Interactive data visualization is today one of the key enabling data analysis tools in a wide range of disciplines in academic research as well as in commercial work flows. Visualization capitalizes on the unprecedented human capacity to analyze visual information, as well as the rapid development of computer graphics technology that enables generation of images with ever increasing quality and speed. While visual exploration of scalar and vector data visualization techniques has proven to provide a global spatial understanding, analysis of complex and intricate structures of interest can become a major challenge. In this article, we show how data exploration and analysis can be further improved by adding haptics—force feedback that represents, and is derived from, the data. This combination of sensory information input leads to the notion of visuo-haptic visualization.

We demonstrate the concept of visuo-haptics in the context of interactive exploration of the beating heart (see Figure 1). This entails a time-dependent volumetric representation of heart geometry and the blood flow therein. Typical visualizations of such data combine a 3D volume visualization of the heart’s anatomy with flow visualization methods using temporal snapshots. While well suited for a first overview, such visualizations fail to convey the highly dynamic nature of the flow in the heart. Therefore, a current trend is to move to methods specifically tailored to the analysis of unsteady flows. The basis for most of these methods is pathlines, which provide a good physical representation of the flow. However, an intuitive visualization is challenging since pathlines imply a temporal summary of the flow, and thus, the temporal context and the geometric embedding get easily lost. Adding haptics to the exploration of pathlines provides a new means to navigate through time and bridge this gap.

The application used to demonstrate the utility of our approach is exploration of spatiotemporal data—acquired using 4D flow MRI—of patient-specific heart blood flow. The visualization is implemented in the Inviwo open source framework, and the haptic force feedback uses the haptics rendering engine HAPI.
VISUO-HAPTIC VISUALIZATION

In visuo-haptic visualization, the visual rendering is enhanced with haptic feedback to provide an additional channel of information, intuitive representation of nonvisual data and physical guidance of exploration. A haptic instrument such as The Touch from 3D Systems is then used as an input or exploration device, and force feedback representing the data is provided through that same instrument. While the force feedback through the instrument is just a single force, its dynamic behavior, typically modulated at a rate of 1 kHz, can also be perceived as geometrical shapes, micro textures, or even transient vibrations from a virtual impact or event. All these effects can be carefully used and designed to convey structure and information from the volumetric data at hand.

The haptic representation of the data can be chosen to convey the same message as the visual rendering, conveying a stronger message than with either alone. Research on multisensory integration\(^2\) shows that adding a second sense enhances the overall accuracy of the perception as well as the perceived reliability of that perception. Haptics, however, can also provide its own channel of information that can convey properties unrelated to what is shown visually. For example, the scalar value of the volume might be visually rendered while the user simultaneously probes the gradient of the data through the haptic instrument. Some inherently haptic properties, such as hardness and friction, are even more intuitively represented through this sense. Note that while the sense of vision can simultaneously capture a full scene, probing the data with a haptic instrument is a local, interactive process: the user makes an action at a local position in the volume and gets local feedback to that action, representing the data at that position.

Early implementations of volume haptics applied static force representations, which are generally easy to implement. However, research has shown\(^3\) that static force mappings are less effective than geometrical representations, even in cases when they are more intuitive representations. Such geometrical representations\(^4\) of volumetric data are typically implemented by introducing an internal, virtual representation of the haptic probe, called a proxy. The proxy is moved by a control algorithm to follow the probe. The force feedback is then calculated as a spring pulling the probe toward the proxy. Geometrical representations can, depending on their implementa-
tion, define impenetrable features, which hinders exploration of the full domain. Some techniques, however, provide an optional strength property, controlling the amount of force any such geometrical feature should withstand.4

Just as there is a wide range of visual representations, different modes of haptic feedback might fit depending on the data or task at hand. For scalar data, the feedback can be as simple as a viscous drag. Viscous drag might give a sense of relative densities but fails to provide a direct notion of structures. Mapping the data gradient to force direction will pull the user toward high or low density regions if a positive or negative gain is used, respectively. The gradient-directed pull is most effectively used to find local maxima or minima, but if the structure of the scalar field needs to be conveyed then, as has already been established, a geometrical representation is preferable. Isosurface rendering is the most straightforward example of this, implemented by letting the proxy point follow an iso-alue.

For vector data the most straightforward haptic representation is to use the local vector at the haptic instrument as feedback force. For geometrical representation the follow mode can be used, providing guidance along the vectors in the data.5,6 This is easily implemented by confining the proxy motion to follow the field. A yielding follow mode, to avoid occlusion and convey strength, is implemented by allowing cross-field motion as a response to larger forces.

Other representations might also be suitable, depending on the underlying data and the application at hand. For example the guidance can be provided perpendicular to the vectors instead, creating a surface representation of the vector field3 or tube-like representations of vortices.7

HAPTIC RENDERING IN THE INVIWO FRAMEWORK

Inviwo is a modern, open-source framework for rapid visualization prototyping.8 It supports a wide range of standard visualization techniques such as direct volume rendering and various vector field visualization techniques. It is built upon a core in C++ and is easily extendable through a modular, object-oriented interface.

Inviwo Architecture

Inviwo utilizes a dataflow network graph to load and process data and to generate visualizations. Nodes in this acyclic graph are called processors and are designed to perform a specific task. Data is passed between processors in the network through ports; each processor has a set of in-ports to access data and outports to output the processed data. If a processor has no in-port, it is considered a source processor. Source processors create new data, often by loading it from disk, but it is also possible to create data directly in the processors. Similar to source processors, there are sink processors, or processors without any output. The most common sink processor is the canvas processor, which takes an image and renders it to the screen. Processor properties control the parameters used while processing the data. The properties are exposed to the user through a graphical user interface. It allows users to change parameters without recompilation or restart of the application. An example network used to visualize flow around a cuboid can be seen in Figure 2, where the gray rectangles represent processors, the small colored squares represent ports, and lines between ports represent connections. The color of the ports and connections indicates the type of data they handle.

While data flows through the network from top to bottom, events are propagated in the opposite direction. For example, when a user interacts with a visualization, the canvas processor detects it and creates a mouse event. This event is recursively propagated to processors upwards in the network through connections on the in-ports. Any processor that should act upon an event registers an event listener that catches the event when it reaches that processor and updates its parameter respectively.
Figure 2. Inviwo network used to produce images of flow around a cuboid. (a) Network consisting of
a data-loading part, in which the cuboid mesh and a volumetric dataset describing flow around the
cuboid are loaded; a data-processing part, where seed points are being generated and streamlines
are integrated through the field from the seed points; and a visual-rendering part, which renders the
streamlines as tubes around the cuboid and are finally displayed on the screen through the canvas.
(b) Resulting image. The color of the streamlines represent the velocity magnitude.
Inviwo comes with a graphical user interface built upon Qt. This interface consists of the network editor that supports constructing processor networks using visual programming. To build new applications, the user adds processors to the network by dragging and dropping from a list of available processors.

Processors in the network are connected in a similar way, by creating connections between output and input ports. In this way, algorithms can be easily reused and extended, allowing rapid experimentation with different techniques.

### Haptic Integration in Inviwo

While human vision can perceive changes as smooth with frame rates as low as 20 updates per second, the sense of touch is much more sensitive and can discriminate changes with 1,000 updates per second. In other words, in our application we need to update the haptic feedback once every millisecond while the graphics might take up to 50 ms to update. Hence, we cannot just include haptic rendering in our graphics rendering loop. Therefore, we created a separate thread for each haptic device that runs independent of the graphics thread.

To integrate haptics into Inviwo and its processor network, we have introduced a new port type, called haptic effect, and a new sink processor. When this sink processor is added to the network, it initializes the haptic device and the asynchronous haptic thread. The processor has a single input port that accepts an arbitrary number of haptic effects. New data pushed to this port are transferred to the haptic thread. In contrast to regular processors, processors that output a haptic effect do not do the actual processing, but rather create instances of a haptic effect that are passed to their respective output port. This haptic effect object contains all the information needed to calculate its haptic forces, which is later used when the object has been transferred to the haptic thread. A processor network that combines haptics and visual rendering and its resulting visualization can be seen in Figure 3.

In Inviwo, graphical rendering is done by evaluating the network. This is done lazily, meaning only when needed, for example when a property has changed or after interaction events have been handled. Therefore, the haptic thread must create haptic interaction events that are propagated through the network to update the graphics processors about the status of the haptic device. These events contain all information about the device, including its position, orientation, velocity, and active forces. Since the frame rates of the haptic rendering are much higher than what the graphics can handle, events should not be sent for every haptic interface update. Instead, the haptic thread has a clock keeping track of the time since the last event and only creates and propagates a new event if this time exceeds a certain value. The time between these events can be controlled by the user through a property, and is set to 20 ms by default.

### Haptic Modes in Inviwo

Inviwo has been equipped with a set of haptic modes, based on the initial needs in visualization of vector volumes. The first is **force mode**, where the vector field is sampled at the location of the haptic probe and directly mapped to the direction and strength of the applied force.

The second is **follow mode**, where a resistance is provided when the haptic probe is moved perpendicular to the vector field. This resistance guides the user to move the probe freely with the field, which makes it suitable for exploring features and structures in the field.
One of the most common ways to visually represent vector fields are integral lines, for example, streamlines or pathlines. These integral lines are always tangential to the underlying field. When seeded in the location of the haptic interface, the resulting integral line will represent the path on which the haptic interface would move freely when in force mode or follow mode. Hence, the use of integral lines together with haptic force or follow mode will stimulate both the visual and touch senses.
While the follow mode prevents movements perpendicular to the field, they can still be achieved by applying a large enough force. Thereby the integral line that is followed is changed. In some applications this is desired, while in others a specific integral line or trajectory shell should be explored. For such cases, we introduce an additional mode, the line follow mode, which follows one selected integral line (see Figure 3b). Upon activation of this mode, the haptic interface is attached to a proxy point on the current integral line. When the haptic interface is moved, the location of the proxy position is updated by moving it along the integral line toward the haptic interface. A force pulls the device back towards the direction of the proxy position with a strength proportional to the distance between the proxy position and the haptic interface. Hence, when the haptic interface is located directly on the integral line, and when it is moved along the line, no force is applied. When the haptic interface is moved away from the line, it will be pushed back toward it. An example of this mode can be seen in Figure 3b, in which the underlying vector fields describe the flow around a cuboid. The red arrow represents the force currently rendered to the haptic interface.

**PROOF-OF-CONCEPT: BLOOD-FLOW EXPLORATION**

Cardiovascular diseases are the leading cause of death worldwide. In this light, much effort is put into understanding heart diseases, such as cardiovascular disorders, and their impact on the heart’s hemodynamics. Advanced imaging techniques, like 4D Flow MRI, make it possible to directly measure the blood flow in the beating heart, which opens many new opportunities for diagnostics.

**4D Flow MRI**

In this proof-of-concept, all examples used a dataset acquired with a clinical 3T Philips Ingenia scanner. During acquisition, the subject spent 10–15 min in the scanner, which continuously captures data. The captured data was processed to construct a spatiotemporal flow field with complete volumetric coverage over a full cardiac cycle. Besides the flow field, other related quantities, such as the turbulent kinetic energy (TKE) and regular MR-images were acquired. Additionally, for each dataset a segmented heart geometry consisting of its chambers and vessels was provided. It was given as a time-dependent mask that was used to filter out noise outside of the heart.

**Visual Representation**

Typically flow data is visualized using pathlines or streamlines. While streamlines represent the flow for a given time step, pathlines approximate particle traces advected with the flow. When analyzing cardiovascular flows, which are highly dynamic, pathlines are in general preferred over streamlines. However, such a visualization can be challenging to interpret. Since pathlines represent a long-time exposure picture, the context of time can easily be lost. Therefore, time plays multiple roles: seeding time, integration period, and parameter for the lines. Thus, it is not clear how to integrate time-dependent context information into a pathline visualization. This especially concerns the geometry of the beating heart, which is in constant motion that varies in both volume and location during contraction and relaxation (see Figure 4). A common approach is to render the geometry of one selected time step, for example, for the largest geometry of the heart. However, each point in a pathline represents different time steps and the visual registration between pathline and geometry is only true for a single point on the pathline. Haptic guidance during exploration provides an intuitive means to communicate time and other time-varying contexts. Thereby, we use the pathlines to navigate through time to explore dynamic features.
Visuo-Haptic Exploration

We use the haptic device in two modes for blood flow exploration: seeding of pathlines, and navigation along pathlines for exploration.

The basis for the seeding interaction is a rendering of the heart geometry at a specific, fixed time. While exploring the space with the device, for each frame, a pathline is integrated and drawn at the location of the haptic interface. This supports a smooth exploration of the space while keeping the seeding time fixed. When the user locates a pathline of interest, the device is switched to line follow mode. In this mode, time is directly linked to the location on the pathline. Thus, moving along the pathline also changes the heart geometry correspondingly. Technically this is realized using a proxy following the pathline returning the current time parameter. This enables the user to get an improved understanding of the relation between the time component of the pathline to the deforming heart geometry. In Figures 1 and 5, we can see examples of this mode being used to explore the left ventricle and the aorta.

To strengthen the temporal context information, we further augment the pathline in the vicinity of the proxy location with a set of short streamlines (Figure 5b). This can aid the user in gaining insight about the field at that location. A strong deviation of the streamlines from the pathlines indicates a strong acceleration of the flow at this position.

Figure 4. The cardiac cycle provides an important temporal context for blood flow analysis. It can be represented by using heart geometry as a reference. The image shows a diagram of the ventricle volume over two cardiac cycles, together with the geometry rendered at the isovolumetric relaxation and contraction (that is, smallest and largest ventricular volumes).
Since the time stamp in relation to the cardiac cycle is essential for the understanding of the flow, the proxy position is marked in a diagram of the cardiac cycle. This plot shows the currently selected time step as a vertical bar overlaying a line plot representing the ventricular volume, which is an essential property commonly shown in cardiac cycle diagrams. In Figure 6, we see the visualization for six different proxy locations along one pathline, together with the streamlines around each of the positions and the time step marked in the heart-cycle diagram.

Figure 6. From left to right, the heart geometry changes when moving the haptic interface along the pathline, starting in the early diastole (ventricular filling) and ending in the mid systole (ventricular contraction). In each image, the current geometry is shown as a red surface and the bounding geometry of all time steps as a gray surface for context. The current vector field in the vicinity of the haptic interface is highlighted using short streamlines. The visuo-haptic exploration of the pathline begins at the mitral valve in (a) and is guided into the ventricle during the diastole in (b) and (c). During the diastole we see the heart expanding as more blood is being pushed into the ventricle, reaching the maximum in (d), representing the isovolumetric contraction. In (e), we have entered systole, the phase where the ventricle starts to contract and pushes the blood out of the ventricle and into the aorta through the aortic valve. The pathline reaches its max velocity in (f), where it has just left the ventricle and enters the aorta.

Visuo-Haptic Filtering

The exploration of one pathline provides only a small part of the flow inside the heart. However, the representation of a complete set of pathlines entering the heart during one time step leads to
clutter and occlusion. Typically, a set of filtering options can improve the visualization essentially. Instead we provide an extended seeding option that renders pathline bundles exhibiting similar behavior.

A first step towards this feature is the implementation of a cleanup function to discard unreasonable pathlines. A property of 4D Flow MRI is that it has a relatively low spatial resolution and a low signal-to-noise ratio. This, together with the constantly deforming geometry, leads to a high probability of particles being integrated through the wall. Such pathlines do not describe the real blood flow and should preferably be avoided. To address this issue, we use the fact that blood enters and exits the various chambers of the heart only through valves. For example, for the left ventricle, blood enters through the mitral valve during the diastole (relaxation) and leaves through the aortic valve during systole (contraction). By exploiting this knowledge, we can filter out pathlines that either enter or exit the heart at an unexpected time and location. However, the data does not provide the exact position of the valves, so it must be estimated. This is done by a density analysis of the entry and exit points of the pathlines. Therefore, pathlines are seeded densely in the ventricle during isovolumetric contraction, and the time steps of the cardiac cycle in which the ventricle has the largest volume are traced forward and backward until they leave the ventricle (see Figure 7). In Figure 7b, we can see all traced pathlines; in Figure 7a, we see the start and endpoint of each pathline, with green and purple points indicating valid and invalid entry points, respectively, and red and blue indicating valid and invalid exit points, respectively. In Figure 7c, we see the remaining pathlines after discarding the lines that have either an invalid entry or exit point.

The cleaned results shown in Figure 7c are, however, still cluttered, and important flow features are hidden due to occlusion. Traditional visualization approaches could here include filtering and clustering, but with the aid of the haptic interface, we prefer to provide a guided exploration. Similar to the visuo-haptic exploration explained above, this approach utilizes both the follow mode and the line follow mode. In the follow mode, the user is guided through the flow, but instead of tracing a pathline at the location of the haptic interface, we select a pathline bundle close to the pen. The number of pathlines rendered is user controllable. This can be done either by determining a radius $r$ of a sphere around the haptic interface or a number $N$ of nearest pathlines. If $N$ is set to one, we will always see the nearest pathline. It is always possible to switch to the line follow mode when the user wants to explore a specific pathline, and the haptic interface will follow that pathline. The initial position of the proxy is selected as the point on the pathline that is closest to the haptic interface upon activation. Once active, this works exactly the same as in the visuo-haptic exploration mode.

Figure 7. Due to the noise in the measured flow, it is important to filter out invalid pathlines. This is done by evaluating the pathline’s entry and exit points. (a) All of the pathlines seeded in each voxel in the left ventricle during isovolumetric contraction. (b) Respective entry and exit points are visualized as point clouds. These points can be classified as valid or invalid by considering their proximity to the valves. Red and blue represent valid and invalid exit points, respectively, while green and purple represent valid and invalid entry points, respectively. (c) The pathlines remaining after pathlines with either an invalid entry or exit point are discarded. These lines enter through the mitral valve, traverse the ventricle, and exit into the aorta through the semilunar valves.
CONCLUSIONS

Our application demonstrates how data exploration can benefit from adding haptic representations in the visualization workflow. In the case of blood flow analysis of a beating heart, haptics not only enhances the augmented visual images, but it also adds new modes of exploration. The navigation along pathlines adds time as a new dimension to the exploration space and renders the concept of pathlines more graspable. However, so far, our application only uses a limited set of possible haptics modes. It is clear that haptic data exploration has much to offer, and further work is needed to explore its potential. An interesting example is friction that can be applied to convey additional scalar quantities from the underlying field, such as velocity or turbulent kinetic energy. We further plan to use the deformation of the geometry heart to guide the exploration. Adding haptic rendering into the framework Inviwo also demonstrates a straightforward way to integrate haptics into an existing open source platform. Besides the extension of the application, we also see the need to extensively evaluate the gain from using haptics for flow field visualization. Therefore, we will invite cardiologists or other users to explore their data with our framework and will use their feedback to guide further developments. While our work was inspired specifically by blood flow exploration, it is extensible to other applications.

ACKNOWLEDGMENTS

This work was supported in part by the Swedish Research Council, the Excellence Center at Linköping and Lund in Information Technology, the Swedish e-Science Research Centre, and the Knut and Alice Wallenberg Foundation through the project Seeing Organ Function. The described concepts have been realized using the Inviwo visualization framework (www.inviwo.org). The authors would like to thank Tino Ebbers and Mariana Bustamante at the Center for Medical Image Science and Visualization—Linköping University for their cooperation and for supplying us with 4D Flow MRI datasets.

REFERENCES

ABOUT THE AUTHORS

Rickard Englund is a PhD student in scientific visualization at Linköping University. His research focuses on visual data analysis and vector field visualization in particular. Contact him at rickard.englund@liu.se.

Karljohan Lundin Palmerius is a senior lecturer in visualization and media technology at Linköping University. His research interests include immersive visualization and virtual reality. Palmerius received a PhD in direct volume haptics from Linköping University. Contact him at karljohan.lundin.palmerius@liu.se.

Ingrid Hotz is a professor of scientific visualization and heads the scientific visualization group at Linköping University. Her research interests include vector field and tensor field visualization. Hotz received a PhD in computer science from the University of Kaiserslauter. Contact her at ingrid.hotz@liu.se.

Anders Ynnerman is a professor of scientific visualization, director of the Norrköping Visualization center, and head of the research division in media and information technology at Linköping University. Over the past 15 years, one of his research foci has been medical volume rendering. Ynnerman received a PhD in physics from the University of Gothenburg. Contact him at anders.ynnerman@liu.se.