Eurographics Workshop on Visual Computing for Biology and Medicine (2015)
K. Bühler, L. Linsen, and N. W. John (Editors)

Motion-moderated Transfer Function for Volume Rendering 4D CMR Data

Simon Walton[1] and Min Chen[1] and Cameron Holloway[2]

[1]Oxford University, Oxford, UK
[2]St Vincent's Hospital, Sydney, Australia

Abstract

Cardiovascular Magnetic Resonance (CMR) produces time-varying volume data by combining conventional MRI techniques with ECG gating. It allows physicians to inspect the dynamics of a beating heart, such as myocardium motion and blood flows. Because the material intensity changes over time in a typical CMR scan, this poses a challenging problem in specifying an effective transfer function for depicting the geometry of a beating heart or other moving objects. In this paper, we propose to moderate the traditional transfer function based on intensity and intensity gradient. This enables us to depict the exterior boundary of a beating heart in a temporally consistent manner. We examine several different ways of moderating an intensity-based transfer function, and evaluate these designs in conjunction with practical CMR data. We present a ray-casting pipeline which includes optional flow estimation and a mechanism to assist temporal coherence in animation.

1. Introduction

Cardiovascular Magnetic Resonance (CMR) imaging has rapidly gained popularity with cardiologists wishing to study temporal function of the heart due to its attractive qualities (non-invasive, lack of x-ray radiation, and mapping of blood flow). CMR allows for the capture of 4D data of a patient's heart, as well as the internal velocities of the volume concerned. CMR images are acquired by combining MRI imaging with ECG gating techniques. The demands of a high temporal resolution combined with the rapid movement of the heart can produce blurring artefacts in standard MRI output where heart motion occurs during signal capture. To compensate, the ECG signal of the heart is used to average many heartbeats together into one image by aligning ECG signal peaks through time and combining the MRI signals [PCC*10].

However, the use of direct volume visualisation in clinical CMR is relatively rare compared to many other modalities such as CT and single step MRI. The main difficulties are: *(a)* the measurement of some tissue materials is not consistent over the time as the tissue-blood mixing rate changes in relation to the motion; *(b)* the resolution in the z-dimension is typically very low; *(c)* the captured data is noisy especially in the region where blood flows. Indeed, the adoption of more advanced visualization techniques in the area is relatively rare. A review of the challenges and opportunities for visualization in CMR can be found in [WBT*14].

In this work, we propose to use motion data to moderate a conventional transfer function for direct volume rendering. This motion data is estimated using an optical flow method into a volumetric motion volume. By using motion to moderate a transfer function, our visualisation pipeline is able to filter out static areas of the dataset and depict only areas of the data under motion (such as the walls of the heart) without resorting to data segmentation that would have to be done step by step in a semi-automatic manner. The contributions of this work include:

- We have compared methods for moderating an intensity-based transfer function using properties of motion.
- We have developed a visualisation pipeline based on ray casting, which includes optional flow estimation and a mechanism to ensure temporal coherence in animation.
- We applied the technique to a set of clinical CMR data, demonstrating that the technique can operate under the low-resolution and noisy conditions of such data.

2. Related Work

Cardiovascular Magnetic Resonance (CMR) is an imaging modality "for clinical studies of the heart and vasculature, offering detailed images of both structure and function with

DOI: 10.2312/vcbm.20151212

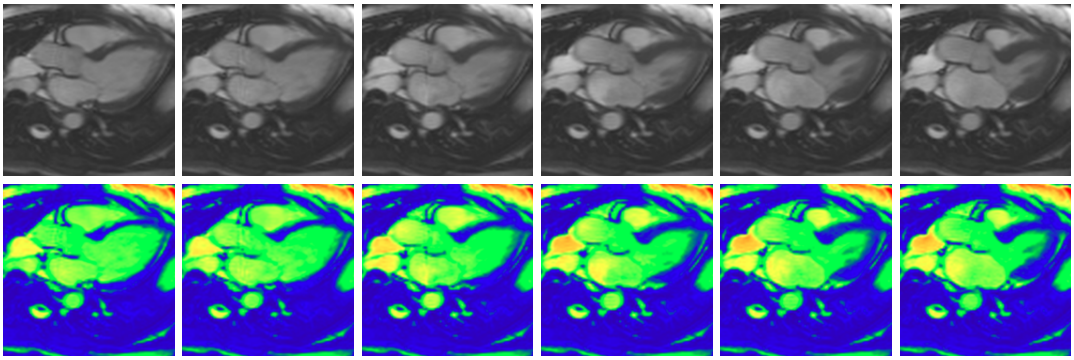

Figure 1: *A selection of time intervals for a long-axis slice through the heart obtained using CMR. The raw intensity values (top) are mapped to a rainbow lookup table (bottom).*

high temporal resolution" [PCC*10]. It is sometimes referred to as the "one-stop-shop for cardiac imaging" [PFF02] as it enables cardiac doctors to assess a comprehensive set of conditions, including "ventricular function, cardiac morphology, vasculature, perfusion, viability, and metabolism" [Poh08]. Its development began as early as the 1970s. The introduction of ECG gating in the later 1990s to handle cardiac motion in 4D CMR imaging significantly improved temporal resolution, making it a routine technology in clinical studies today.

Volume visualisation of 4D CMR data remains a challenge due to the change of intensity of soft tissues in motion. Segmentation of many important structural features, such as endocardial (interior) and epicardial (exterior) borders of a heart, are challenging for automatic methods, and usually require a substantial amount of manual user intervention. In spite of this, segmentation is widely used in clinical settings to isolate the left ventricle and render using a standard surface rendering method.

Typically visualisations in the CMR literature are straightforward greyscale image slices extracted from the corresponding CMR data. Some research applications superimpose streamline representations of blood flow (e.g., [AFM07, TCS*11]) where motion data is available. Occasionally, a proxy cup or cone shape is used to approximate the surface of a heart [Ley10]. Additional data available such as strain can be superimposed as colour mapping [GZM97], or more advanced visualizations such as line integral convolution and tensor ellipsoids [WLY04]. The challenge in rendering a 4D CMR volume requires new techniques in transfer function design and time-varying volume visualisation.

Transfer functions play a central role in direct volume rendering [Lev37] and can aid the user in visually segmenting important regions of the dataset (commonly the left ventricle). The design space of transfer functions can be roughly divided into three aspects: input arguments, rendering effects, and role of users. Because of the large volume of the literature on this topic, here we give a brief overview of a collection of representative works in the literature.

Transfer functions may take a variety of *input arguments*, some of which have to be computed from the raw volume data. These may include, intensity, colour, gradient, shape characteristics [SWB*00], histogram and adaptive histogram [SVSG06, TME15], texture [CR08], object size [CM08], local statistical properties [HPB*10], frequency domain properties [VPG11], and visibility [CM09].

Some transfer functions were designed to deliver specific *rendering effects*, including feature highlighting (e.g., [RE01]), lighting effects (e,g., [WC01, LM04]), pen-and-ink illustration (e.g., [TC00]), deformation (e.g., [CSW*03]), and animated flow motion [CS05]. The modes of user involvement [PLB*01] include interactive specification (e.g., [RSKK06]), automatic formulation (e.g., [RBB*11]), image-based editing [FBT98, WQ07], and template-based style transformation [BG07].

Time-varying volume visualisation has been studied extensively in the literature, though the dominant focus was placed on simulation data. Research efforts can be classified into three categories: computational and interaction support, feature visualisation, and visual mapping, Works on *computational and interaction support* include data structure for efficient computation [WGLS05], functional representation [JEG11], and interactive exploration [BBP08]. Works on *feature visualisation* include object tracking [SW97], isosurface and isovolume extraction [ZXCW05], visualisation of trend relationships [LS09], and correlation [CWMW11]. Works on *visual mapping* include rendering methods [NM02, AEW00], temporal coherence [GSHK04], context and focus [JW06], illustration [JR05, SJEG05], and storyboarding [LS08].

One of the most relevant pieces of work is that of Tory et al. [TRM*01] who studied this subject in the context of medical imaging, and compared three visualisation techniques, isosurface rendering, direct volume rendering and glyph-

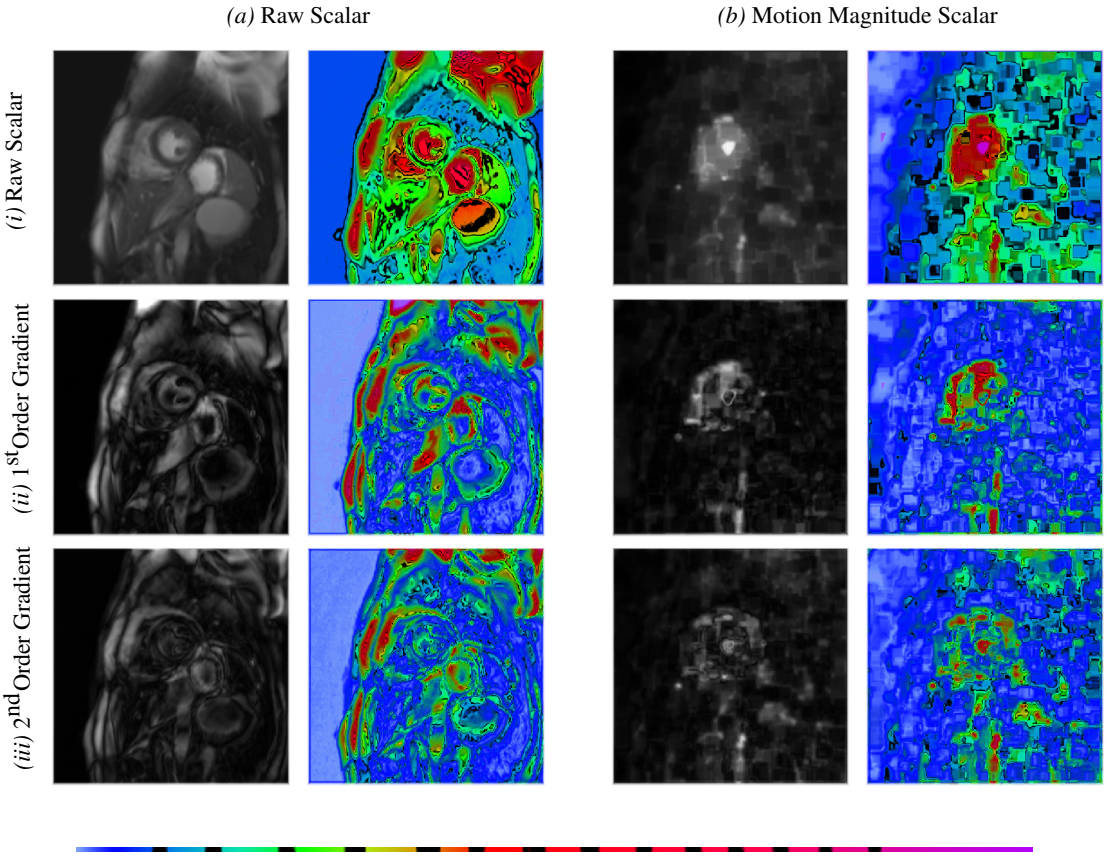

Figure 2: *Example attributes derived from the intensity and motion data. (a) Raw scalar values and (b) motion magnitude values from an optical flow method are modified as (i) unchanged sample value; (ii) 1^{st} order gradient magnitude; (iii) 2^{nd} order gradient magnitude, and used as either greyscale output or as input to a colour lookup table (displayed at bottom – banded to show contours in the data).*

based flow visualisation. In the context of transfer function for simulation data, Jankun-Kelly and Ma discussed the relative merits of single and multiple transfer functions [JM01]. Younes et al. proposed a differential time-histogram table to assist the design of transfer functions [YMC05]. Akiba et al. proposed to specify transfer functions based on a dynamic time histogram. [AFM06]. A number of researchers proposed to use information measurement to highlight features in the data [JWSK07, WYM08, WYG*11].

Whilst a number of these works partially address individual challenges, our work presents a more general solution to improving the visual saliency of temporal CMR imagery by highlighting regions of the dataset that are in motion rather than attempting to infer higher-level semantic information about the data. Such a technique can be applied generally to wider domains.

3. CMR Data Characteristics

This work was carried out in collaboration with a clinical research unit that is specialised in using medical imaging technologies to study the function of the heart and brain. The visualisation researchers in the team have access to anonymised CMR data for over 50 subjects. A CMR scan may result in the following data components:

- *Intensity Volumes* – This is a sequence of volume datasets, $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_T$, and each data set is a scalar volume at $N_x \times N_y \times N_z$ resolution. The data that we are working on typically display $T \approx 25$, $N_x = 256$, $N_y = 256$ with $10 \leq N_z \leq 12$. The scalar values in the data represent the measurement captured by magnetic resonance imaging (MRI), representing the different rates at which the protons in different tissues return to their equilibrium state after the applied magnetic field is turned off.
- *Velocity Data* – Velocity data can be obtained using phase-contrast velocity mapping (discussed below), which encodes velocity into each slice's pixel intensity. Mid-values represent stationary tissue, high values represent flow out of the image in one direction, and low values represent flow in the other direction. Due to the necessity of taking both reference and phase images, each with both refer-

ence and velocity encoding, effort for the same number of slices is increased fourfold. Due to poor z-resolution of this velocity data in our research data, we instead employ an optical flow technique to estimate motion.

- *Colour Map* – The scalar data is usually accompanied by a colourmap, which most commonly is a rainbow scheme. The rainbow colour scheme is carefully designed to map particular measurement values representing different facets of the data (e.g., muscle, blood, air) to particular colours in order to assist the viewer in quickly identifying the structure of the data.

The conventional use of the data is to select an image slice, map the raw scalar measurement to grey scale values, and play an animation sequence of that slice over time. This animation can also be viewed in conjunction with a default colourmap accompanying the data. Figure 1 shows such an image slice at a selection of six time intervals, in both the raw measurement and the colour mapped form.

Because of this common operation mode and the time constraint in relation to the motion of a patient, the resolution in the z-direction is significantly lower than most other imaging modalities. From the perspective of volume visualisation, on one hand, we have to be realistic about the quality of visualisation results. On the other hand, it also poses an interesting challenge to deliver usable visualisation under such conditions. By nature, data obtained via an MRI process contains sampling noise adhering to a Rice distribution [SdDL*99] once the final intensity values are computed. For viewing the data in animated slices, this noise does not present much of a problem. When attempting however to apply any volume visualisation techniques, the artefacts due to temporal incoherence and large spatial separation in the z-direction are clearly visible. These noisy artefacts seriously affect the estimation of gradients and motion flows.

The most significant problem is the fact that the measurement of some materials is not consistent over the time. Because expanding myocardium changes the material properties, the same tissue will appear in different intensity in the grey scale image slices, or be mapped to different colours after colour mapping. This phenomenon can be observed in Figure 1. This issue poses a serious problem in any classification scheme (e.g., segmentation), and causes a serious difficulty in designing a transfer function for rendering a CMR volume as a time-varying data set.

4. Motion-Moderated Transfer Functions

We present a series of motion-moderated transfer function designs for temporal volume rendering that incorporate available dense motion data into the volume rendering pipeline. A transfer function incorporating motion data can present many benefits for the visualisation of temporal volume data, such as bringing the viewer's attention to areas of the data in motion that could otherwise be occluded by static areas of similar intensity attributes, and enabling faster identification the highest-velocity areas of the data (in both animated and static imagery).

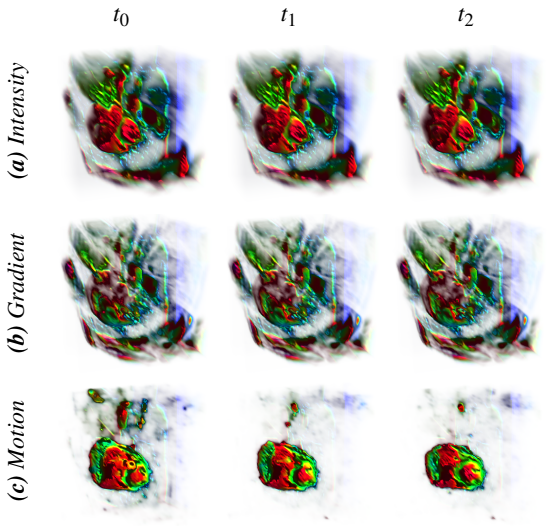

Figure 3: *Three animation frames at times t_0, t_1 and t_2 with differing attributes mapped to opacity:* ***(a)*** *raw intensity;* ***(b)*** *raw intensity gradient magnitude;* ***(c)*** *motion magnitude.*

From Figure 2*(a)*, it is clear that neither the raw scalar intensity nor the gradients of the intensity are adequate in capturing the effective boundary of the heart alone considering that the surrounding tissues have similar intensity-derived properties. The key differentiator in a CMR context is the motion of the heart relative to its neighbouring voxels (Figure 2*(b)*), and it is this attribute of the data that we can introduce to the transfer function space of direct volume rendering. Methods for the visualisation of motion data are typically quite limited and often rely on presenting a sparse vector field or streamline representation to indicate flows within the data. Whilst such visualisations have merit in their respective domains, they are not applicable to CMR due to the relatively poor signal / noise ratio verses simulation data. In addition, such schemes are more suited to surface rendering methods due to the projection of motion vectors to the framebuffer having an association with one particular isosurface.

Given a volumetric dataset representing the intensity values from CMR data, $I_1, I_2, \ldots, I_n \in \mathbb{R}^3$, we can define a transfer function that takes intensity as input and produces another value, or set of values (such as an *RGB* triplet) to aid data interpretation. The transfer function can map to both attributes of the ray composition equation: colour and opacity. Inferred attributes of the intensity can also be mapped; most commonly, the gradient of the intensity ∇I computed using central differences. Now assume that we have an associated motion magnitude field $\mu_1, \mu_2, \ldots, \mu_n \in \mathbb{R}^3$ for each of the n voxels in in the CMR data representing the velocity of each voxel. If we now input motion magnitude into our transfer

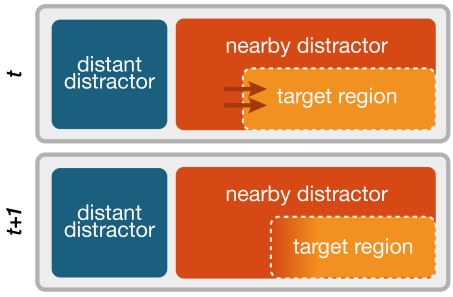

Figure 4: *Motivation for motion-modulation: static distractions, and misclassification of moving materials.*

function, we can map those areas of the data undergoing different velocities to different perceptual values.

Figure 3 demonstrates the transfer of three different input attributes, intensity I, intensity gradient ∇I, and motion magnitude μ to opacity. The colours come from a rainbow lookup table. Note that both intensity and gradient alone give unsuitable results as they include the static portions of the data that possess higher intensity and gradient values. Figure 4 illustrates not only this issue of occluding objects obstructing visualisation of the heart, but also the core classification problem that our work helps to avoid: motion of the target region causing a change in intensity value in the region (e.g. myocardium density change under strain), which causes misclassification. In the case of using the intensity to map to carefully-designed colour schemes (such as the common 'rainbow' scheme), step $t+1$ would result in a change of intensity input into the transfer function, resulting in a potentially large perceptual change in the target region.

When we use motion to modulate the opacity of the ray composition however, we find that the surface of the heart is revealed with minimal occlusion from other features. Not only this, but we mitigate the problem of misclassification as the movement of the heart becomes the heart's signature rather than a static snapshot of its predicted intensity value. Figure 3*(c)* shows the magnitude of the motion vector mapped to opacity. We can compute other attributes of the motion, such as the gradient of the motion magnitude, and use the magnitude and gradient motion attributes in conjunction with intensity attributes. These attributes can also be mapped to either colour or opacity depending on the purpose of the visualisation.

5. Visualisation Pipeline

We now introduce the pipeline for our motion-moderated transfer functions, which is shown in Figure 5, and is split into three stages:

(**a**) *Build*: The generation of a 3D *flow volume* for input to our motion-moderated transfer function;
(**b**) *Raycast*: The generation of a 2D *colour buffer* and a 2D *motion buffer* from the input volume and flow volumes;

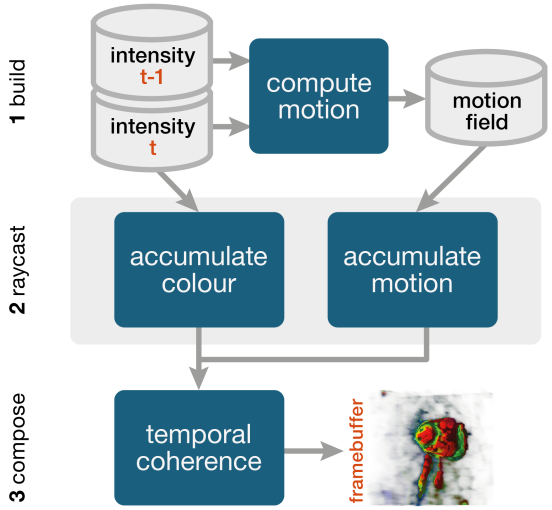

Figure 5: *The three-stage pipeline for our motion-moderated transfer function visualisations.*

(**c**) *Compose*: The composition of the colour and motion buffers as output to the framebuffer.

This section discusses stages 2 & 3 of our pipeline. For reference, the intermediate datasets used in the pipeline are shown in Table 1.

As discussed in Section 3, data obtained from an MRI source typically contains noise that is not only apparent in the final raw information obtained from the reverse Fourier transform of the original signal, but also in the computation of the velocity using phase-contrast velocity mapping. When performing statistical measures of the data, the noise can be presumed Gaussian (though it is Rice distributed) and taken into account. Noise during visualisation however (in particular, temporal noise) can be distracting and produces misleading results where noise artefacts and temporally incoherent intensities are mapped to different isosurfaces and transfer function values. Our work utilises an optical flow algorithm in conjunction with screen-space temporal coherence methods to improve these temporal inconsistencies.

5.1. Raycasting Intensity and Motion

The main raycasting operation, including evaluation of the motion-moderated transfer function, is performed and written to a bound *colour buffer* (see Table 1). Each pixel is composited using the standard volume rendering 'under' operator, evaluating the *intensity volume* from front to back.

In addition to the colour composition, the *motion volume* is composed along the ray into a *motion buffer* using the same 'under' operator. The blending of motion attribute values occurs using the same opacity component as with the colour buffer, but uses the motion vector sampled from the

	Usage
Intensity Volume (**3D**)	The CMR volume dataset; each voxel is a scalar intensity value (3)
Motion Volume (**3D**)	Generated from above using optical flow estimation (5.3)
Colour Buffer (**2D**)	Intermediate per-frame composition of colour/opacity (5.1)
Motion Buffer (**2D**)	Per-frame motion vectors projected to framebuffer space (5.1)
Reverse Buffer (**2D**)	Stores reversed motion vectors for chain-of-flow (5.2.1)

Table 1: *A reference for the datasets used in our method.*

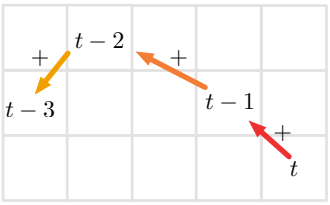

Figure 6: *Improving temporal coherence by following the 'chain of flow' backwards in time.*

motion volume instead. In this way, we obtain an indication of the effect of the motion volume on each pixel given the current transfer function and viewing parameters. After the main ray loop exits, this composited motion vector gives an indication of the visible direction of flow for the pixel given the viewing parameters. Its magnitude similarly gives an indication of the magnitude of motion in that projected area. Figure 7 shows an illustrative maximum intensity projection rendering of the motion magnitudes from the motion volume. Next, we explain why the motion is composited to the motion buffer, and how it is used.

5.2. Composing Colour and Motion & Improving Temporal Coherence

The final stage of our visualisation pipeline is to blend the colour and motion buffers for display on screen. Displaying the colour buffer on its own would display the result of raycasting the motion-moderated transfer function using the intensity and motion information. Due to the generally poor temporal coherence of motion data from frame to frame however, we wish to temporally smooth the resulting motion contributions from each pixel to not only give the user a more balanced and stable representation of the current velocities, but also to reduce the presence of noisy motion in the image output.

Our strategy for improving the temporal coherence of the motion data and reducing noise is performed entirely in image-space, and evaluated cheaply in a GPU shader operating upon a series of $\{colour, motion\}$ buffer pairs comprising not only the buffers for the current frame, but also a fixed-length series of buffers from previous frames. These $\{colour, motion\}$ buffer pairs are maintained in a fixed-size queue buffer. The raycasting operations from Stage 2 of the pipeline insert their buffer pair into the end of the queue, with the front buffers of the queue removed to remain size constancy. The composition stage now has access to n $\{colour, motion\}$ buffer pairs on which to weight its final colour framebuffer results.

We can improve the temporal coherence of our composition using the observation that the image-space motion information from the previous frame can be used to predict the image-space motion of colour buffer contents from the previous frame into the current frame. The colour at a pixel p in the framebuffer can include not only the colour in the current framebuffer, but also colour from previous buffers by following the trail of flow back in time (see Figure 6). Assume that for each pixel in the framebuffer, a vector $\vec{r}$ exists that denotes the reverse motion (we discuss the creation of this vector next). Starting at the current framebuffer, the colour at p is sampled and weighted. p's position is now incremented by r so that $p' = p + \vec{r}$, and the colour sampled at the previous framebuffer at location p' is weighted and added to the contribution. This process continues for all past framebuffers (in the queue) until a weighted total colour is produced. This scheme favours coherent motion, and disfavours noisy motion.

5.2.1. Reversing Motion Vectors

In order to produce a framebuffer containing reversed flow vectors from the motion buffers, an OpenCL kernel reads vectors $\vec{v}$ from the motion buffer at each pixel p, and computes $p' = p + \vec{v}$. The vector is now reversed to become $\vec{r}$, and stored in a *reverse buffer* at pixel location p'. In the case of an existing vector in p', the new and existing reverse vectors are averaged. This strategy creates obvious holes in the buffer as some pixels will have no motion vectors that terminate within, for which we use kernel density estimation to fill these areas. A window of fixed size is placed around each pixel, and for n neighbours containing reverse vectors, the contribution for central pixel p becomes:

$$\frac{1}{n}\sum_{i=1}^{n} K_h(||\, p - p_i \,||)\vec{r}_i$$

where $||\, p - p_i \,||$ is the normalised distance between central pixel p and intra-window pixel p_i, $\vec{r}_i$ is the reverse vector at pixel position i, and K is the Epanechnikov kernel:

$$K(u) = \frac{3}{4}(1-u^2)\mathbf{1}_{\{|u|\leq 1\}}$$

where $\mathbf{1}_{\{|u|\leq 1\}}$ is the indicator function. The window size is usually defined as to contain the maximum expected motion

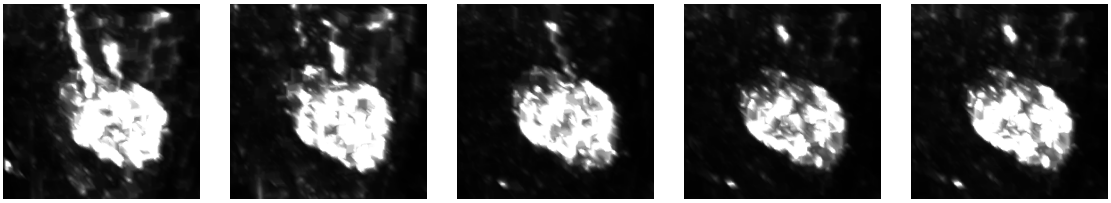

Figure 7: *An illustrative set of animation frames showing the magnitudes of the motion vectors from the motion field.*

magnitude. For motion data generated by an optical flow algorithm, the window size can be defined as the optical flow window size. For motion data of unknown origin (such as from phase-contrast velocity mapping), the above process must keep track of the largest motion vector found in the first stage to judge the window size for the second stage.

5.2.2. Final Composition using Reverse Motion

The reverse motion vectors from the previous step give an indication of the 'chain of flow' through the past n frames. Using this information, a colour composition can be achieved by weighting the contribution of the colour buffer with the contributions of previous colour buffers by following the flow for each pixel in reverse and compositing the colour found at each new location into the final composition. Algorithm 1 gives a recursive solution, where parameter i is the current iteration level (initial: 0), *samplePt* is the current pixel location, and *max* is the maximum depth. *max* is set to the number of past-history buffer pairs that are accessible. In our implementation, we use four buffers. For a pixel p, the function is called with the pixel's current buffer position, and returns a composited colour based on the motion-corrected contributions from previous colour buffers.

First, reverse motion vector $\vec{r}$ is sampled from the *reverse motion* buffer at the current sample point. If the current iteration level is zero, then the colour contribution is simply defined as the current sample point. If the iteration level is greater, then the reverse motion vector $\vec{r}$ sampled at *samplePt* is followed to its endpoint, which becomes the new colour sampling point. The colour is weighted according to a weighting factor ω_i (where $\omega_1 + \omega_2 + \ldots + \omega_{max} = 1$) that gives a higher weighting to the most recent data.

Algorithm 1 motionChain(i,samplePt,max)

```
if i = max then
  return 0
else
  r ← sample2D(reverseMotionBuffer_i , samplePt)
  motionEndPt ← samplePt
  if i > 0 then
    motionEndPt ← samplePt + r
  end if
  c ← sample2D(colourBuffer_i , motionEndPt)
  return ω_i c + motionChain(i + 1 , motionEndPt , max)
end if
```

5.3. Implementation Details

Our visualisation framework was developed in C++ with the use of OpenGL/GLSL for graphical operations and OpenCL for our GPU implementation of optical flow.

To provide real-time flow information to the visualisation pipeline, a custom GPU implementation of pyramidal three-dimensional Lucas-Kanade optical flow was developed using OpenCL. For each frame, two volume datasets are provided to the OpenCL kernel representing the current volume and the previous volume in the sequence. The kernel first uses a per-workgroup shared memory scheme to cache the current workgroup's texture values which provides a significant reduction in computation time. The optical flow kernel first computes the gradient sums to construct the main optical flow matrix G, and checks its determinant to determine whether the matrix can be solved. If it can, G's inverse G^{-1} is computed and used to iteratively determine the optical flow within the current window. Once the computed mismatch vector's contribution is less than a small threshold ε, or the maximum number of iterations is achieved (we use 5 iterations), the flow vector is written to a floating point buffer, and the buffer is provided to OpenGL as a 3D texture.

6. Results & Conclusions

We have experimented with our method using 12 of the research patient CMR datasets available that represent a variety of pathologies. Based on the data in the colour buffer and motion buffers, these attributes can be combined by the user in order to reveal the heart structure by mapping to colour or opacity. Figure 8 shows the user gradually revealing the heart by customising these attribute combinations (intensity, intensity gradient, and motion magnitude) together and mapping the final value to opacity.

Figure 9 shows a wider range of combinations of parameters, with the final value mapped to *(top)* input to a colour lookup table; and *(bottom)* the opacity channel. Figure 10 compares *(top)* a standard direct volume rendered representation of the intensity volume; and *(bottom)* a basic lookup table used with the motion magnitude mapped to opacity. Figure 11 shows the gradient of the motion field mapped to

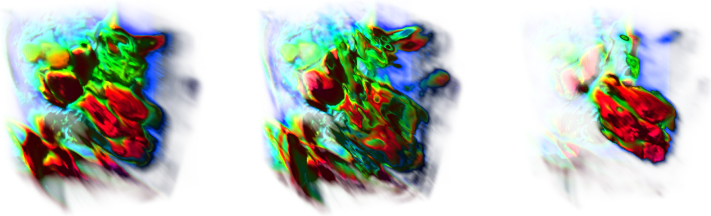

Figure 8: *The user reveals the moving heart boundary by mapping intensity and motion attributes to opacity. Attributes from left to right: intensity I; intensity gradient ∇I; motion magnitude multiplied by intensity $\mu \cdot I$.*

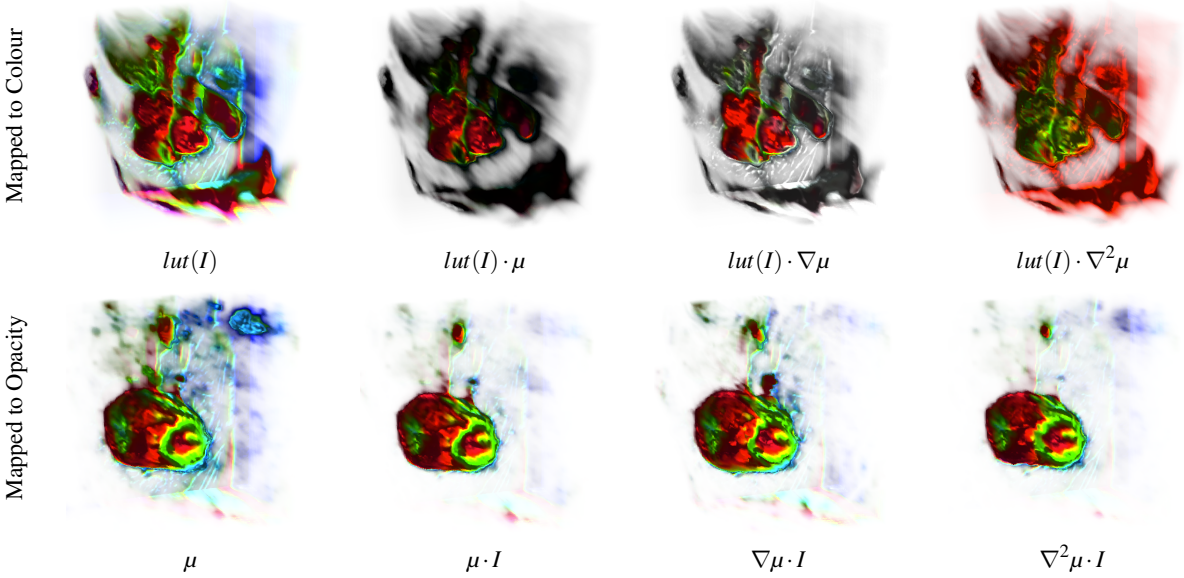

Figure 9: *Here we show the effect of using different combinations of computed attributes as a final output scalar value. The attributes shown are: μ: The magnitude of the motion vector from the motion buffer; I: the original intensity value from the intensity field; $lut(I)$: the RGB triplet in the lookup table for intensity value I. The attributes are mapped to (top row) colour, and (bottom row) opacity.*

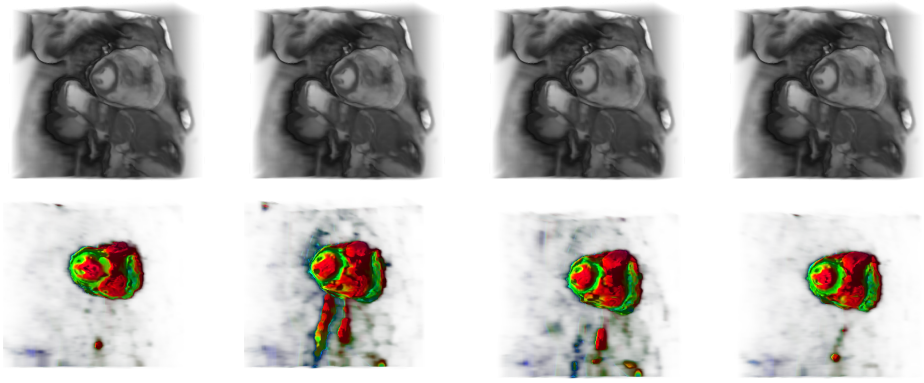

Figure 10: *Comparing (top) a limited-slice direct volume rendering of the intensity volume with (bottom) the motion magnitude μ mapped to opacity and RGB values from a red/green lookup table. The structure of the heart is revealed by its motion.*

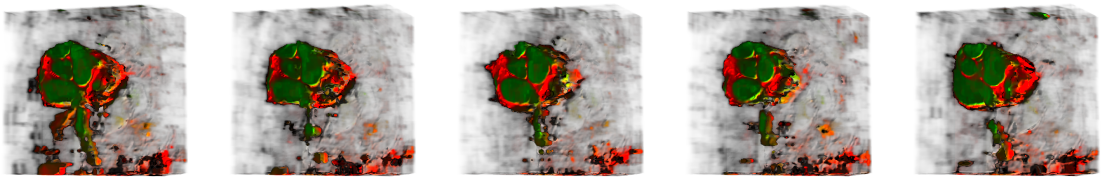

Figure 11: *Using the gradient of the motion magnitude $\nabla\mu$ mapped to opacity in order capture the heart, with a red $\rightarrow$ green LUT. The motion magnitude highlights the 'rate of change' in the data according to the computed motion field.*

opacity to show the heart surface's rate of change. Although relatively noisy, the areas under acceleration are visible. By manipulating combinations of motion and intensity parameters, including their derivatives, and mapping the results to colour/opacity, the user is able to highlight and reveal the structures of motion within the dataset for known contexts in which those parameters are the most effective. In this manner, such a system provides a set of tools for visually segmenting temporal volume data in a simple manner.

In this paper, we tackled the challenging problem of visualising 4D CMR data. We proposed a method that addresses a number of technical difficulties, including change of scalar quantities of tissue due to motion; low volume resolution along the z-axis, a high level of noise; and lack of volumetric velocity field in clinical data. We have successfully used motion to moderate the traditional transfer function, enabling us to display a beating heart in a time sequence consistently and with minimal occlusion of distant or nearby tissues. In addition, a GPU implementation supports the estimation of volumetric optical flow, volume rendering and composition, and temporally coherent animation.

We hope that the work will enable our clinical research partner to *(a)* investigate the feasibility of volume visualisation in a clinical setting, and *(b)* to create a justifiable demand for future CMR scans to support higher resolution data capture in the z-dimension. Our future work includes an investigation into combined volume and flow visualisation for studying blood flows featured in CMR data.

References

[AEW00] A. E. WATERFALL T. J. ATHERTON K. A.: 4D volume rendering with the shear warp factorisation. In *Proc. IEEE Symposium on Volume Visualization*. 2000, pp. 129–137. 2

[AFM06] AKIBA H., FOUT N., MA K.-L.: Simultaneous classification of time-varying volume data based on the time histogram. In *Proc. EuroVis*. 2006, pp. 171–178. 3

[AFM07] A. FRYDRYCHOWICZ T. A. BLEY S. D. J. H. M. L., MARKL M.: Visualization of vascular hemodynamics in a case of a large patent ductus arteriosus using flow sensitive 3D CMR at 3T. *Journal of Cardiovascular Magnetic Resonance 9*, 3 (2007), 585–587. 2

[BBP08] BLAAS J., BOTHA C. P., POST F. H.: Extensions of parallel coordinates for interactive exploration of large multi-timepoint data sets. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1436–1451. 2

[BG07] BRUCKNER S., GRÖLLER M. E.: Style transfer functions for illustrative volume rendering. *Computer Graphics Forum 26*, 3 (2007), 715–724. 2

[CM08] CORREA C. D., MA K.-L.: Size-based transfer functions: a new volume exploration technique. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1380–1387. 2

[CM09] CORREA C. D., MA K.-L.: The occlusion spectrum for volume classification and visualization. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1465–1472. 2

[CR08] CABAN J. J., RHEINGANS P.: Texture-based transfer functions for direct volume rendering. *IEEE Transactions on Visualization and Volume Graphics 14*, 6 (2008), 1364–1371. 2

[CS05] CORREA C. D., SILVER D.: Dataset traversal with motion-controlled transfer functions. In *Proc. IEEE Visualization*. 2005, pp. 359–366. 2

[CSW*03] CHEN M., SILVER D., WINTER A. S., SINGH V., CORNEA N.: Spatial transfer functions - a unified approach to specifying deformation in volume modeling and animation. In *Proc. Volume Graphics*. 2003, pp. 35–44. 2

[CWMW11] CHEN C.-K., WANG C., MA K.-L., WITTENBERG A. T.: Static correlation visualization for large time-varying volume data. In *Proc. IEEE Pacific Visualization*. 2011, pp. 27–34. 2

[FBT98] FANG S., BIDDLECOME T., TUCERYAN M.: Image-based transfer function design for data exploration in volume visualization. In *Proc. IEEE Visualization*. 1998, pp. 319–326. 2

[GSHK04] GAO J., SHEN H.-W., HUANG J., KOHL J. A.: Visibility culling for time-varying volume rendering using temporal occlusion coherence. In *Proc. IEEE Visualization*. 2004, pp. 147–154. 2

[GZM97] GUTTMAN M. A., ZERHOUNI E. A., MCVEIGH E. R.: Analysis of cardiac function from MR images. *IEEE Computer Graphics and Applications 17*, 1 (1997), 30–38. 2

[HPB*10] HAIDACHER M., PATEL D., BRUCKNER S., KANITSAR A., GRÖLLER M. E.: Volume visualization based on statistical transfer-function spaces. In *Proc. IEEE Pacific Visualization Symposium*. 2010, pp. 17–24. 2

[JEG11] JANG Y., EBERT D., GAITHER K.: Time-varying data visualization using functional representations. *IEEE Transactions on Visualization and Computer Graphics Pre-publication* (2011), DOI 10.1109/TVCG.2011.54. 2

[JM01] JANKUN-KELLY T. J., MA K.-L.: A study of transfer function generation for time-varying volume data. In *Proc. Volume Graphics*. 2001, pp. 51–68. 3

[JR05] JOSHI A., RHEINGANS P.: Illustration-inspired techniques for visualizing time-varying data. In *Proc. IEEE Visualization*. 2005, pp. 679 – 686. 2

[JW06] J. WOODRING H.-W. S.: Multi-variate, time varying, and comparative visualization with contextual cues. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 909–916. 2

[JWSK07] JÄNICKE H., WIEBEL A., SCHEUERMANN G., KOLLMANN W.: Multi-field visualization using local statistical complexity. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1384–1391. 3

[Lev37] LEVOY M.: Display of surfaces from volume data. *IEEE Computer Graphics and Applications 8*, 3 (29-37). 2

[Ley10] LEYVA F.: Cardiac resynchronization therapy guided by cardiovascular magnetic resonance. *Journal of Cardiovascular Magnetic Resonance 12* (2010), 64. 2

[LM04] LUM E. B., MA K.-L.: Lighting transfer functions using gradient aligned sampling. In *Proc. IEEE Visualization*. 2004, pp. 289–296. 2

[LS08] LU A., SHEN H.-W.: Interactive storyboard for overall time-varying data visualization. In *Proc. IEEE Pacific Visualization*. 2008, pp. 143–150. 2

[LS09] LEE T. Y., SHEN H.-W.: Visualization and exploration of temporal trend relationships in multivariate time-varying data. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1359–1366. 2

[NM02] NEOPHYTOU N., MUELLER K.: Space-time points: 4D splatting on efficient grids. In *Proc. IEEE/ACM SIGGRAPH Symposium on Volume Visualization and Graphics*. 2002, pp. 97–106. 2

[PCC*10] PRICE A. N., CHEUNG K. K., CLEARY J. O., CAMPBELL A. E., RIEGLER J., LYTHGOE M. F.: Cardiovascular magnetic resonance imaging in experimental models. *The Open Cardiovascular Medicine Journal 4* (2010), 278–292. 1, 2

[PFF02] POON M., FUSTER V., FAYAD Z.: Cardiac magnetic resonance imaging: a one-stop-shop evaluation of myocardial dysfunction. *Curr Opin Cardiol 17* (2002), 663–70. 2

[PLB*01] PFISTER H., LORENSEN B., BAJAJ C., KINDLMANN G., SCHROEDER W., AVILA L. S., MARTIN K., MACHIRAJU R., LEE J.: The transfer function bakeoff. *IEEE Computer Graphics and Applications 21*, 3 (2001), 16–22. 2

[Poh08] POHOST G. M.: The history of cardiovascular magnetic resonance. *JACC Cardiovascular and Imaging 1* (2008), 672–678. 2

[RBB*11] RUIZ M., BARDERA A., BOADA I., VIOLA I., FEIXAS M., SBERT M.: Automatic transfer functions based on informational divergence. *IEEE Transactions on Visualization and Computer Graphics 17*, 2 (2011), 1932–1941. 2

[RE01] RHEINGANS P., EBERT D.: Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics 7*, 3 (2001), 253–264. 2

[RSKK06] REZK-SALAMA C., KELLER M., KOHLMANN P.: High-level user interfaces for transfer function design with semantics. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1021–1028. 2

[SdDL*99] SIJBERS J., DEN DEKKER A., LINDEN A. V. D., VERHOYE M., DYCK D. V.: Adaptive anisotropic noise filtering for magnitude mr data. *Magnetic Resonance Imaging 17* (1999). 4

[SJEG05] SVAKHINE N. A., JANG Y., EBERT D., GAITHER K.: Illustration and photography inspired visualization of flows and volumes. In *Proc. IEEE Visualization*. 2005, pp. 687–694. 2

[SVSG06] SEREDA P., VILANOVA-BARTROLI A., SERLIE I. W. O., GERRITSEN F. A.: Visualization of boundaries in volumetric data sets using LH histograms. *IEEE Transactions on Visualization and Computer Graphics 12*, 2 (2006), 208–218. 2

[SW97] SILVER D., WANG X.: Tracking and visualizing turbulent 3D features. *IEEE Transactions on Visualization and Computer Graphics 3*, 2 (1997), 129–141. 2

[SWB*00] SATO Y., WESTIN C.-F., BHALERAO A., NAKAJIMA S., SHIRAGA N., TAMURA S., KIKINIS R.: Tissue classification based on 3d local intensity structures for volume rendering. *IEEE Transactions on Visualization and Computer Graphics 6*, 2 (2000), 160–179. 2

[TC00] TREAVETT S., CHEN M.: Pen-and-ink rendering in volume visualization. In *Proc. IEEE Visualization*. 2000, pp. 203–209. 2

[TCS*11] TÖGER J., CARLSSON M., SÖDERLIND G., ARHEDEN H., HEIBERG E.: Volume tracking: A new method for quantitative assessment and visualization of intracardiac blood flow from three-dimensional, timeresolved, three-component magnetic resonance velocity mapping. *BMC Medical Imaging 11* (2011), 10. 2

[TME15] THENNAKOON W. K., MADHAVI W., EKANAYAKA R.: Automatic classification of left ventricular function of the human heart using echocardiography. *Proceedings of the Technical Sessions 31* (2015), 83–90. 2

[TRM*01] TORY M., ROBER N., MÖLLER T., CELLER A., ATKINS M. S.: 4D space-time techniques: a medical imaging case study. In *Proc. IEEE Visualization*. 2001, pp. 473–592. 2

[VPG11] VUÇINI E., PATEL D., GRÖLLER M. E.: Enhancing visualization with real-time frequency-based transfer functions. In *Proce. IS&T/SPIE Conference on Visualization and Data Analysis*. 2011, pp. 78680L:1–12. 2

[WBT*14] WALTON S., BERGER K., THIYAGALINGAM J., DUFFY B., FANG H., HOLLOWAY C., TREFETHEN A. E., CHEN M.: Visualizing cardiovascular magnetic resonance (cmr) imagery: Challenges and opportunities. *Progress in biophysics and molecular biology 115*, 2 (2014), 349–358. 1

[WC01] WINTER A. S., CHEN M.: vlib: A volume graphics API. In *Proc. Volume Graphics*. 2001, pp. 133–147. 2

[WGLS05] WANG C., GAO J., LI L., SHEN H.-W.: A multiresolution volume rendering framework for large-scale time-varying data visualization. In *Proc. Volume Graphics*. 2005, pp. 11–19. 2

[WLY04] WÜNSCHE B. C., LOBB R., YOUNG A. A.: The visualization of myocardial strain for the improved analysis of cardiac mechanics. In *Proc. GRAPHITE* (2004), pp. 90–99. 2

[WQ07] WU Y., QU H.: Interactive transfer function design based on editing direct volume rendered images. *IEEE Transactions on Visualization and Computer Graphics 13*, 5 (2007), 1027–1040. 2

[WYG*11] WANG C., YU H., GROUT R. W., MA K.-L., CHEN J. H.: Analyzing information transfer in time-varying multivariate data. In *IEEE Pacific Visualization*. 2011, pp. 99–106. 3

[WYM08] WANG C., YU H., MA K.-L.: Importance-driven time-varying data visualization. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1547–1554. 3

[YMC05] YOUNESY J., MÖLLER T., CARR H.: Visualization of time-varying volumetric data using differential time-histogram table. In *Proc. Volume Graphics*. 2005, pp. 21–30. 3

[ZXCW05] ZHANG C., XUE D., CRAWFIS R., WENGER R.: Time-varying interval volumes. In *Proc. Volume Graphics*. 2005, pp. 99–108. 2