FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN-NÜRNBERG
INSTITUT FÜR INFORMATIK (MATHEMATISCHE MASCHINEN UND DATENVERARBEITUNG)

Lehrstuhl für Informatik 10 (Systemsimulation)

Videocoding using a variational approach for decompression

Münch, P. and Köstler, H.

Lehrstuhlbericht 07-1

Videocoding using a variational approach for decompression

Münch, P. and Köstler, H.

June 26, 2007

Abstract

This paper describes the coding of video data and presents a variational approach to decompress the data. For encoding we select an adapted sub selection of all points in the domain based on a recursive subdivision of each frame. The idea for the decompression scheme comes from image inpainting, where variational approaches that require the solution of a partial differential equation based on isotropic or anisotropic diffusion are used to reconstruct the source video. To evaluate our results we compare certain aspects with the Mpeg and Mjpeg coding scheme.

1 Introduction

The aim of this work is twofold. On the one hand, there is the compression of the source video data, on the other hand the reconstruction of the video sequence. The used compression scheme is based on the selection of some points in the domain and saving their color value and coordinates. The information of all the other points is discarded, what means that we have no lossless compression.

Nevertheless, the decompressed data should be as close as possible to the original data. To achieve this goal we use variational approaches that are quite common in image processing, e. g. in the calculation of optical flow [6, 15, 9, 1], image segmentation [8, 11] or in-painting methods [7, 3].

The basic idea is to fix the given points in the video sequence and to use diffusion to fill in the missing information at unknown points [5]. Thus, one can think of a flow of the information given by the color and the coordinates into the unknown areas around them.

The paper is organized as follows. In section 2 the compression of the video data is described. Since we want to store only a subset of the points in the domain to achieve compression we chose B-Tree Triangular Coding (BTTC) to select the points to be stored in order to guarantee an optimal reconstruction later on. Then we compress their information by using Huffman compression. The variational approaches for decompression [14] are explained in section 3. Besides simple and fast homogeneous diffusion we consider more complex approaches, taking into account discontinuities in the domain and thus not blurring the edges in the image, namely nonlinear isotropic and anisotropic diffusion. The idea of these more evolved approaches is to reduce the diffusion in the vicinity of discontinuities in the image.

In section 4 we show some experimental results comparing the compression rate of our method with Mpeg and the different methods for decompression by quality and speed.

2 Compression Scheme

Some points are selected from each frame, such that they are sufficient for a good reconstruction of the original video sequence. We call these points *landmarks*. The rest of the points is dropped. The quality of the reconstructed video highly depends on this subset of points of the domain we chose to store.

For a good decompression, there are no further rules given other than that the distribution of these landmarks should lead to a maximum of information for a certain number of points. The information each point provides, is its color and the corresponding coordinates. To illustrate what is meant by this condition imagine a picture with objects of different colors. To give a good representation one would need points at the borders of these objects to know where their boundaries are and which color every object has.

The simplest scheme to select some landmarks would be a random distribution of points over the domain. This would have two obvious disadvantages. First the distribution does not care about the information each frame holds, so the maximization condition is not fulfilled in most cases. The second downside is the space that would be used for saving this information, since for each point three values have to be saved, the two coordinates in a frame and its color. Although this could be improved up to a certain level by different approaches a structured method for a well chosen subset of points is definitely superior. To cover both weaknesses of the random scheme, we use a selection based on B-Tree Triangular Coding as in [5].

2.1 B-Tree Triangular Coding

B-Tree Triangular Coding is a fast recursive subdivision scheme and was introduced for image coding by Riccardo Distasi in [4]. The image is first split up into two main triangles, further referred to as $T1$ and $T2$, defined by a diagonal, e.g. from top left to the lower right corner of the domain. The main goal of this scheme is to subdivide these triangles such that one gets a set of triangles where the differences of the color values in each triangle are below a certain threshold. Analogous to [4] we call an right angled triangles RAT, if it lies on the x,y plane PRAT and URAT if we refer to the upper face. The slightly modified algorithm used here works as follows. The triangles $T1$ and $T2$ are put on a stack. As long as there is one or more PRATs on the stack, one takes the topmost and checks if a further subdivision is necessary. Therefore the URAT-plane $G(x, y)$ is calculated by using linear interpolation between the three height values, represented by the gray values, on the corners.

$$G(x,y) = c_1 + \alpha(c_2 - c_1) + \beta(c_3 - c_1) \quad (1)$$

$$\alpha = \frac{(x - x_1)(y_3 - y_1) - (y - y_1)(x_3 - x_1)}{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)} \quad (2)$$

$$\beta = \frac{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x - x_1)}{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)} \quad (3)$$

The coordinates of the three corners of the current PRAT are referred as $x_1, y_1 - x_2, y_2$ and x_3, y_3. The condition that decides whether the PRAT has to be divided into two smaller triangles or not is: All Vales of the height function, which describes the surface $A(x, y, z)$, may not diverge more than a given tolerance from the interpolated plane $G(x, y)$

$$err(x,y) = F(x,y) - G(x,y) \quad (4)$$

$$err(x,y) \leq \epsilon \quad (5)$$

with $\epsilon > 0$. If the condition (5) is violated, the perpendicular is dropped to the hypotenuse of the current PRAT which creates the two child PRATs. These are now put onto the stack. If a PRAT fulfills the condition it is added to the list of leaf-PRATs. These steps will be repeated as long as there are PRATs on the stack. The subdivision creates a tree structure, where the leafs are represented by triangles which must not be subdivided, and the inner knots are representing a subdivision of a PRAT. We now have a domain represented by a set of triangles. The more detail a part of the domain has the more triangles are there. The colors are taken from each model point where the perpendicular meets the hypotenuse when subdividing a triangle and put them in a list. The order of these height values is given by a breadth first search through the tree. In parallel one creates a bitlist where a subdivision, or inner knot, is represented by a 1 and a leaf-PRAT by 0. From these two list one can recreate the tree and insert the colors from the first list at the right points of the domain.

These two lists which hold the information needed for decompression are called the bitlist from which the coordinates can be restored, and the list of height values, where in most cases some colors appear more often than others. By compressing each of them using a Huffman coding extra space can be saved. So the amount needed for storing the height list is reduced to $\sim 60\%$ and $\sim 20\%$ for the bitlist.

Additionally to these lists some extra information is needed to reconstruct a sequence that is summed up as meta information and stored before the compressed lists when saved to file. The most important ones are the height and width of a frame and the number of frames in a sequence.

Without the height and width the coordinates can not be calculated correctly. Other things which are stored are if it is a color sequence and a list of positions in the file where the beginning of a frame is, in order to be able to start decoding from there, without reconstructing every frame before.

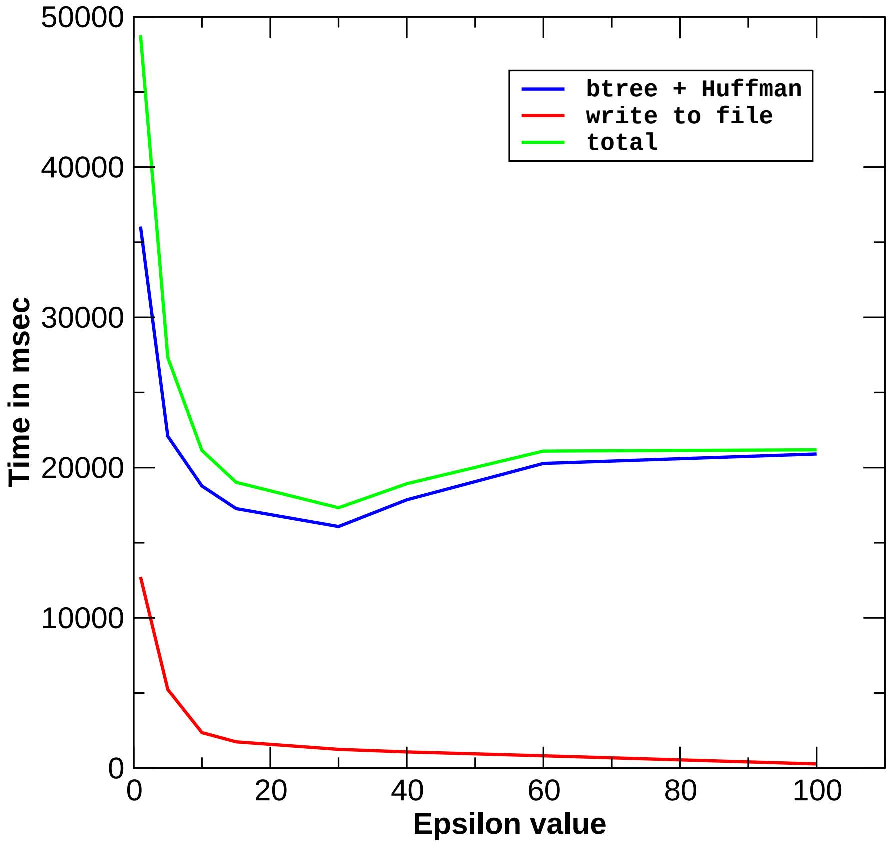

Figure 1: Influence of ϵ on the time needed for compression.

As can be seen in Fig. 1 the main factor for the time needed for compressing one frame is the creation of the tree using the BTTC-scheme and applying Huffmann compression to the lists. The time needed for writing the data to the file can nearly be neglected. The x-scale in this image *comprdiff* represents the tolerance value ϵ. So a greater ϵ leads to less subdivision and therefore less time is needed creating the tree. But at a ϵ-value of 30 in this case there is a increase in the time needed. This is based on the fact that if the condition is broken the current PRAT must not be tested further and will be subdivided. But with great ϵ the probability of a violation is lower or not existent at all. And much more or all points in a PRAT have to be tested. Fig. 2 gives an overview of achievable file sizes with this scheme compared to mpeg and mjpeg compression.

3 Decompression

To decompress the video data we first read the metadata and decompress the Huffman coded lists. For each 1 in the bitlist the current PRAT will be subdivided and the intersection point of the perpendicular with the hypotenuse is given the corresponding color value from the height list while the child PRATs are queued in. When a 0 is read one moves along to the next PRAT. When the queue is empty all landmarks are set at the corresponding coordinates with their color. Thus the domain is mainly empty with some sparkles of information.

Next we describe the variational approach to reconstruct the missing color values in the video sequence. On a n-dimensional domain $\Omega \subset \mathbb{R}^n$ of an image the height function $v : \Omega \to \mathbb{R}$ is described by a scalar valued function. In this case v is only known at some points which are given through the subset $\Omega_1 \subset \Omega$. We are now searching for a smooth function $u : \Omega \to \mathbb{R}$ which is a good approximation to v. With the condition that u has to be identical to v at these landmarks this

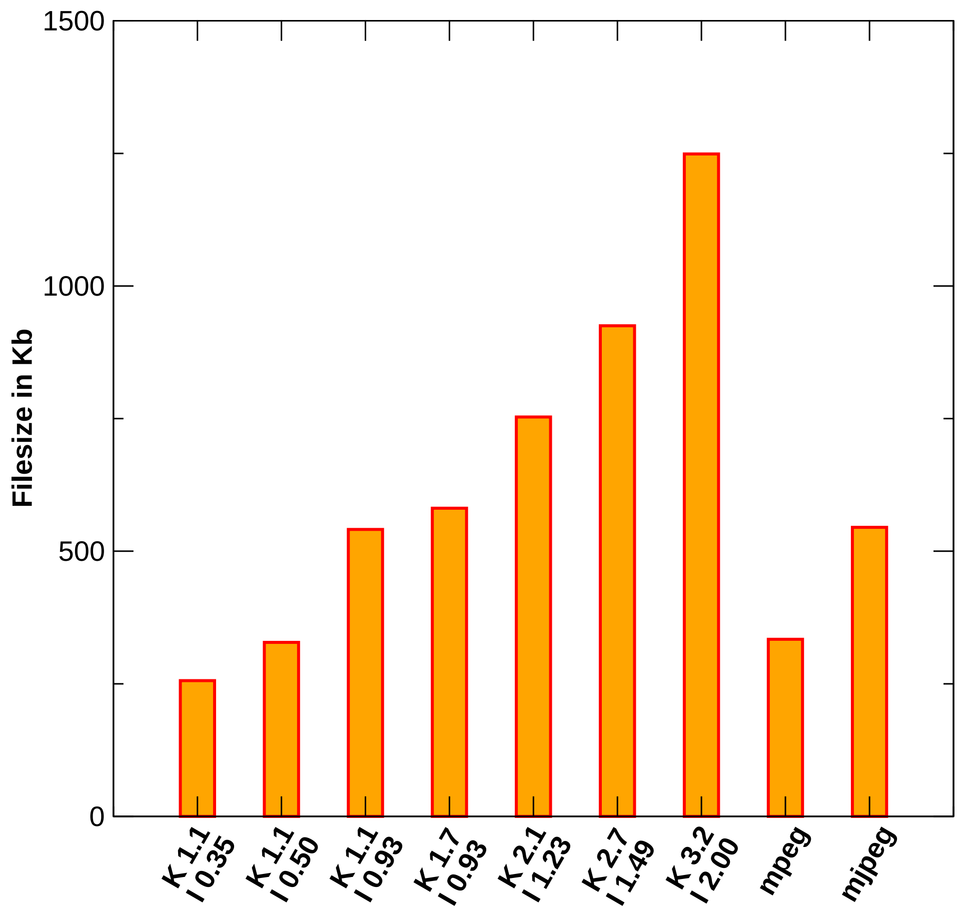

Figure 2: Filesizes of Mpeg, Mjpeg and Pdevc - K: Keyframe Bit per pixel, I: Bit per pixel in inner frames.

results to 6, 7 and 8. The characteristic function c defines which term is valid at a certain position in 6 and represents the trust in each term. The landmarks in Ω_1 get their values from f when initializing. L is an elliptic differential operator which represents the used smoothing operator. In the following L will also be referred to as regularizer.

$$(1 - c(x))Lu \; - \; c(x)(u - f) = \delta_t u \tag{6}$$

$$c(x) = \begin{cases} 1 & x \in \Omega_1 \\ 0 & else \end{cases} \tag{7}$$

$$f(x) = \begin{cases} v(x) & x \in \Omega_1 \\ 0 & else \end{cases} \tag{8}$$

In this PDE the landmarks have to be treated as fixed Dirichlet boundary values. Note that for a second order PDE these landmarks introduce singularities, which can be detected in the resulting images, since the analytical solution tends to infinity at these locations.

The PDEs are discretized with finite differences or finite volumes which results in a sparse system of equations. These can be solved using a Gauss-Seidel iteration or multigrid methods. In the discrete setting we use the stencil notation to describe the influence of a point to its neighbors or the other way round. Landmarks get a so called pointstencil which saves them from the influence of the surrounding points by setting the weight of every neighbor for this point to zero.

3.1 The Different Smoothing Operators

There is a great variety of possible smoothing operators for the equation (6) [14]. We chose here three second order PDE models, homogeneous, nonlinear isotropic and nonlinear anisotropic (or edge-enhancing) diffusion. The simplest one is the *homogeneous diffusion* where the smoothing is independent of directions and image information. It is described by the Laplace equation (9)

$$Lu := \alpha \Delta u \ . \tag{9}$$

This linear PDE is discretized by finite differences and solved by a multigrid method. When solving a 3D block of frames the emphasis between frames can be adjusted with the parameter α in each direction. The intention is here to prevent the creation of ghost effects along the time dimension, where objects shine through at areas where they no longer are. Withing a single frame α has no influence. Note that in 1D the homogeneous diffusion corresponds to linear interpolation, in higher dimensions to using radial basis functions [2].

The general weighting of every point to its surrounding, which is the same for every point no matter where in the domain it is, allows its information to equally spread in every direction resulting in a blur effect such that edges appear smeared. To avoid this information from the image has to be included in the diffusion model.

The *nonlinear isotropic diffusion* model does this by reducing the strength of the diffusion at high gradients in the image that correspond to edges in order to allow there discontinuities in the solution. Nonlinear isotropic diffusion is described by

$$Lu = div(g(|\nabla u|^2)\nabla u) \tag{10}$$

and was introduced by Pietro Perona and Jitendra Malik [10]. As diffusivity function we use the Charbonnier-diffusion

$$g(s^2) = \frac{1}{\sqrt{1 + \frac{s^2}{\lambda^2}}} \ . \tag{11}$$

that decreases with increasing parameter s^2. λ denotes a contrast parameter setting the amount of reduction at discontinuities. The operator (12) has to be discretized by finite volumes instead of finite differences to deal with the jumping coefficients. For the finite volumes discretization the domain is split up in an limited amount of cells called volumes

$$-\int_{\Omega_{i,j}} \nabla(g\nabla u) \quad d\Omega = \int_{\Omega_{i,j}} f \quad d\Omega \ . \tag{12}$$

To calculate the integral of such a volume it is split up into four integrals, which are defined by the four boundaries of a volume and have to be summed up so that the left part of (12) becomes (13):

$$\int_{SW}^{SE} g\frac{du}{dy} \quad dx - \int_{SE}^{NE} g\frac{du}{dx} \quad dy - \int_{NE}^{NW} g\frac{du}{dy} \quad dx + \int_{NW}^{SW} g\frac{du}{dx} \quad dy \ . \tag{13}$$

Each of the variables SW, SE, NE, NW is for one corner of a volume and is the short form of a orientation. SE defines the south-east corner and the integral $\int_{SE}^{SW}$ covers the south or lower edge of the volume. The right side is also discretized with finite volumes, but with the chosen edge length of one the value is the same as for finite differences. To compensate the already mentioned problem with jumping coefficients it is necessary that the limit for g is the same regardless of the side used to determine its value. This is achieved by taking the geometric or the arithmetic mean which can be seen in the coefficients of the stencils (15 - 19). Following this, the updating step can be described by the stencil

$$\begin{bmatrix} & s_N & \\ s_W & s_m + \frac{1}{\tau} & s_O \\ & s_S & \end{bmatrix} \quad u^{k+1} \quad = \frac{u^k}{\tau} \ . \tag{14}$$

$$s_M = 2\frac{g_M}{2h_x^2} + 2\frac{g_M}{2h_y^2} + \frac{(g_W + g_O)}{2h_x^2} + \frac{(g_N + g_S)}{2h_y^2} \tag{15}$$

$$s_N = \frac{(g_M + g_N)}{2h_y^2} \tag{16}$$

$$s_W = \frac{(g_M + g_W)}{2h_x^2} \tag{17}$$

$$s_O = \frac{(g_M + g_O)}{2h_x^2} \tag{18}$$

$$s_S = \frac{(g_M + g_S)}{2h_y^2} \tag{19}$$

Function values at the point or its neighbors are represented by the variables g_M, g_N, g_W, g_O, g_S where the directions are marked with the indices. h is the cell spacing which is defined as 1.

To solve the nonlinear problem numerically we use a lagged diffusivity method [1], what means we use the standard linear Gauss Seidel method as smoother but update the nonlinear stencil entries after each iteration.

Next we again extend the model by not only considering the location of edges but also their orientation. This results in the *edgde-enhancing diffusion* model. Thus when there is a edge of an object with a different color than the background, the smoothing process is done along the edge but only minimized across which preserves the structure of the edge. This can be achieved with the following equation introduced by Joachim Weickert [13]

$$\delta_t u = div(D(\nabla u_\sigma)\nabla u) \ . \tag{20}$$

D is a diffusion tensor which depends on ∇u_σ. While ∇u_σ represents the strength of a edge it is multiplied with the transposed $(\nabla u_\sigma)^T$ to consider the direction of the edge. To make the method more robust in the case of strong discontinuities in the image a smoothing is applied to u by convolution with a Gaussian of standard deviation σ. The diffusivity function for the dampening must be applied to the matrix resulting from the product of ∇u_σ with $(\nabla u_\sigma)^T$. We denote the entries of D by

$$D = \begin{pmatrix} a & b \\ b & c \end{pmatrix} \tag{21}$$

In this tensor one of the eigenvectors has the direction to the highest increase of gray value and the other is perpendicular to it. The diffusivity function g is applied to the eigenvalues μ_1 and μ_2 without changing the eigenvectors.

$$\lambda_1 = \quad g(\mu_1) \tag{22}$$

$$\lambda_2 = \quad g(\mu_2) \tag{23}$$

Then the new tensor

$$J = \lambda_1 v_1 v_1^T + \lambda_2 v_2 v_2^T$$

is built with the new eigenvalues and the old eigenvectors. This leads to the equation

$$Lu = div\left(J(\nabla u_\sigma)\nabla u\right) = div\left(g(\nabla u_\sigma \nabla_{u_\sigma}^T)\nabla u\right) \tag{24}$$

for the regularizer.

The discretization and solution of the PDE is similar to the nonlinear isotropic diffusion case also done by finite volumes leading to a 9-point stencil in 2D and using a multilevel inexact lagged diffusivity method.

4 Results

First we take a look at the gain we achieve by a BTTC-based subdivision compared to random selection. The mentioned distribution of information in an image is accounted with the level of

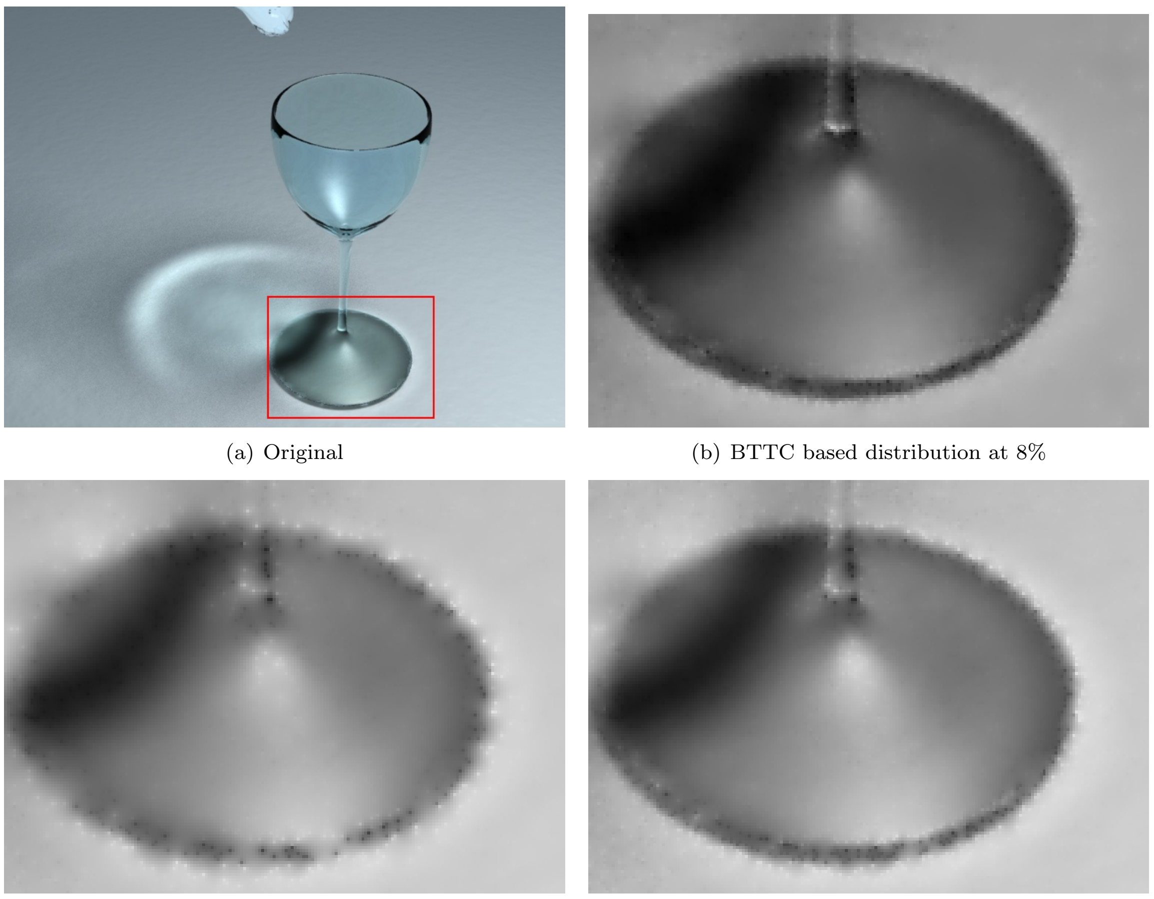
(a) Original (b) BTTC based distribution at 8% (c) Random distribution at 8% (d) Random distribution at 20%

Figure 3: Improvement achieved by BTTC based selection.

subdivision in all parts of the domain and therefore the specific selection of landmarks. This effect is illustrated in the following images showing an empty glass. The sequence is taken from [12].

For all images the same settings and decompression method (homogeneous diffusion) were used. If the same count of landmarks is used as in 3(b) and 3(c) where 8% of all points are landmarks, the only difference is their distribution. By the BTTC-based scheme the density of landmarks is higher in areas with more details, like the bottom of the glass in the pictures above. Using such a adapted distribution, structures can be restored at higher quality. In this example one can achieve the same quality with a BTTC-based selection and eight percent landmarks as using a random distribution and a ratio of 20% landmarks.

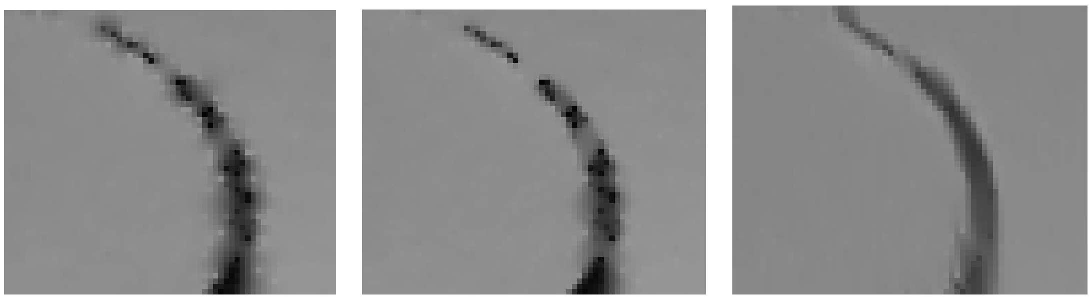
(a) HD (b) NID (c) EED

Figure 4: Comparison of HD, NID and EED at a bezel.

Next we give an impression in which way the different regularizers improve the quality of the reconstructed images. A rate of 20% random landmarks was chosen for this test which still creates some gaps, where no point is chosen from the darker edge as landmark. This weakness of the random selection strengthened the difference between the three used regularizers. In Fig. 4(a) where the *homogeneous diffusion* was used the edge is almost non interrupted. Is shows that the diffusion spreads uniformly in all directions, thus the edge is blurred. Landmarks with the bright background color near the edge are clearly visible like small islands being surrounded by the darker blurred edge. The difference to the *nonlinear isotropic diffusion* in Fig. 4(b) is clearly visible. At the edge the smoothing is damped which separates them much better from the background but also highlights the spaces between, where the amount of landmarks with information from the edge is too low.

Fig. 4(c) shows the effect of the *edge enhancing diffusion* and copes with the difficulties of the two other regularizers. The "flow" of the information is directed along the edge, which is still separated with a clear border. Unlike with the *NID* regularizer the areas with little information inside the edge are filled up. The effect can be summed up to a smoothing along edges or in between areas with the same or similar grey value. The computational costs effort are about nine times higher for *NID* as for the *homogeneous diffusion HD* and another 30% more for the *EED* or twelve times more effort than *HD*. To give a example of the speed a sample is calculated with a resolution of 320x240 pixel. All computations were done on a *Pentium 4* machine with 3.0 GHz, 512 KB L2 Cache and a total of 2 GB RAM. In Table 1 average frame rates per second (fps) are shown for different regularizers. When using the nonlinear diffusion models not only fewer iterations can be calculated in one second (40 with *NID* and about 28 with *EED*) but also more iterations (50 to 70) are needed to reconstruct one frame. Note that the number of iterations depends on the number of landmarks in the given frame.

Regularizer	fps
HD	10
NID	0.6
EED	0.4

Table 1: Comparison of decompression times (in fps) for different regularizers using Gauss-Seidel iteration.

Currently this method needs up to ten times more space than a frame compressed using Mpeg-1 video compression as can be seen in Fig. 5. In contrast to the Mpeg-1 frame there is yet no benefit from correspondencies between frames.

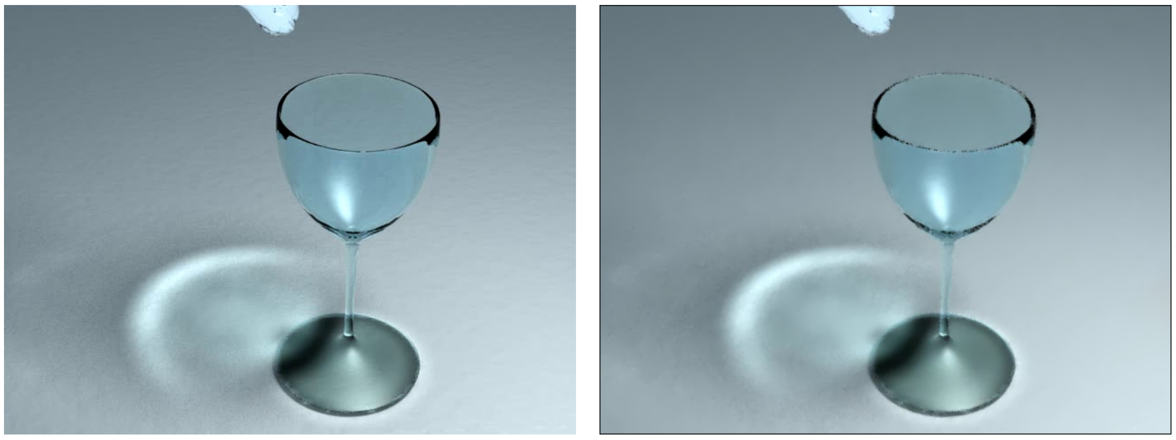
(a) MPEG 0.245 bpp (b) PDEVC 2.2 bpp

Figure 5: Image quality with Mpeg and Pdevc.

5 Conclusions

A new method for video compression and decompression was introduced and implemented. Although only a first step towards a real video codec was made the compression rates were in the range of common coding schemes. The potential for some more optimizations is, e.g. using the YCbCr colorspace instead of the RGB-colorspace, which allows subsampling of the chrominance channels. While this is not visible for the human eye it saves a huge amount of space. As done by most other coding schemes we should make use of correspondencies between frames when compressing the source data. One idea here would be to store information about the motion field in addition to the landmarks and to use this information for decompression. Additionally, if the smoothing along the time axis could be enabled based on this step, a lower amount of landmarks has to be present in frames, where the difference to its neighbors is low. Thus, similar to areas with the same color, which are saved very efficient, this method would benefit from low differences between multiple frames.

References

[1] A. Bruhn, *Variational Optic Flow Computation: Accurate Modeling and Efficient Numerics*, PhD thesis, Department of Mathematics and Computer Science, Saarland University, Saarbrücken, 2006.

[2] M. Buhmann, *Radial Basis Functions*, Cambridge University Press,Cambridge, UK, 2003.

[3] T. Chan and J. Shen, *Non-texture inpaintings by curvature-driven diffusions*, 2001.

[4] R. Distasi, M. Nappi, and S. Vitulano, *Image compression by b-tree triangular coding*, (1997).

[5] I. Galic, J. Weickert, M. Welk, A. Bruhn, A. Belyaev, and H. Seidel, *Towards PDE-based image compression*, Proceedings of Variational, geometric, and level set methods in computer vision, Lecture notes in computer science, (2005), pp. 37–48.

[6] B. Horn and B. Schunck, *Determining optical flow*, Artificial Intelligence, (1981), pp. 185–203.

[7] F. Lauze and M. Nielsen, *A variational algorithm for motion compensated inpainting*, in British Machine Vision Conference, September 2004.

[8] J.-M. Morel and S. Solimini, *Variational methods in image segmentation*, Birkhäuser, Boston, 1995.

[9] P. Münch, *Optischer Fluss für Videosequenzen.* Studienarbeit, Sep 2005. Betr. H. Köstler.

[10] P. Perona and J. Malik, *Scale-space and edge detection using anisotropic diffusion*, Tech. Report UCB/CSD-88-483, EECS Department, University of California, Berkeley, 1988.

[11] C. Schnörr, *Variational methods for adaptive image smoothing and segmentation*, Academic Press, San Diego, 1999, pp. 451–484. Buchbeitrag.

[12] N. Thürey and U. Rüde, *Stable free surface flows with the lattice boltzmann method on adaptively coarsened grids*, Computing and Visualization in Science, (2006).

[13] J. Weickert, *Theoretical foundations of anisotropic diffusion in image processing*, 1996.

[14] J. Weickert, *Anisotropic Diffusion in Image Processing*, Teubner, Stuttgart, 1998.

[15] J. Weickert and C. Schnörr, *Variational Optic Flow Computation with a Spatio-Temporal Smoothness Constraint*, Journal of Mathematical Imaging and Vision, 14 (2001), pp. 245–255.