Fast and High Precision Volume Haptics

Karljohan Lundin Palmerius
Norrköping Visualization and Interaction Studio
Linköping University, Sweden
E-mail: karlu@itn.liu.se

Abstract

Volume haptics has shown itself an effective way of enhancing precision and speed in interaction with medical or scientific visualization. This paper presents a mixed solver approach for the primitive-based volume haptics problem, to provide the best performance for every volume haptics application. For situations where the constraints are orthogonal we present a fast and high precision analytical solver. For complex configurations requiring support for non-orthogonal constraints we allow for fall-back on a numerical solver. The results show significantly improved performance with the analytical solver, allowing for higher stiffness and thus feedback of higher quality, while still allowing for transparent support for non-orthogonal constraints.

1 Introduction

Methods for producing haptic feedback from volumetric data have shown themselves to be a powerful addition to volume visualization in exploratory tasks[5, 13, 10]. The feedback can both provide physical guidance to help the user find a feature, and generate haptic representations of important information in the data, thus providing better understanding.

The first method available for interaction with volumetric data was the force functions-based approach[5, 12, 1, 2, 4, 6]. With this method the force is expressed as a vector-valued function of the volumetric data. It is easy to implement and therefore popular, but representing features as forces of varying strength and direction can, in some cases, be too simplistic an approach. Also, force functions add energy to the haptic control system, which provokes unstable behaviour. The constraint-based methods[11, 14, 10, 3] avoid this by applying a decoupling scheme where each haptic instrument, or probe, is internally represented by a proxy point describing the point of interaction in a stable and predictable manner. This proxy is used to simulate passive constraints, representing the volumetric data, throughout the volume. By letting the constraint yield to a material specific force, occlusion is avoided and a continuous representation of the data can be presented, avoiding the need for the user to explicitly select the region to probe.

The functionality of force functions and constraints have been integrated in a single framework by the introduction of haptic primitives in [9]. The haptic primitives form both an abstraction layer for implementing haptic modes and a way of removing a fundamental limitation of earlier methods[8]. The approach allows for free selection, configuration and combination of haptic modes and intuitive and effective means for implementing new modes. A core concept in this approach is the force balancing, the minimization of a vector-valued function.

This paper presents a mixed solver approach for minimizing the force balancing equation, to provide the best performance for every volume haptics application. For the common situation where all constraints are orthogonal we present a fast and high precision analytical solver. For complex configurations requiring support for non-orthogonal constraint we allow for fall-back on a numerical solver. The following section describes the important principles of the primitives-based approach. In section 3 the analytical solver is presented and section 4 then gives an account of the numerical solver. The results and performance are presented in section 5.

2 Primitives-based Volume Haptics

The primitives-based approach to volume haptics forms a hierarchical structure where low-level haptic primitives are used as an abstraction of general haptic effects, and haptic modes use these primitives to form haptic interaction schemes. The modes implemented using these primitives can then be used to guide users through exploration of scientific or medical data, or even to generate haptic feedback from surgery simulators.

2.1 Haptic Primitives

Haptic constraints are in this approach represented using primitives of one, two and three degrees of freedom: *plane*, *line* and *point*, respectively. Active forces and other force functions are included through a fourth primitive: *directed force*. A wide range of haptic effects can be generated by simply superposition selected primitives from this set.

The approach is proxy-based, meaning that an internal proxy point, $\vec{x}_{\text{proxy}}$, is used to represent the interaction point, the point of contact, and to control the haptic behaviour. The force feedback is then calculated using the coupling equation

$$\vec{f} = -k\left(\vec{x}_{\text{probe}} - \vec{x}_{\text{proxy}}\right) \tag{1}$$

Each haptic primitive is characterized by a simple vector-valued force function of this proxy position as a part of the process of finding the proxy position for the new time-frame. The primitives have the parameters strength, s, position, $\vec{x}$, and direction, a unit vector $\vec{q}$. With $\vec{e}$ defined as the displacement of the primitive relative the proxy, $\vec{e} = \vec{x}_i - \vec{x}_{\text{proxy}}$, the force functions are defined as:

- Directed force, a position-independent force:

$$\vec{F}_i\left(\vec{x}_{\text{proxy}}\right) = s_i\vec{q}_i \tag{2}$$

- Point, an attractor to a point in space:

$$\vec{F}_i\left(\vec{x}_{\text{proxy}}\right) = \begin{cases} \vec{0}, & \text{if } |\vec{e}| = 0 \\ s_i\frac{\vec{e}}{|\vec{e}|}, & \text{if } |\vec{e}| \neq 0 \end{cases} \tag{3}$$

- Line, an attractor towards the closest point on a line:

$$\vec{F}_i\left(\vec{x}_{\text{proxy}}\right) = \begin{cases} \vec{0}, & \text{if } |\vec{m}| = 0 \\ s_i\frac{\vec{m}}{|\vec{m}|}, & \text{if } |\vec{m}| \neq 0 \end{cases} \tag{4}$$

  where $\vec{m}$ is the vector to the closest point on the line defined by $\vec{x}$ and $\vec{q}$,

$$\vec{m} = \vec{e} - \vec{q}(\vec{q}\cdot\vec{e}) \tag{5}$$

- Plane, a directed force which exists only on one side of the plane defined by $\vec{x}$ and $\vec{q}$:

$$\vec{F}_i\left(\vec{x}_{\text{proxy}}\right) = \begin{cases} 0, & \text{if } \vec{e}\cdot\vec{q}_i \leq 0 \\ s_i\vec{q}_i, & \text{if } \vec{e}\cdot\vec{q}_i > 0 \end{cases} \tag{6}$$

The proxy position for each time frame is then found by balancing the force feedback, $\vec{f}$, from the coupling equation against the force from the primitives, by minimizing the residual $\vec{\varepsilon}$ in

$$\vec{\varepsilon} = -\vec{f}\left(\vec{x}_{\text{proxy}}\right) + \sum_i \vec{F}_i\left(\vec{x}_{\text{proxy}}\right) \tag{7}$$

with respect to $\vec{x}_{\text{proxy}}$. All primitive parameters (position, strength and orientation) are constant when estimating $\vec{\varepsilon}$.

To simplify the expressions of $\vec{\varepsilon}$ we let $\odot^s_{\vec{x}}$, $\oslash^s_{\vec{q},\vec{x}}$, $\oplus^s_{\vec{q},\vec{x}}$ and $\ominus^s_{\vec{q}}$ represent the force functions for the point, line, plane and directed force primitives, respectively.

2.2 Haptic Modes

Haptic modes for volume exploration are implemented by selecting haptic primitives and controlling their parameters, such as the orientation, as functions of the local data at the proxy position, $V(\vec{x}_{\text{proxy}})$. The primitives are placed at the location of the proxy point from the previous time-frame, so that the primitive generates the local haptic shape at the probed position in the volume. This produces a haptic representation of the data at any position in the volume. Here we show three examples of haptic modes for vector data and one example of how these can be combined. Observe that these are just three examples of the wide range of both possible and readily available haptic modes. For a more comprehensive list of haptic modes, see [7].

Force mode The force mode pushes the haptic instrument in the direction of the underlying vector field. It is easily simulated using a force primitive with orientation specified by the local field value, $\vec{V}(\vec{x}_{\text{proxy}})$. The strength of the primitive is, to allow for extra control, controlled using a transfer function, τ_{force} ($\tau : \mathcal{R} \rightarrow \mathcal{R}$), of the magnitude of the vector value, $\tau_{\text{force}}(\left|\vec{V}(\vec{x}_{\text{proxy}})\right|)$. The residual to minimize is then expressed as

$$\vec{\varepsilon} = -\vec{f}(\vec{x}'_{\text{proxy}}) + \ominus^{s=\tau_{\text{force}}(\left|\vec{V}(\vec{x}_{\text{proxy}})\right|)}_{\vec{q}=\frac{\vec{V}(\vec{x}_{\text{proxy}})}{|\vec{V}(\vec{x}_{\text{proxy}})|}}(\vec{x}'_{\text{proxy}}) \tag{8}$$

with respect to the new proxy position, $\vec{x}'_{\text{proxy}}$.

Follow mode The follow mode generates an anisotropic 3D friction, so that moving the haptic instrument perpendicular to the vector field produces a resistance that requires a certain force to overcome. This makes it easy to follow the flow of the field, and provides a sense of the strength property through the feeling of resistance. The follow mode is implemented using a line primitive with the orientation parameter set to the normalized local vector of the data, $\vec{V}(\vec{x}_{\text{proxy}})$. Also in this mode the strength is controlled through a transfer function, τ_{follow}, from the magnitude of the vector value, $\tau_{\text{follow}}(\left|\vec{V}(\vec{x}_{\text{proxy}})\right|)$. The residual to minimize for this mode is then expressed as

$$\vec{\varepsilon} = -\vec{f}(\vec{x}'_{\text{proxy}}) + \oslash^{s=\tau_{\text{follow}}(\left|\vec{V}(\vec{x}_{\text{proxy}})\right|)}_{\vec{x}=\vec{x}_{\text{proxy}},\ \vec{q}=\frac{\vec{V}(\vec{x}_{\text{proxy}})}{|\vec{V}(\vec{x}_{\text{proxy}})|}}(\vec{x}'_{\text{proxy}}) \tag{9}$$

Surface mode To generate a surface-like feedback from scalar data we use a plane primitive with the orientation set to the gradient of the data. The primitive strength is controlled through a transfer function from the magnitude of the gradient, $\tau_{\text{surface}}(\left|\vec{\nabla} V(\vec{x}_{\text{proxy}})\right|)$. The balancing equation for this mode is then

$$\vec{\varepsilon} = -\vec{f}(\vec{x}'_{\text{proxy}}) + \mathop{\bigcirc\!\!\!\!\parallel}_{\vec{x}=\vec{x}_{\text{proxy}},\, \vec{q}=\frac{\vec{\nabla} V(\vec{x}_{\text{proxy}})}{|\vec{\nabla} V(\vec{x}_{\text{proxy}})|}}^{s=\tau_{\text{surface}}(\left|\vec{\nabla} V(\vec{x}_{\text{proxy}})\right|)} (\vec{x}'_{\text{proxy}}) \quad (10)$$

Combined modes When two or more modes are used simultaneously their individual force function contributions are combined linearly, as expressed by equation 7. The combined residual to minimize for the force and follow modes above becomes

$$\begin{aligned} \vec{\varepsilon} = -\vec{f}(\vec{x}'_{\text{proxy}}) \quad &+ \quad \mathop{\bigcirc\!\!\!\!\!\Rightarrow}_{\vec{q}=\frac{\vec{V}(\vec{x}_{\text{proxy}})}{|\vec{V}(\vec{x}_{\text{proxy}})|}}^{s=\tau_{\text{force}}(\left|\vec{V}(\vec{x}_{\text{proxy}})\right|)} (\vec{x}'_{\text{proxy}}) \\ &+ \quad \mathop{\oslash}_{\vec{x}=\vec{x}_{\text{proxy}},\, \vec{q}=\frac{\vec{V}(\vec{x}_{\text{proxy}})}{|\vec{V}(\vec{x}_{\text{proxy}})|}}^{s=\tau_{\text{follow}}(\left|\vec{V}(\vec{x}_{\text{proxy}})\right|)} (\vec{x}'_{\text{proxy}}) \end{aligned} \quad (11)$$

3 Analytical Solver

We have designed an analytical method for solving the balancing equation (equation 7) and so find the position of the proxy. This solver makes use of the common situation where haptic primitives, co-located at a position, $\vec{x}_{\text{pp}}$, are in configurations that produces orthogonal constraints. The situation is then similar to that of earlier constraint-based methods[11, 14, 10, 3]. These general prerequisites for this solver are frequently fulfilled since

1. all primitives are generally placed at the old proxy position to generate a shape representation of the data at the currently touched position
2. the orientation of the primitives are generally controlled by the same data and sometimes even from the same feature in the data.

An example of this is shown in the combination of the force mode and the follow mode, equation 11, on the same vector data. The orthogonality requirement is not complete, though, as there are other special but strictly defined allowed situations, similar to the viscosity feedback in [11]. The exact requirements for each haptic primitive are described in the following section.

The analytical solver is based on iterative movements the proxy point in accordance with the haptic primitives in turn. An example of the proxy movements with two haptic primitives is shown in figure 1. During these iterations the proxy position represents the force exerted by the applied haptic

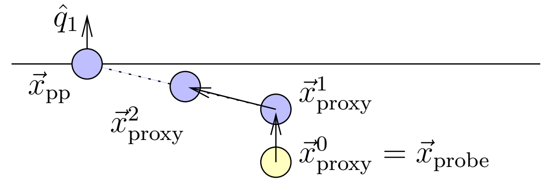


Figure 1. The proxy movements from first a plane primitive and then a point primitive.

primitives, onto the currently processed primitive. Thus, the proxy is initially placed at the probe position, which represents a zero force feedback through the coupling equation (equation 1). The haptic primitives are then applied in the following order.

3.1 Proxy Movements

Force primitives The first haptic primitives to be applied are the force primitives. These must be applied first, so that any primitive that represents a constraint can modify the forces applied by these force primitives. The order in which the constraint primitives are applied, however, is not important, but the order used here is more convenient than many alternatives.

For each force primitive the proxy point is moved according to

$$\vec{x}^{n+1}_{\text{proxy}} = \vec{x}^{n}_{\text{proxy}} + \frac{s_i}{k}\vec{q}_i \quad (12)$$

and so ends up in a position that through the coupling equation represents the sum of the individual force vectors of these primitives. Since the force field of the force primitive is linear there are no restrictions on how the primitives are configured or in which internal order they are applied.

Plane primitives The proxy position now represents the force from the force primitives, and the proxy is moved to simulate the plane primitives. The proxy is moved towards the plane defined by the primitive, but never past it, so the proxy movement is defined by

$$\vec{x}^{n+1}_{\text{proxy}} = \vec{x}^{n}_{\text{proxy}} + \vec{q}_i \min\left(s_i/k,\ \vec{q}_i \cdot \vec{e}\right) \quad (13)$$

where $\vec{e}$ is the primitive position relative to the current proxy position, $\vec{e} = \vec{x}_i - \vec{x}^n_{\text{proxy}}$. Also, since the plane primitive is single sided, this is only applied if the proxy currently is on the right side of the plane, that is if $\vec{q}_i \cdot \vec{e} > 0$.

The plane primitives may be applied in any order, however all active plane primitives must be either mutually orthogonal, a proof is provided in [8], or coincide, either in the same direction or opposing. In the case of orthogonal plane primitives, the proxy movements of the primitives are

linearly independent. Two co-directional plane primitives are merged into a single primitive by adding their strengths, and two opposing plane primitives will never act simultaneously, since a plane primitive is only active if the proxy is on the right side of the plane.

Observe that with support for the force primitive, it is not known if a plane primitive is active until all force primitives have been applied — a force primitive may push the proxy to the other side of the plane and so deactivate a previously active plane or vice versa. This is analogous with a flow pushing the probe against the surface of an object — the flow and surface forces may then cancel each other — or a flow pushing the probe out of an object, the surface will then not contribute to the force feedback.

Line primitives When the proxy has been moved to represent the force and plane primitives, the proxy is moved to generate feedback from the line primitives. The proxy is moved towards the line defined by the primitive position and orientation,

$$\vec{m} = \vec{e} - \vec{q}_i\,(\vec{q}_i \cdot \vec{e}) \tag{14}$$

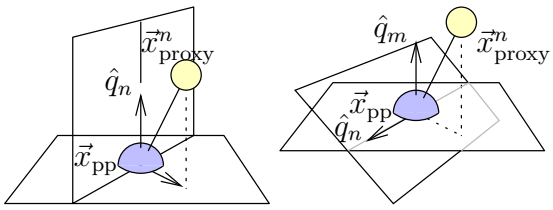
$$\vec{x}^{\,n+1}_{\text{proxy}} = \vec{x}^{\,n}_{\text{proxy}} + \frac{\vec{m}\min\left(s_i/k,\ |\vec{m}|\right)}{|\vec{m}|} \tag{15}$$

The line primitive generates a constraint facing the proxy, in a manner similar to that used to implement the follow mode in [3, 10], see figure 2. Thus, to guarantee that only orthogonal constraints are generated, all concurrently active line primitives must be parallel. Such parallel line primitives are then merged into a single primitive by summarizing their individual strengths. The line primitives must also be orthogonal to any plane primitives, to guarantee orthogonal constraints, see figure 2(a). Since the line primitive always moves the proxy towards the common primitive position, $\vec{x}_{\text{pp}}$, in the plane defined by the current the proxy position and the line orientation, $\vec{q}_i$, the line primitive may also be parallel to the plane primitives, see figure 2(b). A proof that shows the validity of non-orthogonal constraints under these premises is provided section 3.2.

Point primitives Last, the proxy point is moved to generate feedback from point primitives. Since all primitives are co-located and point primitives are without orientation, all point primitives can be merged by summarizing their individual strengths. The proxy is then moved towards the primitive position,

$$\vec{x}^{\,n+1}_{\text{proxy}} = \vec{x}^{\,n}_{\text{proxy}} + \frac{\vec{e}\min\left(s_i/k,\ |\vec{e}|\right)}{|\vec{e}|} \tag{16}$$

The iterative movements of the proxy according to the equations provided above produces a proxy position theoretically minimizing the balancing equation. Any differences



(a) The line constraint faces the proxy, so it is always orthogonal to a plane orthogonal to the line.

(b) When the line primitive lies in a plane primitive, the proxy is never moved past either constraint.

Figure 2. The two allowed line/plane primitive configurations.

are due to numerical errors in the floating point arithmetics. The necessary proof is provided below.

3.2 Proof of Correct Force Balancing

The success of this algorithm relies on the principle that iterative ordered application of the primitives actually minimizes the balancing equation (equation 7). Below follows a proof for the special case where one plane primitive, with strength s_1 and orientation $\vec{q}_1$, and one point primitive, with strength s_2, both located at $\vec{x}_{\text{pp}}$, produces a proxy position that is in free balance outside their respective constraint, see figure 1. Other configurations can be seen as special subsets of this case. For example, the cases where the proxy ends up at one or more constraints makes the proxy locked to the primitive position in the constrained dimension, which produces a trivial solution in that dimension. In the combination of multiple orthogonal plane primitives, or the line/plane primitive case illustrated in figure 2(a), all proxy movements are linearly independent. Also, the line/plane primitives example illustrated in figure 2(b) is identical to this example in the 2D subspace defined by the orientation of the line primitive.

Iteratively moving the proxy position according to first the plane primitive, from $\vec{x}_{\text{probe}}$ to $\vec{x}^{\,1}_{\text{proxy}}$, and then the point primitive, from $\vec{x}^{\,1}_{\text{proxy}}$ to $\vec{x}^{\,2}_{\text{proxy}}$ provides these equations:

$$\vec{x}^{\,1}_{\text{proxy}} = \vec{x}_{\text{probe}} + \frac{s_1}{k}\vec{q}_1 \tag{17}$$

$$\vec{x}^{\,2}_{\text{proxy}} = \vec{x}^{\,1}_{\text{proxy}} + \frac{s_2}{k}\frac{\vec{x}_{\text{pp}} - \vec{x}^{\,1}_{\text{proxy}}}{\left|\vec{x}_{\text{pp}} - \vec{x}^{\,1}_{\text{proxy}}\right|} \tag{18}$$

and using the resulting position, $\vec{x}^{\,2}_{\text{proxy}}$, to calculate the residual gives us

$$\vec{\varepsilon} = -k\left(\vec{x}^{\,2}_{\text{proxy}} - \vec{x}_{\text{probe}}\right) + s_1\vec{q}_1 + s_2\frac{\vec{x}_{\text{pp}} - \vec{x}^{\,2}_{\text{proxy}}}{\left|\vec{x}_{\text{pp}} - \vec{x}^{\,2}_{\text{proxy}}\right|} \tag{19}$$

By putting together the different parts and simplifying we get, with $\alpha = \vec{x}_{\text{pp}} - \vec{x}_{\text{probe}} - \frac{s_1}{k}\vec{q}_1$,

$$\vec{\varepsilon} = -s_2\frac{\vec{\alpha}}{|\vec{\alpha}|} + s_2\frac{\vec{\alpha} + \frac{s_2}{k}\frac{\vec{\alpha}}{|\vec{\alpha}|}}{\left|\vec{\alpha} + \frac{s_2}{k}\frac{\vec{\alpha}}{|\vec{\alpha}|}\right|} \quad (20)$$

$$= -s_2\frac{\vec{\alpha}}{|\vec{\alpha}|} + s_2\frac{\left(1 + \frac{s_2}{k|\vec{\alpha}|}\right)}{\left(1 + \frac{s_2}{k|\vec{\alpha}|}\right)}\frac{\vec{\alpha}}{|\vec{\alpha}|} = \vec{0} \quad (21)$$

Thus, the iterative proxy movements produces a correct minimization of the balancing equation.

4 Numerical Solver

If any of the requirements for the analytical solver is not fulfilled, the solver fails and the system needs to fall back on a more general solver that is capable of handling any combination of haptic primitives, even non-orthogonal configurations. The force balancing equation can not be easily solved analytically, as is shown in [8], but there are several available numerical methods for finding the solution to similar problems, such as the Nelder-Mead Simplex Method. The demands on high precision, to minimize accumulated errors, however, narrows the number of alternatives. We have chosen an approach based on steepest descent.

The steepest descent optimization approach iterates towards the solution with a direction defined by the gradient of the function, which is not available in equation 7. Since the equation describes the balancing of forces, however, the residual vector describes the direction towards a lower residual magnitude and is thus the direction for steepest descent. Thus, the proxy position from the previous haptic frame is used as initial position and the iteration towards the solution is expressed as

$$\vec{x}^{n+1}_{\text{proxy}} = \vec{x}^{n}_{\text{proxy}} + l\,\vec{\varepsilon}/|\vec{\varepsilon}| \quad (22)$$

where l is the current step length.

Since the formula for $\vec{\varepsilon}$ is not linear, the iterations can not be assumed to proceed directly towards the solution. In fact, the path towards the solution in a setup with any constraint is often a zigzag line, which complicates the specification of the step length. We use the following heuristics to allow for adjusting the step length during the iteration towards the solution. First the step length is initialized to a constant value that works as a representative for the local space, currently 1 mm. This value is then used until the intermediate values turn 90° or more, or the precision is too low for accurate estimation of the angle, that is

$$\vec{\varepsilon}_n \cdot \vec{\varepsilon}_{n-1} < \epsilon \quad (23)$$

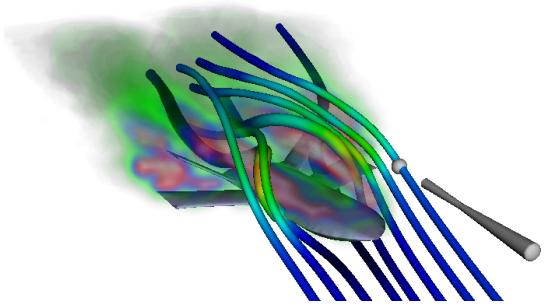


Figure 3. A screen shot from the virtual wind tunnel setup.

where ϵ is the machine epsilon. When this occurs the step length is lowered by a reduction value, empirically optimized to 0.7. The loop is terminated when the step length becomes lower than ϵ.

This approach converges and provides a solution that gives consistent and smooth haptic feedback. Many operations, however, are performed close to the precision limit of the floating point numbers used by the software, however, which makes the result only accurate within a certain limit. The discontinuities of the balancing equation causes the convergence to be slow compared to the analytical solver, which put extra strain on the CPU. Values on timings and precision are listed in the results section.

5 Results

Both solvers have been integrated in the Volume Haptics Toolkit (VHTK)[7]. These solvers constitute a priority chain so that the analytical solver is tried first but if it fails the system can fall back on the numerical solver. This also allows for additional special cases to be handled by individual solvers and allows the deactivation of a solver in case it is known not to be able to provide a correct solution.

The solver chain has been tested on a virtual wind tunnel setup, see figure 3. In this setup we first use the follow mode and the force mode on the flow data, a configuration is supported by the analytical solver. The follow mode provides guidance and information about the orientation and strength of the flow, and the force mode pushes the probe in the direction of the flow and so provides a representation of the direction of the flow. We then add the surface mode to provide feedback from the pressure data of the wind-tunnel simulation. The combination of these three modes with these two data sets does not generate only orthogonal constraints, so the analytical solver fails. The system then automatically falls back on the numerical solver and so successfully generates a correct solution and consistent force feedback.

	Time	Residual
Numerical Solver	71 μs	$38.0 \cdot 10^{-6}$ m
Analytical Solver	11 μs	$10.7 \cdot 10^{-14}$ m

Table 1. The time needed for the two solvers.

To compare the precision of the solvers we measure the residual magnitude when the solver returns the resulting proxy position. This value is not supposed to be zero (just minimized) except for the case where the proxy is in balance, not following a constraint. Thusly, the residual in free balance is what is tested here. The residual magnitude for the analytical solver is always below $3 \cdot 10^{-13}$ during the test, while the numerical iterative descent-based solver yields a residual between $8 \cdot 10^{-8}$ and $9 \cdot 10^{-5}$. The machine epsilon is, in comparison, $2 \cdot 10^{-16}$. The mean residual magnitude for the two solvers are listed in table 1.

This significantly increased precision of the analytical solver allows an application to run at a higher stiffness without unwanted vibrations. The practical stiffness depends on the type of device, but with our Desktop PHANToM we can increase the stiffness from about 400 N/m to about 800 N/m when working in most configurations.

The analytical solver also finds the solution considerably faster than the numerical solver. Results from the windtunnel, listed in table 1, shows that it takes about 85% less time.

6 Conclusions

The haptic primitives is a powerful contribution to volume haptics — the approach provides an abstraction layer that makes it easier to design and implement new haptic modes, and the technique allows for free combination of different haptic modes in a scene-graph structure. The mixed solver approach presented in this paper ensures that the best performance is provided for every volume haptics application. A fast and high precision analytical solver render the haptic feedback in the most common haptic configurations. The high precision allows for stable feedback with higher stiffness constant in the coupling equation, which gives better feeling of location and shape of features in the volumetric data. In special situations where the analytical solver fails, the system falls back on the more general numerical solver, capable of estimating the haptic feedback also for non-orthogonal constraints.

Acknowledgement

Professor Anders Ynnerman and Dr Matthew Cooper are gratefully acknowledged for the fruitful discussions in the topic and aid in the completion of the work in this paper and the paper itself.

References

[1] R. S. Avila and L. M. Sobierajski. A haptic interaction method for volume visualization. In *Proceedings at IEEE Visualization*, pages 197–204, October 1996.

[2] W. Hashimoto and H. Iwata. A versatile software platform for visual/haptic environment. In *Proceedings of ICAT'97*, pages 106–114, 1997.

[3] M. Ikits, J. D. Brederson, C. D. Hansen, and C. R. Johnson. A constraint-based technique for haptic volume exploration. In *Proceedings of IEEE Visualization '03, pp. 263–269*, 2003.

[4] F. Infed, S. V. Brown, C. D. Lee, D. A. Lawrence, A. M. Dougherty, and L. Y. Pao. Combined visual/haptic rendering modes for scientific visualization. In *Proceedings of 8th Annual Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 1999.

[5] H. Iwata and H. Noma. Volume haptization. In *Proceedings of IEEE 1993 Symposium on Research Frontiers in Virtual Reality*, pages 16–23, October 1993.

[6] D. A. Lawrence, C. D. Lee, L. Y. Pao, and R. Y. Novoselov. Shock and vortex visualization using a combined visual/haptic interface. In *Proceedings of IEEE Conference on Visualization and Computer Graphics*, 2000.

[7] K. Lundin, M. Cooper, A. Persson, D. Everstedt, and A. Ynnerman. Enabling design and interactive selection of haptic modes. *Virtual Reality*, 2006. DOI: 10.1007/s10055-006-0033-7.

[8] K. Lundin, M. Cooper, and A. Ynnerman. The orthogonal constraints problem with the constraint approach to proxy-based volume haptics and a solution. In *Proceedings of SIGRAD Conference*, pages 45–49, Lund, Sweden, November 2005. SIGRAD.

[9] K. Lundin, B. Gudmundsson, and A. Ynnerman. General proxy-based haptics for volume visualization. In *Proceedings of the World Haptics Conference*, pages 557–560, Pisa, Italy, March 2005. IEEE.

[10] K. Lundin, M. Sillen, M. Cooper, and A. Ynnerman. Haptic visualization of computational fluid dynamics data using reactive forces. In *Proceedings of Conference on Visualization and Data Analysis, part of IS&T/SPIE Symposium on Electronic Imaging 2005*, pages 31–41, San Jose, CA USA, January 2005.

[11] K. Lundin, A. Ynnerman, and B. Gudmundsson. Proxy-based haptic feedback from volumetric density data. In *Proceedings of Eurohaptics*, pages 104–109. University of Edinburgh, United Kingdom, 2002.

[12] A. Mor, S. Gibson, and J. Samosky. Interacting with 3-dimensional medical data: Haptic feedback for surgical simulation. In *Proceedings of Phantom User Group Workshop'96*, 1996.

[13] L. Pao and D. Lawrence. Synergistic visual/haptic computer interfaces. In *Proceedings of Japan/USA/Vietnam Workshop on Research and Education in Systems, Computation, and Control Engineering*, 1998.

[14] E. Vidholm, X. Tizon, I. Nyström, and E. Bengtsson. Haptic guided seeding of MRA images for semi-automatic segmentation. In *Proceedings of IEEE International Symposium on Biomedical Imaging*, 2004.