5-2018

Phase unwrapping of 4D-flow MRI data with graph cuts.

Andrew Justice
University of Louisville

Title: Phase Unwrapping of 4D-flow MRI Data with Graph Cuts

First Author: Andrew Justice

Co-authors: Sean Callahan, Jungwon Cha, Amir Amini

Institutions:

[1]Medical Imaging Lab, Department of Electrical and Computer Engineering, University of Louisville, Louisville, KY 40292.

Corresponding Author: Amir A. Amini

Contact information: 409 Lutz Hall, University of Louisville, Louisville, KY, 40292

Tel: (502)852-4767, E-mail: amir.amini@louisville.edu

Word count: _____

Keywords: Phase Contrast MRI; 4D flow MRI; Phase Unwrapping; Graph Cuts; Max-flow Min-cut Thereom

Source of Funding: This research has been supported by NIH Grant R21HL132263

Abstract

A common issue when measuring velocity utilizing 4D flow magnetic resonance imaging (MRI) is aliasing that occurs because of a low velocity encoding parameter (V_{enc}). Aliasing can be avoided if the velocity encoding parameter is set above the largest velocity quantity. However, when this is done the velocity to noise ratio is lowered less detail is acquired in the image. Thusly, it is sometimes desirable to have a V_{enc} below the maximum velocity to acquire higher quality data.

Consequently, an efficient and robust algorithm is needed to unwrap the aliased data. This paper proposes an iterative graph cuts algorithm to perform the phase unwrapping on the 4D flow data. This graph cuts algorithm utilizes a global energy minimization framework and is largely based on the Phase Unwrapping Max-Flow Algorithm (PUMA) proposed in [1]. However, the algorithm presented in this paper is extended to account for the added dimensions of data. Previous methods that have been utilized for 4D phase unwrapping include the Laplacian method and the Spatial Gradients method presented in [2] and [3] respectively. Of these, the Laplacian method has proven the most successful in phase unwrapping quality and time. The proposed graph cuts method is shown to unwrap aliased 4D data more accurately than these existing techniques. This includes unwrapping synthetic data with V_{enc} values down to 20% of the max velocity and SNRs down to 2.

Introduction:

Phase unwrapping is a mathematical problem that has been studied intensely for decades. Its research implications impact various fields including optical interferometry, synthetic aperture radar interferometry (InSar), and magnetic resonance imaging (MRI). In these applications, phase is a quantity that is constrained to a specific range. Typically that range is $[-\pi, \pi)$. When the phase extends outside of this range, the phase will wrap until it is back in the $[-\pi, \pi)$ range. Consequently, information is recorded incorrectly and must be reversed. This is done by finding the integer value, n, that will return the wrapped phase (φ_w) to its true phase φ.

$$\varphi(r) = \varphi_w(r) + 2\pi n(r) \tag{1}$$

When the velocity encoding parameter (V_{enc}) is less than the peak fluid velocity measured phase wrapping will occur at all velocities past that velocity encoding parameter. When this phase wrapping occurs, the aliased data does not accurately represent the true velocity of fluid being measured. Ideally, a quick and effective phase unwrapping solution could immediately be applied to accurately and precisely unwrap the data.

There is, indeed, a quick and simple solution to the phase unwrapping problem derived by Itoh in the [4]. In [4], an algorithm was derived that largely exploited the Itoh condition (see equation 2 and 3) to unwrap data quickly. While this solution works well for data without noise, it quickly loses effectiveness when applied to more complex examples with noisy data and undersampling. Thusly, it is not a practical algorithm for the real world. However, the Itoh condition is still a foundation for many of the existing phase unwrapping algorithms. The Itoh condition states that if the phase gradient is less than π (equation 2), then the change in phase for the unwrapped image is equal to the change in phase for the wrapped image (equation 3).

$$|\Delta\varphi| < \pi \tag{2}$$

$$\Delta\varphi = \Delta\varphi_w \tag{3}$$

The phase unwrapping problem has been a challenge for many decades and while the 4D phase unwrapping is a relatively new subset of the problem, many of the original techniques used for

unwrapping of fewer dimensions still hold value. These techniques include: L^P-Norm minimization algorithms, path-following algorithms, region-growing algorithms, and Fourier transform method algorithms.

The first of these is the L^P-Norm minimization algorithm. This technique is largely based on the assumption that unwrapped phase values will satisfy the Itoh condition (equation 2). Consequently, the phase gradient of the wrapped and unwrapped images should be equal (equation 3). The L^P-Norm minimization methods essentially adopt a global optimization strategy that solves for the unwrapped phase [5]. While this technique works with various norm calculations, typically the L^2-Norm is the norm of choice due to the computational efficiency (utilizes the Fast Fourier Transform) of the calculation and the accuracy of the result. The primary disadvantage of this method is that occasionally true discontinuities within the phase image get smoothed over. For the least squares norm (L^2-Norm) the equation can be seen below (equation 4) for a 2D image.

$$E = \min\left(\sqrt{\sum_{x,y}\left|\frac{\partial\varphi_w(x,y)}{\partial x} - \frac{\partial\varphi(x,y)}{\partial x}\right|^2 + \sum_{x,y}\left|\frac{\partial\varphi_w(x,y)}{\partial y} - \frac{\partial\varphi(x,y)}{\partial y}\right|^2}\right) \qquad (4)$$

Here, gradients in the x and y direction are estimated using the wrapped phase differences. We then want to minimize the difference error between the gradient of the wrapped and unwrapped phase estimates based on the data quality at each point. Phase unwrapping here minimizes the squares of the differences between the gradients of the wrapped input phase and those of the unwrapped phase. An essential concept in phase unwrapping is comparison to near neighbor voxels. If the difference between two neighboring voxels is greater than π, then typically one can be suspicious that it might be wrapped based on the Itoh condition. However, one cannot be sure because of noise, discontinuities, etc. Thus, the L^2-Norm is a way to analyze this voxel and see how quickly it changes [6].

The Path-Following Algorithm is another common technique utilized for phase-unwrapping. The Path-Following Algorithm, like the L^P-Norm minimization technique attempts to utilize the gradient of the phase map to unwrap the wrapped image. The primary equation for the path-following algorithm is $\varphi(r) = \int_c \nabla\varphi(r) * dr + \varphi(r_0)$ where C is the path in domain D

connecting points r and r_0. Path independence is broken when the line integral of the phase gradient around a closed loop (4 voxels) is not equal to zero (i.e. $\oint_C \nabla_\emptyset(r) * dr \neq 0$). The Path-Following Algorithm (which is occasionally called the Branch Cut Method) relies on the residue detection. For phase maps, there is a basic assumption that unwrapped phase has local phase gradients of less than π everywhere. However, with noise, we occasionally get residues that make this unwrapping inconsistent. Thusly, branches are placed to avoid unwrapping around these residues. Once branches are placed based on residue points, the algorithm will prevent the integration path from crossing the branches [6].

Region-Growing Algorithms are another popular method for phase unwrapping of traditional 2D images. With region-growing algorithms a seed point in the phase image is first chosen. This seed point typically is a point of high-quality signal. Once the seed point is chosen, phase unwrapping commences and nearest neighbors are unwrapped. Thus, the region of unwrapped phase "grows" until the entire image is unwrapped. Again, the phase gradient is important for determining whether the nearest neighbors are unwrapped or not. A threshold value (α) is chosen and if $\Delta\varphi > \alpha$ then 2π is added to the nearest neighbor. This region-growing is done until the entire image has been unwrapped [6].

This paper will primarily focus on the phase unwrapping problem as it pertains to 4-dimensional flow MRI i.e. 4D Phase Contrast(PC) MRI. This is the type of flow imaging seen in [7] and [8]. In both [7] and [8], higher velocities happen at the stenosis of the phantom (where diameter shrinks). Consequently, these stenosis areas are the most likely area in which aliasing will be found.

The 4-dimensions account for data in the x, y, z dimensions along with the temporal dimension. PC-MRI uses bipolar gradient pulses to create a linear relationship between fluid velocity, v, and the phase shift, φ, of the MRI. This linear relationship is dependent on the velocity encoding parameter, V_{enc}, and can be seen in the following equation.

$$v(\varphi) = \frac{\varphi * V_{enc}}{\pi} \tag{5}$$

V_{enc} relates to the velocity-to-noise ratio (VNR) via the equation $VNR = \frac{|v|*\pi}{V_{enc}}$ where v is the fluid velocity. For 4D flow imaging, velocity is relational to signal strength as the intensity of the image should correspond linearly with velocity of the fluid (provided no aliasing occurs). Clearly, when analyzing the VNR equation, as the V_{enc} increases the SNR will decrease which is not ideal. This suggests that the V_{enc} should be kept just above the highest fluid velocity to be measured to maintain a serviceable SNR while also avoiding aliasing. However, if the aliasing of an image can be unwrapped with accurate results, a better SNR can be achieved and as a result, better image quality [2].

To date, few traditional phase unwrapping techniques have been applied to perform 4D flow phase unwrapping when velocity aliasing is present. Previously proposed techniques include Laplacian methods [2]and spatial gradient [3] techniques. This paper explores a 4-dimensional graph cuts unwrapping technique for unwrapping aliased velocities in 4D flow MRI.
The spatial gradient technique mentioned above was one of the first attempts at unwrapping 4D data. It focused on iteratively calculating the probability of a voxel being wrapped [3]. The algorithm calculates the probability of a wrapped voxel based on weighing constants and the measured differences between the voxel of interest and its neighboring voxels. A threshold is then decided on to unwrap voxels above a certain probability level.

The most recent technique that has demonstrated a capacity to unwrap 4D data is the Laplacian method by Loecher, etc. [2]. This technique manipulates the standard offset equation (see equation 1) for phase unwrapping and solves for the phase integer offset. The Laplacian of the true phase is then solved for via the euler identity. The result is seen in equation 6.

$$\nabla^2 \emptyset(r) = cos\varphi(r)\nabla^2 sin\varphi(r) - sin\varphi(r)\nabla^2 cos\varphi(r) \quad (6)$$

The true phase is finally solved for algorithmically via a series of fast-fourier transforms (FFT) and inverse fast-fourier transforms (IFFT). The goal of this study was to develop an algorithm that allowed for more accurate unwrapping of 4D PC-MRI data.

Theory:

The 4-dimensional graph cuts unwrapping algorithm is largely an extension of the algorithm developed in [1] and previously applied to MRI in [9]. This algorithm takes the graph cuts algorithm (called PUMA for Phase Unwrapping Max-Flow Algorithm) developed in [1] and extends it from 2-dimensions (x and y) to 4-dimensions (x, y, z, and time). Graph cuts leverage the max-flow min-cut theorem and have been applied widely to image segmentation. At its core, this algorithm is largely a subset of the L^P-Norm minimization technique as the energy minimization is a generalization of the classical L^P-Norm [10].

Graph cuts are exceptionally popular in the realm of image segmentation, however, in recent papers they are gaining notoriety for their ability to unwrap wrapped phase in MR phase images. Generally, within graph cut theory, a graph is simply a set of vertices and edges that represent the image of interest. A graph can be represented in the notation $G = \langle V, E \rangle$ where V is the set of vertices and E is the set of edges [11]. In this case, the graph is representative of the MR image being analyzed. Furthermore, within graph cut theory, there is a source node, S, and a sink node, T. For our purposes, a cut separates the voxels into two disjoint subsets, S (set with the source node) and T (set with the sink node) [11]. One set represents voxels that will be further unwrapped while the other set represents voxels that will not be unwrapped further. Whenever a cut is made, the cost of the that cut is determined by the summation of all the weighted edges that the cut slices. The goal of the graph cuts algorithm is to find the cut that gives the minimum cost among all cuts. According to the Max-Flow Min-Cut Theorem, this minimum cut corresponds to the route that produces the maximum flow from the source node to the sink node. Regarding edges, one can consider two different types (n-link edges and t-link edges) as defined in [11]. N-links connect pairs of neighboring voxels while the T-links will connect a voxel to the source or sink. N-link edge costs are typically derived from the discontinuity penalty between voxels while the T-link edge costs are derived from the data term in the energy. See figure 1 for an illustration of cut from source to sink.

To leverage the max-flow min-cut theorem for phase unwrapping, edges between all voxels must be mathematically given a weight. Once these edge weights are determined, a minimum cut is made that also defines the maximum flow. Thusly, the algorithm determines the cut that will lead to the minimum energy for the entire dataset. The functions used to determine these weights are first-order Markov random fields (MRF). This minimum energy, once reached, corresponds to the generalized L^P-Norm [1]. Ideally, once the minimum energy is obtained the data will be unwrapped and the true velocity values will be revealed.

The Phase Unwrapping Max-Flow Algorithm (PUMA) unwraps images via a sequence of binary optimizations. Following each iteration, every voxel label is kept at its current value or increased by one.

$$K^{i+1} = K^{i} + n^{i+1} \tag{7}$$

where K is the wrap count integer value, n is a binary variable (0 or 1), and i represents the iteration count [1]. The wrap count integer determines the final unwrapping product using the equation $\varphi(r) = \varphi_w(r) + 2\pi K$ where φ is the unwrapped phase at voxel, r, and φ_w is the wrapped phase at voxel, r. For the 4D algorithm used in this paper the local energies are weighted for neighboring voxels in the spatial (x, y, z) and temporal dimensions. Each binary optimization decreases the total energy of the system until the total energy of the system ceases to decrease with the optimization.

This relationship between edge potential and energy function can be further understood by analyzing equations 8 and 9 (see below).

$$E_{xy}\left(n_x^{i+1}, n_y^{i+1}\right) = \left(\varphi_x - \varphi_y + 2\pi\left(K_x^i - K_y^i\right) + 2\pi\left(n_x^{i+1} - n_y^{i+1}\right)\right)^2 \tag{8}$$

In equation 8, the edge potential between voxels x and y (E_{xy}) depends on several factors. Generally, the edge potential will increase with larger differences between the wrapped phase of the neighboring voxels x and y. The edge potential equation also uses previous iterations to further fine-tune its value. If the neighboring integer offset values (K) have a difference between

them, this will increase the edge potential while a difference in the binary variable, n, between neighboring voxels will also increase the edge potential. The energy equation (equation 9) is merely a summation of all the edge potentials in an image. More details about the specifics of the PUMA algorithm will be revealed in the next section

$$E(n) = \sum_{xy} E_{xy}(n_x^{i+1}, n_y^{i+1}) \quad (9)$$

Phase Unwrapping Max-Flow Algorithm (PUMA):

1. *Load wrapped phase image*
2. *Initialize iteration count (i), initialize energy list, initialize kappa (K)*
3. *While Possible Improvement*
    a. *i++;*
    b. *Update energy list;*
    c. *Calculate Wrapped phase change for images and change in neighboring kappa values*
    d. *Perform energy calcuations to obtain E(0,0), E(1,0), E(0,1), and E(1,1)*
    e. *Construct graph by Defining various edges*
    f. *Perform Max-Flow Min-Cut Calcuation using mex file*
    g. *Obtain new kappa_aux value*
    h. *If (new energy < previous energy)*
        i. *Previous energy = new energy*
        ii. *Kappa = kappa_aux*
    i. *Else*
        i. *Possible_improvement = 0*
        ii. *Unwph = 2*pi*kappa+psi*
    j. *End if*
4. *End while*

The above pseudocode represents the PUMA algorithm utilized to unwrap the wrapped 4D-data. The PUMA algorithm was initially developed in [1] for 2D phase unwrapping and was extended to 4D for this paper.

The first few steps simply include loading the phase image into the program and initializing important variables and constants such as the iteration counter, the energy list, and the kappa integer. Next, a while loop is started in step 3. This loop will continue as long as there is a possibility for the total system energy to decrease with another iteration of the binary optimization. In step 3a and 3b, the energy list is updated with the newest energy calculation and the iteration counter is incremented by one. Step 3c takes the original phase image and

calculates the change in wrapped phase for neighboring voxels and the change in kappa for neighboring voxels. This highlights one area in which the algorithm of this paper differs from that of the Bioucas-Dias paper [1]. While the original 2D PUMA algorithm was only concerned with neighboring voxels in the vertical and horizontal direction (x and y directions), the 4D PUMA algorithm will look at neighboring voxels in the x, y, z, and t dimensions. The general equations used for step 3c can be seen below in equation 10 thru 17. In equations 10-13, the change in the wrapped phase image is denoted by the difference of two neighboring phase images [1]. In the equations below h, v, d, and t are representative of horizontal, vertical, depth, and time respectively.

$$\Delta\varphi^{h}_{xyzt} = \varphi_{x-1,yzt} - \varphi_{xyzt} \tag{10}$$

$$\Delta\varphi^{v}_{xyzt} = \varphi_{x,y-1,zt} - \varphi_{xyzt} \tag{11}$$

$$\Delta\varphi^{d}_{xyzt} = \varphi_{xy,z-1,t} - \varphi_{xyzt} \tag{12}$$

$$\Delta\varphi^{t}_{xyzt} = \varphi_{xyz,t-1} - \varphi_{xyzt} \tag{13}$$

The above equations denote the change in wrapped phase that is essential for calculating the edge potentials of the phase image. In equations 10-13 x, y, z, and t represent the directions that the phase image is shifted to perform the neighboring calculations. The same is true for the directions in equations 14-17 except, for that case, the difference in the kappa unwrapping integer is being calculated. [1].

$$\Delta K^{h}_{xyzt} = K_{x-1,yzt} - K_{xyzt} \tag{14}$$

$$\Delta K^{v}_{xyzt} = K_{x,y-1,zt} - K_{xyzt} \tag{15}$$

$$\Delta K^{d}_{xyzt} = K_{xy,z-1,t} - K_{xyzt} \tag{16}$$

$$\Delta K^{t}_{xyzt} = K_{xyz,t-1} - K_{xyzt} \tag{17}$$

Once the $\Delta\varphi$ and ΔK values are determined, some mathematical manipulation can be done to map the binary optimizations to the max-flow min-cut algorithm. To understand this, one must first obtain the change of corrected phase equation as seen in equation 18 [1].

$$\Delta\varphi = \left[2\pi\left(K^i - K^i_{nbhr}\right) - \Delta\varphi\right] \tag{18}$$

When $K^{i+1} = K^i + n^{i+1}$(equation 7) is substituted into equation 18, the result is seen in equation 19.

$$\Delta\varphi = \left[2\pi\left(K^i + n^{i+1} - K^i_{nbhr} - n^{i+1}_{nbhr}\right) - \Delta\varphi\right] \tag{19}$$

This further simplifies to equation 20.

$$\Delta\varphi = \left[2\pi\left(n^{i+1} - n^{i+1}_{nbhr}\right) + 2\pi(K^i - K^i_{nbhr}) - \Delta\varphi\right] \tag{20}$$

For simplicity the kappa and wrapped phase difference terms are combined into one term ($a = 2\pi(K^i - K^i_{nbhr}) - \Delta\varphi$) as seen in equation 21.

$$\Delta\varphi = \left[2\pi\left(n^{i+1} - n^{i+1}_{nbhr}\right) + a\right] \tag{21}$$

Utilizing these equations previously developed in [1] one can use the variable, a, to calculate the energy for all the edges. These equations for the different edge energies can be seen below in equation 22-25. One should also note that as the number of dimensions being analyzed increases, the number of edges will increase as well. For example, when PUMA was extended to 4D for this paper, it calculates the edges for 4 different directions whereas the original PUMA algorithm only calculated edges for the 2 primary directions of x and y. Energy values can be calculated in step 3d of the algorithm using equations 22-25.

$$E^{xy}(0,0) = V(a)d_{ij} \tag{22}$$

$$E^{xy}(1,1) = V(a)d_{ij} \tag{23}$$

$$E^{xy}(0,1) = V(-2\pi + a)d_{ij} \tag{24}$$

$$E^{xy}(1,0) = V(2\pi + a)d_{ij} \tag{25}$$

Once energy values are calculated, the various edges can be defined and thusly utilized to construct the graph. Here, the N-link and T-link edges are constructed. The N-link, or directed edge, which exists between neighboring pairs of voxels is calculated to be $E(0,1) + E(1,0) - E(0,0) - E(1,1)$. The T-link edges which exists between a voxel and a terminal can be defined

in a few different ways. The edge between terminal and voxel can be defined as $E(1,0) - E(0,0)$ or $E(0,0) - E(1,0)$ depending on which gives a positive result. $E(1,0) - E(0,0)$ represents an edge between the source and voxel while $E(0,0) - E(1,0)$ represents an edge between the sink and voxel.

Following the graph construction, step 3f follows though with the Max-Flow Min-Cut calculation. This was completed in Matlab via a MEX file that allows the algorithm to take advantage of a C++ function for processing. This MEX file comes from the [1]. The MEX file outputs a binary image that determines whether each voxel needs to increase in phase value or not. Thusly, this binary image is added to the kappa image to obtain an auxiliary kappa value that more accurately depicts the unwrapped image (step 3g).

Step 3h calculates an updated system energy which is then compared with the system energy before the max-flow min-cut algorithm was run. Because minimum energy is ideal for a system that most closely matches the true velocity values, the kappa is only updated if the new system energy is less than the old system energy. If this is the case, the algorithm returns to step 3a where at least one more iteration of energy minimization will be performed. Otherwise, as seen in step 3i, the flag that indicates possible improvement within the system energy is tripped. If this is the case, that is the last iteration for the graph cuts energy minimization function. The unwrapped image is then calculated using equation 1 and the final result can be displayed and saved.

Experimental Results:

Testing was done on synthetic and phantom data. The 4D synthetic data is representative of pulsatile flow through cylindrical tubes – diameters: 10mm, 20mm, 30mm, and 40mm. The peak velocity was 100 cm/sec while the V_{enc} was between 10 cm/sec and 100 cm/sec in increments of 10 cm/sec to create varying degree of aliasing. Complex Gaussian noise was added to the data to match the desired SNR level [1].

Real data was also collected using a rigid phantom of the LV outflow tract including an aortic valve on a 1.5 T Phillips Scanner (see figure 2) [8]. The phantom was then placed in a flow circuit through which a blood mimicking fluid (60% water, 40% glycerol) was pumped. The pump generates an ECG signal that is connected to the MRI scanner's ECG monitor. The pump was then programmed to drive a pulsatile waveform through the flow circuit. This simulates the physiologic flow that can be scanned with 4D imaging. The phantom was then scanned with a flip angle of 8 degrees, TR of 8.5 ms, TE of 4.1 ms, simulated heart rate of 72 beats/min, and FOV of 100x100x60 mm. A SENSE Knee coil was used to collect 4D flow images in the phantom. Real pulsatile flow velocity through the phantom was obtained with V_{enc} of 40, 80, 125, and 250 cm/sec. This was done to create various levels of aliasing when the V_{enc} was lower than the maximum velocity in the flow. 20 slices were collected through a phantom with a 1" diameter with in-plane resolution 1.5 mm x 1.5 mm and slice thickness = 3 mm. 16 temporal phases were collected. The peak systolic flow rate was set at 200 ml/sec while the peak velocity was observed at approximately 155 cm/sec. The representative excitation waveform and the measured waveform can be seen in figure 3 [8]. Wrapping was observed in all the scans in the systolic phase with the exception of the 250 cm/sec V_{enc} setting. The 250 cm/sec V_{enc} data was used as the true reference velocity data. The relative root mean squared error (RRMSE) between the reference data and the unwrapped data was used as an error metric. A personal computer with an Intel i7 processor and 8 GB RAM was used to perform all the experiments.

Results on Synthetic Data:

Ultimately, the graph cuts unwrapping technique proved completely successful (meaning all aliased voxels were accurately unwrapped) for $V_{enc}{}'s$ down to 20% of the max velocity and SNRs down to 2 (see figure 4). In comparison, the 4D Laplacian method was only completely effective for $V'_{enc}s$ down to 40% and SNRs down to 3. An example of the wrapped and unwrapped synthetic data can be seen in figure 5.

[INSERT FIGURE 4]

[INSERT FIGURE 5]

Results on Real Data:

Figure 6 shows aliased systolic and diastolic images from an in-vitro flow phantom experiment, displaying images at the level of a synthetic aortic valve and corresponding unwrapped data. Table 1 summarizes results from the in-vitro phantom study.

[INSERT TABLE 1]

[INSERT FIGURE 6]

Discussion:

In this paper, an extended version of the PUMA algorithm is applied to 4-dimensional data in order to undo the effects of phase wrapping caused by low V_{enc}. The algorithm accomplishes phase unwrapping via graph cuts which aim to globally optimize the energy of the system.

The graph cuts algorithm has proven a reliable method for unwrapping smaller datasets of PC-MRI data. When compared with the standard 4D PC-MRI Laplacian unwrapping algorithm utilizing the RRMSE metric, the graph cuts algorithm proved marginally more accurate. As seen in the case of synthetic data, especially in cases with low SNR, the performance was superior. The method however is computationally more expensive than the Laplacian method.

In order to limit the degrees of freedom, each binary iteration must have a positive step size. This results in an overall image shift of 2kπ in the final phase map. This shift can be undone by choosing a seed point that will not be unwrapped, comparing it with the final value of that seed point after unwrapping, and then subtracting the differences in the images [9].

Regarding algorithm complexity, the Laplacian algorithm in [2] proves superior. Because the Laplacian algorithm manipulates the Laplacian equations with FFTs to unwrap the data, the complexity of the algorithm is reducible to 4 FFTs [12]. The complexity of the algorithm is $O(n\log(n))$ where n represents the number of voxels in the 4D image. Utilizing the extended PUMA algorithm leads to a complexity of $O(KT)$ where K is the number of max-flow min-cut iterations and T is the complexity of one binary optimization [1]. Because the number of max-flow min-cut iterations is variable depending on the number of wraps that must be unwrapped,

the run-time will be variable. Moreover, because the complexity of one binary optimization is heavily dependent on the number of voxels being analyzed (which is significant in 4D images), the run-time is even more variable. As shown in [1], the worst complexity of the PUMA algorithm comes out to $O(n^{2.5})$ where n is the number of voxels in the 4D image.

Overall, the 4D phase unwrapping problem proves challenging in several areas. The two primary parameters that need to be balanced are algorithm speed and algorithm unwrapping accuracy. For the graph cuts method presented in this paper, the algorithm proves accurate relative to all other methods (specifically the Laplacian method) utilized for 4D phase unwrapping. However, improvement is needed in regards to speed for the graph cuts algorithm to become a more feasible algorithm for real-time phase unwrapping as data is collected.

Figures:

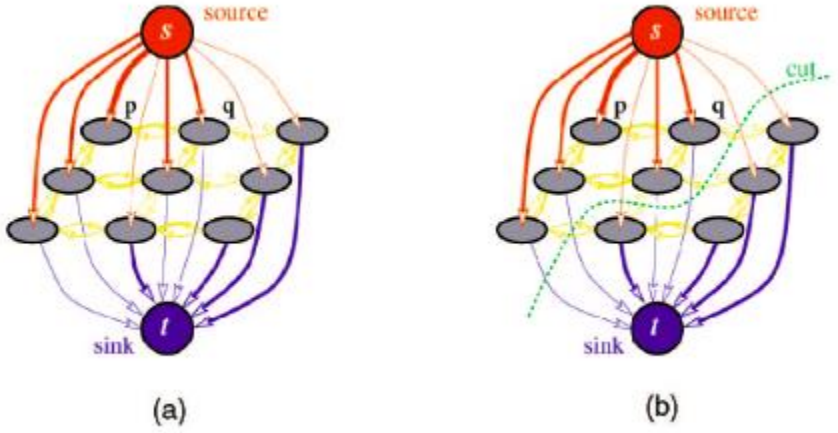
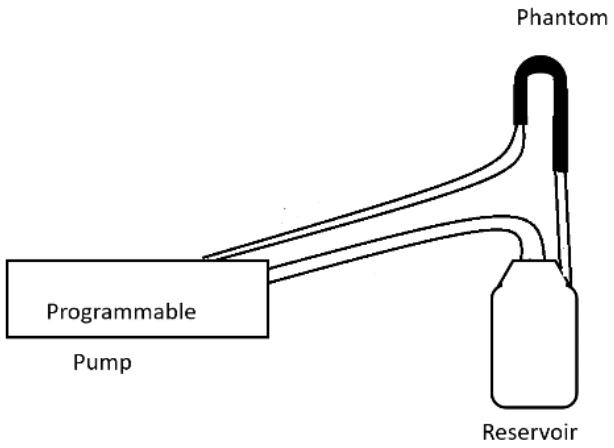


Figure 1 – Visual showing the source (s), sink (t), and voxels (p and q). (a) represents an example graph. (b) represents the same example graph with a cut taken through it. Line thickness between voxels is representative of the edge costs. Came from [11].

(A)



(B)

(C)

Figure 2 - (a) Schematic of the aortic valve flow circuit used in MRI experiments. (b) Calcified aortic valves used with varying degrees of stenosis (95%, 75%, 0%, and 50% respectively). (c) Aortic Arch Phantom Setup. The arrow points to the valve block. Taken from [13].

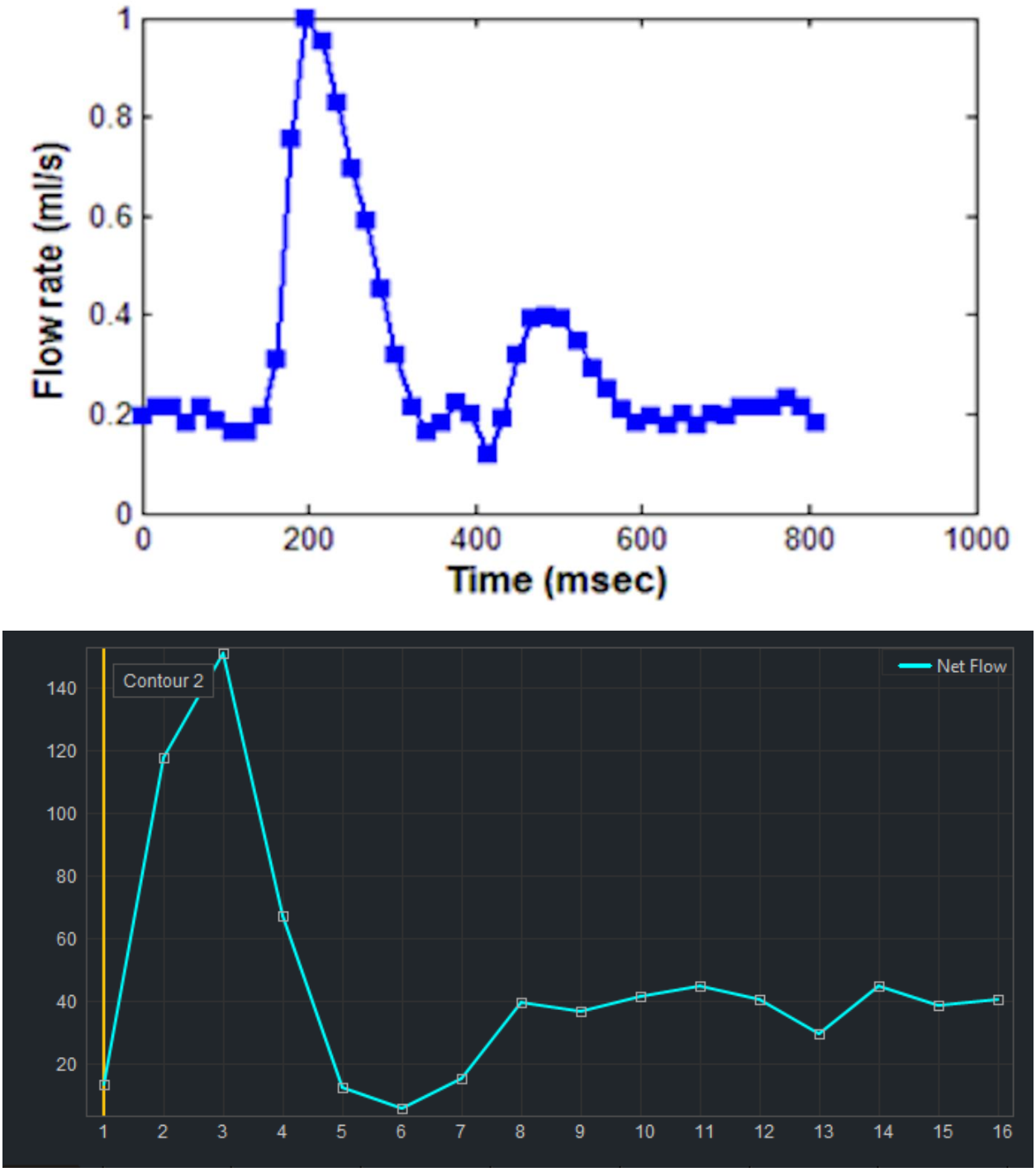


Figure 3 – Top: Normalized representative excitation waveform prescribed at the pump for pulsatile phantom experiments. The peak flow rate was adjustable and was set to 200 ml/s. Bottom: The measured resultant waveform acquired via 4D MRI scan with VENC=250 cm/s while the VMAX was measured at 158 cm/s. [8]

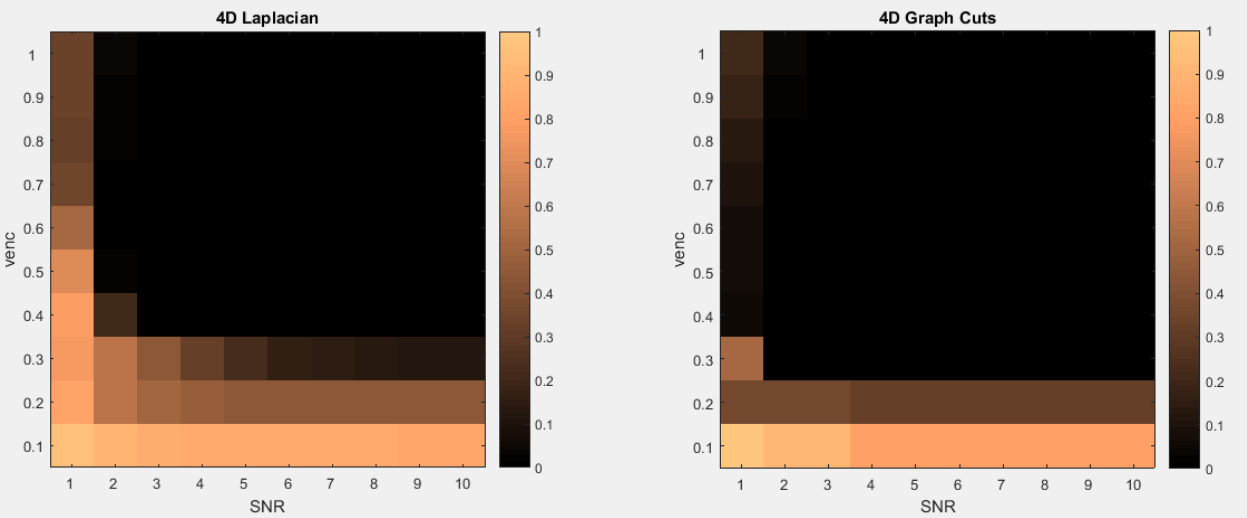


Figure 4 – Graphic of unwrapped voxel percentages for various SNR and VENCs. SNRs range from 1 to 10 while VENCs range from 0 to 100% of the top velocity. The color of each pixel corresponds to the percentage of voxels that have not been unwrapped successfully. When the color is black, it means all the voxels have been successfully unwrapped. The top image represents the success of the Laplacian algorithm while the bottom image represents the success of the graph cuts algorithm.

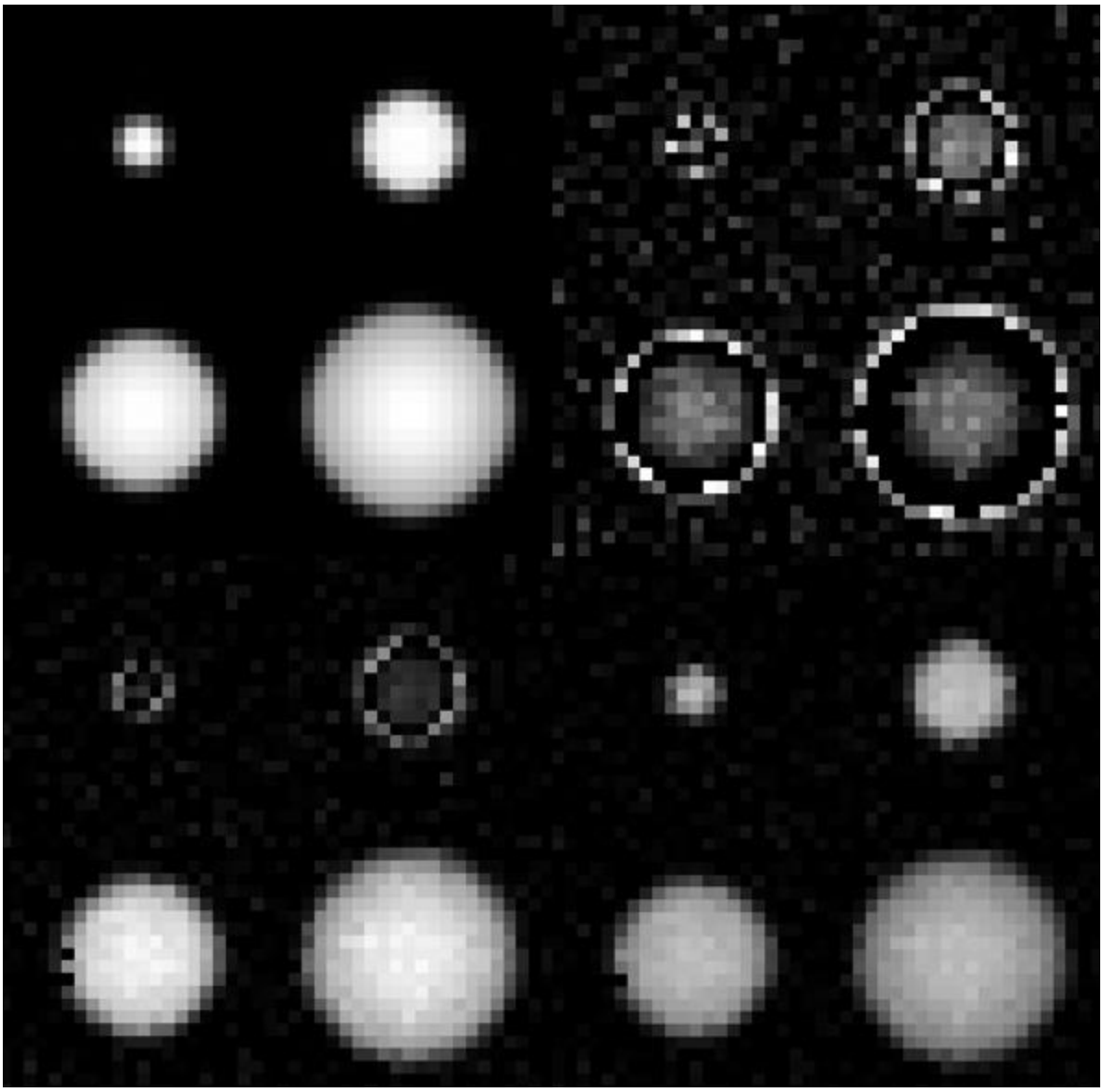


Figure 5 – Synthetic Images with SNR of 8 and VENC of 0.4. Images are size 44x44x40x24. The cross-section looks at slice 7 in both the time dimension and the z dimension. Top Left: True reference image. Top right: Wrapped Image with noise added. Bottom Left: Unwrapped with Laplacian Algorithm. Bottom Right: Unwrapped with PUMA Graph Cuts Algorithm

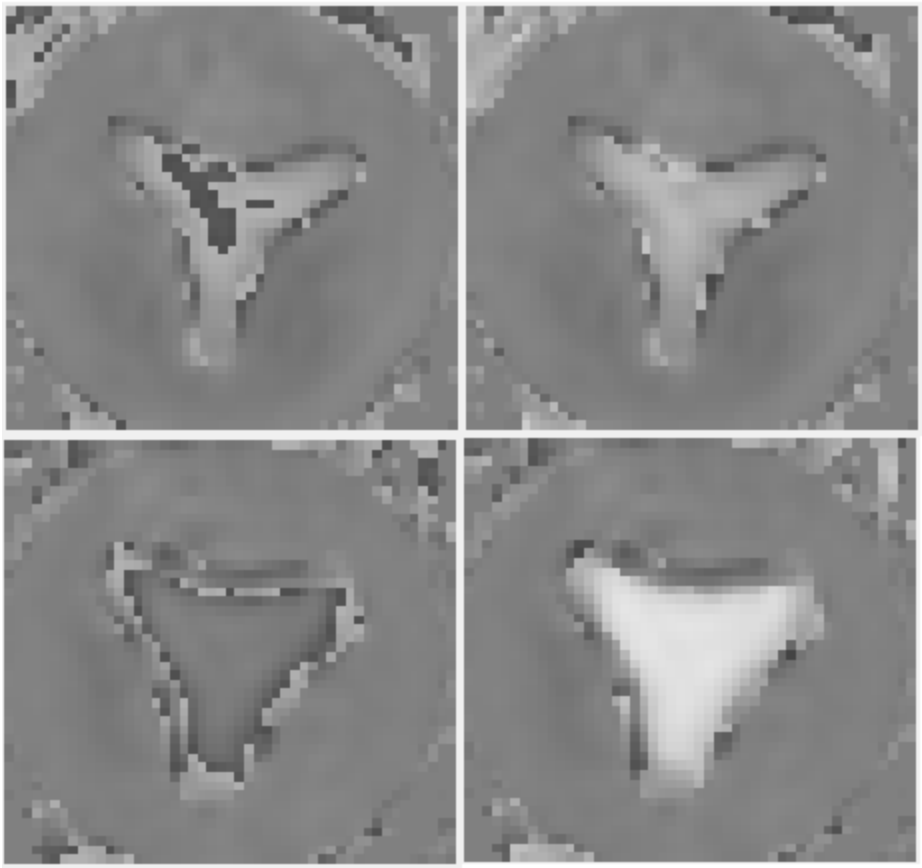


Figure 6 – 4D flow through aortic valve phantom. Peak flow rate = 200 ml/sec, Venc = 40 cm/sec. Top row: diastolic flow, Bottom row: systolic flow. Aliased z-component of the velocity data (left) and unwrapped data (right) after application of the Graph Cuts Unwrapping Algorithm.

Tables:

V_{enc} (cm/sec)	4D GC RRMSE	Laplacian RRMSE	Wrapped Data RRMSE
40	79.01	81.32	103.19
80	72.00	74.03	89.26
125	61.25	61.88	94.05

Table 1 – RRMSE Results for Phantom Data at different VENCs when compared with the ground truth data collected with VENC 250 cm/s. Wrapped Data RRMSE is the RRMSE for the wrapped data before any unwrapping algorithm is applied.

References

[1] J. Bioucas-Dias and G. Valadão, "Phase Unwrapping via Graph Cuts," *IEEE TRANSACTIONS ON IMAGE PROCESSING,,* vol. 16, pp. 698-709, 2007.

[2] M. Loecher, E. Schrauben, K. Johnson and O. Wieben, "Phase Unwrapping in 4D MR Flow With a 4D Single-Step Laplacian Algorithm," *Journal of Magnetic Resonance Imaging,* vol. 43, pp. 833-842, 2016.

[3] M. Loecher, K. Johnson, B. Landgraf and O. Wieben, "4D Gradient Based Phase Unwrapping for PC-MR Flow Data," *International Society for Magnetic Resonance in Medicine,* p. 3284, 2011.

[4] K. Itoh, "Analysis of the Phase Unwrapping Problem," *Applied Optics,* 1982.

[5] G. R. Valadão, "2D Phase Unwrapping via Graph Cuts," 2006.

[6] D. Ghiglia and M. Pritt, Two-Dimensional Phase Unwrapping Theory, Algorithms, and Software, John Wiley & Sons, Inc., 1998 .

[7] M. Kadbi, M. Negahdar, J. Cha, M. Traughber, P. Martin, M. Stoddard and A. Amini, "4D UTE Flow: A Phase-Contrast MRI Technique for Assessment and Visualization of Stenotic Flows," *Magnetic Resonance in Medicine,* vol. 73, no. 3, 2015.

[8] M. Negahdar, M. Kadbi, M. Kendrick, M. Stoddard and A. Amini, "4D Spiral Imaging of Flows in Stenotic Phantoms and Subjects with Aortic Stenosis," *Magnetic Resonance in Medicine,* vol. 75, no. 3, pp. 1018-1029, 2016.

[9] J. Dong, F. Chen, D. Zhou, T. Liu, Y. Zhaofei and Y. Wang, "Phase Unwrapping with Graph Cuts Optimization and Dual Decomposition Acceleration for 3D High-Resolution MRI Data," *Magnetic Resonance in Medicine,* vol. 77, no. 3, pp. 1353-1358, 2017.

[10] D. Ghiglia and L. Romero, "Minimum L norm two-dimensional phase unwrapping," *Journal of the Optical Society of America,* vol. 13, no. 10, pp. 1999-2013, 1996.

[11] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 26, no. 9, pp. 1124 - 1137, 2004.

[12] M. Loecher, K. Johnson, B. Landgraf and O. Wieben, "4D Gradient Based Phase Unwrapping for PC-MR Flow Data (abstract)," *Proc. Intl. Soc. Mag. Reson. Med.,* p. 3284, 2011.

[13] A. Henn, S. Callahan, M. Kendrick and A. Amini, "An Aortic Arch Phantom for Optimization of 4D Flow Imaging Protocols," 2017.