

Design and Psychophysical Study of Volume Compression for Haptic Rendering

Nils Jensen, Gabriel Gaus, Gabriele von Voigt
L3S Research Center / RRZN
Leibniz University Hannover
Schloßwender Str. 5, 30159 Hannover, Germany
{jensen, gaus, vonvoigt}@l3s.de

Stephan Olbrich
University Computer Center
Heinrich-Heine-University Düsseldorf
Universitätsstr. 1, 40225 Düsseldorf, Germany
olbrich@l3s.de

Abstract

The paper specifies a novel coder/decoder that compresses sequences of 3D texture data to support the haptic rendering of animated volumes. Lossy compression gives 25% of the original size. Lossy compression and clipping gives 3% but requires the renderer to query for new data when the point of contact moves. We evaluated the codec by means of rendering volumes on a 3-DOF Phantom 1.5A that simulated viscosity for each texture element. In this setting, we studied the impact of information loss in controlled experiments. We validated that our codec preserves haptic perception. An area of application is the development of network-distributed virtual environments that generate visual and haptic feedback.

1. Introduction

Human beings use force cues to explore the world if visual feedback is limited. Such a limitation can arise from a lack or overload of visual information. One of our long-term research goals is to simplify visual cues by way of fusing graphic and haptic devices to promote insight.

Apart from limitations in processing and device technology, software-related performance limitations affect the fidelity of visual-/haptic renderings. Many approaches are concerned with the accelerated rendering of polygonal scenes [15][14], but only few treat the rendering of *non-polygonal* scenes [1][5][13]. Most importantly, they exclude scene transfer to the haptic renderer [4][8].

The paper discusses a technique to accelerate haptic rendering, when one or more 3D textures, called a "stream" of scenes, must be transferred to the haptic renderer. In this case it is insufficient to retrieve each complete scene from the source and apply the standard haptic rendering techniques [5]. This has the following reasons: first, a complex scene is larger than the bandwidth of the link between the source and the renderer. The user notices the delay that

is caused by each transfer. Second, complex preprocessing is a disadvantage if new scenes arrive frequently at short intervals. The difficulty is that unprocessed scenes exceed the memory and processing power of the renderer, and such scenes prevent high rendering rates [12].

To solve these problems, we must understand that each of these latency problems is caused by overloaded processing capacity and bandwidth, not by the inherent latency of hard- or software components. Therefore the objective is to drop some scene data, which reduces to the task of removing exactly data that do not contribute to the haptic sensation. Our approach to this problem is to combine clipping, quantisation, and lossless compression to a lossy, sensation-preserving codec.

The objective in the paper is to specify the development and evaluation of our new codec and to describe why it preserves full control over the quality of service delivered to the haptic device, and thus, over the accuracy of the synthesised force cues. The results follow:

- An efficient codec that is specifically designed to support data preprocessing for direct haptic volume rendering with bandwidth shortages in the rendering pipeline, for example in networked systems.
- Codec integration in an existing visualization system. We have validated robustness, space-/time-performance, and customisability. The system inter-actively renders a sequence of volumes from a remote source.
- A study of the psychometric function of sensing synthesised viscosity in the haptic software component of the visualization system.

Section 2 discusses related work, Section 3 specifies the extended rendering pipeline. Section 4 describes the design and implementation of the codec. Sections 5–6 describe the integration and test. Section 7 specifies a psychophysical study to determine the impact of lossy compression on haptic sensation. Section 8 concludes and gives future work.

2. Previous and Related Work

Compared to forcefield- or polygon-based haptic rendering [2][12], haptic direct volume rendering is the rendering of scenes that are represented by 3D textures [1][13]. [5] report an extension to the GHOST Software Development Kit for this purpose.

Some researchers discuss applications of haptic direct volume rendering, for example [15][9][16].

There is no work on compression for haptic rendering. Most related is [14]’s discussion of simplification because they put sensation-preservation in the research context of haptics. Their approach reapplies mesh simplification algorithms [11]. They remap a geometry to fewer polygons and render them with a haptic device. The quality loss compared with the non-simplified geometries is unnoticeable. Compared to this, our approach follows the goal of sensation-preservation, but it is different in two important aspects: it works for networked systems that separate the data source from the renderer, and it is concerned with 3D texture-representations of the data.

Our approach combines clipping, quantisation, and lossless data compression. Clipping is to remove all parts of a scene outside the viewing frustum by way of testing the geometry parts against an intersection with the volume that represents the frustum. Quantisation maps attributes of a scene to fewer bits. Clipping and quantisation permanently erase information. The loss is easy to quantify.

Data compression is a technique to reduce the amount of data that conserves information. Lossless compression guarantees the decompressed data are identical to the uncompressed data. Lossy compression does not guarantee this because it permanently erases information. Generally, the induced loss is hard to quantify because it is spread between more than one part of the algorithm and because it can affect some data inhomogeneously.

Many algorithms can be used to support lossless volume compression, for example the PRO algorithm we developed [7]. PRO uses predictors because they are space and memory efficient. A predictor p estimates values v_i from n previous values. It has the general form $v_i = p_i = \sum_{j=1}^n d_{ij}v_j$. d is a weighting factor. Compression iterates the prediction over all values and writes a '1' if the predicted value matches the true value and writes a '0' in the other case. The bit stream is saved. Unpredicted values are saved in a second list. The predictors are specifically designed to keep the number of values in the second list small. Decompression reads the bit stream and repeats to iterate the prediction p_i over all values. If it reads a '1' from the first list then it outputs the predicted value, otherwise it moves the first value from the second list to the output. This fully reconstructs the data.

PRO follows this scheme for a setting of $n = 13$ and

it uses eight different predictors. PRO uses the predictors in raster-scan order on each scalar, data slice by data slice. Dependent on which predictor triggers for a local data pattern (the first-order continuation along the 12-neighbourhood), PRO appends the code of the predictor and optionally writes the difference between the predicted and the correct value in one bit. [7] specifies the predictors and their efficiency.

The impact of lossy volume compression on haptic perceptibility has not been investigated, partly due to the difficulty to guarantee an upper limit on the error. However, the error is exactly quantifiable if we build a lossy compressor from a primitive component that erases information within a strict error limit and a lossless compressor that post-processes the data. This has the advantage to encapsulate information loss in the primitive component and to improve the compression in the post-processor.

3. Overview

As indicated, our approach combines clipping, quantisation, and prediction-based encoding and decoding based on the data and the probing behaviour of the user. To support high rendering rates and high update rates of the animation, it is important to minimise the size of the data that arrive at the haptic renderer. Our combined approach offers important advantages over simple lossy compression, the most prominent being the ability to select the loss rate in accordance with the desired haptic accuracy compared to the rendered original data. The compression chain must be flexible to ensure intact data delivery if parts of the codec are bypassed.

By way of extending the haptic "rendering pipeline" [12], we split "rendering" to the following components:

1. a data source that generates or loads data in memory and drops unnecessary volume data,
2. a communication link over which either compressed or uncompressed volume data are sent,
3. a renderer that decodes the data and renders them in accordance with a user-specified rendering style. The renderer backpropagates which data will be required in the next time step.

In the next Section, we specify the processes of the extended rendering pipeline in more detail.

3.1. Distributed Architecture

The data source (for example a numerical simulation) runs on single- or multiprocessor-computers (see Figure 1). The first part of the codec is a clipping routine that filters data not needed at the time t . A quantiser drops a fixed

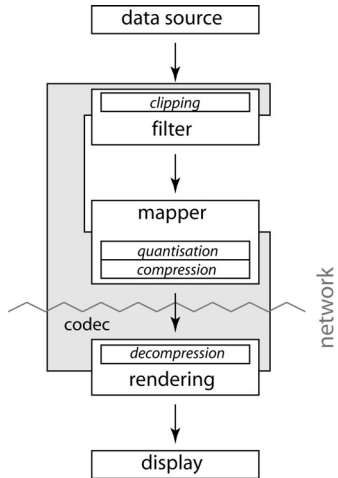


Figure 1. Visualization and haptic rendering pipeline with clipping, quantisation, compression, and decompression

number of bits for each byte that represents a texture element (texel). Then the codec either uses PR0 or bit packing, dependent on the requirements. The communication link sends the data to the renderer which is a separate workstation. The renderer decodes the data appropriately and maps them to force cues in accordance with the user-specified rendering style. The position of the end-effector is continuously reported to the filter at the data source. The haptic loop and the data receiver are threads on the renderer.

4. Codec Specification

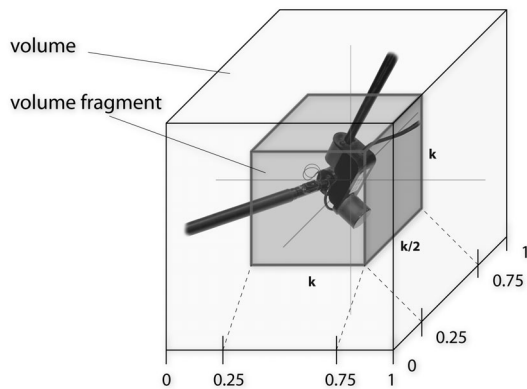


Figure 2. Clipping ensures only the data in the fragment is further processed

The codec uses a volume fragment that contains $k^2 \cdot k/2$ texels as a stream of bytes (Figure 2). The centre of this

cube is the position of the end-effector attached to the haptic device, and k is the maximum distance that the end-effector is expected to move between two time steps. However, this is an example and the codec could use more and differently shaped fragments. The data source generates or loads only texels that are inside the fragment. This reduces the data by 87.5% if the volume fragment is, for example, 1/8 of the total volume extent.

Then the quantiser eliminates the m least significant bits for each byte that represents a texel and appends the left bits to the output stream, resulting in a relative reduction between 12.5% and 87.5%. PR0 can read the stream and process it further. This results in a reduction of approximately 50% in accordance with conservative estimations. In total, the size of the original data is reduced to 3% on average.

The data can be stored or transmitted and then decoded. Exactly that information will be lost that depends on the choice of k and m . If a bi-directional link between the data source and the renderer is available, then this can be used to progressively transmit the volume data and then only m will induce a loss of information, with $m = 0$ meaning no and $m = 8$ meaning full loss (being hardly useful).

Section 7 describes controlled experiments to measure which values of m can be noticed by users.

5. Integration

We have integrated the codec in the Distributed Simulation and Virtual Reality Environment (DSVR)[6].

The renderer must calculate a force f which is returned by the method `gstEffect::calcEffectForce(...)`. GHOST takes the return value to generate a force vector in Newton. The renderer calculates f as follows. The equation is $f = -u_{xyz} \cdot v$, with x, y, z being the coordinates, u_{xyz} the viscosity of the texel and v the velocity of the end-effector. The viscosity of u_{xyz} is in the range of integers [0..255]. This is then mapped to [0..0.003] to be given as the force parameter to the driver to ensure that the device acts in the physical limits of its hardware capabilities. One may think of it as the useful force range of our Phantom. Higher values would generate noise, vibrations, and finally stop the device.

6. Performance Test

As an example, a data generator computed a sequence of volume data, see Figure 3 on the next page. It is a direct visual rendering of the Cayley surface defined by $g(x, y, z) = t/4(x^2 + y^2 + z^2) + txyz$ and modulated with time $t = [0..48]$. For each t , g was discretised and calculated for 48^3 voxels (to speed up data generation) and then

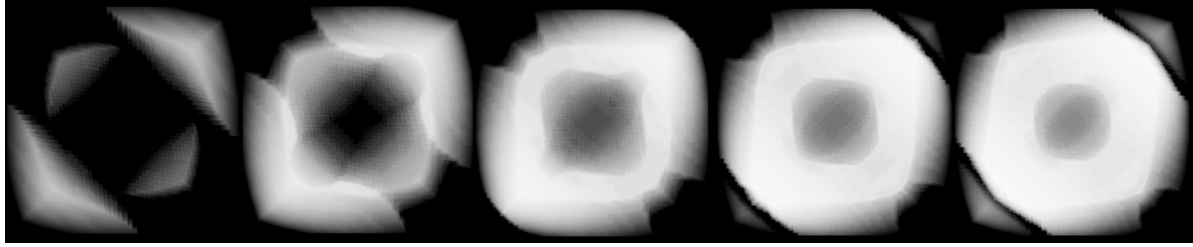


Figure 3. An example for a series of volumes processed by our codec, where $t = 9, 19, 29, 39, 48$

scaled to a resolution of $256^2 \cdot 128$ voxels. The accumulated size was 392 MB (uncompressed), 151 MB (compressed by PR0), and 117 MB (quantised, compressed). The data generator forwarded the data to a streaming server that replicated the data concurrently to the haptic renderer.

k	Size (texels)	Avg ups	Min ups	Max ups
16	$16^2 \cdot 8$	27.8	25.1	32.3
32	$32^2 \cdot 16$	22.3	20.8	23.3
64	$64^2 \cdot 32$	12.7	12.5	13.0
128	$128^2 \cdot 64$	3.7	3.5	3.9
256	$256^2 \cdot 128$	1	1	1
16	$16^2 \cdot 8$	12.8	11.8	14.1
32	$32^2 \cdot 16$	13.6	13.2	14.1
64	$64^2 \cdot 32$	10.5	10.5	10.5
128	$128^2 \cdot 64$	3.9	3.7	4.2
256	$256^2 \cdot 128$	0.6	0.6	0.7
16	$16^2 \cdot 8$	17.7	15.6	20.8
32	$32^2 \cdot 16$	16.1	15.5	16.5
64	$64^2 \cdot 32$	10.5	10.4	10.8
128	$128^2 \cdot 64$	5.3	5.1	5.5
256	$256^2 \cdot 128$	0.8	0.7	0.8

Figure 4. Updates per second for some codec settings

The following computers were connected via 100 Mbit/sec Ethernet:

- The data generator was a processor on the HLRN, a high-performance computing cluster with 1024 processors running AIX.
- The streaming server was a Pentium III with 0.8 GHz, 256.0 MB memory, and 254.9 GB disk space, running Redhat Linux 3.2.3-47.
- The haptic renderer was a Pentium 4 with 2.4 GHz, 512.0 MB memory, and 37.1 GB disk space under Windows XP. Users operated a 3-DOF Phantom 1.5A.

TCP was chosen to protect against bit errors which prevent data decompression. The socket buffer size was set

to 262144 bytes and NODELAY was set to 0.

The development tools were MS Visual Studio .Net 2003, Ghost SDK 3.1, Ghostwrap, and the standard Gnu C compilers from the Cygwin and Linux installations.

For the tests we chose combinations of clipping, compression, and 8-to-4-bit quantisation. Section 7 will identify this as the maximum unnoticeable error. For each combination, we varied the resolution of the volume between $16^2 \cdot 8$ and $256^2 \cdot 128$ uniformly arranged texels. We measured the updates per second (ups) on the renderer. This is the number of received and decompressed volumes per second. It is 0 for static volumes and differs from the number of rendered volumes per second. Decompression is the only part of our codec that affects rendering speed.

Figure 4 shows the measured ups for one clipping factor k per row. Clipping and quantisation use no computation on the renderer. Decompression uses 1.5 sec to process 8 MB [7], scaling linearly with the size of the volume. The first part of the Table shows the ups when compression and quantisation are disabled. The second part shows the ups when compression is enabled and quantisation is disabled. The last part shows the ups when compression and quantisation are enabled. We did each test three times and took the arithmetic average, minimum, and maximum.

We added graphic rendering to zoom in, rotate, and translate the volumes. This can take place on the haptic workstation (as in our case) or on a second workstation that transfers the image to the haptic workstation.

The results show that $k = 32..64$ supports interactive retrieval in a typical Intranet. Clipping improves the average ups but volumes that are too small affect update rates. The reason could be that the delay to fill the TCP packets increases in an irregular way. The correct solution is to set the socket's NODELAY option which would improve latency but affect data throughput because some packets would not be filled completely. Further, this solution is more prone to temporary jitter or to an inherently slow network. A compromise is to choose a volume size $32 \leq k \leq 128$ dependent on the bandwidth and on the number of cores on the processor.

Quantisation improves the performance because it is reliable, efficient, and completely done on the data generator.

Contrary, compression/decompression introduces overhead that does not compensate for the saved bandwidth in a 100 MBit/sec LAN. We estimate suitability for a network with a bandwidth of 2 MBit/sec and less.

7. Output Quality

We chose a 3-interval-forced-choice setup (3IFC) to understand the impact of quantisation losses $m = [1..7]$ on haptic sensation [10]. 3IFC gives accurate results in an efficient way when compared with the method of adjustment and with the method of limits.

The experimental setup follows. A participant receives every second a new presentation, three per trial. At two times per trial, the software presents a random, *uniform* volume density of $d = [0..(2^8 - 2^m - 1)]$. The third time in a trial, the software presents a *uniform* volume density of $d + 2^m$, called a "signal". For each presentation, an integer between 1 and 3 is displayed on the screen. The user enters the number that was displayed when (s)he has identified the signal. The order of presentations per trial is randomly assigned. After each trial, the participant reads a message whether (s)he guessed the signal correctly. To identify the 50% point on the psychometric function (X_{50}), m is incremented after each wrong answer, and decremented after each correct answer. To identify the 84% point on the psychometric function (X_{84}), m is incremented after each wrong answer, and decremented after four correct answers in sequence. In each case, we terminate the experiment with the 10th reversal from weaker to stronger stimuli. This is a compromise between expected experimental accuracy and fatigue of the participants.

Two men and a woman, associates of our institute, volunteered to use the system. They were 31, 34, and 19 years old. Each participant conducted the experiments at a different time than the others. The first author instructed on the objectives and type of the study and the participant began to use the system for approximately 3 minutes to train him-/herself. At the end of the training, the participant experimented to determine X_{50} , and then to determine X_{84} . We estimated the absolute (AL) and the difference threshold (DL) for each participant.

Figure 5 specifies the results. Participant A has approximately an AL of $X_{50} = 16$ and a DL of $X_{84} - X_{50} = 16$. Participant B and C have an AL of 48 and a DL of 16.

Incorrect measurements are inconsistent with the s-shaped psychometric function and cannot be reproduced between the participants. Participant A's estimation rate for a stimulus intensity of 4 in X_{50} is because this stimulus intensity was only reached three times. At each time the estimation was correctly guessed. This was obviously by chance because this effect could not be reproduced neither for 8 in X_{50} nor for 4..8 in X_{84} . Further, none of the other

participants achieved this result. A similar argument can hold for participant C's estimation rate for 16 or 32 in X_{50} , where the discrepancy is that one would expect both probabilities to be swapped to more closely approximate an s-shaped curve. This single effect could be attributed to a temporary distraction or fatigue of participant C. Overall, the data indicate that the experiment successfully measured the relevant interval, that all participants follow overall the same psychometric model in evaluating viscosities and that the data resolution with which viscosities are discriminated is close to 4 bit on the test device. This value is less than the resolution of 6 bit that was reported and cited in [3] because we

- selected 3IFC instead of the method of adjustment
- chose a haptic device with different resolution
- restricted force to an interval being manageable by our haptic device
- used another force formula to synthesise viscosity

From the new result we learn about the most probable sensation-preserving configuration of our codec. Specifically, the quantisation factor can be set to $8 - 4 = 4$ bit without degrading haptic sensation, in other words 50% loss of data does not affect haptic sensation in the subsequent haptic rendering process.

8. Conclusion and Future Work

The paper has described the design, integration, and test of a novel codec to process sequences of 3D texture data to support the efficient transfer to a haptic renderer. Users trade space-/time-usage with output quality. Experiments indicate that lossy compression preserves haptic sensation.

Future work is the application of our system to support the design and use of multimodal 3D virtual environments where users benefit from coupled visual and haptic renderings of synthesised volumetric media.

More future work is the evaluation of the codec with other rendering styles and haptic devices to further study the impact of information loss on haptic sensation.

9. Acknowledgements

Thanks to the reviewers for comments on improving the paper. We thank the Deutsche Forschungsgemeinschaft (DFG-EVITA No OL241/1-1), and the HPC-EUROPA project (RII3-CT-2003-506079 No 326), with the support of the European Community – Research Infrastructure Action under the FP6 "Structuring the European Research Area" Programme, for partial funding.

References

- [1] R. S. Avila and L. M. Sobierajski. A haptic interaction method for volume visualization. In *VIS '96: Proceedings of the 7th conference on Visualization '96*, pages 197–ff., Los Alamitos, CA, USA, 1996. IEEE Computer Society Press.
- [2] F. Brooks Jr., M. Ouh-Young, J. Batter, and P. Kilpatrick. Project GROPE – haptic displays for scientific visualization. In *SIGGRAPH 90*, pages 177–185, New York, NY, USA, 1990. ACM Press.
- [3] G. Burdea. *Force and Touch Feedback for Virtual Reality*. Wiley, 1996.
- [4] R. Haber and D. McNabb. *Visualisation Idioms: a Conceptual Model for Scientific Visualization Systems*, pages 74–93. IEEE Computer Society Press, Los Alamitos, CA, USA, 1990.
- [5] C. Huang, H. Qu, and A. Kaufman. Volume rendering with haptic interaction. In *3rd Phantom User Group Workshop*, pages 14–17. Massachusetts Institute of Technology, 1998.
- [6] N. Jensen, S. Olbrich, H. Pralle, and S. Raasch. An efficient system for collaboration in tele-immersive environments. In *Eurographics Workshop on Parallel Graphics and Visualization 2002*, pages 123–131. Eurographics, 2002.
- [7] N. Jensen, G. von Voigt, W. Nejd, and J. Bernarding. Efficient 1-pass prediction for volume compression. In *SCIA 2005*, pages 302–311, Berlin, Germany, 2005. Springer.
- [8] C. C. Law, W. J. Schroeder, K. M. Martin, and J. Temkin. A multi-threaded streaming pipeline architecture for large structured data sets. In *VIS '99: Proceedings of the conference on Visualization '99*, pages 225–232, Los Alamitos, CA, USA, 1999. IEEE Computer Society Press.
- [9] D. A. Lawrence, C. D. Lee, L. Y. Pao, and R. Y. Novoselov. Shock and vortex visualization using a combined visual/haptic interface. In *VIS '00: Proceedings of the conference on Visualization '00*, pages 131–137, Los Alamitos, CA, USA, 2000. IEEE Computer Society Press.
- [10] M. Leek. Adaptive procedures in psychophysical research. *Perception & Psychophysics* 63, pages 1279–1292, 2001.
- [11] D. Luebke and C. Erikson. View-dependent simplification of arbitrary polygonal environments. In *SIGGRAPH 97*, pages 199–208, New York, NY, USA, 1997. ACM Press.
- [12] W. Mark, S. Randolph, M. Finch, J. van Verth, and R. Taylor. Adding force feedback to graphics systems: Issues and solutions. In *SIGGRAPH 96*, pages 447–452, New York, NY, USA, 1996. ACM Press.
- [13] W. McNeely, K. Puterbaugh, and J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. In *SIGGRAPH 99*, pages 401–408. ACM Press, 1999.
- [14] M. A. Otaduy and M. C. Lin. Sensation preserving simplification for haptic rendering. *ACM Transactions on Graphics*, 22(3):543–553, 2003.
- [15] D. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. In *Computer Graphics Proceedings 1997*, pages 345–352, New York, NY, USA, 1997. ACM Press.
- [16] E. Vidolm and I. Nystrom. A haptic interaction technique for volume images based on gradient diffusion. In *World Haptics Conference 2005*, pages 336–341, Los Alamitos, CA, USA, 2005. IEEE Computer Society Press.

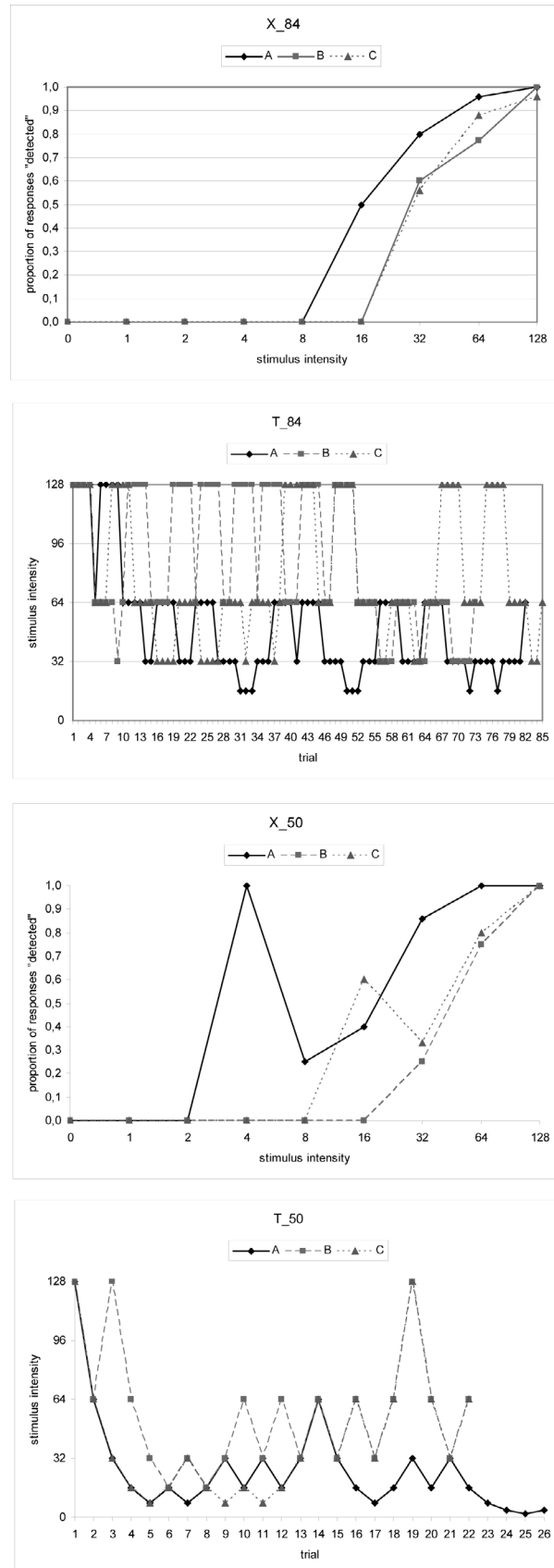


Figure 5. Results of the user study