Compression of Vector Field Changing in Time*

Tomáš Golembiovský[1] and Aleš Křenek[2]

1 Faculty of Informatics
Masaryk University, Czech Republic
nyoxi@ics.muni.cz

2 Institute of Computer Science
Masaryk University, Czech Republic
ljocha@ics.muni.cz

Abstract

One of the problems connected with a real-time protein-ligand docking simulation is the need to store series of precomputed electrostatic force fields of a molecule changing in time. A single frame of the force field is a 3D array of floating point vectors and it constitutes approximately 180 MB of data. Therefore requirements on storage grow rapidly if several hundreds of such frames need to be stored.

We propose a lossy compression method of such force field, based on audio and video coding, and we evaluate its properties and performance.

Digital Object Identifier 10.4230/OASIcs.MEMICS.2010.40

1 Introduction

Proteins play an essential role in many biological processes in cells and thus are an important subject of studies of many biochemists. Biological activity of a protein is frequently described in terms of the docking problem — under what conditions and where on the surface of the protein a small molecule (ligand) can be attached in an energetically favoured position. Various automated methods of solving the problem exist [3]. Recently haptic-assisted methods (the inter-molecular forces are delivered to the user with a force-feedback device) were introduced [6].

We have developed an improved docking simulator [5] with emphasis on realistic feeling as well as accuracy of the interaction. Unlike previous methods, the ligand is connected to the haptic device by visco-elastic link which is known to improve the feeling of manipulation with real objects [7].

The principal forces involved in the interaction are van der Waals and electrostatic. The latter have long range influence, therefore the only feasible approach is precomputing the resulting force field on a 3D grid of sufficient resolution (typically 250 points in each dimension). Because the molecule can change its shape in time, the induced force field changes as well, resulting in a long sequence of 3D “frames” whose size becomes unmanageable. Here we propose a compression method on this data inspired by common audio and video compression.

2 Compressed Data and Its Properties

The data we are interested in are formed by a series of frames. A single frame is a 3D array of 3D vectors, having 250 cells in each dimension. The force vectors are represented as single

* This work was done as a part of the research intent MSM0021622419. The access to the MetaCentrum supercomputing facilities provided under the research intent MSM6383917201 is highly appreciated.

Sixth Doctoral Workshop on Math. and Eng. Methods in Computer Science (MEMICS'10)—Selected Papers.
Editors: L. Matyska, M. Kozubek, T. Vojnar, P. Zemčík, D. Antoš; pp. 40–46
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

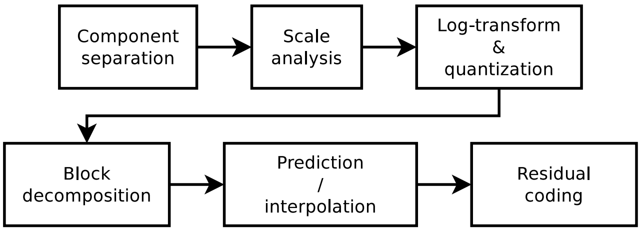

Figure 1 Schema of lossy compression of force field

precision floating point numbers. Altogether it leads to approx. 180 MB ($250^3 \cdot 3 \cdot 4$B) of raw data per one frame.

The force field is a sum of contributions of all charged atoms of the macromolecule on a unit charge, given by the Coulomb law $F_i = k_e q_i e/r^2$. This yields steep peeks in near vicinity of the macromolecule atoms while the field is relatively smooth farther. The dynamic range is as many as 18 orders of magnitude for the same reasons. On the other hand, due to the complementary van der Waals interaction in the haptic model, which generates even steeper barriers, it is virtually impossible for any two atoms to get closer than approx. 1 Å. Therefore we do not have to insist on accurate encoding of the force field peeks.

3 Compression Method

According to the reasoning given above, accurate encoding of high values of the force field is not required, therefore we trade off accuracy for compression ratio, and we aim at a rather aggressive lossy compression scheme.

Currently, the Cartesian components of force vectors in each frame are processed separately, exploiting their possible correlation would be a subject to further work.

Figure 1 outlines the whole compression process. Its principal steps, heavily inspired by audio and video codecs, are:

1. Component separation: Cartesian components are treated independently.
2. Scale analysis: for each frame set an appropriate scaling factor (which maps the field domain onto $[-1; 1]$) for subsequent μ-law transform is determined.
3. Logarithmic transform and quantization
4. Decomposition into blocks
5. Prediction: best-fitting parameters of some prediction model are found
6. Coding of residuals w.r.t. the predictor

The rest of this section deals with detailed description of those steps.

3.1 Frame Types

The frames of the force field are classified in a manner defined in MPEG-1 and later adopted in many video codecs.:

1. I-frame — uses linear filters with fixed coefficients
2. P-frame — uses one previous I- or P-frame for prediction
3. B-frame — uses one previous and one following I/P-frame for prediction

Specific assignment of frame type is controlled by configuration. One example of such a scheme is `IBBPBBP` that is commonly used by MPEG video compression.

3.2 Logarithmic Transform and Quantization

The first approach of compressing the floating point numbers directly lead to low compression rates and high decompression times due to large amount of floating point arithmetics. Therefor quantization was introduced combined with logarithmic transform to deal with the large dynamic range of the data (Sect. 2). Values are scaled down in order to fit into the interval [-1;1]. Then continuous version of μ-law companding algorithm is applied:

$$F(x) = \operatorname{sgn}(x)\frac{\ln(1+\mu|x|)}{\ln(1+\mu)}$$

We allow adaptive scaling to accommodate different value range. However, since I- and P-frames are used for prediction later, the scale cannot change arbitrarily. Therefore we split frames into sets by the location of I-frames and let all frames in the set share one scale.

The transformed values are subsequently quantized — expressed as integer value with configurable number of bits. Finally, the whole field is split into smaller blocks of equal size that are further processed separately.

3.3 Prediction

3.3.1 I-frames

I-frames start each frame set and they represent the corresponding frame accurately (up to the loss due to logarithmic quantization). This is to allow seeking in the compressed file and to prevent accumulation of further error throughout the whole frame sequence.

In I-frames the values (each Cartesian component of the force) in the block (cube of points) are ordered into a sequence according to a fixed pattern first, and those are passed to predictor. The force fields are formed by continuous wave-like patterns that makes them in some aspects similar to audio waves. For prediction we use linear order predictors with fixed coefficients, originally described in the Shorten[9], where they were used for compression of audio signal, and were later adopted also by AudioPaK[2] or Flac[1]. The predictor attempts to fit a p-order polynomial to the last p points:

$$\hat{x}_0[n] = 0 \tag{1}$$
$$\hat{x}_1[n] = x[n-1] \tag{2}$$
$$\hat{x}_2[n] = 2x[n-1] - x[n-2] \tag{3}$$
$$\hat{x}_3[n] = 3x[n-1] - 3x[n-2] + x[n-3] \tag{4}$$
$$\dots \tag{5}$$

where $x[n]$ denotes signal sample at point n and $\hat{x}_k[n]$ the k^{th} order prediction for the sample at point n. The prediction error can be also computed recursively, making the method attractive from the performance point of view. We chose 5 as the maximal order used. Higher order predictors showed only little contribution to the compression while slowing down the compression significantly. The reader is suggested to refer to [9] for further details.

3.3.2 P-frames

P-frames use previous I- or P- frame as the prediction that is then subtracted from current frame to obtain the prediction residual (encoded in the same way as I-frames, cf. Sect. 3.4). Unlike MPEG compression we don't make motion prediction yet; so far we were not able to elaborate a promising scheme.

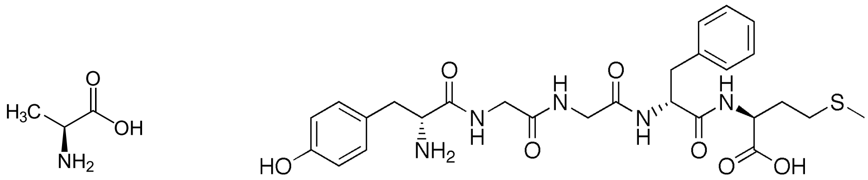

Figure 2 Testing molecules alanine and enkephalin

3.3.3 B-frames

B-frames use the nearest preceding and following I- or P-frame from which weighted average is computed. The weights used are distances (number of frames) from current frame to the respective I- or P-frame:

$$\hat{X} = \frac{\triangle f \cdot P + \triangle p \cdot F}{\triangle p + \triangle f}$$

where P and F are the preceding and following frame, and $\triangle p$ and $\triangle f$ are the distances from them. The same residual encoding (Sect. 3.4) is used again.

3.4 Residual Coding

The distribution of residual values resembles a two-sided geometric distribution. Therefore we choose Rice codes [8], which are known to perform well in this case, for the residual coding.

Rice codes are designed to code non-negative integer numbers, we have to map the error codes to this domain. If all error values in a block are either non-positive or non-negative (which turns to be more than 50% of cases on real data), we encode just absolute values and keep the block-wide information on the sign. This trick saves one bit per single value. Otherwise the values are interleaved — positive values map to even numbers and negative values to odd numbers.

3.5 Frame Promotion

It turns out that for some B-frames the P-frame prediction would lead to a better compression. Moreover, evaluation of the experimental data shows that it typically happens on all Cartesian components of the vector field simultaneously. Therefore we introduce *frame promotion*, ad-hoc replacement of the B-frame by P-frame in the specific frame set.

The method requires no extra work on the decompression side and only small additional effort during the compression. The overall compression ratio is improved by 0.5%–2% (see Sect. 4.2).

Similarly, sometimes I-frames deliver better compression than P-frames as well. However, we do not promote P- to I-frames due to much higher requirements on decompression time.

4 Results

Experimental implementation of the presented compression method is available. We used it for quantitative analysis of the method properties. Rather than evaluating the method directly with large macromolecules, we chose smaller molecules — alanine amino-acid and met-enkephalin peptide (Fig. 2). Alanine itself and the amino-acids forming enkephalin occur frequently in large macromolecules, and due to their flexibility (there are two freely rotatable bonds in alanine, and many more in enkephalin) they represent the worst case of

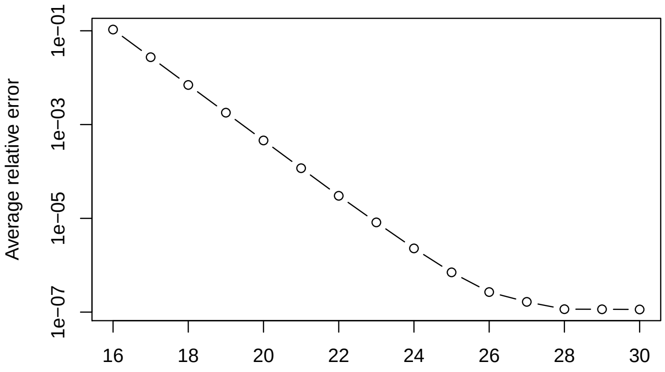

Figure 3 Average relative error for enkephalin at various quantization levels

type	alanine	enkephalin
I	17%	16%
P	6%	6%
B	7%	7%

Table 1 Compression ratio per frame type. Only P-frames of compression schemes without B-frames are included (see text)

local behaviour of the force field for the compression. Testing data sets were generated by molecular dynamics simulation, interpolated with a method we developed before [4]. The force field was sampled on a grid of 250 points in each dimension, a resolution used in the haptic application, with approx. 200 frames in both sets.

4.1 Data Loss

Figure 3 shows typical relative loss observed. Data loss is only a result of floating point arithmetic involved in μ-law transform and the quantization. Tests were performed with quantization ranging from 16 to 30 bits. Under the assumption that the compression inaccuracies are located close to the macromolecule atoms (Sect. 2) we conclude that the relative error of 10 % at 16 bits is still acceptable.

4.2 Compression Ratio

Table 1 shows the achieved compression ratio per frame type and testing data set, using 16bit quantization. The dispersion of the compression ratio for I- and B-frames is negligible regardless of their position in the compression scheme. On the other hand, for P-frames it holds only when they follow an I- or P-frame immediately in the scheme. This is a consequence of the prediction used for P-frames — the farther a P-frame is from the preceding I- or P-frame, the bigger is the prediction error, making the space occupied by encoded residuals to grow almost arbitrarily.

We didn't observe any significant difference between behaviours of the compression of the two molecules, therefore we further refer to enkephalin only.

Figure 4 shows results for several compression schemes applied on the enkephalin data set. The testing frame sequence was split fruther into four phases — two are fairly stable,

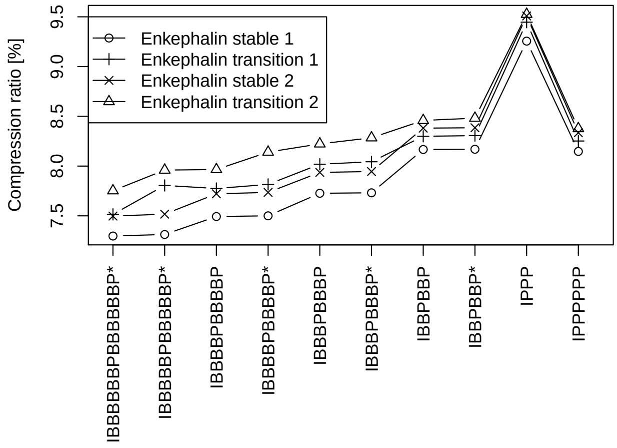

Figure 4 Comparison of compression schemes Asterisk marks disabled frame promotion.

where the shape changes are small in either unfolded or folded shape of the molecule, while two others represent folding and unfolding of the molecule. The absolute compression ratios differ between these segments, however, the trends are quite parallel.

We tested two schemes without B-frames and several with two groups of B-frames (the *IBBPBBP* type), each group having two to six B-frames. The trend visible in the figure indicates that adding groups of more than three B-frames improves the overall compression ratio w.r.t. P-frames only. The overall compression ratio approaches the ratio of the B-frames themselves (Tab. 1).

The effect of the frame promotion is also clearly observable.

4.3 Decompression time

Decompression times are, due to the algorithm, independent on the actual data sets. Using the data sets of 250 grid points in each dimension and 16bit quantization the current experimental implementation achieves 8, 2, and 3 seconds[1] for I-, P-, and B-frames at AMD Opteron 885 at 2.6 GHz with sufficiently large memory to eliminate the effects of disk I/O. The measured times correspond to the complexity of the decoding formulae of the frame types.

The times may look too large for a single frame, and they are rather far from the target real-time requirement of the haptic simulation, however, they correspond to the size of the data (approx. 100× more than HDTV).[2] However, as the current implementation is a proof of concept only, there is still room for conventional optimization of the numeric code as well as leveraging apparent data parallelism.

[1] The times do not include the expensive final inverse of the μ-law transform, which is necessary in the haptic application on only a few points — current positions of the interacting ligand atoms.

[2] Standard gzip compression takes approximately the same time with signifficantly worse compression ratio of $\sim 90\%$.

4.4 Further quantization of P- and B- frames

We carried experiments with further, more aggressive quantization of residuals of the P- and B-frames. The outcome is very destructive on P-frames because of introducing unacceptable error. The effect is not so apparent on B-frames, however, compression ratio is not improved significantly. Therefore we conclude that further quantization is not usable in the method.

5 Conclusion

The problem we approached, compression of vector field changing in time, exhibits similar properties to audio and video compression, however, it has three spatial dimensions, therefore the involved data size exceed HDTV video by approx. 100 times. We proposed a compression method inspired by both audio and video codecs, and we evaluated its properties.

The achieved compression ratio is about 7–8 % of the original data size. Despite the compression method is lossy, the induced relative error seems to be acceptable, however, the real effect on the haptic simulation still has to be evaluated. The compression ratio and data loss can be traded off by tuning compression parameters, though. Decompression times of the current naïve implementation are rather high, corresponding to the data sizes, however, we foresee room for fairly straightforward improvements by optimization of the numeric code and parallel implementation as well.

Despite the method was designed to compress electrostatic force field of molecular interaction, it does not rely on many of its specific properties. It is generally applicable on similar vector fields in other applications where lossy behaviour is allowed and where the assumptions of high dynamic range of the signal and continuous values hold.

References

1 FLAC. Flac – format, 2008. Online; last access 2009/12/23; available on: `http://flac.sourceforge.net/format.html`.

2 Mat Hans and Ronald W. Schafer. Lossless compresion of digital audio. Technical Report HPL-99-144, November 1999.

3 Esther Kellenberger, Jordi Rodrigo, Pascal Muller, and Didier Rognan. Comparative evaluation of eight docking tools for docking and virtual screening accuracy. *Proteins: Structure, Function, and Bioinformatics*, 57(2):225–242, 2004.

4 Aleš Křenek. *Towards interactive molecular models.* PhD thesis, 2005. Faculty of Informatics, Masaryk University, Brno.

5 Aleš Křenek and Jiří Filipovič. Haptics-assisted docking simulation using virtual coupling, 2010. in preparation.

6 Hiroshi Nagata, Hiroshi Mizushima, and Hiroshi Tanaka. Concept and prototype of protein-ligand docking simulator with force feedback technology. *Bioinformatics*, 18(1):140–146, 2002.

7 Miguel A. Otaduy and Ming C. Lin. Stable and responsive six-degree-of-freedom haptic manipulation using implicit integration. *World Haptics Conference*, pages 247–256, 2005.

8 R. Rice and J. Plaunt. Adaptive variable-length coding for efficient compression of spacecraft television data. *IEEE Transactions on Communication Technology*, 19(6):889–897, 1971.

9 Tony Robinson. SHORTEN: Simple lossless and near-lossless waveform compression. Technical Report TR156, December 1994.