# Patient-Specific Analysis of Left Ventricular Blood Flow

Timothy N. Jones and Dimitris N. Metaxas

VAST Lab, Department of Computer and Information Science
University of Pennsylvania, Philadelphia, PA 19104-6389, USA
tnj@graphics.cis.upenn.edu, dnm@central.cis.upenn.edu
http://www.cis.upenn.edu/~tnj/

**Abstract.** We use a computational fluid dynamics (CFD) solver to simulate the flow of blood through the left ventricle (LV). Boundary conditions for the solver are derived from actual heart wall motion as measured by MRI-SPAMM. This novel approach allows for the first time a patient-specific LV blood flow simulation using exact boundary conditions.

## 1 Introduction

The development of a patient specific LV blood flow simulator is an open and challenging research problem. There are imaging techniques such as phase velocity MR which can indirectly measure velocity, but the resulting data must be reconstructed and tends to be very noisy and requires significant post-processing to be of any use. A detailed description of these methods is beyond the scope of this paper. There has been a large body of work in the mechanical engineering literature over the past few decades related to the computational simulation of fluid dynamics (CFD). Some of the more approachable texts are those of Ferziger and Perić [3], Fletcher [4], Patankar [7], and Roache [11]. There have been many varieties of numerical methods developed for simulating different types of fluids (liquids and gases) with various physical properties (density and viscosity) undergoing varying types of flows (laminar, turbulent, supersonic) in different types of physical environments (those with external temperature and chemical influences). While the general problem of CFD is one of the more challenging in computational science, good results have been obtained using techniques designed for specific problems.

Several researchers have developed CFD techniques to simulate blood flow through the heart with varying degrees of realism. Peskin and McQueen [9][5] developed the immersed fiber method, in which the heart wall is modeled by a woven stand of fibers immersed in a viscous, incompressible fluid. The fibers are arranged in a shape which approximates that of the heart, and exert forces in the tangential direction. These forces are applied to the Navier-Stokes equations to induce motion in the fluid. The result is an effect similar to that caused by the contracting heart wall. While the results are convincing from a computational point of view, the fibers have no physical connection to the actual heart. Yoganathan et al. [15] later addressed the anatomical issue by constructing the fibers to represent the shape of the LV. Pelle et al. [8] used the Laplace equation

to simulate velocity in a cylindrical model of the LV, neglecting viscous effects and assuming irrotational flow. The Bernoulli equation computes pressure from velocity in a linked model. The velocity equations were augmented with terms to more closely approximate the local structure of the heart wall. Thomas and Weyman [14] used a Navier-Stokes solver to simulate ventricular filling in a simplified geometrical model. Dubini et al. [2] used a Navier-Stokes solver to simulate the installation of a by-pass device in a simplified model of the right ventricle. Taylor et al. [13] used the shape of a digitized canine LV in a Navier-Stokes simulation. The motion of the wall was described by having it moved toward the center of the aortic outlet.

While these contributions and others have allowed important quantitative analysis of simplified models of hearts, none of them have used realistic data from actual patients. The goal of our work has been to address this shortcoming. In this paper, we adapt a computational fluid dynamics (CFD) solver to develop for the first time a patient specific blood flow simulator for the left ventricle (LV). Boundary conditions for the solver are derived from actual heart wall motion as measured by MRI-SPAMM. Visualization of preliminary results is presented.

## 2 Boundary Data Extraction

The model of blood flow described in the next section uses as input boundary data extracted from the LV's endocardium motion. The volumetric analysis of the LV from MRI-SPAMM data was developed by Park et al.[6]. The method uses as input boundary and tagged data to fit a volumetric deformable model with parameter functions that can capture the motion of the LV. These parameters capture the contraction and twist of the LV which are essential for modeling correctly the blood flow within the LV. In this paper we use the 3D time-varying position of points on the LV endocardium that are sampled from the volumetric LV model. These data are used as boundary conditions in the blood flow simulation described next.

## 3 Modeling Blood Flow

In this section we describe the equations that govern the dynamics of our blood flow model.

We use an Eulerian approach for the description of the flow field, in which the variables (pressure and velocity) of the fluid are solved for across the domain of interest. The Lagrangian formulation, in which variables of the fluid are tracked at the position of an object (such as a blood molecule) as it moves through the domain, or mixed Eulerian-Lagrangian formulation, are useful in other situations such as solid body dynamics or mixed media interfaces and will not be used here.

The fundamental laws to be considered in the description of fluid flow are the conservation of mass (continuity) and momentum, repectively:

$$\frac{\mathrm{d}m}{\mathrm{d}t} = 0 \quad \text{and} \quad \frac{\mathrm{d}(m\mathbf{v})}{\mathrm{d}t} = \sum \mathbf{f}, \tag{1}$$

where $m$ is mass, $t$ is time, $\mathbf{v}$ is velocity, and $\mathbf{f}$ are forces.

Our Eulerian approach deals not with parcels of matter, but rather with parcels of space known as control volumes. We therefore need to convert the control mass-oriented conservation equations above into control volume-oriented forms. The amount of mass in a control mass CM can be defined as:

$$m = \int_{\Omega_{CM}} \rho \mathrm{d}\Omega, \tag{2}$$

where $\Omega_{CM}$ is the volume of the control mass and $\rho$ is the density of the matter. Inserting this definition into Eq. 1 and applying Green's theorem yields the control volume equation for mass conservation (for control volumes fixed in space):

$$\frac{\partial}{\partial t}\int_{\Omega_{CM}} \rho \mathrm{d}\Omega = \frac{\partial}{\partial t}\int_{\Omega_{CV}} \rho \mathrm{d}\Omega + \int_{S_{CV}} \rho \mathbf{v} \cdot \mathbf{n} \mathrm{d}S = 0, \tag{3}$$

where $\Omega_{CV}$ is the volume of the control volume CV, $S_{CV}$ is the surface of the control volume, and $\mathbf{n}$ is the outward-facing normal to the surface. This equation states that the rate of change of the amount of mass of a control volume is the rate of change within the volume plus the next flux of mass through the boundaries of the volume due to fluid motion (convection). For incompressible fluids (liquids) such as blood, density is constant and the first term in Eq. 3 disappears, leaving:

$$\int_S \rho \mathbf{v} \cdot \mathbf{n} \mathrm{d}S = 0. \tag{4}$$

Applying a similar technique to the momentum equation yields the control volume form:

$$\frac{\partial}{\partial t}\int_{\Omega_{CV}} \rho \mathbf{v} \mathrm{d}\Omega + \int_{S_{CV}} \rho \mathbf{v}\mathbf{v} \cdot \mathbf{n} \mathrm{d}S = \sum \mathbf{f}. \tag{5}$$

The forces $\mathbf{f}$ in the momentum equation are surface forces such as pressure and stress and body forces such as buoyancy and gravity. The blood is treated as a Newtonian fluid, that is, one in which stress is linearly dependent on velocity. The stress tensor, which represents the rate of transport of momentum, with constant viscosity is then:

$$\mathsf{T} = -p\mathsf{I} + \mu(\mathrm{grad}\mathbf{v}), \tag{6}$$

where $\mu$ is the dynamic viscosity, $\mathsf{I}$ is the unit tensor, and $p$ is the static pressure. Ignoring body forces and utilizing Eq. 6 in Eq. 5 the momentum conservation equation becomes:

$$\frac{\partial}{\partial t}\int_{\Omega_{CV}} \rho \mathbf{v} \mathrm{d}\Omega + \int_{S_{CV}} \rho \mathbf{v}\mathbf{v} \cdot \mathbf{n} \mathrm{d}S = \int_{S_{CV}} \mathsf{T} \cdot \mathbf{n} \mathrm{d}S. \tag{7}$$

The continuity and momentum equations are collectively referred to as the Navier-Stokes equations.

## 4 Numerical Methods

We describe the technique we use to approximate the continuous equations of our mathematical model with discrete versions suitable for solution on a computer.

### 4.1 Discretization

Any system of partial differential/integral equations such as those in our mathematical model derived above must be approximated by algebraic equations for solution on a computer. This is done by defining a numerical grid, which samples points in space with nodes. The original equations are approximated by a system of linear algebraic equations in which the values of the variables at the grid nodes are the unknowns. Each node provides one algebraic equation which relates variable values at that node with the values at neighboring nodes on the grid. This is assembled into a system of the form:

$$a_P \phi_P + \sum_k a_k \phi_k = q_P, \tag{8}$$

where $\phi$ represents the unknown, P denotes the node at which the equation is being solved, $k$ spans the neighboring nodes involved in the approximations, the $a$ are coefficients involving geometrical and fluid properties, and $q$ contains the terms in which all variable values are known (source terms). The system can be written in matrix form as:

$$\mathbf{A}\phi = \mathbf{q}, \tag{9}$$

where $\mathbf{A}$ is the coefficient matrix, $\phi$ is the vector containing the variables at the grid nods, and $\mathbf{q}$ is the vector containing the source terms. The solution of this system will be addressed below. For now we are only interested in its construction.

There are several methods for discretizing partial differential equations. The finite difference method is the oldest and simplest to implement. The partial derivatives in the differential form of the conservation equations are replaced by approximations in terms of nodal values of the variables. A structured grid is used. In the finite element method, the domain of interest is divided into a set of discrete volumes, or elements, that may be unstructured. The equations in the mathematical model are multiplied by a weight function before being integrated over the domain. The weight function describes how the variable varies over an element. While this method provides the most accuracy in irregular geometric domains, the formulation of the element equations does not have an easy physical interpretation.

The finite volume method, which we employ, divides the domain of interest into a finite number of small control volumes (cells), and places the numerical grid nodes at the center of the cells[3][7]. The grid is constructed such that adjacent cells share a face. The sum of the integral equations derived in the previous section for the individual cells is then equal to the global equation since surface integrals for adjacent cells cancel out. To obtain algebraic equations for each cell, the integrals must be approximated by quadrature formulae. A diagram of a 2D finite volume is shown in Fig. 1.

In Cartesian coordinates, 3D cells have six faces, denoted e, w, n, s, t, and b. Net boundary flux through a cell is the sum of integrals over the faces:

$$\int_S f \mathrm{d}S = \sum_k \int_{S_k} f \mathrm{d}S, \tag{10}$$

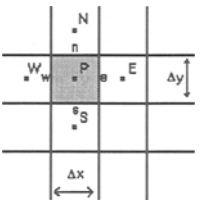


**Fig. 1.** 2D finite volume around node P, neighbors E, W, N, S, faces e, w, n, s

where $k$ spans the six faces and $f$ is the component of the convective or diffusive vector in the direction normal to the face $k$. The surface integral in the right side of Eq. 10 is approximated in terms of variable values at a finite number of locations on the face. The values on the face are approximated in terms of nodal values at the two cells which share the face. We use the midpoint rule, a second-order accurate method in which the integral is approximated as the product of the variable value at the face center and the area of the face. For the face 'e' we have:

$$F_e = \int_{S_e} f \mathrm{d}S \approx f_e S_e. \tag{11}$$

Since variables are solved for at nodes which lie in the center of cells, the value at the face centers must be interpolated. These values must be obtained with at least second-order accuracy to maintain that of the midpoint rule. Central differencing, described above, is a suitable approach.

Approximation of volume integrals is performed similarly to that of surface integrals; the volume integral is replaced by the product of the variable value at the center of the cell and the volume of the cell.

### 4.2 Matrix System Construction

For the unsteady term of Eq. 7, we use a second order three time level scheme. This leads to the approximation:

$$(\frac{\partial}{\partial t}\int_{\Omega} \rho u_i \mathrm{d}\Omega)_P \approx \frac{\rho \Delta \Omega}{2\Delta t}(3u_i^{n+1} - 4u_i^n + u_i^{n-1}) = A_P^t u_{i,P}^{n+1} - Q_{u_i}^t, \tag{12}$$

where:

$$A_P^t = \frac{3\rho\Delta\Omega}{2\Delta t} \quad \text{and} \quad Q_{u_i}^t = \frac{\rho\Delta\Omega}{2\Delta t}(4u_i^n - u_i^{n-1}). \tag{13}$$

The index $n$ refers to the time iteration step.

We split the surface integrals in the convective, viscous, and pressure terms into six faces as described above. The nonlinear terms are approximated by the product of an old value (from the preceding iteration) and a new value. The mass flux across the 'e' face is:

$$\dot{m}_e^m = \int_{S_e} \rho \mathbf{v} \cdot \mathbf{n} \mathrm{d}S \approx (\rho u)_e^{m-1} S_e, \tag{14}$$

where $m$ refers to the implicit solver iteration step. The convective flux of $u_i$ momentum is then:

$$F^c_{i,e} = \int_S \rho u_i \mathbf{v} \cdot \mathbf{n} \mathrm{d}S \approx \dot{m}_e u_{i,e}. \tag{15}$$

The diffusive flux requires the stresses $\tau_{xx}$, $\tau_{yx}$, and $\tau_{zx}$ at the face 'e'. Central difference approximations give:

$$(\tau_{xx})_e = \mu(\frac{\partial u}{\partial x})_e \approx \mu \frac{u_E - u_P}{x_E - x_P}, \tag{16}$$

and similarly for $(\tau_{yx})_e$ and $(\tau_{zx})_e$.

The pressure term is approximated by:

$$Q^P_u = -\int_S p\mathbf{i} \cdot \mathbf{n} \mathrm{d}S \approx -(p_e S_e - p_w S_w)^{m-1}, \tag{17}$$

which leads to the complete approximation for the momentum equation:

$$A^t_P u_{i,P} + F^c_i - F^d_i = Q^t_i + Q^P_i, \tag{18}$$

where:

$$F^c = F^c_e + F^c_w + F^c_n + F^c_s + F^c_t + F^c_b, \tag{19}$$
$$F^d = F^d_e + F^d_w + F^d_n + F^d_s + F^d_t + F^d_b. \tag{20}$$

The complete algebraic system for the $u$ momentum has the form:

$$A^u_P u_P + \sum_k A^u_k u_k = Q^u_P \tag{21}$$

The coefficient $A_E$ in the $A$ matrix is:

$$A^u_E = \min(\dot{m}^u_e, 0) - \frac{\mu S_e}{x_E - x_P}, \tag{22}$$

and the coefficients for the other faces are analogous. The $A_P$ coefficient is:

$$A^u_P = A^t_P - \sum_k A^u_k, \tag{23}$$

and the $Q$ vector is:

$$Q^u_P = Q^t_u + Q^p_u. \tag{24}$$

The construction of the system for the $v$ and $w$ equations are analogous.

**Pressure Correction** Because the solution for the momentum equations uses old values of pressure for the calculation of the pressure term (Eq. 17), the result does not in general satisfy the continuity equation. To resolve this problem, we use the SIMPLE algorithm first developed by Caretto et al. [1]. We first solve

the momentum equations as described above, and then gradually "correct" the pressure, and hence the velocity by way of the pressure term, until the continuity equation is satisfied to within some small numerical tolerance.

For the continuity equation to be satisfied, the net mass flux through a cell must be zero. After solving the momentum equations above, the net mass flux is:

$$\Delta \dot{m}_P = \dot{m}_e + \dot{m}_w + \dot{m}_n + \dot{m}_s + \dot{m}_t + \dot{m}_b. \tag{25}$$

The velocity corrections $u'$, $v'$, and $w'$ will cause this term to become zero.

The pressure correction $p'$ is related to the velocity correction $u'$ by:

$$u'_e = -\frac{\sum_k A^u_k u'_k}{A_P} - \frac{S_e}{A^u_P}(p'_E - p'_P). \tag{26}$$

Substituting this into the continuity equation leads to the pressure-correction equation:

$$A^p_P p'_P + \sum_k A^p_k p'_k = -\Delta \dot{m}_P, \tag{27}$$

where the $A$ coefficients are:

$$A^p_P = -\sum_k A^p_k \quad \text{and} \quad A^p_E = -(\frac{\rho S^2}{A^u_P})_e, \tag{28}$$

and similarly for the other faces.

### 4.3 Matrix System Solution

The construction of our discrete approximation to the Navier-Stokes equations results in a matrix system in the form of Eq. 9. Many techniques have been devised to solve such systems, but they vary greatly in their utility (see Press et al. [10] for an excellent overview). Direct methods are applicable to any type of matrix, but their poor performance prevents their use in most interesting engineering problems. Gauss elimination, the most basic method, reduces a matrix to an upper triangular form through a series of linear operations on rows or columns. A back substitution phase then computes the unknowns from the upper triangular matrix. LU decomposition improves on the Gauss technique by constructing the triangular matrix in a way that does not require the source vector, resulting in considerable speedup for systems that use the same coefficient matrix repeatedly. Structured matrices, in which non-zero coefficients occur in small, structured areas of the matrix such as along the diagonal, reduce the computational and storage complexity of solvers for them substantially, but direct solvers are frequently still too slow for practical application.

Iterative solvers start out with an approximate solution to the system (for example, the solution at the previous time iteration), and add a residual to the solution until the unknown reaches zero (to within a small tolerance). Using large residuals allows the solver to complete faster, but too large a residual induces divergence, in which the solution error increases. Our numerical method uses the

algorithm of Stone's "strongly implicit solver" [12], an iterative solver specifically designed for approximations to partial differential equations. The matrices resulting from these approximations are highly structured, with non-zero coefficients occuring only along the main diagonal and a few nearby neighboring diagonals (depending on the neighborhood used in the approximation of derivatives). An approximate LU decomposition is performed in which the resulting triangular matrices have a similar sparsity to the original matrix. Stone's method takes advantage of the smoothness of the PDEs being solved to greatly improve convergence.

# 5 Boundary Conditions

In this section we describe the boundary condition requirements of our mathematical model and numerical implementation, and how we derive them from MRI-SPAMM LV wall motion data.

The Navier-Stokes equations require boundary conditions at all points on the boundary of the solution domain for uniqueness. These are either Dirichlet conditions, in which the velocity at a boundary point is specified, or Neumann conditions, in which the derivative of velocity at a boundary point is specified. In the latter case, velocity values at interior points adjacent to the boundary are extrapolated to satisfy the derivative condition.

For our finite volume discretization, this means that the solution domain must be surrounded by boundary cells. We utilize three types of boundary cells: inlet (Dirichlet), outlet (Neumann), and wall (zero normal velocity). At walls, the convective flux in the normal direction and normal viscous stress are zero. At inlet boundaries, the velocity normal to the boundary (mass flux) is usually specified. This specification of velocity at wall and inlet boundaries means that the pressure correction across these boundaries should be zero. This is implemented by extrapolating pressure at the exterior cells along the boundary so that the pressure gradient across the interior boundary cells is zero. Outlet cells generally have a zero velocity gradient in the normal direction. These boundaries are ignored in the momentum equations, and their values copied from the adjacent interior cells. Their mass flux is allowed to adjust during the pressure correction step to ensure continuity.

We approximate sloped or curved boundaries by blocking off cells in our regular grid which lie outside the true boundary. While this introduces a discretization error, good results can be obtained by using a sufficiently fine grid resolution. Alternative coordinate representations such as cylindrical can also reduce this error for appropriate shapes such as the LV.

## 5.1 Heart Wall Boundary Conditions

Our goal in this work has been to simulate the flow through the LV of specific patients. The availability of MRI-SPAMM imaging tools provides us with a motion model of the walls of the LV. We use the motion data computed for the inner wall as boundary conditions as we now describe.

The output of the MRI-SPAMM tracking algorithm is a 4D time-series of the positions of fixed points on the LV wall throughout the cardiac cycle. Dividing

the displacement of point positions by the time period computed from the pulse rate gives us the velocity at a finite number of positions on the wall. Unfortunately, these points are irregularly sampled, so we must interpolate them at regular intervals corresponding to our numerical grid. We do this by connecting the points on the wall to form a surface using splines or line segments. This wall surface is then intersected with planes which cut through the numerical grid at regular intervals along a specified axis. This intersection gives us a 2D cross section of the wall surface. The intersection plane is then divided into cells corresponding to the spacing of our numerical grid along the other two axes. A series of point-in-polygon tests is used to compute the grid cells which the wall surface intersects. The union of the intersection cells across all cut planes through the dataset provides us with the boundary cells required to solve the mathematical model. Because the boundary cells correspond to points on the wall surface, and the MRI-SPAMM data provides us with the velocity of these points, the boundary cells are of the inlet type. Recalling that "inlet" refers to the Dirichlet condition, the fact that the wall may be expanding at these points does not present a problem - the term inlet merely indicates a specified velocity and not the direction of fluid flow. Because the velocities at the boundary cells are interpolated from the irregularly sampled MRI-SPAMM data points, the velocities on the numerical grid boundaries are scaled to yield the same total flux across the wall.

The "outlet" (Neumann) boundaries are more complex. Because the MRI-SPAMM method does not track the base of the LV containing the valve, our input model is "open-ended." The results presented here treated the entire open area as an outlet area, but any valve model is straightforward to implement. Different valve configurations can be simulated by reconfiguring the boundary cells at the base, or improved MRI techniques in the future which image the base will be handled automatically by our method.

## 6 Results

We show the results of our method on a sample MRI-SPAMM dataset from a normal subject of a half cardiac cycle from end diastole to end systole.

Figure 2 shows a visualization of the blood flow in three frames of a half cardiac cycle. A small number of points was seeded across a plane perpendicular to the long axis and integrated in both time directions with the velocity field. The lines indicate the paths of the points, and the cones indicate the direction of motion. Speed is indicated by color. A 40x20x20 numerical grid was used with 40 slices perpendicular to the long axis of the LV. Approximately 1500 boundary points (depending on the frame) were interpolated from 91 MRI-SPAMM data points. Figure 2(a) is at end diastole, as the LV starts to contract. We see that the velocity shows a strong outflow pattern to the center of the base. Flow along the walls is toward the apex, which is consistent with the contraction of the walls. Figure 2(b) is a bottom view (from the apex) of the flow at end diastole. Here we can clearly see the rotational pattern induced in the blood flow by the twisting motion of the endocardium. The flat side of the cones indicates the outward (toward the base) motion of the flow in the interior region, while

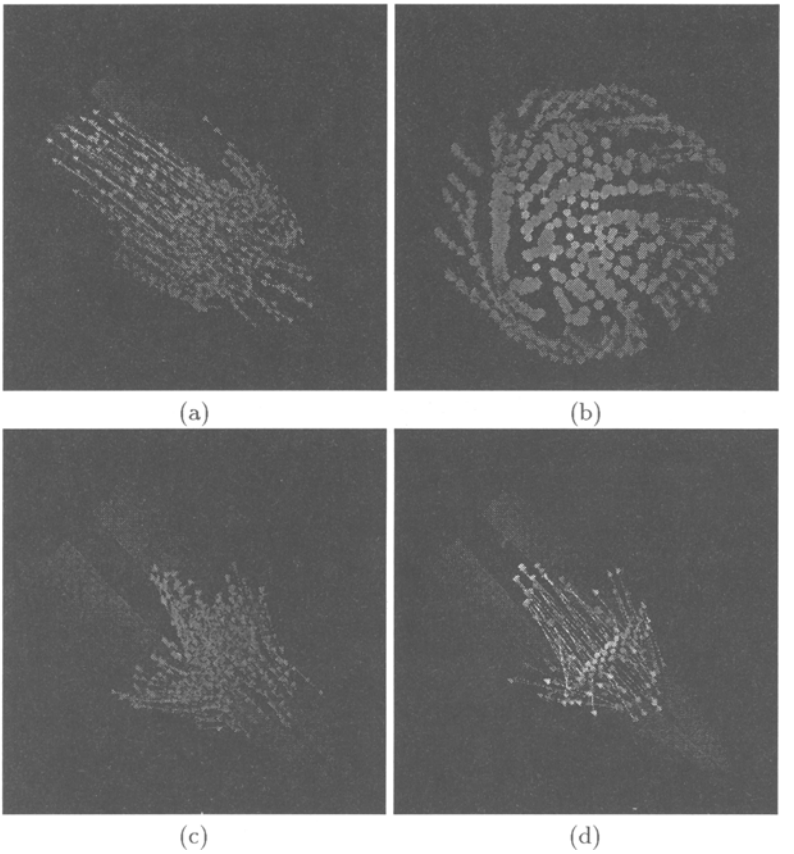


**Fig. 2.** Visualization of blood flow: (a) end diastole, (b) end diastole from apex, (c) mid systole, (d) end systole

the motion along the walls is in the opposite direction. Figure 2(c) is midway through systole, and we see the outflow pattern beginning to diminish as the LV approaches full contraction. The shorter lines indicate reduced speed as the flow begins to change direction. Figure 2(d) is at end systole, as the LV is beginning to expand again. Here we see the flow pattern from the base reversed, as blood begins to rush in again. Flow along the walls is directed away from the apex, consistent with the wall motion as the LV begins to expand.

## 7 Conclusions

In this paper we have described a novel approach to the patient-specific simulation of blood flow within the left ventricle. By utilizing MRI-SPAMM data as boundary conditions to a Navier-Stokes based CFD solver, we are able to produce convincing results which capture the full 3D contraction and twist of endocardial motion. Results from a sample dataset qualitatively consistent with clinical experience are presented. Ongoing work focuses on the application of

our method to various types of cardiac disease and the extraction of relevant quantitative information.

## 8 Acknowledgements

This research has been funded by grants from the NSF (Career Award 96-24604), the Whitaker Foundation, and the NIH/NLM (LM06638-01).

## References

1. L.S. Caretto, A.D. Gosman, S.V. Patankar, and D.B. Spalding. Two calculation procedures for steady, three-dimensional flows with recirculation. In *Proc. 3rd Int. Conf. Num. Methods Fluid Dyn.*, 1972.
2. G. Dubini, M.R. de Leval, R. Pietrabissa, F.M. Montevecchi, and R. Fumero. A numerical fluid mechanical study of repaired congenital heart defects. application to the total cavopulmonary connection. *J. Biomechanics*, 29(1):111–121, 1996.
3. J.H. Ferziger and M. Perić. *Computational Methods for Fluid Dynamics.* Springer-Verlag, 1996.
4. C.A.J. Fletcher. *Computational Techniques for Fluid Dynamics.* Springer-Verlag, 1991.
5. D.M. McQueen and C.S. Peskin. A three-dimensional computational method for blood flow in the heart: Ii. contractile fibers. *J. Computational Physics*, 82:289–297, 1989.
6. J. Park, D. Metaxas, and L. Axel. Analysis of left ventricular wall motion based on volumetric deformable models and mri-spamm. *Medical Image Analysis*, 1(1):53–71, 1996.
7. S.V. Patankar. *Numerical Heat Transfer and Fluid Flow.* Hemisphere Publishing, 1980.
8. G. Pelle, J. Ohayon, and C. Oddou. Trends in cardiac dynamics: Towards coupled models of intracavity fluid dynamics and deformable wall mechanics. *J. de Physique III*, 4(6):1121–1127, 1994.
9. C.S. Peskin and D.M. McQueen. A three-dimensional computational method for blood flow in the heart: I. immersed elastic fibers in a viscous incompressible fluid. *J. Computational Physics*, 81:372–405, 1989.
10. W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C.* Cambridge, 2nd ed. edition, 1992.
11. P.J. Roache. *Computational Fluid Dynamics.* Hermosa Publishers, 1972.
12. S.H. Stone. Iterative solution of implicit approximations of multidimensional partial differential equations. *SIAM J. Numerical Analysis*, 5:530–558, 1968.
13. T.W. Taylor, H. Suga, Y. Goto, H. Okino, and T. Yamaguchi. The effects of cardiac infarction on realistic three-dimensional left ventricular blood ejection. *Trans. ASME*, 118:106–110, 1996.
14. J.D. Thomas and A.E. Weyman. Numerical modeling of ventricular filling. *Annals Biomedical Engineering*, 20(1):19–39, 1992.
15. A.P. Yoganathan, Jr. J.D. Lemmon, Y.H. Kim, P.G. Walker, R.A. Levine, and C.C. Vesier. A computational study of a thin-walled three-dimensional left ventricle during early systole. *J. Biomechanical Engineering*, 116:307–314, 1994.