Virtual angioscopy in human coronary arteries with visualization of computational hemodynamics

Andreas Wahle,[a] Steven C. Mitchell,[a] Sharan D. Ramaswamy,[b]
Krishnan B. Chandran,[b] and Milan Sonka[a]

The University of Iowa, [a]Department of Electrical and Computer Engineering,
and [b]Department of Biomedical Engineering, Iowa City, IA 52242–1527, USA

ABSTRACT

We are presenting a comprehensive system for fusion of intravascular ultrasound (IVUS) data and x-ray angiography, aiming to create a geometrically accurate 3-D or 4-D (3-D plus time) model of the coronary vasculature. For hemodynamic analyses, methods of computational fluid dynamics (CFD) are applied to the reconstructed data, resulting in quantitative estimates of the wall shear stress. Visualization is performed using the Virtual Reality Modeling Language (VRML). Lumen and adventitia borders are modeled as surfaces using indexed face sets; quantitative results are encoded as color per vertex. The endoscopic mode (virtual angioscopy) allows an interactive fly-through animation with variable speed along with arbitrary positioning within the vessel. Since this functionality exceeds those of the standard VRML animation nodes, an external prototype library containing VRML and JavaScript definitions has been developed that provides a 3-D graphical user interface to navigate within the endoscopic mode. The control panel is available on demand, but does neither obstruct any vessel features when not needed, nor does it limit the viewport for the scene. Preliminary results showed a good feasibility of the overall procedure, and a high reliability of the fusion and CFD methods as well as the visualization with the virtual endoscopy VRML library.

Keywords: Virtual Angioscopy, Virtual Reality Modeling Language, Coronary Atherosclerosis, 3-D Computational Hemodynamics, Intravascular Ultrasound, X-ray Angiography

1. INTRODUCTION

In coronary artery diseases, intravascular ultrasound (IVUS) is an established complement to conventional selective x-ray angiography.[1,2] The accurate cross-sectional data provide a good basis for visualization and analyses of wall properties and the vessel hemodynamics.[3] Previous studies of computational hemodynamics have implicated regions of flow reversal and separation, as well as relatively low wall shear stresses and oscillating shear stresses, as causes for initiation and growth of atherosclerotic plaques. Since the cardiovascular system is a site of frequent initiation and growth of atherosclerotic lesions, a number of experimental and computational simulation studies has been reported on the fluid mechanics in arteries.[4–7] Especially, coronary arteries are embedded on the epicardial layer of the cardiac chamber and hence are subjected to translation, bending, and twisting as the cardiac muscles contract and relax during a cardiac cycle. In attempting to correlate the fluid mechanics of blood flow in the coronary vessels and atherosclerotic initiation, these effects must be included. However, conventional 3-D reconstructions from IVUS stack the frames as acquired during the pullback of the catheter to form a straight volume, thus completely neglecting the vessel curvature. Therefore, we are developing a comprehensive system for fusion of the IVUS data with the pullback path determined from x-ray angiography.[8–10] This results in a geometrically accurate 3-D or 4-D (3-D plus time) model of the coronary vasculature, such as shown in Figure 1. With the fusion of angiographic and intravascular ultrasound images, we have demonstrated the capability of reconstructing morphologically realistic 3-D geometry of the coronary vessels and its motion during a cardiac cycle.[3,11] The long-term goals of our study are to incorporate the motion of the coronary vessels in the computational simulation to describe the fluid mechanics in the coronary vessels in detail.

This paper focuses on the visualization aspect, specifically on the presentation of the vessel segment and the derived quantification results in an endoscopic way (virtual angioscopy). The computational fluid dynamics calculations are the primary topic of a companion paper of this symposium.[3] Virtual reality (VR) methods are gaining increasing importance in the visual presentation of complex medical data and quantifications on those. Endoscopic approaches

E-mail: <andreas-wahle@uiowa.edu>; http://www.engineering.uiowa.edu/~awahle; Fax: +1-319-335-6028

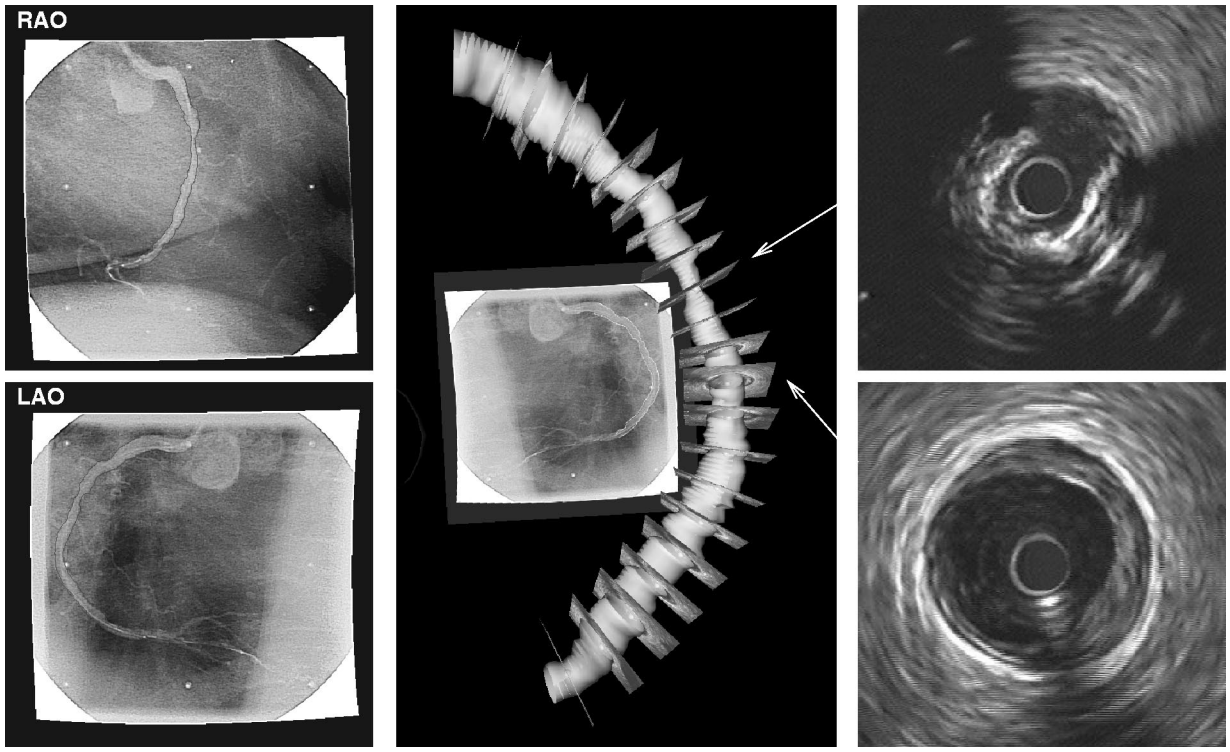


Figure 1. The fusion process: Angiograms of a stenosed human right coronary artery pre intervention are shown on the left hand side, a subset of the corresponding IVUS frames on the right hand side; center shows the reconstructed lumen surface in VRML along with some of the IVUS frames in their geometrically correct location and orientation, where the locations of the two IVUS frames depicted on the right are indicated with arrows.

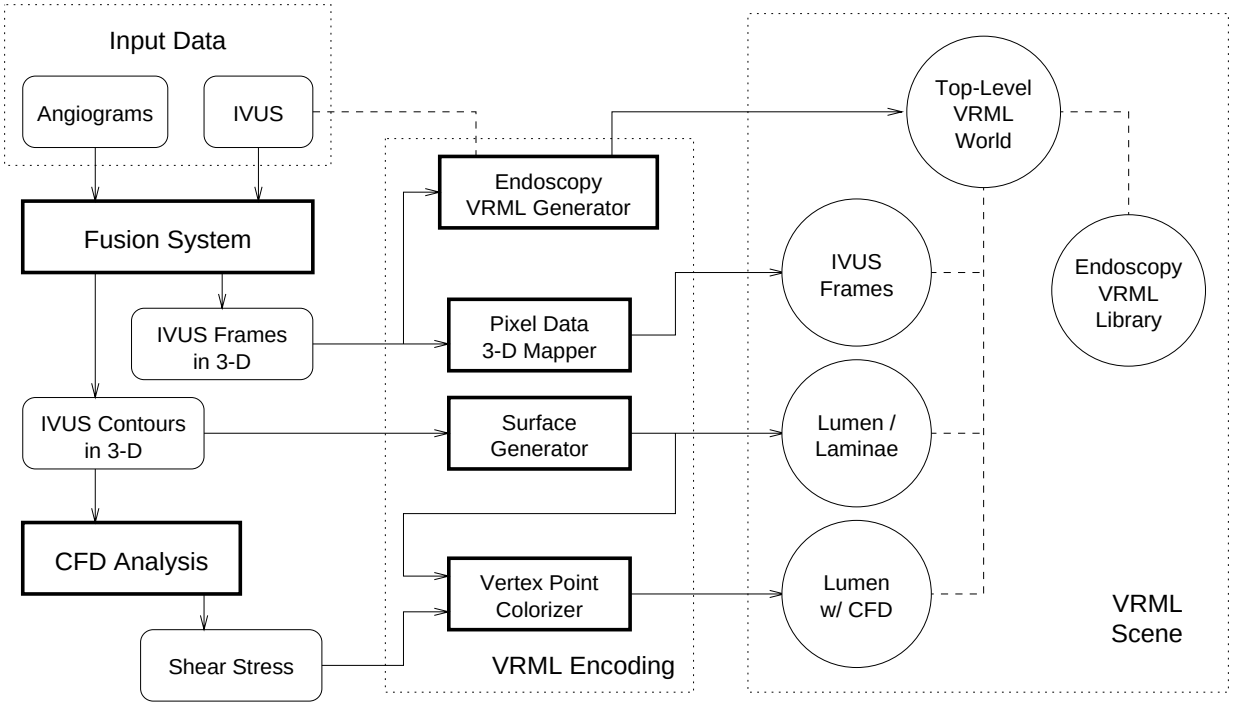


Figure 2. Processing scheme: The output from the fusion is used for computational fluid dynamics calculations, the VRML scene is generated automatically; ⟶ data flow, −− references.

allow the visualization from inside of tubular structures, thus there already exist several systems for virtual endoscopy or angioscopy, simulating the actual procedure. Some common examples include virtual colonoscopy,[12,13] virtual bronchoscopy,[14,15] and virtual angioscopy in aorta[16] or liver vessels.[17] These methods are usually based on either computed x-ray tomography (CT) or magnetic resonance imaging (MRI) modalities.

In contrast to the 3-D reconstruction from parallel CT or MRI slices, which require to solve problems of finding the correct vessel centerline as well as the accurate surface,[13] IVUS images are already acquired almost perpendicular to the vessel centerline. This simplifies surface generation as well as trajectory calculations, where on the other hand it has to be kept in mind that the 3-D reconstruction from biplane projections and fusion with the IVUS data to obtain the 3-D contours along the catheter path is a complex task.[8–10] For the presentation of the 3-D model, a standardized VR description language is utilized, the *Virtual Reality Modeling Language* (VRML).[18,19] VRML has the advantage that interaction and user interface can be realized independently from any specific platform, once the 3-D model is created. To provide sufficient flexibility in interaction, VRML allows the encapsulation of Java or JavaScript nodes. This concept was used to build a re-usable VRML library from which the necessary code for each scene can be automatically derived.

2. METHODS

2.1. Overview

A schematic overview of the individual parts of the fusion, quantification, and visualization system is given in Figure 2. At the beginning, the fusion subsystem generates a single 3-D model (or a set of models in the 4-D case) derived from biplane angiography and intravascular ultrasound. It provides a function that maps the 2-D frames into the 3-D space along with the 3-D coordinates of the segmented contours. From these data, a hemodynamic quantification is performed by the computational fluid dynamics (CFD) subsystem. The frames and contours are used to automatically create a VRML scene; the quantification results (e.g., shear stress) are included into the 3-D surface data encoded as colors.

For the specific task of endoscopic viewing to perform a virtual angioscopy on the reconstructed and evaluated data (Fig. 3), the VRML scene also contains a comprehensive external prototype library. This library provides the necessary functionality during the endoscopic mode, including a graphical user interface to navigate through the vessel. The primary focus of this paper is the implementation of the endoscopic mode in VRML.

2.2. Data acquisition and fusion process

The image data acquired from biplane angiography, which is usually needed for catheter guidance during the IVUS intervention, can be utilized to accurately reconstruct the path and orientation of the IVUS catheter in 3-D and 4-D (i.e., 3-D plus time, where *time* specifies any phase of the heart cycle). During this fusion process, the relative and absolute orientations of the IVUS frames are determined using our previously reported system for establishing the absolute orientation of still images in 3-D.[8–10] The fusion process can be split into the following list of tasks: (1) Acquire IVUS and angiographic data and sort them by the heart phase; (2) segment IVUS and angiographic data for vessel and tissue structure; (3) determine the 3-D location for each feature in each considered heart phase; (4) use the reconstructed 3-D or 4-D data for CFD analysis and visualization.

As outlined in the companion paper,[3] it is essential that the data be sorted correctly for the heart phase with respect to the ECG signal. Frequently, the ECG signal is not available in either angiographic or IVUS data. In these cases, the phases need to be identified manually, which is tedious and prone to error. Use of the DICOM standard as archiving medium is encouraged, since it provides the means to store the ECG signal as well as acquisition parameters (imaging geometry, frame rate, and field of view) along with the image data.[2]

The biplane angiograms are taken immediately at the start of the pullback and cover at least one heart cycle. They are used to automatically extract the catheter path along the expected pullback trajectory, and to find the vessel outline as a reference for the following fusion process. As minimum requirement for matching between angiograms and IVUS in 3-D, the lumen contour has to be segmented from the IVUS data as well.[10] Well-established graph-search-based segmentation methods are used,[2] which were fine-tuned for the specific image data. The fusion process itself consists of two parts, (a) identifying the location of each IVUS frame in each phase, and (b) estimating the orientation of the frame in 3-D space. Part (b) furthermore splits up into (b1) calculating the relative twisting of the IVUS catheter from its path based upon a differential-geometric model (Frenet–Serret), and (b2) estimating the absolute orientation of the entire frame set using a non-iterative statistical approach.[8,9]

Figure 3. Endoscopic view from the most distal end towards the proximal end for the artery shown in Figure 1.

2.3. Computational fluid dynamics analysis — Shear stress quantification

For this study, the reconstructed geometries of a typical right coronary artery at end-systole and end-diastole (the latter of which is shown in Fig. 1) were used to compute the distribution of wall shear stress in the vessel during the systolic and diastolic configurations. The presence of stenosis in the imaged vessel is also clearly visible in the reconstruction. A finite-volume numerical procedure (U2RANS* CFD software) for 3-D fluid flow problems was used in the analysis of the reconstructed coronary arteries at both end-diastolic and end-systolic cardiac phases. Constant fluid property parameters for whole human blood were used based on a Reynold's number of 300. The density of blood was specified as $\rho = 1035\,\mathrm{kg/m^3}$ with a kinematic viscosity $\nu = 3.381 \cdot 10^{-6}\,\mathrm{m^2/s}$. For the inlet velocity boundary condition, a flat velocity profile under steady-state conditions was used ($u \approx 0.52\,\mathrm{m/s}$). Blood was treated as an incompressible, homogenous, and Newtonian fluid. Further, the no-slip boundary condition was assumed at the vessel wall and a constant pressure outlet condition was utilized in the CFD model.[3]

The procedure for solving flows in the 3-D coronary geometry starts with independent grid generation (using GRIDGEN† grid generation software) of the coronary vessel in each of the considered cardiac phases. The derivation of a 3-D grid is initiated by converting the image segment into a closed surface. Next, this surface is split and meshed into six independent domains that make up one volumetric 3-D mesh. In this format, it is now possible to subject this mesh to 3-D CFD analysis. As part of the geometry development, the grid density has been optimized based on a compromise between the grid size and the structural integrity of the mesh. An iterative procedure is performed to solve the equations of flow and space conservation. An initial pressure is assumed at the outlet boundary and subsequently adjusted based on a pressure-correction scheme, thus ensuring mass conservation.[3]

*U2RANS was developed by Yong G. Lai, Iowa Institute of Hydraulic Research at The University of Iowa
†Pointwise Inc., Fort Worth TX, USA

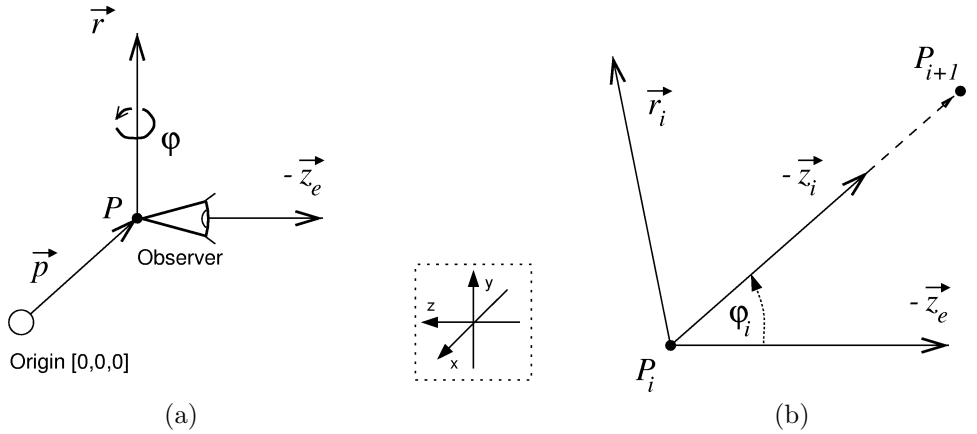


Figure 4. Specification of a viewpoint: (a) as defined in VRML; (b) derived from catheter path or vessel centerline.

2.4. Computation of the fly-through trajectory

In general, a fly-through path has to be calculated as a sequence of viewpoints through the vessel to be observed. Obviously, jumping from viewpoint to viewpoint is not appropriate, thus interpolation must be employed. This section will describe how a viewpoint is generated for each frame location, while Sec. 2.6.4 provides the information about the actual implementation as VRML interpolation node.

A viewpoint in VRML is defined by a `position` and an `orientation`, which may be expressed as a triple $(\vec{p}, \vec{r}, \varphi)$ containing a position vector $\vec{p}$, a rotation axis $\vec{r}$, and a rotation angle φ in a right-handed coordinate system. The initial viewing direction is the negative z-axis (Fig. 4a).[18,19] This *axis-and-angle* definition for the orientation is well-known from robotics.[20] For each point P_i of the trajectory, a respective set of parameters $(\vec{p}_i, \vec{r}_i, \varphi_i)$ has to be calculated. The vectors $\vec{p}_i$ can directly be derived from either the catheter location for frame i (for following the catheter path), or from the centroid of the segmented lumen contour (thus following the vessel centerline). The first method is the default and usually leads to a smoother animation due to the relatively high stiffness of the catheter. For the orientation, rotation axis $\vec{r}_i$ and angle φ_i are calculated as follows:

$$\vec{d}_i = \vec{p}_{i+1} - \vec{p}_i = [\,(x_{i+1} - x_i)\,,\,(y_{i+1} - y_i)\,,\,(z_{i+1} - z_i)\,] = [\,dx_i\,,\,dy_i\,,\,dz_i\,] \tag{1}$$

$$-\vec{z}_i = \frac{\vec{d}_i}{\|\vec{d}_i\|} \tag{2}$$

$$\vec{r}_i = \frac{-\vec{z}_e \times -\vec{z}_i}{\|-\vec{z}_e \times -\vec{z}_i\|} = \left[\left(\frac{dy_i}{\sqrt{dx_i^2 + dy_i^2}}\right),\,\left(\frac{-dx_i}{\sqrt{dx_i^2 + dy_i^2}}\right),\,0\right] \tag{3}$$

$$\cos\varphi_i = -\vec{z}_e \cdot -\vec{z}_i = -\left(\frac{dz_i}{\|\vec{d}_i\|}\right) \tag{4}$$

where $\vec{p}_{i+1}$ is the positional vector from P_i to the next point P_{i+1} of the trajectory path; $\vec{d}_i$ is the directional vector from points P_i to P_{i+1}; $-\vec{z}_i$ is the desired viewpoint direction, with $-\vec{z}_e = [0, 0, -1]$ indicating the initial view (Fig. 4b). Note that the z-component of $\vec{r}_i$ is always zero, i.e., the axis of rotation is in the x/y plane. In a special case, no point P_{i+1} exists for $i = (n-1)$ in a set of n points $\{P_0 \cdots P_{n-1}\}$; therefore,

$$\begin{aligned} \vec{r}_{n-1} &= \vec{r}_{n-2} \\ \varphi_{n-1} &= \varphi_{n-2} \end{aligned} \tag{5}$$

are assumed, thus yielding a viewpoint definition for each point P_i. This scheme so far does not reflect the actual axial orientation of the corresponding IVUS frame, i.e., the view-up vector $\vec{v}_i$ will always be the same for each given viewing direction $-\vec{z}_i$. Thus, $\vec{v}_i$ may be aligned with the view-up vector of the IVUS frame i by rotation around $\vec{z}_i$.

2.5. VRML graphical user interface for virtual angioscopy

The initial scene after startup shows the object (vessel) in a frontal view, and indicates the location of the endoscopic observer (usually at the most distal end) by a pyramid representing the observer's eye and viewing direction (Fig. 5). The object may have several different representations, e.g., just the semi-transparent IVUS frames in their correct orientations and locations, the lumen surface with or without the frames, or the surface along with the color-coded results from CFD quantification. To switch between these representations, clicking on the currently visible object will advance to the next representation. Any set of objects can be defined for a scene.

While the initial viewpoint allows handling of the scene as with any other VRML world, the endoscopic view requires substantially more features for navigation. An automated fly-through with variable speed along with arbitrary positioning within the vessel cannot be accomplished by the standard VRML animation nodes alone. Another problem is the presentation of the control panel itself, which should be available on demand, but must neither obstruct any vessel features when not needed, nor limit the viewport for the scene.

The solution of this problem was to put the control panel within the endoscopic scene itself, attached to the observer similar to optical glasses. By default, the control panel is switched off, but can be activated by clicking on a proxy shape directly in front of the observer. Since the control panel moves in 3-D with the observer, it seems to be at a constant location relative to the viewer. Once the user moves the mouse outside of the proxy shape, the control panel disappears, thus allowing an unobstructed view of the scene in full size.

The control panel provides the following features to influence the scene (Fig. 6a):

- With the `PROX` and `DIST` buttons, the viewing direction of the observer can be switched between proximal and distal, respectively.
- The upper slider indicates the *speed* of the observer in either proximal or distal directions; the observer does not move when the sphere is in the center. Clicking the `STOP` button above the slider stops observer motion.
- The lower slider indicates the current *location* of the observer; it moves whenever the observer is moving, or can be dragged directly to any location.
- The `IVUS` button switches the panel into a different mode, where the actual IVUS frame corresponding to the current location is shown; with the two arrows, the user can move into either proximal or distal directions frame by frame (Fig. 6b).
- The `VIEW` button switches the object representation, as was described above.
- To leave the endoscopic mode, the `EXIT` button will change the scene to a different viewpoint and reactivate the browser's control panel.

The `EXIT` button may be redundant, since the VRML browser usually provides some mechanism to change the viewpoint. However, given that the browser console is deactivated during the endoscopic mode, it may not be possible to switch to another viewpoint, thus leaving the endoscopic mode, depending on the browser used. Therefore, this additional button has been included into the user interface.

2.6. Implementation

2.6.1. Fusion from biplane angiography and intravascular ultrasound

A comprehensive overview on the fusion system can be found elsewhere.[8–10] While the functional core in general is platform-independent, the user interface has been developed for the PC-based environment usually available in hospitals. The newest version is based on the *Microsoft Foundation Classes* (MFC/4.2) and includes all functionalities of segmentation, 3-D reconstruction, fusion, and basic 3-D visualization in VRML (i.e., not including CFD analysis or the endoscopic mode).[21]

2.6.2. Mapping of shear stress data to contour points

For the hemodynamic analysis, the CFD system as outlined in Sec. 2.3 is directly fed with the results from the fusion program. A postprocessing step assigns the results of the analysis, e.g., the shear stress distribution, to the respective contour points of the reconstructed lumen surface. Using a color look-up-table, each quantification value can be associated with a specific color. In VRML, this color is applied as color-per-vertex encoding within the `IndexedFaceSet` created for each surface.[3]

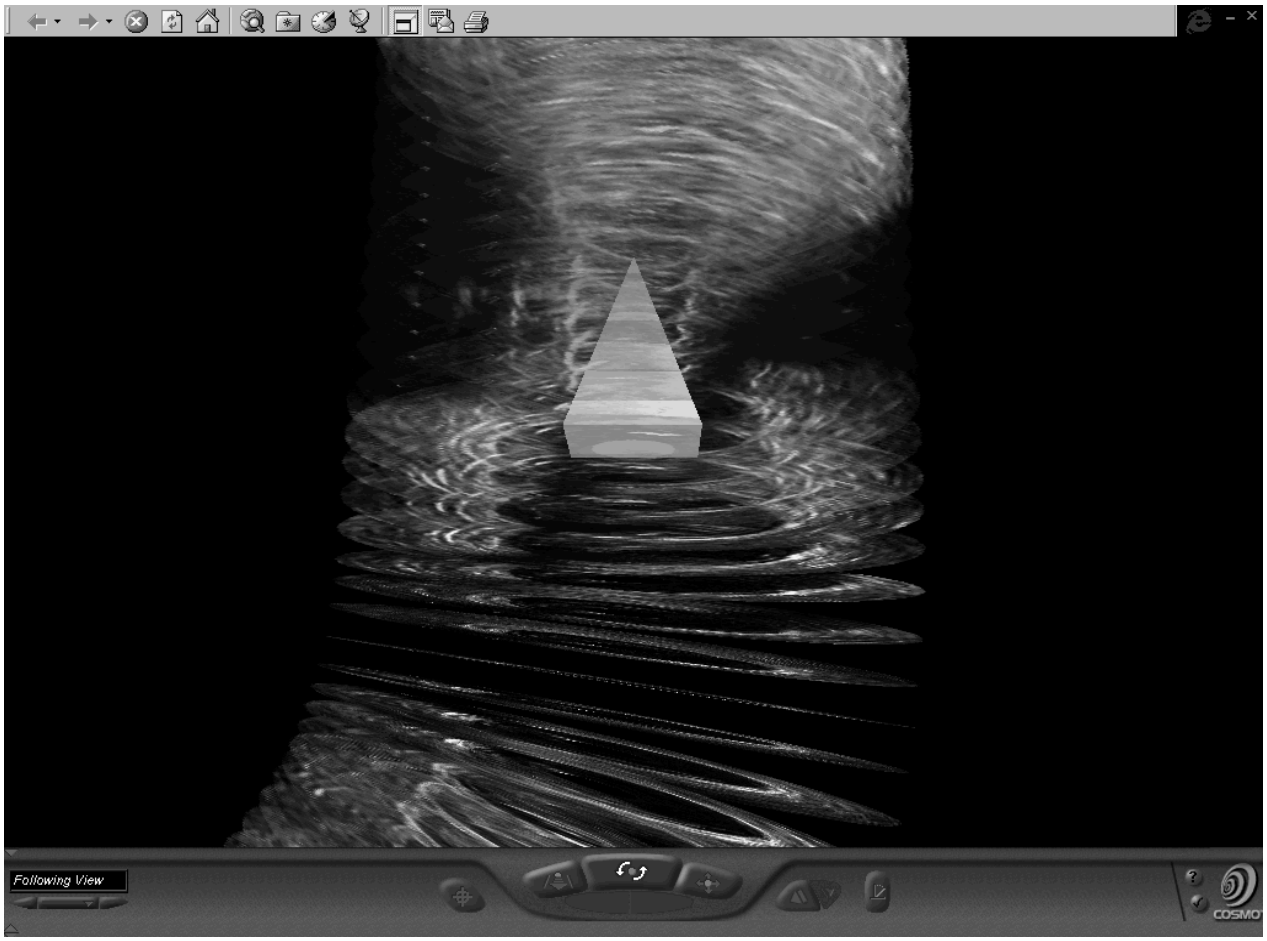
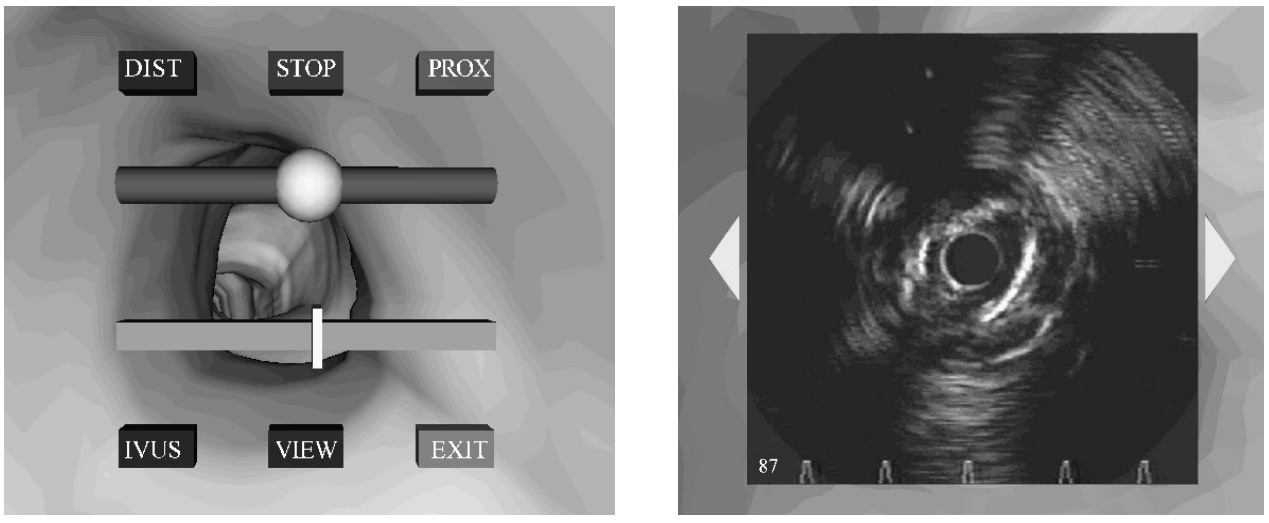


Figure 5. The IVUS data is represented as semi-transparent frames; the pyramid indicates the current location of the observer at a stenosis.



(a) (b)

Figure 6. Endoscopic view, with frames replaced by lumen surface representation: Control panel (a) in standard view and (b) to examine IVUS image data (cropped, but depicted at approximately the same scale as Fig. 5).

2.6.3. Automated generation of VRML code for surfaces and IVUS frames

As already mentioned in the previous paragraph, the discrete 3-D contours are combined to one `IndexedFaceSet` shape in VRML. This requires a triangulation of the non-parallel contours. Since the vessel contours are downsampled to a reduced set of representative vertex points,[22] the number of points per contour is fixed, thus making the triangulation straight-forward.[3]

For the default representation of the IVUS frames in the VRML scene, the raw pixel data is simply mapped onto the frames using the `ImageTexture` subnode of `Appearance`. This is however insufficient in the endoscopic mode, where they would obstruct the view through the vessel. On the other hand, a constant transparency value over all frames would not be appropriate either. Thus, in the semi-transparent IVUS frames seen in Figs. 3 and 5, a pixel-by-pixel transparency has been defined using the `PixelTexture` subnode. Its `image` field allows a direct definition of a grayscale image by indicating grayvalue and opacity for each pixel explicitly.

The transparency is based on (a) the location of the pixel within the image, and (b) its grayvalue. Structures not related to the vessel wall or plaque should not be visible, or just included up to a certain extent. This implies masking of the image, which is in the simplest case a ring defined around the catheter, excluding the catheter itself and possible catheter-related artifacts as well as any annotation outside of the transducer's field of view. A more sophisticated masking can be applied by using the 2-D segmentation information, thus excluding the lumen from being visible as well as any echoes beyond the external lamina. Excluded pixels are encoded with full transparency and allow therefore an unobstructed view through the frame at this point. For the relevant data, the transparency is based on the grayvalue; e.g., a bright echo may indicate a calcification, and will thus be encoded with a high opacity value, while dark portions of the image may indicate the lumen of another vessel or other structures not providing any echo and therefore have a lower opacity.

2.6.4. VRML external prototype library for virtual endoscopy and angioscopy

To allow efficient reuse of common VRML and JavaScript code for all scenes to be generated, and to effectively restrict the size of the top-level VRML file, an external prototype library[18,19] is used. The prototype definitions of the library along with the semantics of the new node types are shown in Table 1; the structure of the top-level VRML file is shown in Table 2. The two major node types provided by the library are the `EndoscopyInterpolator` and `EndoscopyObserver` nodes, controlling the path of the observer through the vessel and providing the interaction with the user, respectively. As was indicated in Fig. 2, for each object that may be part of a representation, a separate VRML file exists. These files are actually accessed through the VRML library as well, specifically by using the `EndoscopyObjInclude` node, to define for each representation individually if they are to be included or not.

In the top-level VRML file (Tab. 2), the `Mover` node defines the fly-through trajectory and controls its animation, followed by the navigation control. The `MoveIt` transformation node is controlled by `Mover` and defines the common coordinate system for the point light, the endoscopic viewpoint, and the observer node with the user interface included. Note that the endoscopic viewpoint `ControlView` is constant, while its coordinate system is animated through `MoveIt`; the nested `Transform` node results from the need to switch between proximal and distal views. The `FollowView` viewpoint allows a view from outside the object, but following the observer (see an example of this view in Fig. 5). The final `Group` node contains all the objects that may be included into the scene, along with a `TouchSensor` to switch between the representations. No further nodes are created at the highest level, all other functionality is provided by the library and the inlined code for the VRML objects.

3. RESULTS

The overall fusion study currently includes seven patients undergoing routine stent placement, in addition to phantom and in-vitro validation on twelve cadaveric pig hearts.[3,8–10] For the purpose of endoscopic visualization, one of the in-vitro pig arteries and one of the in-vivo human arteries have been chosen, the latter of which shown in the figures of this paper. The pig data with 72 IVUS frames included at a sampling rate of 1.5 mm distance between frames resulted in the automated generation of about 54 MB of VRML code. The in-vivo right coronary artery in the end-diastolic phase, as shown in Fig. 1, included 153 ECG-gated frames at 0.5 mm/s pullback speed. The generated CFD mesh contained 15,000 grid points; the shear stresses between $\tau = 0.74$ Pa and $\tau = 11.2$ Pa were color-coded from blue to red, respectively.[3] The resulting VRML scene required approximately 130 MB of disk space. Since all the data are loaded into main memory during startup of the VRML browser, respective hardware requirements

```
EXTERNPROTO EndoscopyInterpolator [
    field    MFVec3f    keyPosition
    field    MFRotation keyOrientation
    field    MFFloat    key
    field    SFFloat    baseSpeed
    eventIn  SFFloat    set_speed
    eventIn  SFFloat    set_fraction
    eventOut SFFloat    fraction_changed
    eventOut SFVec3f    position_changed
    eventOut SFRotation orientation_changed
] "libEndoscopy.wrl#EndoscopyInterpolator"


EXTERNPROTO EndoscopyObserver [
    field    SFInt32    frames
    field    MFString   frameUrls
    eventIn  SFFloat    set_fraction
    eventOut SFFloat    fraction_changed
    eventOut SFFloat    speed_changed
    eventOut SFRotation view_changed
    eventOut SFBool     object_changed
    eventOut SFBool     reset_binding
    eventOut SFBool     exit_binding
] "libEndoscopy.wrl#EndoscopyObserver"
```

```
EXTERNPROTO EndoscopyNavigator [
    eventIn  SFBool     set_binding
    eventOut MFString   type_changed
    eventOut SFBool     isHeadlight
    eventOut SFBool     isCandlelight
] "libEndoscopy.wrl#EndoscopyNavigator"


EXTERNPROTO EndoscopyObjSwitch [
    field    SFInt32    nObjects
    eventIn  SFBool     set_newObject
    eventOut SFInt32    object_changed
] "libEndoscopy.wrl#EndoscopyObjSwitch"


EXTERNPROTO EndoscopyObjInclude [
    field    MFString   url
    field    MFInt32    show
    eventIn  SFInt32    set_object
] "libEndoscopy.wrl#EndoscopyObjInclude"
```

`EndoscopyInterpolator` — This node defines and controls the fly-through trajectory by combining the standard `PositionInterpolator` and `OrientationInterpolator` nodes. The location (fraction) can either be explicitly set using the `set_fraction` input event, or the internal `TimeSensor` can be activated by sending a non-zero value to `set_speed`. In both cases, `fraction_changed` will indicate the new position along the trajectory.

`EndoscopyObserver` — This node provides the 3-D user interface and is sharing the same coordinate system as the endoscopic `Viewpoint` node. References to the original IVUS data are provided through the `frameUrls` field containing `frames` entries. It generates events if the user modifies speed, viewing direction (proximal/distal), or the current location. For synchronization, the events `set_fraction` and `fraction_changed` are cross-connected with those of the `EndoscopyInterpolator` node. Because the default browser controls are turned off in the endoscopic mode, the `reset_binding` and `exit_binding` events are used to switch between viewpoints. Finally, the `object_changed` event is created when the `VIEW` button is hit, since the controls usually cover the object itself, and it otherwise would not be possible to change the representation during the endoscopic mode.

`EndoscopyNavigator` — There are some special actions taken when the endoscopic mode is entered: The lighting scheme is changed from the directional headlight to a point light at the location of the observer, and the browser controls are switched off to prevent interference with the controls of the `EndoscopyObserver` node. The input is hardwired to the `isBound` output event of the endoscopic `Viewpoint` node.

`EndoscopyObjSwitch` — This node models a looping switch between `nObjects` different object representations. With each input event, either from clicking at the currently visible object or the `VIEW` button of the endoscopic panel, a counter $[0 \cdots \texttt{nObjects} - 1]$ is incremented by one and the new representation sent to `object_changed`.

`EndoscopyObjInclude` — For each external object, e.g., IVUS frames, vessel wall surfaces, color-coded surfaces, one node of this type exists. It implements an extension of the standard `Inline` node by enabling switching of its visibility. The `show` field consists of an array for which the length equals the number `nObjects` of the corresponding `EndoscopyObjSwitch` node; for any `0`-entry, the VRML code given in `url` is *not* included, for each `1` it is. In this way, any combination of external objects can be defined for each representation.

Table 1. External prototype definitions to access the VRML endoscopy library.

```
Transform {
        DEF Mover EndoscopyInterpolator { }

        DEF NaviSwitch EndoscopyNavigator { }
        DEF Navigate NavigationInfo { }

        Viewpoint { }

        DEF MoveIt Transform {
                DEF FollowView Viewpoint { }
                DEF ViewIt Transform {
                        DEF Candle PointLight { }
                        DEF ControlView Viewpoint { }
                        DEF Controls EndoscopyObserver { }
                }
        }

        Group {
                DEF ObjTouched TouchSensor { }
                DEF ObjSelect EndoscopyObjSwitch { }

                DEF Object0 EndoscopyObjInclude { }
                          |<n>
                DEF Object[n-1] EndoscopyObjInclude { }
        }
}
```

Table 2. Node hierarchy of the automatically generated top-level VRML file.

have to be met. For display, the CosmoPlayer 2.1[‡] with Netscape 4.7 or MS Internet-Explorer was used on either SGI Onyx 2 (IRIX 6.5 operating system, 576 MB main memory) and Pentium III (NT 4.0/Win2000, 256/512 MB) machines, showing excellent performance.

4. DISCUSSION

Our preliminary results showed a good feasibility of the overall procedure, and a high reliability of the fusion and CFD methods. The visualization in VRML along with the endoscopic mode allow a good assessment of normal and diseased vessel segments, directly indicating the 3-D morphology as well as hemodynamic data such as the local shear stress. The new prototype library extents the capabilities of the user to navigate within the vessel and to control the automated fly-through animation substantially as compared to the regular controls provided by the VRML browser. The huge amount of data however extends the loading time considerably, taking up to a minute or more for the initial view to appear. Once the data is loaded, navigation can be performed in real time.

The extension of the endoscopic mode into 4-D is in progress, as already done for the plain views.[3] It would require an extension of the `EndoscopyInterpolator` node to accept input events allowing the real-time modification of the fly-through trajectory. It must be ensured that the endoscopic observer stays within the vessel while it is cycling through the heart phases, and that the viewing direction is adjusted accordingly. Furthermore, a synchronization between the animations of the fly-through trajectory and the vessel objects has to be implemented. This implies that the functionality of the `EndoscopyObjInclude` node has to be extended to either export any internal timing information or to allow timing events to influence the object's shape and appearance.

While the VRML standard allows a user-friendly manipulation of the scene and ensures a platform-independent visualization, it is limited with respect to interaction. Alternative methods include Open-GL or the *Visualization*

[‡]Cosmo Software, Computer Associates International Inc., Islandia NY, USA

Toolkit (VTK),[23] and are currently under investigation. The major advantage of VRML is of course the platform-independence of the code generated, while Open-GL or VTK would require to include the visualization tool into the fusion system itself or as a separate program.

5. CONCLUSIONS

The virtual angioscopy approach enables a better assessment of the reconstructed and calculated data in a local context, and can be used for didactical purposes as well. Data fusion of biplane angiography and intravascular ultrasound along with computational hemodynamics and interactive 3-D visualization offers a maximum of information for diagnosis and assessment of any intervention. By utilizing the standardized VRML 3-D modeling language, it is ensured that the created scenes can be distributed and viewed across platforms. The combination of external prototypes with JavaScript nodes allows the implementation of a complex graphical user interface for the endoscopic mode, while the default browser controls apply in all other views. Without obstructing the actual viewport for the scene when not needed, the user's control panel is included into the scene on demand and allows interaction with the scene in real time.

ACKNOWLEDGMENTS

This work is supported in part by grant 1 R01 HL63373-01 of the *National Institutes of Health* (NIH), Bethesda MD, USA; previous support was provided by grant Wa 1280/1-1 of the *Deutsche Forschungsgemeinschaft* (DFG), Bonn, Germany. The image data used within the scope of this study was acquired at the University of Iowa Hospital and Clinics, Iowa City IA, USA; the Department of Cardiology at the University Hospital of Essen, Germany; and the Brigham and Women's Hospital at Harvard Medical School, Boston MA, USA.

REFERENCES

1. C. von Birgelen, E. A. de Vrey, G. S. Mintz, A. Nicosia, N. Bruining, W. Li, C. J. Slager, J. R. T. C. Roelandt, P. W. Serruys, and P. J. de Feyter, "ECG-gated three-dimensional intravascular ultrasound: Feasibility and reproducibility of the automated analysis of coronary lumen and atherosclerotic plaque dimensions in humans," *Circulation* **96**, pp. 2944–2952, Nov. 1997.
2. J. H. C. Reiber, G. Koning, J. Dijkstra, A. Wahle, B. Goedhart, F. H. Sheehan, and M. Sonka, "Angiography and intravascular ultrasound," in *Handbook of Medical Imaging — Volume 2: Medical Image Processing and Analysis*, M. Sonka and J. M. Fitzpatrick, eds., pp. 711–808, SPIE Press, (Bellingham WA), 2000.
3. A. Wahle, S. C. Mitchell, S. D. Ramaswamy, K. B. Chandran, and M. Sonka, "Four-dimensional coronary morphology and computational hemodynamics," in *Medical Imaging 2001: Image Processing*, M. Sonka and K. M. Hanson, eds., vol. 4322, SPIE Proceedings, (Bellingham WA), Feb. 2001.
4. K. B. Chandran, M. J. Vonesh, A. Roy, S. Greenfield, B. Kane, R. Greene, and D. D. McPherson, "Computation of vascular flow dynamics from intravascular ultrasound images," *Medical Engineering and Physics* **18**, pp. 295–304, June 1996.
5. A. Santamarina, E. Weydahl, J. M. Siegel, and J. E. Moore, "Computational analysis of flow in a curved tube model of the coronary arteries: Effects of time-varying curvature," *Annals of Biomedical Engineering* **26**, pp. 944–954, Nov. 1998.
6. J. L. Berry, A. Santamarina, J. E. Moore, S. Roychowdhury, and W. D. Routh, "Experimental and computational flow evaluation of coronary stents," *Annals of Biomedical Engineering* **28**, pp. 386–398, Apr. 2000.
7. K. B. Chandran, "Morphologically realistic 3D reconstruction of cardiovascular segments and hemodynamic analysis," in *Proc. 11th International Conference on Mechanics in Medicine and Biology, Maui HI*, pp. 161–162, ICMMB, 2000.
8. A. Wahle, G. P. M. Prause, S. C. DeJong, and M. Sonka, "Geometrically correct 3-D reconstruction of intravascular ultrasound images by fusion with biplane angiography — methods and validation," *IEEE Transactions on Medical Imaging* **18**, pp. 686–699, Aug. 1999.
9. A. Wahle, G. P. M. Prause, C. von Birgelen, R. Erbel, and M. Sonka, "Fusion of angiography and intravascular ultrasound in-vivo: Establishing the absolute 3-D frame orientation," *IEEE Transactions on Biomedical Engineering — Biomedical Data Fusion* **46**, pp. 1176–1180, Oct. 1999.

10. A. Wahle, S. C. Mitchell, M. E. Olszewski, R. M. Long, and M. Sonka, "Accurate visualization and quantification of coronary vasculature by 3-D/4-D fusion from biplane angiography and intravascular ultrasound," in *European Biomedical Optics Week (EBiOS 2000): Biomonitoring and Endoscopy Technologies*, C. F. P. van Swol *et al.*, eds., vol. 4158, SPIE Europto, (Bellingham WA), 2000.
11. M. E. Olszewski, R. M. Long, S. C. Mitchell, A. Wahle, and M. Sonka, "Quantitative measurements in geometrically correct representation of coronary vessels in 3-D and 4-D," in *Proc. 4th IEEE Southwest Symposium on Image Analysis and Interpretation, Austin TX*, pp. 259–263, IEEE-CS Press, (Los Alamitos CA), 2000.
12. G. Wang, E. G. McFarland, B. P. Brown, and M. W. Vannier, "Gastrointestinal tract unraveling with curved cross sections," *IEEE Transactions on Medical Imaging* **17**, pp. 318–322, Apr. 1998.
13. Y. Samara, M. Fiebich, A. H. Dachman, J. K. Kuniyoshi, K. Doi, and K. R. Hoffmann, "Automated calculation of the centerline of the human colon on CT images," *Academic Radiology* **6**, pp. 352–359, June 1999.
14. K. Mori, Y. Suenaga, J. I. Toriwaki, J. I. Hasegawa, H. Anno, K. Katada, and H. Natori, "Automated display of anatomical name of bronchial branches in virtual bronchoscopy system and its application as a training tool for medical students," in *Medical Imaging 1999: Physiology and Function from Multidimensional Images*, C. Chen and A. V. Clough, eds., vol. 3660, pp. 301–312, SPIE Proceedings, (Bellingham WA), 1999.
15. G. McLennan, E. A. Hoffman, R. D. Swift, and W. E. Higgins, "Virtual bronchoscopic assessment of major airway obstructions," in *Medical Imaging 1999: Physiology and Function from Multidimensional Images*, C. Chen and A. V. Clough, eds., vol. 3660, pp. 313–320, SPIE Proceedings, (Bellingham WA), 1999.
16. J. Beier, T. Diebold, H. Vehse, G. Biamino, E. Fleck, and R. Felix, "Virtual endoscopy in the assessment of implanted aortic stents," in *Computer Assisted Radiology and Surgery (CAR '97)*, H. U. Lemke, M. W. Vannier, and K. Inamura, eds., vol. 1134 of *Excerpta Medica International Congress Series*, pp. 183–188, Elsevier, (Amsterdam), 1997.
17. J. Beier, T. Vogl, T. Rohlfing, D. Strahlendorf, N. Hidajat, and R. Felix, "Virtual endoscopy in the CT-based assessment and control of TIPS (transjugular intrahepatic portosystemic shunt)," in *Computer Assisted Radiology and Surgery (CARS '99)*, H. U. Lemke, M. W. Vannier, K. Inamura, and A. G. Farman, eds., vol. 1191 of *Excerpta Medica International Congress Series*, pp. 181–185, Elsevier, (Amsterdam), 1999.
18. A. L. Ames, D. R. Nadeau, and J. L. Moreland, *The VRML 2.0 Sourcebook*, John Wiley and Sons, New York, 2nd ed., 1997.
19. ISO/IEC 14772, "Information technology — computer graphics and image processing — the virtual reality modeling language." International Organization for Standardization, Geneva CH, 1998.
20. K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, *Robotics: Control, Sensing, Vision, and Intelligence*, CAD/CAM Robotics and Computer Vision, McGraw-Hill, New York, 1987.
21. S. C. Mitchell, A. Wahle, C. von Birgelen, R. Erbel, and M. Sonka, "Real-time visualization of coronary interventions using VRML," in *Medical Imaging 1999: Physiology and Function from Multidimensional Images*, C. Chen and A. V. Clough, eds., vol. 3660, pp. 279–287, SPIE Proceedings, (Bellingham WA), 1999.
22. A. Wahle, S. C. Mitchell, C. von Birgelen, R. Erbel, and M. Sonka, "On-site 3-D reconstruction and visualization of intravascular ultrasound based upon fusion with biplane angiography," in *Computer Assisted Radiology and Surgery (CARS '99)*, H. U. Lemke, M. W. Vannier, K. Inamura, and A. G. Farman, eds., vol. 1191 of *Excerpta Medica International Congress Series*, pp. 56–60, Elsevier, (Amsterdam), 1999.
23. W. J. Schroeder, L. S. Avila, and W. Hoffman, "Visualizing with VTK: A tutorial," *IEEE Computer Graphics and Applications — Visualization* **20**, pp. 20–27, Sept./Oct. 2000.