Rendering techniques for multimodal data

Maria Ferré[1], Anna Puig[2], and Dani Tost[2]

1 Dept. Enginyeria Informàtica i Matemàtiques
Rovira i Virgili University
Carretera de Salou, s/n 43006 Tarragona, Spain
mferre@etse.urv.es
2 Dept. Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Avda. Diagonal, 647,
Barcelona, 08028, Spain.
{anna, dani}@lsi.upc.es

Abstract Many different direct volume rendering methods have been developed to visualize 3D scalar fields on uniform rectilinear grids. However, little work has been done on rendering simultaneously various properties of the same 3D region measured with different registration devices or at different instants of time. The demand for this type of visualization is rapidly increasing in scientific applications such as medicine in which the visual integration of multiple modalities allows a better comprehension of the anatomy and a perception of its relationships with activity. This paper presents different strategies of Direct Multimodal Volume Rendering (DMVR). It is restricted to voxel models with a known 3D rigid alignment transformation. The paper evaluates at which steps of the rendering pipeline must the data fusion be realized in order to accomplish the desired visual integration and to provide fast re-renders when some fusion parameters are modified. In addition, it analyzes how existing monomodal visualization algorithms can be extended to multiple datasets and it compares their efficiency and their computational cost.

1 Introduction and previous work

In scientific studies, it is often required to work on different properties of a 3-D region or on various measurements of the same property captured with different devices and even with the same device but at different moments in time. For simplicity, we will herein use the term of *multimodality* to refer to these three cases, although strictly speaking only the two former cases actually use different capture modalities.

Medical applications are a good example of the increasing demand for multimodal systems able to deal simultaneously with various datasets of a same region [1]. Computer Tomography (CT) together with Magnetic Resonance Imaging (MRI) provide a larger and better segmented perception of anatomical structures. The combination of MRA (Angiography) with MR data of the brain allows physicians to predict eventual cerebral damage produced by vascular accidents such as aneurysms or haemorrhages. Finally, the integration of MR data with functional modalities such as SPECT (Single Photon Emission Computed Tomography) and PET (Positron Emission Tomography) is the key to the localization of small tumors that appear when conjugating anatomical information with physiological abnormalities. The simulations presented in this paper are from medical images. However, the discussion is valid as well for other multimodal applications.

The integration of different datasets presents three main problems:

- *Data Registration*, i.e., finding a geometrical transformation to align the datasets. This step is necessary whenever the different properties are not captured simultaneously. Different approaches exist to solve this problem [16], [6], [7], mostly based on rigid transformations.
- *Data modeling*, i.e., finding accurate data structures able to represent various properties at different resolutions while skipping over irrelevant data. Little research has been done on this problem [18], [17]. Most research has concentrated on dealing with hybrid (surface and volume) data [9] but the proposal of efficient multiple volume models is still an open problem.
- *Multimodal rendering*, i.e. developing rendering strategies able to map different properties in a unique visualization.

This paper focuses at the last problem. It is restricted to rigid transformations between models that are already known or that have been computed in a previous step. In addition, it is supposed throughout the text that the initial datasets are voxel models.

Many different direct volume rendering methods have been developed to visualize 3D scalar fields on uniform rectilinear grids. Two main strategies are used: (i) Indirect Volume Rendering (IVR), which consists of rendering polygonal models of surfaces of interest extracted from the volume dataset [12], and (ii) Direct Volume Rendering (DVR). Indirect Multimodal Volume Rendering (IMVR) is rather straightforward, as surfaces can be extracted from the different datasets and rendered simultaneously using classical surface rendering methods. However, it is not always the best option. On one hand, it is not easy to fit surfaces in all data sets, specially in SPECT, which are very fuzzy. On the other hand, surfaces from different modalities often coincide partially or totally, which may cause artifacts in the visualization and give poor visual clues of the represented structures. Finally, although surface rendering is generally nicer than DVR, it conveys less information.

This paper addresses Direct Multimodal Volume Rendering (DMVR). The main related previous papers are based on the raycasting method. In [3] rays are cast through non overlapping volumes so that no actual data fusion is done. [22] developed the "Normal fusion" strategy for MR and SPECT aligned datasets: each ray is cast through a segmented MRI dataset representing the brain surface. The surface is shaded using a conventional light model. Its location is next used as the starting point of the ray integration through a given depth inside the SPECT dataset. The rendering shows the SPECT values below the MR brain surface. Finally, [2] discuss implementation strategies of the raycasting pipeline for the integration of CT, segmented CT and distribution radiotherapy dose aligned data. Specifically, the authors propose three different intermixing approaches: image-level intermixing (i.e. image merging), accumulation-level intermixing (opacities and intensities mixing along the ray) and illumination model intermixing (optical properties mixing).

The goal of this paper is to address DMVR from a general perspective. We define a general framework that includes the different types of multimodal rendering. DMVR is analyzed from various perspectives: the end-user requirements, the fusion criteria and the rendering algorithms adaptivity. The paper is structured as follows. Section 2

presents the requirements of a multimodal rendering. Section 3 analyzes at which steps of the rendering pipeline should the fusion be done in order to fulfill these requirements. Section 4 discusses how known rendering algorithms can be extended to support multimodal data. Finally, in Section 5, some practical simulations are evaluated, leading to the conclusions.

2 Requirements

2.1 Multimodal rendering modes

The first requirement of multimodal rendering concerns the type of desired visual integration. Two different types of multimodal data visualization should be provided:

- Rendering one property per point. There are two main applications of this type of rendering:
  - Complementary data: the property shown is the same for all samples. The final image looks like a monomodal rendering, although various properties have been used during the rendering pipeline. A typical example of that is the fusion of CT and MR modalities which both represent anatomical information but that outline different tissues. These two types of data can be combined during rendering to better segment anatomical structures.
  - Supplementary data: for each sample of the volume data a decision is made on which property must be shown at this location. Only one property is rendered for each sample, but it does not need to be the same. In some way, this rendering simulates that the volume data is subdivided into disjunct monoproperty regions. An example of supplementary data rendering would be combining MR with MRA, in such a way that MR data are shown at samples where MRA values (which capture vascular structures) are not considered relevant according to a user-defined criterion, and on the contrary, only MRA is shown at relevant locations. From a medical point of view, MR data would give a reference frame to better understand MRA.
- Rendering various properties per point. In this type of rendering, it is assumed that each property represents a different material and an actual fusion of these materials is done during rendering. An example of this would be showing simultaneously anatomical and functional information as for instance MR and PET or SPECT. The difficulty here is to find visual clues to realize a meaningful fusion. Typically, in the fusion of two modalities, gray values are used to map one property while the other one brings hue but more sophisticated combinations can be done.

The second requirement of multimodal rendering is to provide flexibility in changing fusion parameters. Whatever visual integration mode is used, multimodal rendering must give means of modifying the fusion criteria and fast revisualizing the data.

Another desirable feature is the stability of the rendering under modifications of the visualization pipeline parameters: camera, light sources and transfer functions. This latter aspect is a key point in DMVR because of the intrinsic difficulty in fixing them. Although work is on progress to automatically set these functions, the trial-and-error

method is still one of the most used. In multimodal data, classification is even more complex because there are more interrelated parameters.

In addition, an important point is if the rendering needs the initial models to be aligned or if it can work on the original data sets, applying on-line the alignment transformation.

The fulfillment of these requirements depends, on one hand, on the stage of the rendering pipeline in which the fusion is applied, and on the other hand, on the rendering algorithm used.

3 Fusion pipelines

3.1 Monomodal rendering pipeline

Let first discuss the shading model usually used for rendering monomodal scalar data in order to better evaluate how the visual integration modes exposed in the previous section fit into the rendering pipeline. We herein use the common assumption in volume rendering of scientific data that the media has low albedo and thus, an emission+absorption model [13] is sufficient for shading volume data. The amount of light of wavelength λ coming at a point x through a viewing direction w is $I_\lambda(x,w)$:

$$I_\lambda(x,w) = \int_{l=x}^{L} C_\lambda(l,w)\mu(l)e^{\int_{t=x}^{l}\mu(t)\delta t}\delta l \tag{1}$$

where $C_\lambda(l,w)$ is amount of light emitted at l in direction w, $\mu(l)$ is the extinction coefficient at l and the exponential term modelizes the attenuation of light due to absorption between x and l. L is the integration length.

This integral equation is approximated through Riemann sums by subdividing the viewing rays into a set of n intervals, such that the emission of an interval s_i is $C_\lambda(s_i)$ and the accumulated opacity is $\alpha(s_i)$

$$I_\lambda(x,w) = \sum_{i=1}^{n} C_\lambda(s_i,w)\alpha(s_i)\prod_{j=1}^{i-1}(1-\alpha(s_j)) \tag{2}$$

In this work we have considered two types of shading depending on the how the emission term is computed:

- Volume Shading: emission is a function of the property value in the interval s_i: $C_\lambda(s_i,w) = E_\lambda(s_i,w)$. Assuming that the emission is the same in all directions: $C_\lambda(s_i) = E_\lambda(s_i)$
- Surface-and-Volume Shading: emission is computed as the sum of the Volume Shading plus a Lambert or a Phong shading model in the intervals where the surface gradient is significant enough:

$$C_\lambda(s_i,w) = E_\lambda(s_i) + I_{a\lambda}k_aO_{a\lambda} + \sum_{f=1}^{nls} I_{f\lambda}(k_dO_{d\lambda} * N\cdot L_f) + (k_sO_{s\lambda}R_f\cdot w) \tag{3}$$

being $I_{f\lambda}$ $f = 1...nls$ the intensity of the nls light sources reflected in the surface passing through the interval s_i, N the normal vector and R_f the reflected vector at s_i.

The material optical properties of these formula, i.e. the opacity α, the emission E and the surface properties (k_a, k_d, k_s, O_d, O_s and n) are computed through transfer functions represented by look-up-tables indexed with the property value computed for the sample interval.

The monomodal rendering pipeline for a sample interval is represented at the top of Figure 1. It consists of the following steps: (i) property value computation, (ii) gradient computation, (iii) classification i.e. optical properties computation, (iv) shading and (v) composition. Volume rendering algorithms differ one from each other in how they discretize the viewing rays and in the order in which they perform these different steps.

3.2 Multimodal rendering pipeline

Let V be a volumetric region and $p_1, p_2, ..p_n$ a set of n scalar property functions defined over V. For all point x, $x \in V$, the i-th property value of x will be denoted as $p_i(x)$ and its i-th gradient value $g_i(x)$. Note that $p_i(x)$ and $g_i(x)$ can be directly a sample dataset value or an interpolated value.

The fusion of data for multimodal rendering can be realized at any step of the shading pipeline. Therefore, we distinguish five types of multimodal rendering processes: (i) Property fusion, (ii) Property and gradient fusion, (iii) Material fusion, (iv) Shading fusion and (v) Color fusion.

Property Fusion (PF)

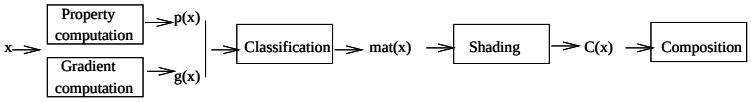
The different property values are composed at the beginning of the rendering process. Once the property resulting from the fusion function has been computed, the rendering pipeline proceeds as for monomodal data (see second drawing in Figure 1). Specifically, the gradient vectors are computed after the fusion, on the basis of the new property values.

The property value at a sample location x of the volume set is computed using an n-parameterized function of the n original property values. A particular case of such a fusion function is a linear combination of the different elements to be mixed:

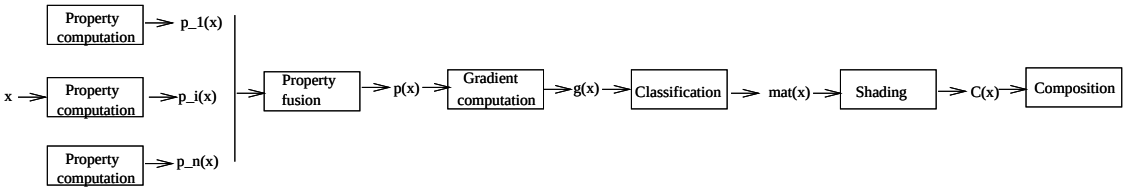
$$p(x) = \sum_{i=1}^{n} \beta_{prop_i}(x) * p_i(x) \tag{4}$$

being $\beta_{prop_i}(x) = f_i(p_j(x), j = 1...n)$ such that $\forall i = 1...n, \beta_{prop_i}(x) >= 0.0$ and $\sum_{i=1}^{n} \beta_{prop_i}(x) = 1.0$

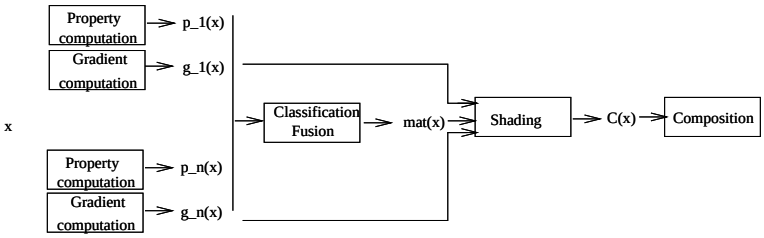
$\beta_{prop_i}(x)$ can be implemented as n valued transfer functions that for a given combination of properties indicate the weight of each of them. The property fusion (PF) fits well the one-property-per-point multimodal rendering model described in section 2. Specifically, for complementary data, the fusion function acts as a segmentation filter. The new computed values represent the material which is more likely to be present at this location taking into account all the original property values. When function β_{prop} is



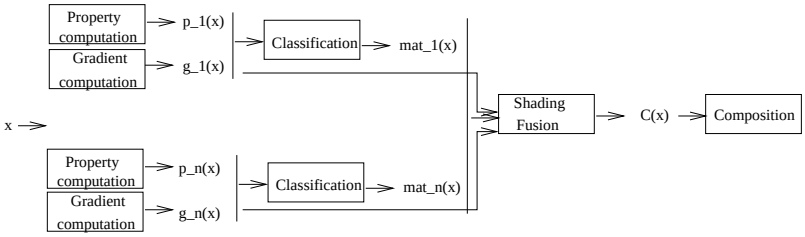
Monomodal shading pipeline



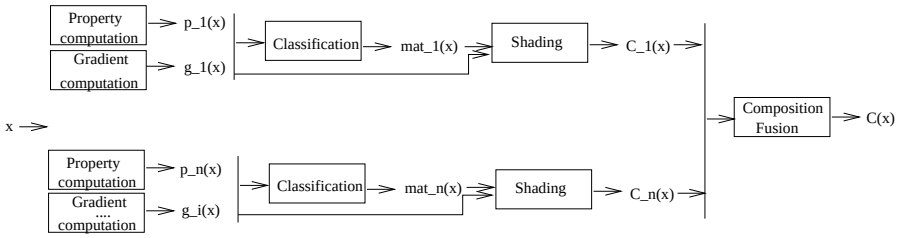
Property Fusion pipeline (PF)



Property and Gradient Fusion pipeline (P&GF)



Material Fusion (MF)



Shading fusion

Figure1. Multimodal Rendering Schemes.

defined as a binary function over the domain $0,1$ only one property is shown in the final rendering and thus, a one-property-per-point of supplementary properties is achieved.

Obviously, PF is not suitable for rendering various properties per point, as the original data are lost at the beginning of the fusion.

This type of fusion is stable under modifications of the camera, the light model and the transfer function because the fusion occurs at the beginning of the pipeline. On the contrary, changes in the fusion parameters, i.e. on the function β_{prop}, require the rendering process to be completely re-done.

Property and Gradient Fusion (P&GF)

n property values and *n* gradient vectors at a sample location, are used to compute unique material properties at this location. The remaining steps of the monomodal shading pipeline are next executed orderly (see third scheme Figure 1). The fusion is achieved by designing transfer functions of $2*n$ parameters i.e. performing a $2*n$-parameterized classification. This can be represented as:

$$mat(x) = clas((p_1(x), g_2(x)), .., (p_n(x), g_n(x)) \tag{5}$$

P&GF is essentially similar to PF. It acts as a segmentation filter, useful for one-property-per-point multimodality but not suitable for multiple properties per point. By opposite to PF, P&GF uses the gradient vectors of all individual properties in the classification. The gradient information eases the edges detection and thus can provide neater fusion.

P&GF shares with PF the facility of camera and lights modifications. Modifications in the transfer function are actual fusion modifications and provoke the whole pipeline to be recomputed.

Figure 2 shows a 2D example of PF and P&GF modes. The original images are from CT (upper left image) and from MR (upper right). The lower left image is an example of PF. Voxels with high CT intensity have replaced MR values. Therefore, the image shows the skull as well as the brain. In the lower right image, only high CT value having a significant gradient have been replaced. The image shows two new contours: the internal and the external skull walls. Voxels between these two boundaries show the MR value.

Material Fusion (MF)

n property values and *n* gradient vectors at a sample location are computed and next *n* classification processes are realized separately. Then, given *n* materials a fusion of them is done that gives one unique material per point (Second pipeline in Figure 1). A simple implementation of this consists of applying a linear combination of materials:

$$mat(x) = \sum_{i=1}^{n} \beta_{mat_i}(x) * mat_i(x) \tag{6}$$

being $\beta_{mat_i}(x) = f(mat_j(x), j = 1...n)$ such that $\forall i = 1...n, \beta_{mat_i}(x) >= 0.0$ and $\sum_{i=1}^{n} \beta_{mat_i}(x) = 1.0$

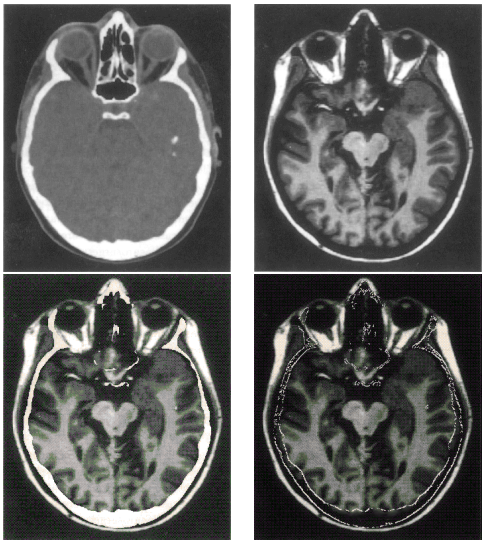


Figure2. PF and P&GF for CT and MRI data.

Note that, as for PF, β_i functions can be constant or may vary. In the latter case, they can can be implemented as *n* valued transfer functions of material combinations. Material fusion is similar to *illumination model intermixing* used in [2]. The meaning of a material fusion is rather different from the two previous modes. In fact, this mode does not intend to simulate a real mixture of materials as it is rather improbable that such a physical model is known. Instead, it tries to simulate the presence of all the materials separately but with different occupancy. It is somewhat the opposite of the classification model proposed in [4]. In that classification scheme, the actual composition of a voxel is determined by computing the most probable combination of materials that could have given the voxel value. On the contrary, in material fusion, it is the user who fixes the weights of the different materials giving more or less importance to some properties over the others.

Like property fusion, material fusion is suitable for one-property-per-point multi-modal rendering when β_{mat} functions are binary. However, it is also usable if more than one property should be mapped at the same location using β_{mat} functions to weight the material quantities at each sample, as exposed above.

Changes in the fusion function (β_{mat}) need only fusion, shading and composition to be re-done. However, modifications of any of the property transfer function change the pre-classified models and may even require tuning the material fusion function. Thus, this modality is worth to be applied on pre-classified models.

As the previous modalities, MF allows camera and light changes because the fusion is previous to color computation.

Shading Fusion (SF)

This method uses *n* optical properties computed separately for all the properties as an input of the shade value computation (fourth scheme in Figure 1). Assuming a linear combination of the optical parameters, the Volume Shading formula is substituted by:

$$C_\lambda(x) = \sum_{i=1}^{n} \beta_{shade_i} E_{i_\lambda}(x). \tag{7}$$

while the Volume and Surface Shading is :

$$C_\lambda(x,w) = \sum_{i=1}^{n} \beta_{shade_i}(E_{i_\lambda}(x) + I_{a\lambda} k_{a_i} O_{a\lambda_i} + \sum_{f=1}^{nls} I_{f\lambda}(k_{d_i} O_{d\lambda_i} * N_i \cdot L_f) + (k_{s_i} O_{s\lambda_i} R_{f_i} \cdot w)) \tag{8}$$

If Volume-Shading is applied, this mode gives the same results as MF. However, when Surface-Shading is applied, as the light sources reflection is computed separately for each surface and next weighted, the surfaces are more distinguishable.

For changes of the fusion parameters or of the viewing parameters, this modality is similar to MF. In particular, fusion modifications are more efficient on pre-classified models.

Color Fusion (CF)

The last method simply mixes the *n* RGBα values computed through *n* shading processes (last scheme in Figure 1). A simple implementation of it is a linear combination of colors:

$$C(x) = \sum_{i=1}^{n} \beta_{C_i}(x) * C_i(x) \tag{9}$$

being $\beta_{C_i}(x) = f(C_j(x), j = 1...n)$ such that $\forall i = 1...n, \beta_{C_i}(x) >= 0.0$ and $\sum_{i=1}^{n} \beta_{C_i}(x) = 1.0$

This type of fusion is well adapted to multiple properties per point rendering but it is not suitable for one-property-per point. Its main advantage is that changes in the fusion parameters are very fast on pre-shaded models. However, changes in the viewing parameters provoke the full pipeline to be recomputed if Surface-Shading with specularity is applied. Light model or transfer function changes provoke the whole pipeline recomputation in all cases.

4 Rendering methods

Nowadays, there are four main DVR paradigms: raycasting [11], splatting [19], 3D texture-mapping [8] and shear-warp [10]. Many improvements on these original papers have been developed in the last years. A practical evaluation of them is presented in [15]. We next analyze their adequacy for multimodal rendering.

4.1 Alignment

As mentioned in Section 1, most research papers on multimodal rendering are based on raycasting. The main advantage of raycasting for multimodality (MDVR) is that, except for discrete raycasting [21], the different models do not need to have the same orientation and resolution. Assuming that the registration transformation between models is known, it is as if each model has its own local coordinate system. Thus, each sample point of a ray is converted into the local coordinate system of each model in order to determine to which voxels the sample belongs. The sample value is therefore interpolated between these different voxel vertices values. If no interpolation is done, i.e. if the property value at a sample point is directly set to the value of the voxel to which the sample belongs (*voxel-based rendering* by opposite to *cell-based rendering* according to [20] notation), sampling would actually integrate different 3D regions, which would not be totally correct. However, *voxel-based rendering* is more often used in object-order methods and it is not very useful in raycasting.

Object-order methods such as splatting and 3D texture-mapping as well as the hybrid shear-warp paradigm face different problems in rendering simultaneously various models with different size and orientations. First of all, the depth composition is not correct in volume slices traversals of models having different orientations. This problem is avoided when rendering is done by rasterizing slices parallel to the image plane, as it is done in texture-mapping. Besides, even if the orientation of the models is the same, if the resolution is different, the optical depth of the pixels onto which voxels project is not the same and thus their composition is not correct. In particular, shear-warp needs pixel per voxel ratios about 1 or 2 and thus, it would not support larger projection in one of the datasets. Up to now, none of these methods are directly applicable to non-aligned datasets. In addition, revoxelizing the models into the same coordinate system and with the same resolution is computationally expensive and it adds an extra interpolation error.

4.2 Adaptivity to multiple properties

As mentioned above, the extension of raycasting to multiple data is straightforward for aligned data sets as well as non-aligned ones.

Splatting on various aligned datasets is also simple. Two main strategies can be applied: either the voxels are sliced in planes parallel to the viewing plane or they are traversed in their original BTF or FTB order. In both cases for PF, P&F, MF and SF the fusion must be done at the voxel level, before actually splatting it. CF instead can be done after splatting by compositing the projection sheets.

The extension of shear-warp to multimodal data is, by now, the most complicated one. On one hand, shear-warp requires three sets of voxel run-length encoding, one for each major viewing directions. This is serious drawback, as in an n multimodal study, the total memory requirement would be of $3 * n$ models. On the other hand, the simultaneous traversal of different RLE should be studied in depth to keep up the efficiency of the original method.

The use of texture-mapping is limited to expensive hardware. However, it is the probably fastest rendering method. Its extension to multiple aligned volumes is straightforward for RGBα volume data [5]. Its main limitation is its high memory consumption,

which provokes a lot of brick swap for large monomodal volumes and much more for multimodal data, that require n volume bricks must be resident in the texture memory simultaneously. Multimodal studies require thus brick sizes to be smaller than in monomodal rendering.

4.3 Suitability for the five fusion strategies

Splatting and raycasting give higher image quality than shear-warp and texture-mapping but at a higher cost. An important feature of ray-tracing is that it actually interpolates in 3D the sample property and gradient values. Thus, benefiting from pre-classified or pre-shaded models for MF, SF and CF, would speed it up but at the cost of reducing image quality. So, raycasting seems more suitable for PF and P&GF. On the contrary, as splatting and shear-warp algorithms actually use one property value per sample and interpolate in 2D, they would take advantage of pre-classified and pre-shaded models for MF, SF and CF. RGBα texture-mapping is suitable for pre-shaded volume and thus is suitable for the CM pipeline. Several techniques have been developed to add shading capabilities to monomodal texturing [14]. However, their adaptation to multimodality, which would extend the usability of this technique to the other fusion pipelines, has not yet been studied.

5 Simulations

The simulations have been realized on a 190x220x178 multimodal study composed of 2 bytes-intensity MR images, 1 byte-intensity labeled images and 1 byte per channel (RGB) SPECT images. The label model represents the segmented anatomical regions of the brain. SPECT and MR were not originally aligned.

Five different fusion modalities have been realized using raycasting and splatting on a Sun Ultra 60 360MHz. CF has also been tested with texture-mapping on SGI Octane with 4MB of texture memory. The rendering time was much higher than in monomodal rendering, as the dataset did not fit in texture memory. The texture-based rendering time for a CF monomodal dataset is around 1.2 seconds but it triplicates (3.7) for the multimodal dataset.

Raycasting has been done on the original, non aligned datasets but SPECT and MR have been aligned for splatting with a rigid afine transformation consisting of several axis-rotation plus scaling. This operation has a low cost but it introduces an additional error in the sampling process. Specifically, in this dataset, the average relative difference between ray sample values interpolated in the original model and in the transformed one is 0.0140 per channel.

For both splatting and raycasting the ratio pixels per voxel is around 4. This is why the computational cost of splatting and raycasting is approximately the same. It should be noted that no early termination criteria have been used in the tests. Specifically, splatting cost is 0.069 % lower than ray-casting in all the tests.

Computational costs are summarized in Table 1. Times correspond to raycasting but apply as well for splatting with the correction factor. For each pipeline except for PF, two costs have been computed: full pipeline (second column) and reduced pipeline

(third column) using (i) precomputed gradients for P&GF, (ii) preclassified data for MF and SF and (iii) preshaded volumes for CF.

Fusion Mode	Original Datasets	Intermediate Models
PF	125.102	–
PGF	129.662	89.102
MF	132.311	35.746
SF	127.251	33.807
CF	148.769	28.209

Table 1. Computational costs

As expected, computational costs without pre-process increase as the fusion is delayed in the rendering pipeline. However, refusion is faster. Changes in the camera, the light sources or the transfer function actually provoke the whole pipelines to be recomputed and thus, the less computationally expensive is PF and P&GF by opposite to CF.

The fusion function used in the simulation are: (i) (PF, Color Plate 3) replacing MR values with high intensity SPECT values, (ii) (P&GF, Color Plate 4) Fusion between labeled data and MR data: labeled values with gradient substitute MR values in the cerebral cortex (red on the right and green on the left), (iii) (MF, Color Plate 5) Weighted average of SPECT and MR. The transfer function only shows surface MR values indicated by the labeled model, (iv)(SF, Color Plate 6) Weighted shading of SPECT and MR Surface-and Volume materials, (v) (CF, Color Plate 7) color blending of shaded MR and SPECT data.

Note that the fusion function cost can affect the overall performance of the pipelines. Here the fusion used in MF is more complex than the simple constant weighting used in CF. This explains why MF and SM are slightly different.

6 Conclusions

This paper has addressed the problem of direct rendering multimodal volume models. It has stated the requirements of this type of visualization and it has proposed five integration methods that differ in the step of the rendering pipeline at which the fusion is done (PF, P&GF, MF, SF, CF). The last is the fusion done in the rendering pipeline, the faster are modifications on the fusion parameters but the less changes in the viewing parameters and light sources are allowed. In addition, PF and P&GF are suitable for rendering one property per point while MF, SF and CF are better for multiple properties per point rendering.

The suitability of different rendering algorithms to apply these five methods has been evaluated. It can be concluded that raycasting, which is the more versatile technique, is probably the most suitable method when one-property per point multimodal PF and P&GF rendering are required. Splatting, shear-warp and texture-mapping don't currently support the visual integration of non-aligned data-sets. However, for aligned

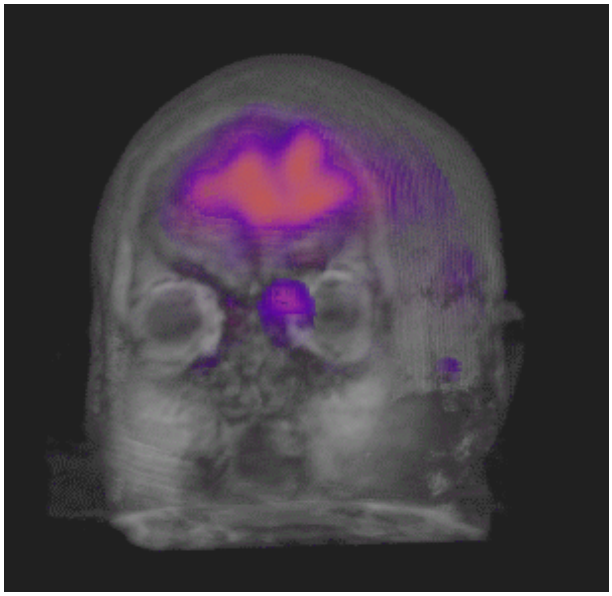


Figure3. PF example.

datasets, splatting conveniently fulfills the requirements of MF and SF multimodal rendering pipeline. Finally, fast CF can be done when hardware texture mapping is available.

Starting from this work, several research lines are opened. First, the shear-warp RLE traversal should be extended to multiple datasets. Next, the composition of non-aligned datasets with object-order and shear-warp methods should be addressed. In addition, much effort is being done in the research of automatic transfer function definition. Special attention should be paid on n-parameterized functions that are use in multimodality. Finally, user-friendly interfaces should be developed for the input of *n* parameterized fusion functions.

Acknowledgments: This work has been funded by the project TIC 99-1230-C02-02 from the Ministerio de Educación y Ciencia.

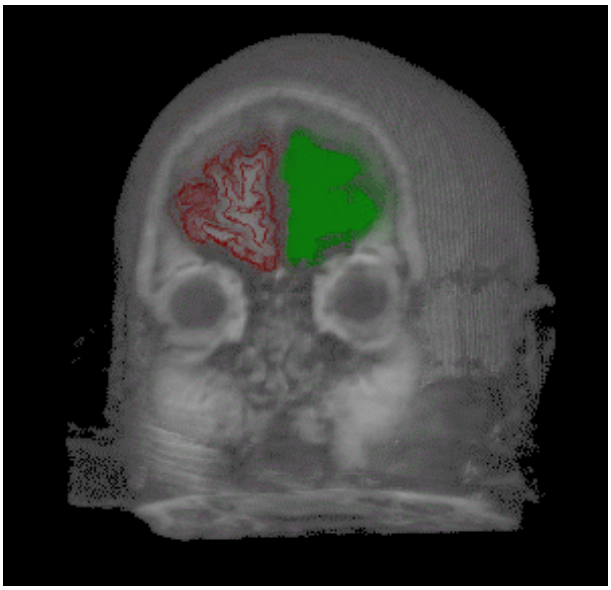


Figure4. P&GF example.

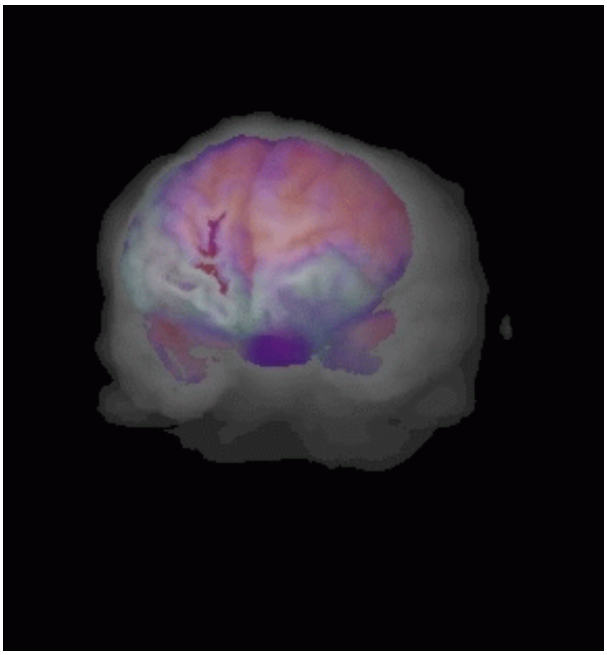


Figure5. MF example.

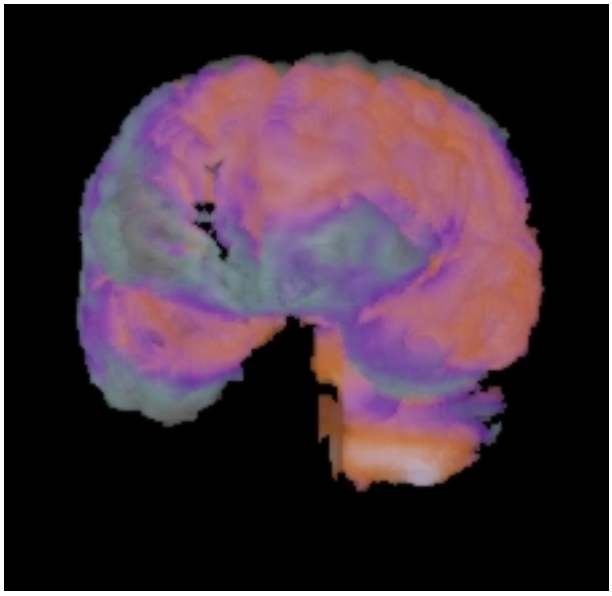


Figure6. SF example.

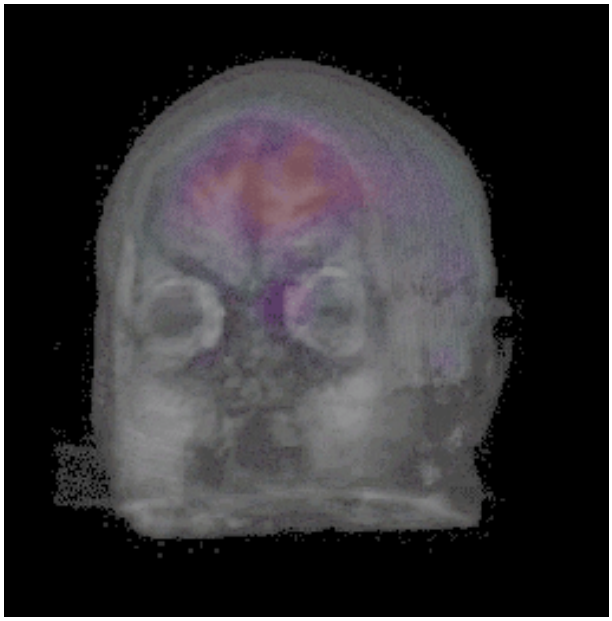


Figure7. CF example.

References

1. V. Barra and J.Y. Boire. A general framework for the fusion of anatomical and functional medical images. *NeuroImage*, 3(13):410–424, 2001.
2. W. Cai and G. Sakas. Data intermixing and multivolume rendering. *Computer Graphics Forum*, 18(3):359–368, 1999.
3. M. Chen and A. Leu. Parallel multi-volume rendering on distributed memory architecture. *First EG Workshop on Parallel Graphics and Visualization*, pages 173–187, 1996.
4. R.A. Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. *Computer Graphics*, 22(4):65–74, August 1988.
5. G. Eckel. Opengl volumizer programmer's guide. *Document n. 0073720001*, 1999.
6. P.A.v.d. Elsen. *Multimodality Matching of Brain Images*. PhD thesis, Utrecht Univerity, June 1993.
7. M. Ferre and D. Tost. Integration of multimodal data based on surface registration. *Proceedings of METMBS01*, pages 73–77, June 2001.
8. A. Van Gelder and K. Kim. Direct volume rendering with shading via 3d textures. *Proceedings of 1996 Symposium on Visualization*, pages 23–30, 1996.
9. K. Kreeger and A. Kaufman. Mixing translucent polygons with volumes. *Proceedings IEEE Visualization*, pages 191–198, 1999.
10. P. Lacroute and M. Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. *ACM Computer Graphics*, 28(4):451–458, July 1994.
11. M. Levoy. Efficient ray tracing of volume data. *ACM Transactions on Graphics*, 9(3):245–261, July 1990.
12. W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM Computer Graphics*, 21(4):163–169, July 1987.
13. N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995.
14. M. Meissner, U. Hoffmann, and W. Straber. Enabling classification and shading for 3d texture mapping based volume rendering using opengl and extensions. *Proceedings IEEE Visualization*, pages 207–214, 1999.
15. M. Meissner, J. Huang, D. Bartz, K. Mueller, and R. Crawfis. A practical evaluation of popular volume rendering algorithms. *Proceedings of Volume Visualization 2000*, pages 81–91, 2000.
16. C.A. Pelizzari, G.T.Y. Chen, D.R. Spelbring, R.R. Weichselbaum, and C.T Chen. Accurate three-dimensional registration of ct, pet, and/or mr images of the brain. *Journal of Computer-Assisted Tomography*, 13(1):20–26, January 1989.
17. Tost D. Puig, A. and I. Navazo. A hybrid model for vascular tree structures. *Data Visualization, Springer Verlag, Eds. W. de Leeuv, R. Van Liere*, 2000.
18. U. Tiede, T. Schiemann, and K.H. Hohne. High quality rendering of attributed volume data. *Proceedings IEEE Visualization*, pages 255–262, 1998.
19. L. Westover. Footprint evaluation for volume rendering. *Computer Graphics*, 24, August 1990.
20. J. Wilhems and A. Van Gelder. A coherent projection approach for direct volume rendering. *ACM Computer Graphics*, 25(4), July 1991.
21. R. Yagel, D. Cohen, and A. Kaufman. Discrete ray tracing. *IEEE Computr Graphics and Applicactions*, 5(12):19–28, 1992.
22. K.J. Zuiderveld, A.H.J. Koning, R. Stokking, J.B.A. Maintz, F.J.R. Appelman, and M.A. Viergever. Multimodality visualization of medical volume data. *Computer and Graphics*, 20(6):775–791, 1996.