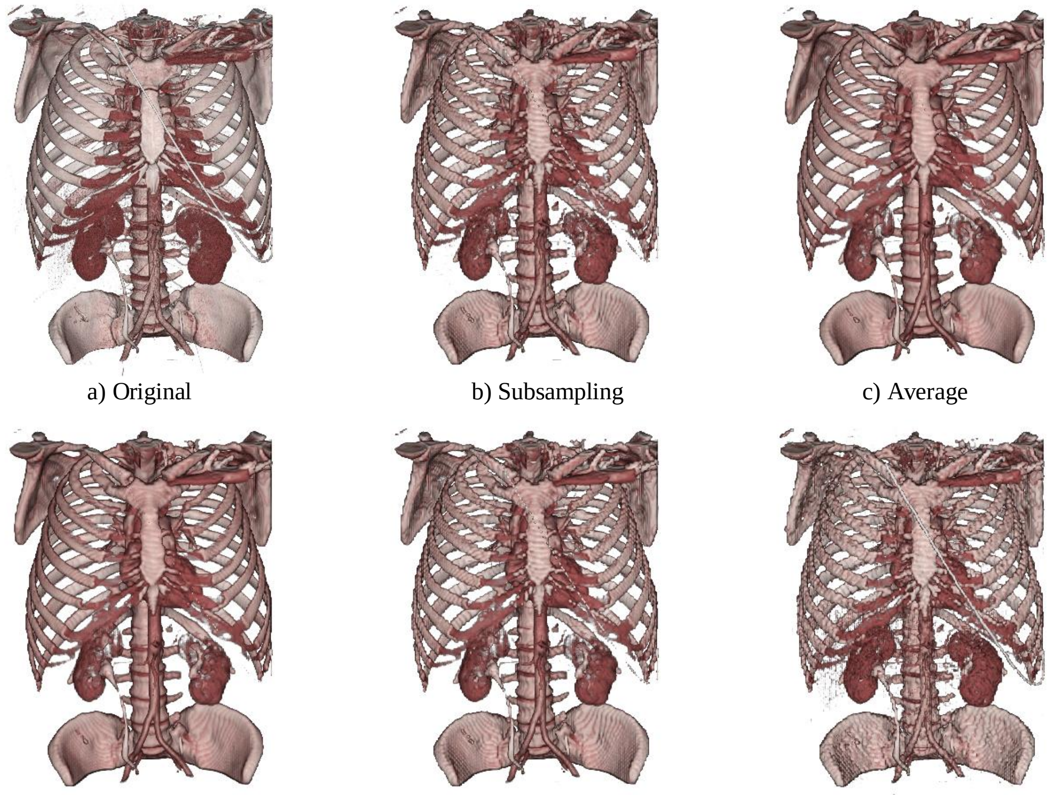


Figure 1. Effect of downsampling on the head model (5123) dataset to 1283. The algorithms applied are b) simple subsampling (taking one of the samples from the original model), c) averaging the 8 voxels from the higher resolution model, d) Gaussian filtering (σ=0.7 in voxel units), e) bilateral filtering (here we take σ=0.7 in voxel units, and σ=6 for the intensity-based Gaussians, in a range from 0 to 255), and f) topology-guided downsampling. The examples show that none of the most elaborate filters improve over the average or Gaussian filtering



a) Original b) Subsampling c) Average

d) Gaussian Filter (σ=0.7) e) Bilateral Filter (σ’s=0.7 & 6) f) Topology-Guided

Figure 2. Effect of downsampling on the body model (5123) dataset to 1283. Compared to the previous example, here the lack of preservation of small features is clearly visible: note how some tube-shaped elements disappear partially or even completely (in b-e). The results for the topology-guided downsampling are far from satisfactory. Even if the small features are in general preserved, the staircase artifacts make the result not usable for medical visualization

The effect of those algorithms onto medical data is illustrated in Figures 1 and 2. Note that the supposedly better choices, such as the bilateral filter and the topology-guided downsampling produce worse results than simpler approaches such as the Gaussian filter or the average filtering. Even just subsampling the model seems to produce comparable or even better results than those approaches. This is in part due to the fact that the relation between the filtered volume and the final rendered image is TF-dependent and highly non-linear, with the consequence that such successful image processing techniques show poor performance in terms of the quality of the final images. These artifacts are more visible for models that have small features such as the body model in Figure 2, where the loss of small features is evident. Furthermore, even though Topology-guided downsampling preserves some fine details, it produces undesirable, bumpy images, due to the fact that its sampling strategy modifies the location of the critical points.

Summarizing, the main disadvantages of the exposed methods for downsampling are the loss of details and the bumpiness of the produced results. The images shown in Figures 1 and 2 are just a subset of the experiments we carried out, but they are representative.

In order to overcome these problems, we have developed a new, Gaussian-based downsampling method that attempts to preserve small features and still produces smooth results at a low cost, and that requires no parameterization, so it runs completely unattended, without any need for user intervention. The implementation details of the proposed technique are presented in the next section.

4. FEATURE-PRESERVING VOLUME DOWNSAMPLING FILTER

4.1 Overview

The main objectives of our downsampling filter are:

- Preservation of details
- Fully automatic
- Low cost

In order to preserve details, our algorithm first simulates a downsampling step based on Gaussian filtering, and uses this information as a guidance image to give more importance to those regions that would previously suffer from excessive degradation so that features are preserved. Then, the real downsampling is performed. Another important factor is that the process should run unattended, that is, it should not require manual tweaking of parameters for good results. The rationale behind this is that the filter is intended for use with medical models, to rapidly downsample larger models that would not fit into the GPU. This downsampling should be done automatically, since no expert supervision can be carried out. As a result, the process must be robust to different models, and automatic.

4.2 Implementation

Given a volumetric scalar field V_n of resolution 2^n, to compute a coarser representation V_{n-k}; $k > 0$, our downsampling technique proceeds in three steps:

- First, a simulated coarser volume S_{n-k} is computed by means of Gaussian-filtering and subsampling.
- Secondly, the distance between V_n and S_{n-k} provides hints about the loss of features in the simulated downsampling step. Using this information we generate a filtered output F_n using Local Feature Kernels, which better preserve features that would otherwise disappear with standard filtering and subsampling.
- Finally, F_n is subsampled to obtain V_{n-k}.

At a more detailed level, to compute the filtered value F_n at each discrete sample position x we perform the following convolution:

$$F_n(x) = \sum_{i \in B_r} V_n(x+i) \cdot f_x(i)$$

where f_x is the normalized Local Feature Kernel at the position x with support B_r, a ball of radius r centered at the origin. f_x is in turn a product of a normalized global Gaussian kernel g and a distance kernel d_x:

$$f_x(i) = \frac{1}{\alpha} \cdot g(i) \cdot d_x(i), \quad \forall i \in B_r$$

The denominator is computed as:

$$\alpha = \sum_{i \in B_r} g(j) \cdot d_x(j)$$

and it ensures the sum of weights in f_x equals one. The distance kernel d_x is defined as the normalized absolute distance of values in the neighborhood of x between the original scalar field V_n and the simulated Gaussian-downsampled scalar field S_{n-k}:

$$d_x(i) = \frac{1}{\beta} |V_n(x+i) - S_{n-k}(x+i)|, \quad \forall i \in B_r$$

Again, the denominator ß ensures the normalization of weights in d_x. Note that V_n and S_{n-k} have different resolutions; sample positions in the kernel domain happen to be aligned with the center of V_n's voxels, but density values from S_{n-k} must be computed by tri-linear interpolation. The distance kernel assigns larger weights to those samples in V_n that are prone to disappear (those which most differ with S_{n-k}). As both g and d_x are combined into f_x, smoothing or sharpening is done depending on their weights, which are local to the filtered sample position x. Homogeneous regions will provoke homogeneous distance kernels, thus letting g_x filter the most part, whereas feature regions will provide more characteristic distance kernels for the feature selection task. Figure 3 shows a visual overview of the algorithm.

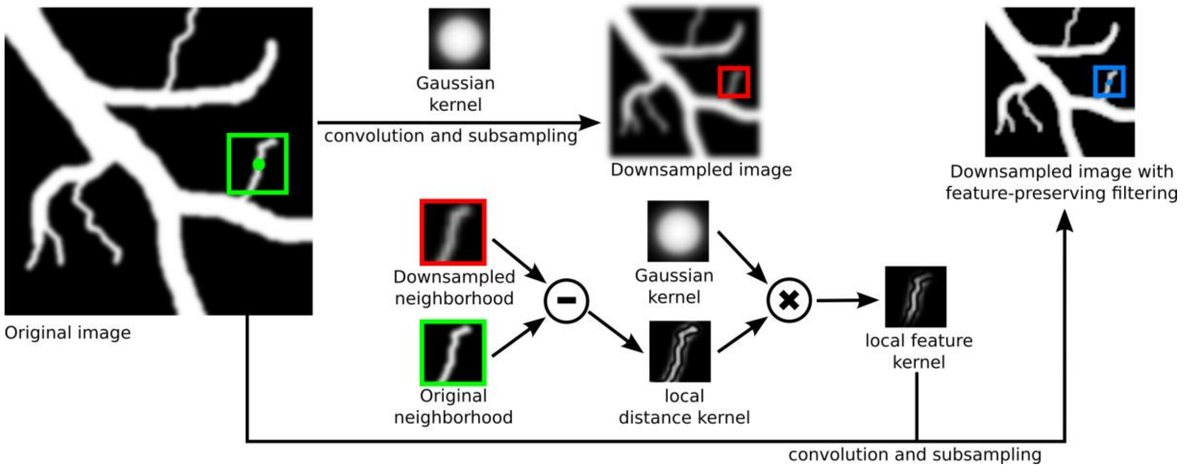
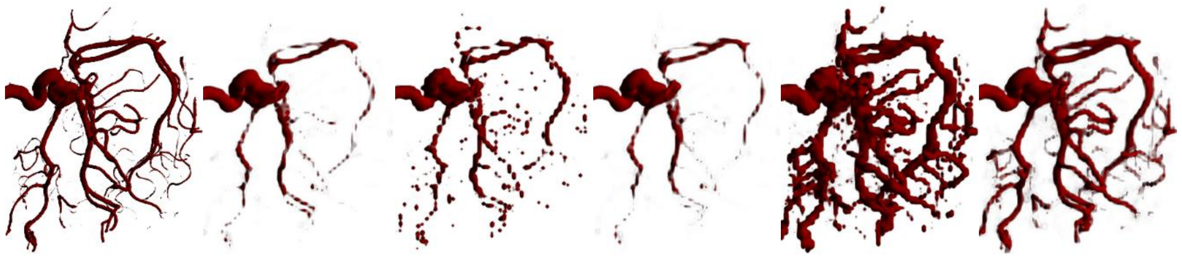


Figure 3. Overview diagram of our feature-preserving downsampling filter using Local Feature Kernels. As seen in the image, the technique first needs to perform a Gaussian-based downsampling to generate a guidance downsampled dataset. Then, the original dataset can be filtered with Local Feature Kernels, which are made of a combination of a Gaussian kernel and a distance kernel computed from the sample-by-sample neighbourhood differences between the original and the guidance datasets

4.3 Results

Without any optimization, running on a commodity PC (Intel Core i7 CPU, 8GB RAM) our algorithm takes up to 4 minutes in order to compute the downsampling for the thorax model from 512^3 to 128^3 voxels. Although it is a few times more costly than standard Gaussian-downsampling, it's affordable for a preprocess and scales linearly with the input dataset resolution, plus it clearly achieves higher quality results.

In all the comparisons, we reduce the models a ratio of 64:1, going from 512^3 to 128^3 in Figures 1, 2 and 5, and from 256^3 to 64^3 in Figure 4, for an aneurysm model with a lot of small features. Note how our method improves over all the others.



a) Original b) Average c) Bilateral filter d) Gaussian e) Topology-guided f) Our approach

Figure 4. Performance of our downsampling filter for an aneurysm, a model with a lot of small features. Most methods lose a lot of details, or result in overly exaggerated features, leading to undesirable visual artefact effects. Our system (shown in f) is the one that performs best

Figure 5 shows the effect of our downsampling filter compared to the averaging and Gaussian filtering methods for the previous shown models. We can see that for large models with small features, such as the thorax model on top, many of those details are preserved. Note how the ribs have softer shape, and the kidneys are not so transparent as in Figures 5b and 5c. The catheter going from the left shoulder to the hip is also quite preserved, such as the ureter.

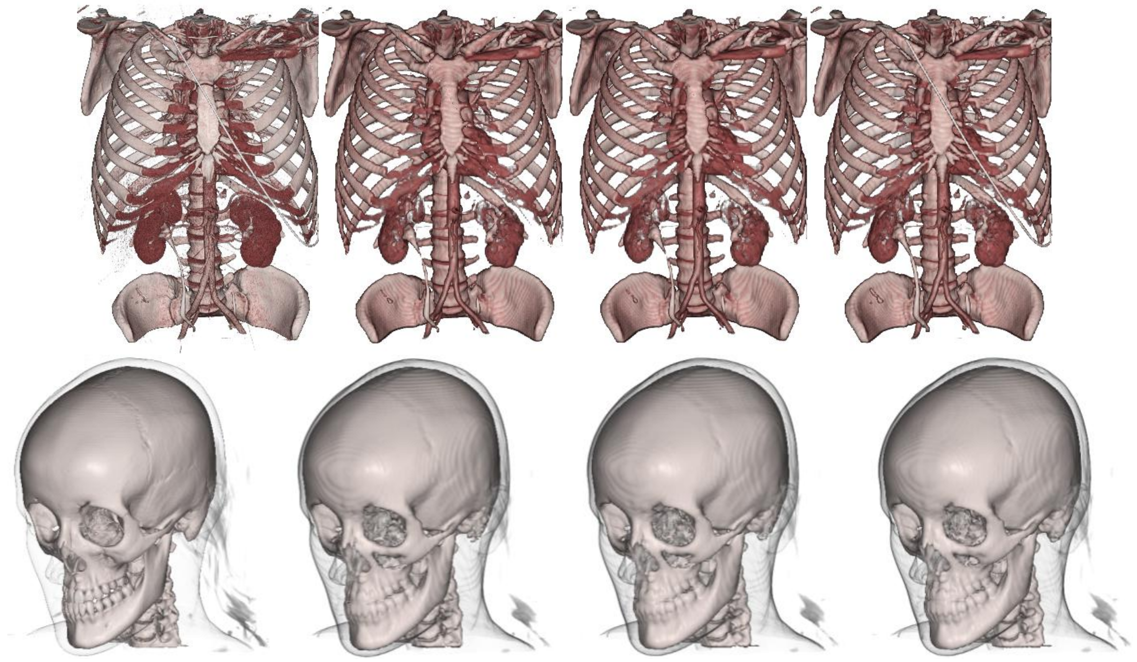


Figure 5. Our improved feature-preserving filter (d) improves the downsampling of the other approaches by preserving more small details and avoiding the ugly staircase effect that may appear using other techniques. The original dataset (a) has 5123 resolution, while the downsampled models have 1283. The Gaussian filter uses again the σ=0.7 as compared to voxel size. Notice how in the thorax model (top) some of the small features such as the catheter and some of the vessels are better preserved than with the other filters. The head model again preserves smaller features such as the creases of the bones better, while partially avoiding the staircase effects that are more noticeable in (b) both in the skin and the skull

5. CONCLUSIONS AND FUTURE WORK

Since medical models are now commonly available (you can get the data of your CT or MRI in a DVD after a medical test), the need of rendering such models in commodity PCs has grown. So downsampling is a key step in several scenarios, such as when the data must fit a small GPU, and when the models are huge enough that even the best GPU in the market cannot hold it. In these last cases, building a multiresolution scheme also involves downsampling the models to build the intermediate nodes.

In this paper we have analyzed several techniques for downsampling volumetric models and we have developed a new filtering method that adaptively smooths or preserves features depending on the topology of the surrounding region of the voxel being filtered. We do this by generating Local Feature Kernels. These are the product of a smoothing Gaussian kernel and a local distance kernel derived from the absolute distance between the full resolution dataset and a previous downsampling simulation. Our proposed heuristic identifies zones which are potentially likely to disappear, and gives them more importance when filtering. As a result, they are preserved after subsampling. We have shown that the technique improves the quality of several models with respect to other downsampling techniques, and it is especially suited for models with small features, such as the aneurysm. In our experiments we also found that the improvement over some models was more limited, but in any case, the quality was at least analogous to Gaussian filtering. Thus, our filter can be safely used in a vast majority of models.

Since the key of this filtering method resides in suitably adjusted Local Feature Kernels, we believe that our process can still be improved. In this line, we plan to further explore additional ways to determine their weights, variance, and radius.

ACKNOWLEDGEMENT

The material in this paper is based upon work supported by the Spanish Ministerio de Economia y Competitividad and by FEDER (EU) funds under Grant No. TIN2014-52211-C2-1-R. We would also like to thank the annonymous reviewers for their contributions and useful comments to improve the contents of this paper.

REFERENCES

Aurich and Weule, 1995. Non-Linear Gaussian Filters Performing Edge Preserving Diffusion. *Proceedings of the DAGM Symposium (DAGM'95)*, pp. 538-545. Springer Berlin Heidelberg.

Balsa-Rodríguez, M. et al., 2014. State-of-the-art in compressed GPU-based direct volume rendering. *Computer Graphics Forum*, 33(6), pp. 77–100.

Boada, I. et al., 2001. Multiresolution volume visualization with a texture-based octree. *The Visual Computer* 17, 3 (2001), pp. 185–197.

Díaz-García, J. et al., 2016. Adaptive Transfer Functions. *The Visual Computer.* 32, 6-8 (June 2016), pp. 835-845.

Hadwiger, M. et al., 2006. *Real-Time Volume Graphics*. A. K. Peters Ltd, Natick.

Kopf, J. et al., 2013. Content-adaptive image downscaling. *ACM Trans. Graph.* 32, 6, Article 173 (November 2013), 8 pages.

Kraus, M. and Ertl, T., 2001. Topology-Guided Downsampling. *Volume Graphics 2001*, pp. 223-234.

Kraus and Bürger, 2008. Interpolating and downsampling RGBA volume data. *Proceedings of Vision, Modeling, and Visualization' 08*, pp. 323–332.

Kwon et al., 2016. A fast 3D adaptive bilateral filter for ultrasound volume visualization, *Computational Methods Programs Biomedecine,* 2016 Sep;133:25-34. doi: 10.1016/j.cmpb.2016.05.008.

Levoy and Whitaker, 1990. Gaze-directed volume rendering. *SIGGRAPH Comput. Graph.* 24, 2, pp. 217–223.

Öztireli, A. C. and Gross, M., 2015. Perceptually based downscaling of images. *ACM Trans. Graph.* 34, 4, Article 77 (July 2015), 10 pages.

Paris et al., 2007. A Gentle Introduction to Bilateral Filtering and its Applications, *ACM SIGGRAPH 2007 course notes*, https://people.csail.mit.edu/sparis/bf_course/.

Paris and Durand, 2009. A fast approximation of the bilateral filter using a signal processing approach. *International journal of computer vision* 81.1 (2009): pp. 24-52.

Smith and Brady, 1997. SUSAN -- A New Approach to Low Level Image Processing. *International Journal of Computer Vision*, Vol. 23, number 1, pp. 45-78.

Sicat et al., 2014. Sparse PDF volumes for consistent multi-resolution volume rendering. *IEEE Transactions on Visualization and Computer Graphics*. (Proc. IEEE Vis.) 20(12), pp. 2417–2426.

Tomasi and Manduchi, 1998. Bilateral Filtering for Gray and Color Images. *Proceedings of the International Conference on Computer Vision* (ICCV'98), pp. 839-846.

Weber, N. et al., 2016. Rapid, detail-preserving image downscaling. *ACM Trans. Graph.* 35, 6, Article 205 (November 2016), 6 pages.

Wang et al., 2011. Feature-preserving volume data reduction and focus + context visualization. *IEEE Transactions on Visualization and Computer Graphics*. 17(2), pp. 171–181.

Wang et al., 2012. Adaptive Speckle Reduction in OCT Volume Data Based on Block-Matching and 3-D Filtering, *IEEE Photonics Technology Letters*. Volume: 24, Issue: 20, pp. 1802-1804.

Younesy et al., 2006. Improving the quality of multiresolution volume rendering. *Proceedings of Joint Eurographics/IEEE VGTC Conference on Visualization*, pp. 251–258. Eurographics Association, Lisboa, Portugal.