EFFICIENT INTERPOLATIONS FOR FUSING SHORT-AXIS AND LONG-AXIS MR IMAGES TO ENHANCE 3-D CARDIAC RECONSTRUCTION

Don-Lin Yang and Ming-Hui Liu
Institute of Information Engineering,
Feng Chia University, Taichung, Taiwan
dlyang@fcu.edu.tw

Chia-Hsien Wen and Jyh-Wen Chai
Taichung Veterans General Hospital,
Taichung, Taiwan
chwen@vghtc.gov.tw

ABSTRACT

A new method is proposed for fusing short-axis and long-axis MR images using efficient interpolations. It is suitable for the enhancement of 3-D cardiac reconstruction. Short-axis cardiac images are similar to regular MR images, which provide information about the cross-section of a heart, and long-axis cardiac images contribute supplementary information in a parallel direction to the long-axis. Our method can produce higher quality interpolated images compared to other methods. It includes five steps: arranging long-axis images in a circular form, finding a cubic spline curve, circular interpolation, fast parametric shape-based interpolation of short-axis images, and fusion of short-axis and long-axis images. From our experiments, the processing time improvement is fifty-three times faster than the gray-level shape-based method with at least 20% less mean square errors. Compared to the linear interpolation, our method has improved a minimal of 34% in mean square errors. This approach can help doctors make their diagnoses faster, easier, and more precise.

***Index Terms*— Cardiac MR Images, Left Ventricular, Interpolation, Short-Axis, and Long-Axis.**

1. INTRODUCTION

There are three common techniques used for medical image acquisition: Computed Tomography (CT), Magnetic Resonance Imaging (MRI), and Ultrasound Imaging (UI). Medical imaging can display human internal organs. They provide useful information for doctors to diagnose patients' condition. Different modalities are suitable for different types of organ tissues or body parts. In general, CT is used for hard objects such as bones, while MRI is suitable for soft tissue organs such as the liver. Computed tomography is also called Computer Aided Tomography (CAT). CT scanners are widely used in hospitals. It emits X-ray onto objects to generate a state change for acquiring data, which is then transferred to a host computer. The host computer converts the data to image slices and displays them on the monitor. For 3-D medical image applications, CT imaging can display the internal organs of a human body. Especially, CT works well with hard tissue objects. For soft tissue, as we will explain in the next paragraph, MRI is more appropriate. However, unlike motionless organs, a beating heart is hard to visualize in digital images. Although UI can display an embryo in real-time, the image quality is very poor.

For soft tissue in the human body such as blood vessels or the heart, CT images contain noises that make them unsuitable in real applications. For cardiac reconstruction, echocardiographic imaging produces noises that are more disruptive than those of CT images. Ultrasound cannot pass through the air to efficiently inspect the lungs either. In early years, the MRI sampling technique could only generate blurred images of a heart because it was too slow. Recently, the sampling rate of MRI scanners has improved. In a very short time, 10 *ms* or less, a clear cardiac image can be acquired. These cardiac MR images can be used to reconstruct 3-D models.

MRI scanners usually work on a slice-by-slice basis. To obtain an image, the patient lies in a torus that generates a strong static magnetic field, usually 0.5 to 4 tesla. The field strength is approximately eight orders magnitude stronger than the Earth's magnetic field. Generated by the torus, this field aligns all individual magnetic fields at an atomic level. This is called static equilibrium. In the human heart, the left ventricle pathological change is the focus of research. Since the left ventricle pumps blood to every part of the body, its ejection fraction (EF) is very important information [7] and has great prognostic value to cardiologists. The ejection fraction is defined as

$$EF = 1 - \frac{\text{end systolic volume}}{\text{end diastolic volume}} \qquad (1)$$

Here the end systole (ES) and the end diastole (ED) have reached their fully contracted and fully expanded cardiac phases, respectively. To measure the volume of the left ventricle more accurately, cardiac imaging has short-axis [1, 4] and long-axis [2, 3] images. Short-axis images can provide cross-section information of a heart. If the distance between two consecutive short-axis images lengthens, the interpolation quality becomes poorer. Long-axis images can reduce this problem by providing supplemental data in the direction parallel to the long-axis.

2. BACKGROUND

2.1 Fusion methods

There are many methods that can fuse the short-axis and long-axis images. Kuwahara and Eiho [5] combine LV boundary contours obtained at three orthogonal cross-sections of the heart and manually trace ventricular boundaries with two transverse, two coronal, and two sagittal cross-sections. Goshtasby [8] fuses short-axis and long-axis cardiac MR images by mapping the image slices back into the MR viewing window and estimating the image entries that are not mapped. It is approximated using a weighted averaging scheme with weights inversely proportional to the distances of voxels with known intensities to the voxel whose intensity is being estimated. This approximation uses a smoothness parameter that can be varied to fuse images with different inter-slice distances. Many researchers use the ellipsoid model [6] to measure and reconstruct the left ventricle. Using the ellipsoid model to reconstruct a whole heart is still a problem because of the complexity of the heart.

2.2 Interpolation methods

Interpolation process generates supplemental data for 3-D reconstruction. Basically, interpolation methods can be divided into two categories. One is the deterministic interpolation technique that assumes a constant variability between the sample points, such as linear interpolation. Linear interpolation is the most popular technique because its processing time is very short. However, it can cause the staircase artifact on the boundary of object images. Another method is called the statistical interpolation, such as shape-based interpolation, which can minimize the estimation error. To meet the demand for higher quality, most researchers focus on the statistical interpolation of recent years. For example, the binary shape-based interpolation approach was proposed in 1992. The prediction accuracy of interpolation has improved, but it is only suitable for binary images. Gray-level shape-based interpolation extends its application to gray-level images. However, the required processing time is much longer.

A. Linear interpolation

Linear interpolation [10, 13] is very simple to implement. The greatest advantage of linear interpolation is that the processing time is short.

It can transform a sequence of medical image slices into an isotropic volume data set. In the interpolation process, two source images are needed. One image is called the reference image I_R, and the other is called the target image I_T . The equation to compute the gray-level of an interpolated image is as follows:

$$I_m(x, y) = \frac{M - m + 1}{M + 1} I_R(x, y) + \frac{m}{M + 1} I_T(x, y), m = 1,2,3,...,M \qquad (2)$$

In Equation (2), $I_m(x, y)$ is the gray-level of the *m*th interpolated image, and M is the total number of interpolated images. $I_R(x,y)$ is the gray-level of the reference image, and $I_T(x, y)$ is the gray-level of the target image. This simple method only requires a single step. A two-step incremental linear interpolation algorithm [11] can produce more accurate images with less time. However, both methods can cause the staircase artifact on the object boundary. Shape-based interpolations [12, 14] can minimize this problem.

B. Binary shape-based interpolation

G. T. Herman [12] proposed binary shape-based interpolation in 1992. In conventional terms, a pixel inside (outside) the object will be assigned a positive (negative) distance value. Initially, one assigns each pixel that is inside (outside) the object along the border a small positive (negative) value indicating the distance from the center of the pixel block to the borderline. All remaining pixels that are inside (outside) the object, but not on the border, are assigned a large positive (negative) value.

For efficiency, the representation of distance values can be implemented in such a way that the actual Euclidean distance is approximated by integers. Two passes are then made through the data to assign the distance values via a mask of values that are minimized to indicate the minimum distance to the border of the object. This method can be used for binary images only. Since most medical images are gray-level images, extensions must be made to use this approach.

C. Gray-level shape-based interpolation

Gray-level shape-based interpolation [14] enhances the approach of binary shape-based interpolation to accommodate gray-level images. It consists of five steps: lifting, distance transform, interpolation, conversion to binary, and collapsing. The process of lifting is to convert an n-dimension (n-D) gray-level scene into an (n+1)-D binary scene. A distance transform converts the (n+1)-D binary scene created in the previous step to (n+1)-D gray-level scene, such that every spatial element contains the minimal Euclidean distance from the spatial element to the boundary of the object. The interpolation step can be a linear interpolation. In the conversion to binary step, it simply converts the (n+1)-D gray-level scene to an (n+1)-D binary scene. The final collapsing step is the complement of lifting. It converts the (n+1)-D binary scene into an n-D gray-level scene. The conversion of (n+1)-D binary scene to (n+1)-D gray-level scene in the distance transform step requires the most processing time.

3. OUR PROPOSED METHOD

In this section, we describe our fusion method for cardiac MR images. The flow chart of the fusion method is shown in Fig. 1. Since our research focuses on interpolation, we assume that the source short-axis and long-axis images have completed the processes of segmentation and registration. One of the twelve short-axis images we used in this study is shown in Fig. 2(a), and one of the nine long-axis images is shown in Fig. 2(b). After the segmentation and registration process, the nine long-axis images are shown in Fig. 3(a). Fig. 3(b) depicts how to acquire the nine slices of long-axis images by rotating it around the center z-axis and taking a slice at every twenty degrees.

Our method has five steps: (1) arranging long-axis images in a circular form, (2) finding a cubic spline curve, (3) circular interpolation, (4) fast parametric shape-based interpolation of short-axis images, and (5) fusion of the short-axis and long-axis images. Steps 1 to 3 are used to map long-axis images to short-axis images. Step 4 generates the interpolated image of short-axis. Step 5 combines the mapped images and short-axis images to produce interpolated images.

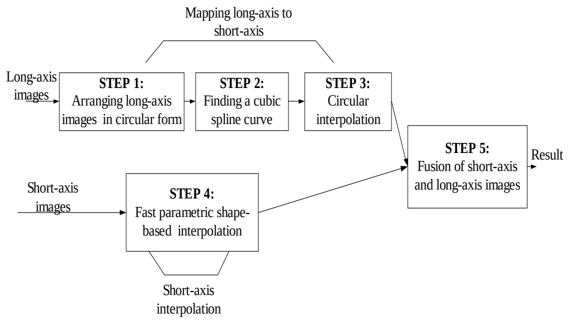
Mapping long-axis to short-axis
Long-axis images
STEP 1: Arranging long-axis images in circular form
STEP 2: Finding a cubic spline curve
STEP 3: Circular interpolation
STEP 5: Fusion of short-axis and long-axis images
Result
Short-axis images
STEP 4: Fast parametric shape-based interpolation
Short-axis interpolation

Fig. 1. The flow chart of our fusion method.

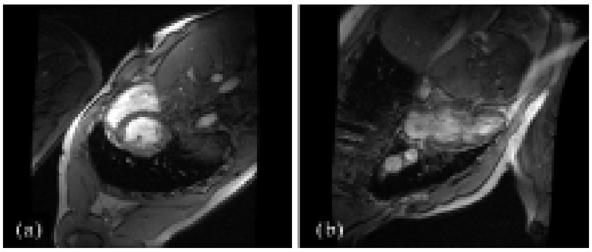

Fig. 2. Two MR images of a heart. (a) A short-axis image. (b) A long-axis image.

Step 1—Arranging long-axis images in a circular form

The arrangement of long-axis images is shown in Fig. 3(b). This way we can reconstruct the actual sampling process of an MRI scanner. The first image of Fig. 3(a) is placed at the number 1 position in Fig. 3(b), and the second image of Fig. 3(a) is placed at the number 2 position in Fig. 3(b), and so on. They are equally distanced by twenty degrees. These nine slices of 2-D long-axis images are positioned to form a partial 3-D image. The *x*-axis and *z*-axis can be viewed as the original long-axis plane, and the *x*-axis and *y*-axis can also be treated as the 2-D coordinates of the short-axis image. Fig. 4(a) shows a cross-section diagram of Fig. 3(b) along the *z*-axis. Fig. 4(b) is an actual cross-section image showing gray-level information.

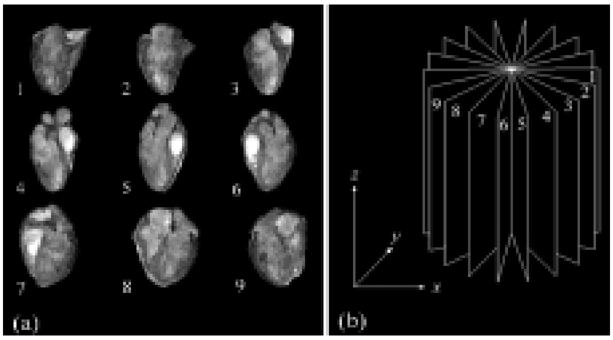

Fig. 3. (a) The nine long-axis images after segmentation and registration. (b) Arranging the nine images circularly to form a partial 3-D image.

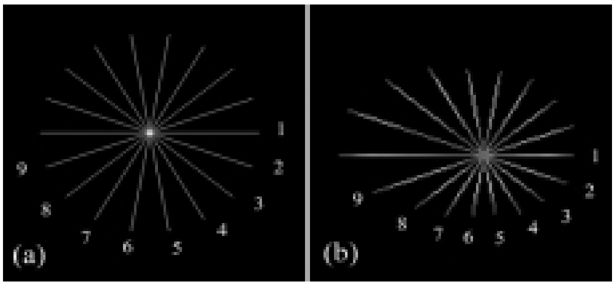

Fig. 4. (a) A cross-section of Fig. 3(b) along the *z*-axis. (b) An actual cross-section image showing gray-level information.

Step 2—Finding a cubic spline curve

A spline function consists of polynomial pieces joined together under certain conditions. To get a smooth curve, a natural cubic spline is the most popular curve function that can be constructed to match the data points. First, starting from the initial *z* value, we extract the two end points from each long-axis image at the cross-section to form the feature points shown in Fig. 5(a). They are arranged in the same order as in Fig. 4(a). Secondly, we map this 2-D information into 1-D as shown in Fig. 5(b), where the *y* value is the distance from the center point to the pixel position. Next, we use the natural cubic spline procedure to generate the interpolated point of the curve (Fig. 5(c)). Lastly, we re-map the 1-D information back into 2-D. The final result is shown in Fig. 5(d).

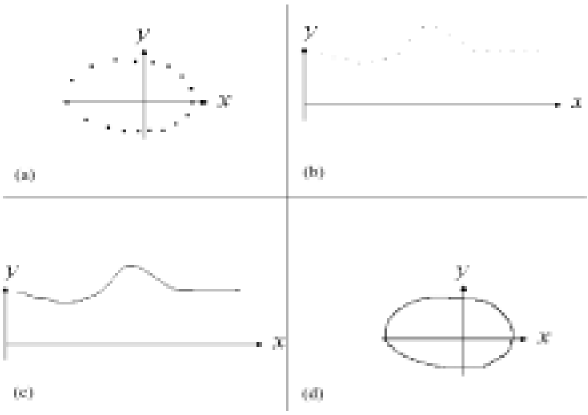

Fig. 5. (a) Extracted feature points from long-axis images. (b) Converted 2-D image data as 1-D data. (c) Generated missing points between the feature points. (d) Converted 1-D data back as 2-D image.

Step 3—Circular interpolation

After finding the cubic spline curve, we can generate the points (pixels) in the fan-shaped area inside the curve by using circular interpolation as shown in Fig. 6. ϕ is the angle between the interpolated line of points and the corresponding reference image line. M_ϕ is the angle between the reference and target image lines. The gray-level of the interpolated image can be calculated as follows:

$$\begin{cases} \begin{cases} I_{CR}(x,y) = I_{CT}(x,y) & , \mathit{if}\ \max[L_R(x,y)] > \max[L_T(x,y)] \\ I_{CT}(x,y) = I_{CR}(x,y) & , \mathit{if}\ \max[L_T(x,y)] > \max[L_R(x,y)] \end{cases} \\ I_C(x,y) = \frac{\varphi}{M_\varphi} I_{CR}(x,y) + \frac{M_\varphi - \varphi}{M_\varphi} I_{CT}(x,y) \end{cases} \tag{3}$$

where $\max[L_R(x,y)]$ and $\max[L_T(x,y)]$ are maximum lengths of the reference and target images' lines respectively. $I_C(x,y)$ is the gray-level of the interpolated point. $I_{CR}(x,y)$ is the gray-level of the reference point, and $I_{CT}(x,y)$ is the corresponding gray-level of the target point. $I_C(x,y)$, $I_{CR}(x,y)$, and $I_{CT}(x,y)$ are equidistant to the center point. Fig. 7 shows example images before and after circular interpolation.

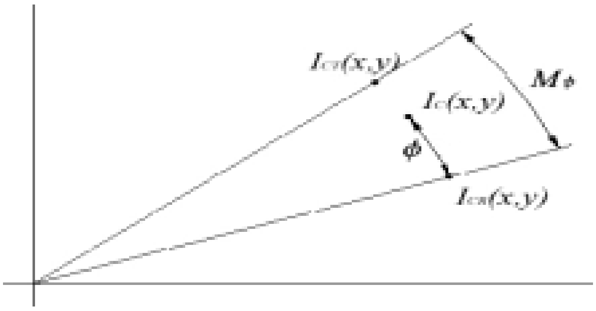

Fig. 6. Circular interpolation.

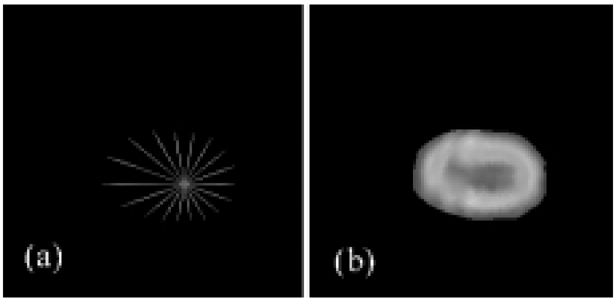

Fig. 7. (a) A cross-section image before circular interpolation. (b) Result of the circular interpolation process.

Step 4—Fast parametric shape-based interpolation of short-axis images

We use the parametric shape-based method [20] for the interpolation of short-axis images. Although linear interpolation requires less processing time, it suffers a deficiency of image quality known as the staircase artifact on the image boundary. For convenience, the terms "reference image" and "target image" are used for the two consecutive slices used to generate the interpolated slices in between. The staircase problem occurs when a pixel of the reference image has a zero gray-level value and the corresponding value of the target image is non-zero, or vice versa. Although shape-based interpolation approaches can resolve this problem, they require a much longer processing time.

First, we examine where the discrepancies between the actual and interpolated images occur. By working on the area that can be improved the most, we can increase the accuracy of interpolation in an effective manner. We define two kinds of image data sets: *Type A* and *Type B*. *Type A* denotes any image data set satisfying the condition that both of its reference and target images have corresponding pixel values that are *non-zero*. For a *Type B* image data set, either the reference image or the target image has a corresponding pixel value that is *non-zero*, but not both of them.

From an early research result [16], we found that the *Type*

B image data has more interpolation errors than the *Type A* image data does. Therefore, we should try to improve the quality of interpolation for *Type B* data sets in order to acquire the best overall result. Since linear interpolation has a rather good performance for *Type A* data sets, we use it in our method. Our main focus will be on improving the interpolation quality of *Type B* data sets. Furthermore, we found that the changes of gray-level values on the object boundaries are mostly smooth and gradual. A sudden drop of gray-level value indicates a possible cause of the staircase effect.

The following notations are used. $S\_R(i,j)_{mxn}$ and $S\_T(i,j)_{mxn}$ arrays store the pixel values of the reference and target images respectively. Let (i,j) denote the array element at the i-th row and j-th column and $f(i,j)$ denote the gray-level of the (i,j) array element. Let m denote the maximum dimension of a row and n denote the maximum dimension of a column.

Step 4.1—Extending boundary gray-level

By extending the boundary gray-level to outside pixels and by coupling it with the distance information from Steps 4.2 and 4.3 to generate a gradually diluted effect, we can improve the interpolation quality of *Type B* images. For illustration purposes, we assume that the source images have been pre-processed to identify the object boundary. Retaining the gray-level values of inside pixels, we extend the gray-level of the object boundary outward to all outside pixels. There are three phases in Step 4.1:

Phase 1: Assign each of the outside pixels along the object boundary to the average gray-level value of its non-zero neighboring pixels.

Phase 2: Assign extended pixel values row by row from top to bottom and left to right within the rows. For pixel (i,j) where $2 \leqq i \leqq m$ and $2 \leqq j \leqq n$, if both pixels $(i-1,j)$ and $(i,j-1)$ have gray-level values, assign $f(i,j)$ value to the average of $f(i-1,j)$ and $f(i,j-1)$. If only one of pixel $(i-1,j)$ and pixel $(i,j-1)$ exists, use that *non-zero* value as the pixel's gray-level value. Otherwise, the gray-level value is set to *zero*.

Phase 3: Assign remaining pixel values row by row from bottom to top and right to left within the rows. For pixel (i,j) where $1 \leqq i \leqq m-1$ and $1 \leqq j \leqq n-1$, if both pixels $(i+1,j)$ and $(i,j+1)$ have gray-level values, assign $f(i,j)$ with their average gray-level as in Phase 2. If only one of the pixels $(i+1,j)$ and $(i,j+1)$ exists, use that *non-zero* value as the pixel's gray-level value. Otherwise, the gray-level value is set to *zero*. $R(i,j)_{mxn}$ and $T(i,j)_{mxn}$ store the extended reference and target image data respectively.

Step 4.2—Distance transform

A distance transform converts source image data to the shortest distance information from each pixel to the object boundary. This step produces a mask to be used in Step 4.4 to generate a gradually diluted effect such that the farther away from the boundary, the smaller the gray-level an interpolated pixel gets. The mask is also used in Step 4.5 to perform linear interpolation. This process is accomplished by using a similar method from [12]. For simplicity, positive values are assigned to pixels inside the object boundary and negative values are for pixels that are outside. First, we convert original images to binary images using a specified threshold. To initiate the distance calculation, replace every *zero* pixel with a large negative number (e.g., -99) and replace every *one* pixel with a large positive number (e.g., 99). There are two exceptions: if a *zero* pixel shares an edge with a *one* pixel, replace it with –5; if a *one* pixel shares an edge with a *zero* pixel, replace it with +5. To speed up the shortest distance calculation from every pixel to the object boundary, two three by three templates are used. These two templates have pre-set distance information that is tunable. The first chamfering updates the pixel row by row from top to bottom with a left to right ordering within the rows. We apply the second chamfering to the result of the first one and update the pixel again row by row from the bottom to the top with a right to left ordering within the rows. The result of distance transform for reference and target images are denoted by $DT\_R(i,j)_{mxn}$ and $DT\_T(i,j)_{mxn}$ respectively.

Step 4.3—Setting decreasing parameter

For fine-tuning the dilution effect of the decreasing process in Step 4.4 to further reduce the staircase artifact, we introduce a decreasing parameter d. Using this parameter, we can also adjust the dilution rate to accommodate various images of organ shapes and tissues. Therefore,

$$SDT\_R(i,j) = DT\_R(i,j) * d$$
$$SDT\_T(i,j)=DT\_T(i,j)*d \qquad (4)$$

$1 \leqq i \leqq m$ and $1 \leqq j \leqq n$, where d is the decreasing parameter.

$SDT\_R(i,j)_{mxn}$ and $SDT\_T(i,j)_{mxn}$ contain the adjusted distance transform data for reference and target images respectively. In general, the range of the d parameter is from 1 to 15, depending on the source images.

Step 4.4—Decreasing process invocation

Step 4 uses the mask of adjusted distance transform data (Step 4.3) over the extended reference and target images (Step 4.1) to generate gradually diluted images. If a pixel is farther away from the boundary, its gray-level is lower. Otherwise, its gray-level is higher. For an interpolated pixel (i,j), if its reference image has pixel value and its target image has no pixel value, $f(i,j)=R(i,j)+SDT\_T(i,j)$. On the other hand, if its reference image has no pixel value and its target image has a pixel value, then $f(i,j)=T(i,j)+SDT\_R(i,j)$. To bind the gray-level within a specified range and to avoid a negative gray-level, a threshold T is used to set *f(i,j)* to zero if it is below T. We use $DM\_R(i,j)_{mxn}$ and $DM\_T(i,j)_{mxn}$ to store the result of Step 4.4.

Step 4.5—Linear interpolation

A conventional linear gray-level interpolation is used in this step. For an interpolated pixel *(i,j)*, if the sum of *DT_R(i,j)* (distance value on the reference image) and *DT_T(i,j)* (distance value on the target image) is greater than *zero*, we assign it with the average gray-level $f(i,j)=\frac{DM\_R(i,j)+DM\_T(i,j)}{2}$. If the sum of *DT_R(i,j)* and *DT_T(i,j)* is equal to *zero*, then $f(i,j)=\frac{DM\_R(i,j)+DM\_D(i,j)}{4}$, otherwise *f(i,j)*=0.

From our study, this procedure can produce results as good as, or even better than, the average case of gray-level shape-based interpolation. Our interpolated result shows that the object boundary is very smooth and no staircase artifact is found. On the other hand, the result of linear interpolation exhibits severe staircase effect.

Step 5—Fusion of short-axis and long-axis images

In this final step, we employ a simple weighting method to combine the interpolated short-axis and long-axis images. Fig. 8 shows the relation between the unsigned gray-level difference per pixel and the distance from the center of axial rotation to the pixel's position. Because the weight of fusing the long-axis images should be increased for the pixels near the center of the cross-section, we use the formula (5) to calculate the weight. This formula came from the result of our experience. In our experiments, we use $W_{start}=0.7$, $W_{stop}=0.3$.

$$\begin{cases} W_L = W_{start} - \frac{Udiff_P}{Max[Udiff_P]} * (W_{start} - W_{stop}) & (5) \\ W_S = 1 - W_L & (6) \\ I = W_S * I_S + W_L * I_L \quad , if\, I_S > 0 & (7) \end{cases}$$

where W_L is the weight of a pixel of the long-axis image, W_S is the weight of a pixel of the short-axis image, W_{start} is the heaviest weight for the center point of axial rotation, $Udiff_P$ is the unsigned gray-level difference per pixel, $Max[Udiff_P]$ is the maximum value of $Udiff_P$, W_{stop} is the lowest weight for the maximum value of $Udiff_P$, I is the gray-level of a pixel, I_S is the gray-level of a short-axis pixel, and I_L is the gray-level of a long-axis pixel.

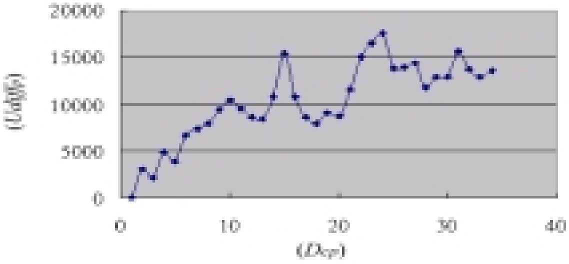

Fig. 8. The relation between $Udiff_P$ and D_{CP}. D_{CP} is the distance from the center point of axial rotation to the pixel's position.

4. EXPERIMENTAL RESULT AND DISCUSSION

In our experiments, we use forty-two slices of 256×256 source images of the human head and twelve slices of 256 ×256 short-axis source images of the human heart. Long-axis images are acquired by partitioning the 3-D image parallel to the direction of *z*-axis as shown in Fig. 3(b). There are nine long-axis images with an angular separation of twenty degrees.

All experiments are performed on an Intel Pentium II 233 machine. Between two adjacent images, we generate one intermediate slice. We use quantitative as well as qualitative measurements to evaluate the interpolation results [17, 18]. The notations are defined as follows: *v* denotes the gray-level of an image pixel, *V* denotes a 2-D set of image pixels, *N* is the total number of 2-D image pixels, *f(v)* denotes the gray-level of the interpolated pixel *v*, and $f_a(v)$ refers to the gray-level of the standard (actual) image pixel *v*. We evaluate interpolated images by using the following measurements: the mean-square difference of gray-level (*Msd*), the number of sites of disagreement (*Nsd*), the largest difference of gray-level (*Ldiff*), the unsigned difference of gray-level (*Udiff*), and processing time (*time*).

1. Mean square difference of gray-level, *Msd(v)*

$$Msd(v) = \frac{1}{N} \sum_{v \in V} [f(v) - f_a(v)]^2 \qquad (8)$$

2. Number of sites of disagreement, *Nsd(v)*

$$Nsd(v) = \sum_{v \in V} \tau(|f(v) - f_a(v)|) \qquad (9)$$

$$\tau(x) = \begin{cases} 1, & \text{if } x > \theta, \\ 0, & \text{otherwise.} \end{cases}$$

θ denotes the tolerant scope of gray-level, and our default value is twenty.

3. Large difference of gray-level, *Ldiff(v)*

$$Ldiff(v) = \max_{v \in V} \{|f(v) - f_a(v)| \qquad (10)$$

4. Total unsigned difference of gray-level, *Udiff(v)*

$$Udiff(v) = \sum_{v \in V} |f(v) - f_a(v)| \qquad (11)$$

Table 1 – Average quality and processing time of 40 interpolated human head images.

	Linear	Gray-Level Shape-Based	Our Method
Msd(v)	187.9	69.4	55.6
Nsd(v)	1663.4	189	187
Ldiff(v)	90.55	74.8	82.5
Udiff(v)	312025	138216	134995
Time (s)	0.25	27.8	0.52

For a qualitative evaluation, human head and heart images are used in our experiments. Forty-two original slices of the human head and twelve original slices of the human heart are selected as short-axis images. Long-axis images are acquired by partitioning the 3-D image parallel to the direction of *z*-axis. There are two sets of nine long-axis images with an angular separation of twenty degrees.

Table 1 lists the average quality measurements and processing time of forty interpolated human head images for the three methods including the linear, the gray-level shape-based, and our fusion method. Some sample resultant images are shown in Fig. 9. In Fig. 9, images (a), (b), and (c) are three consecutive MR slices of the actual human head. Images (d), (e), and (f) are generated by using the linear, the gray-level shape-based, and our fusion method respectively.

Our interpolated result is much more accurate than the other two methods when they are compared to the standard image. Gray-level shape-based interpolation performs better than linear interpolation on the contour, but the gray-level values of the pixels inside the object are different from that of the standard image. The result of linear interpolation is the poorest.

Quantitatively, our fusion method is also better than the other two methods as shown in Table 1. In Table 1, our method has the smallest *Msd*, *Nsd*, and *Udiff* while gray-level shape-based interpolation performs slightly better than ours in *Ldiff*. In addition, our *Msd* value is 20% less than that of shape-based interpolation and 70% better than linear interpolation. When comparing processing speed, our method is only 0.27 seconds slower than linear interpolation, but it is fifty-three times faster than shape-based interpolation. The quality and speed of our fusion method can definitely improve the 3-D reconstruction of a heart. Table 2 lists the average quality measurements of ten interpolated human heart images for the three aforementioned methods. Some images of the sample results are shown in Fig. 10. In Fig. 10, images (a), (b), and (c) are three consecutive MR slices of actual human heart. Images (d), (e), and (f) are generated by using the linear, the gray-level shape-based, and our fusion method respectively. The *Msd* value of our method is 27% and 34% less than the linear and gray-level shape-based methods respectively.

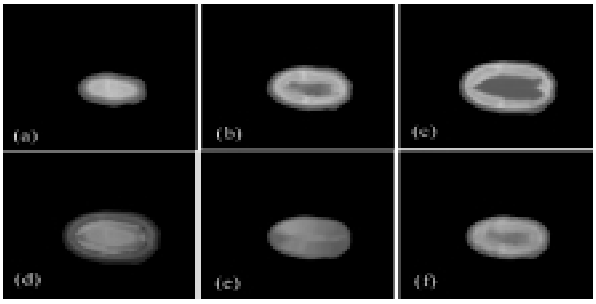

Fig. 9. Experimental results of interpolations (human head).

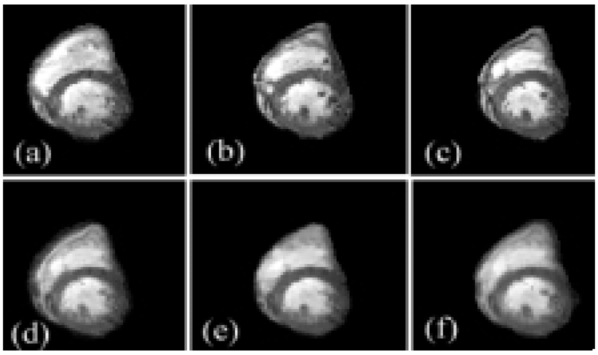

Fig. 10. Experimental results of interpolations (human heart).

Fig. 11 shows the 3-D reconstruction of interpolated images. It uses the ray-tracing technique to render the 3-D models. The reconstructed result of our method (Fig. 11(a)) shows very little staircase artifact, and the surface of the heart is very smooth. On the other hand, the linear interpolated result (Fig. 11(b)) appears with severe staircase artifact, and it loses the bottom portion of the heart image.

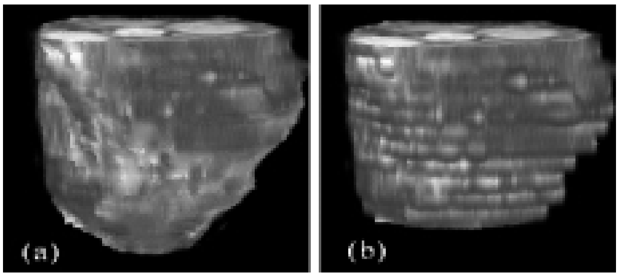

Fig. 11. The 3-D reconstruction of interpolated images. (a) Our method (b) Linear interpolation.

Table 2 – Summary of the average quality of 10 interpolated cardiac MR images.

	Linear	Gray-Level Shape-Based	Our Method
Msd(v)	299.7	270.7	198.3
Nsd(v)	1623.7	1388.4	1290.5
Ldiff(v)	150.8	138.4	141.4
Udiff(v)	92825.1	81474.5	72803.9

5. CONCLUSION

In this paper, we propose a general method to fuse short-axis and long-axis slices of cardiac MR images. It has five steps: (1) arranging long-axis images in a circular form, (2) finding a cubic spline curve, (3) circular interpolation, (4) fast parametric shape-based interpolation of short-axis images, and (5) fusion of the short-axis and long-axis images. Steps 1 to 3 are used to process long-axis images, and Step 4 is used to interpolate short-axis images. Step 5 combines the short-axis and long-axis images to produce the interpolated images. The experimental result shows a processing time speedup of fifty-three times faster than gray-level shape-based interpolation with at least 20% less mean square errors. Compared to linear interpolation, our method has a minimal of 34% improvement in mean square errors.

The circular interpolation has room for improvement since we know that the weight of fusing the long-axis images should increase for the pixels near the center of the cross-section. A better model can be developed to calculate their weights. To speedup processing time, we will investigate parallel interpolation algorithms [19].

REFERENCE

[1] W. J. Stratemeier, R. Thompson, T. J. Brady, S. W. Miller, S. Saini, G. L. Wismer, R. D. Okada, R. E. Dinsmore, “Ejection fraction determination by MR imaging: Comparison with left ventricular angiography,” *Radiology*, vol. 158, no. 3, 1986, pp. 775-777.

[2] R. R. Edelman, R. Thompson. H. Kantor, T. J. Brady, M. Leavitt, and R. Dinsmore, “Cardiac fuction: Evaluation with fast-echo MR imaging,” *Radiology*, vol. 162, no. 3, 1987, pp. 611-615.

[3] P. T. Buser, W. Aufermann, W. W. Holt, S. Wagner, B. Kircher, C. Wolfe, C. B. Higgins, “Noninvasive evaluation of global left ventricular function with use of cine nuclear magnetic resonance,” *JACC*, vol. 13, no. 6, 1989, pp. 1294-1300.

[4] G. B. Cranney, C. S. Lotan, L. Dean, W. Baxley, A. Bouchard, and G. M. Pohost, “Left ventricular measurement using cardiac axis nuclear magnetic resonance imaging, “ *Circulation*, vol. 82, no. 1, 1990, pp. 154-163.

[5] M. Kuwahara and S. Eiho, “3-D heart image reconstructed from MRI data,” *Computerized Medical Imaging and Graphics*, vol. 15, no. 4, 1991, pp. 241-246.

[6] T. S. Denney and J. L. Prince, “Reconstruction of 3-D Left Ventricular Motion from Planar Tagged Cardiac MR Images: An Estimation Theoretic Approach,” *IEEE Trans. on Med. Imag.*, vol. 14, no. 4, Dec. 1995.

[7] S. Jaggi, W. C. Karl, and A. S. Wilsky, “Estimation of Dynamically Evolving Ellipsoids with Applicatins to Medical Imaging,” *IEEE Trans. on Med. Imag.*, vol. 14, no. 2, June 1995.

[8] A. A. Goshtasby and D. A. Turner, “Fusion of Short-Axis and Long-Axis Cardiac MR Images,” *1996 Proceedings of MMBIA.*

[9] J. A. Parker, R. V. Kenyon, and D. E. Troxel, “Comparison of interpolating methods for image resampling,” *IEEE Trans. Med. Imag.*, vol. MI-2, no. 1, March 1983.

[10] A. Goshtasby, D. A. Turner, and L. V. Ackerman, "Matching of tomographic slice for interpolation," *IEEE Trans. on Med. Imag.*, vol. 11, no. 4, Dec. 1992.

[11] J. Rokne and Y. Rao, “Double-step incremental linear interpolation,” *ACM Trans on Graphics*, vol. 11, no. 2, pp. 183-192, April 1992.

[12] G. T. Herman, J. Zheng, and C. A. Bucholtz, “Shape-based interpolation,” *IEEE Comput. Graphics Applicat.*, vol. 12, no. 3, pp. 69-79, 1992.

[13] G. Barequet and M. Sharir, “Piecewise-linear interpolation between polygonal slices,” *10th Computational Geometry*, 1994.

[14] G. J. Grevera, and J. K. Udupa, “Shape-Based Interpolation of Multidimensional Grey-Level Images,” *IEEE Trans. on Med. Imag.*, vol. 15, no. 6, Dec. 1996.

[15] G. J. Grevera and J. K. Udupa, “A objective comparison of 3-D image interpolation methods,” *IEEE Trans. Med. Imag.*, vol. 17, no. 4, Aug. 1998.

[16] D. L. Yang, M. H. Liu, Y. C. Chung, C. H. Wen, “Hybrid Interpolation to Enhance 3-D Medical Image Reconstruction,” *Chinese Journal of Medical and Biological Engineering*, vol. 20, no. 1, March 2000.

[17] J. A. Parker, R. V. Kenyon, and D. E. Troxel, “Comparison of interpolating methods for image resampling,” *IEEE Trans. Med. Imag.*, vol. MI-2, no. 1, March 1983.

[18] G. J. Grevera and J. K. Udupa, “A objective comparison of 3-D image interpolation methods,” *IEEE Trans. Med. Imag.*, vol. 17, no. 4, Aug. 1998.

[19] H. Embrechts and D. Roose., "Parallel algorithms for the distance transformation," *Parallel Processing: CONPAR 92- VAPP V*, pp. 571-582, 1992.

[20] D. L. Yang, M. H. Liu, Y. C. Chung, C. H. Wen, “A fast parametric shape-based interpolation method of gray-level images,” *Proceeding of the Fourth Asian Conference on Computer Vision*, pp. 565-570, Jan. 2000.