

An Investigation of Lossless Human Movement IMU Data Compression Methods

David Chiasson¹, Junkai Xu¹, and Peter Shull¹

¹State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering, Shanghai Jiao Tong University

Abstract—Real-time kinematic data is central to many emerging technology applications, yet to date few techniques have been proposed to process and represent this relatively new information modality in an efficient manner. This paper explores techniques for the lossless compression of human kinematic IMU data. Techniques from traditional multimedia networks are compared with several novel approaches. Simple, computationally cheap techniques are shown to losslessly compress consumer-grade IMU kinematic data by a factor of ten to twenty, depending on movement activity, compared with traditional data representations. Furthermore, these simple techniques are shown to approach the optimal linear compression level. All techniques discussed in this study are implemented and released as open source C code using fixed point computation which can be integrated into a variety of computational platforms.

Index Terms—kinematic data, human movement, compression, codec

I. INTRODUCTION

MANY emerging technology applications rely on real-time kinematic data as a crucial component. These applications include virtual reality, autonomous driving, internet of things, physical therapy, wearable electronics, and human performance among others. These applications have been enabled by the recent explosion of cheap inertial based sensors (IMUs) along with mobile computation power to process this data at high sampling frequency. Kinematic data represents a nascent field of multimedia data which is starkly under-developed compared to the existing maturity of text, audio, and visual type data processing techniques. In order to enable the modern applications listed above, efficient and standard techniques for representing and processing kinematic data are needed.

This study explores a range of both traditional and novel compression techniques applied to human kinematic data gathered by 6-axis IMUs. The best performing techniques are identified and discussed. Compression addresses two common problems present in modern human movement sensing applications: limited transmission bandwidth and limited storage space. Only lossless compression techniques are considered in this study. The reason for this is that lossy compression inherently involves a value judgment about what information is useful and what information is irrelevant. In the absence of a specific application, no justification can be made for discarding any portion of information.

This study focuses on highly practical approaches which can be used in modern applications. To demonstrate this, all

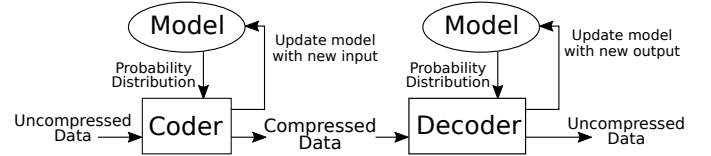


Fig. 1. General compressor and decompressor

compression techniques have been implemented and released as open-source C code using fixed point computations. The hope of the authors is that this code can be a starting point for academic or industry developers implementing some kinematic data application on diverse computational platforms.

A lossless compression algorithm can be divided into two components, modeling and coding as shown in figure 1. The model incorporates knowledge about the signal to be compressed. It estimates a probability mass function (PMF) which represents the likelihood of all possible input symbols. A dynamic model is one which changes its PMF estimate after new input symbols are received. The Coder uses the PMF produced by the model to compute a unique variable length encoding for each possible symbol. Short encodings are assigned to likely symbols, and long encodings are assigned to unlikely symbols such that the total length of the compressed data is minimized. The decompressor uses an identical model to provide the same PMF to the decoder which is used to revert each code back into the original symbol. If the model is dynamic, this output is used to update the model for future predictions.

The coding component is well understood. The theoretical limit of coding performance on a signal with a known PMF is given by Shannon's noiseless coding theorem [7] as first order entropy:

$$H = - \sum_i P_i \log_2 P_i$$

Furthermore, coding techniques have been developed which approach this limit under various conditions as early as 1952 [4]. Many near optimal techniques now exist with varying complexity and for various signal classes. Coding is thus considered a solved problem.

Modeling on the other hand must be revisited for each new signal class and application. Due to the pigeon hole principal, no algorithm can compress every possible input.

Each model must make a implicit decision about the scope of signals that will be compressed. To the authors' knowledge, no previous work has directly addressed the modeling problem for kinematic data signals. This provides the motivation for the current work.

We hypothesize that compression ratios will be comparable to those achieved in lossless audio compression. We also hypothesize that the best performing codec will utilize some cross axis correlation and be informed by physics based models.

II. METHOD

For this study, the Human Gait Database (HuGaDB) [1] is used to meaningfully and repeatably demonstrate the performance of various compression methods. HuGaDB is a public dataset of six-axis IMU signals collected from six different body segments of 18 healthy subjects performing 12 different movement activities sampled at 60Hz. This database was selected because it allows the comparison of compression techniques across body segment, subject, and activity in addition to sensor modality.

The data collected from three random subjects was designated as training data, and only data from the remaining 15 subjects was used in our results. The training data was used to train the lasso regression of our cross-stream FIR filter discussed later.

utilize predicatbility, correlations, smoothness

A. Traditional Methods

In order to provide context for the performance of compression techniques, several traditional data representation formats are chosen as reference. In this study, the baseline data format (exhibiting a compression ratio of one) is comma separated value (CSV), a simple text based format which is the de-facto standard for storing sensor information. Performance of each compression technique is evaluated by computing the compression ratio CR relative to CSV via the following formula:

$$CR = \frac{\text{size of CSV file}}{\text{size of compressed file}}$$

CSV files are ANSI encoded and formatted to have a constant length sample format to eliminate sources of randomness in our CR computation. Due to this decision, Binary format will have the same CR regardless of data properties.

To give further context to performance, established algorithms from both text and audio compression are applied to the kinematic data. In total, the following five reference encodings are used to give context to compression ratios:

- **CSV** Standard text based format. To minimize variance in csv size, each data point is formatted to the the same text length resulting in a file size of sixteen bytes per sample.
- **Binary** The optimal fixed size format. In our corpus, every sample is two bytes. This would be the optimal compression if each sample were an IID random variable uniformly distributed across the sample space.

- **ZIP compression of CSV** ZIP is a general purpose file compression format integrated into all major computer systems. ZIP was executed using the DEFLATE method [2] and a compression level of 6.
- (I could add Binary + rice encoded, FLAC, binary + zip, and binary + rice encoded + zip)

B. Tested Encodings

TODO: can I prove that the optimal IIR linear filter cannot exceed that of the optimal FIR linear filter?

Several restrictions are placed on the algorithms considered in this study. First, a viable algorithm must be causal. This is a basic requirement for an algorithm to be real-time. We also chose to only consider algorithms with zero filter delay. Since our sensors operate at a relatively low sampling frequency of 60Hz, a delay of one sample would be 16ms which is significant for modern information networks.

We also only consider algorithms which are node independent. The model for each signal considers at most the information from the six co-located signals including it's own past input. While it is likely that utilizing inter-node correlation could produce better compression ratios, especially if a human biomechanics model is introduced, such an approach would limit the usefulness of said algorithm to that specific placement of nodes on the body.

Finally, only lossless algorithms are considered. As discussed in the introduction, a lossy algorithm requires an objective metric for determining how much data loss and of what type is acceptable. Without this metric it is trivial to create a compression algorithm with an infinite compression ratio but which loses all information.

In practice, the implementation of a strictly lossless algorithm turns out to be non-trivial. This is because the industry standard for floating point computation IEEE 754 [TO CITE] is not sufficiently stringent to guarantee identical results on various implementations. For example, rounding of results may be slightly different between two computation platforms, or the same computation platform at two points in time. Additionally, a compiler or interpreter which processes the source code for an algorithm implementation will often utilize mathematical properties such as commutativity to optimize computation. This may result in different machines performing floating point operations in a different order which could lead to differing results even if each rounding operation were well defined by IEEE 754. In order to avoid both of these scenarios and guarantee identical results across diverse computational platforms, all algorithms in this study are implemented using integer operations and fixed-point 16.16 precision.[cite fixed point math library?] There is significant cost associated with this technical decision. Namely that quantization error is independent of magnitude. This can lead to significant numerical issues for complex algorithms [TO CITE]. In the context of compression, complex or numerically unstable algorithms may produce compression ratios which underestimate their understanding of the underlying signal since their implementation is not equivalent to their mathematical derivation.

The following lossless compression techniques are proposed in this study:

Derivation of our optimal linear problem and implementation

- **Delta encoding** Current sample is predicted to be equivalent to the previous sample so that the difference between the two is encoded. If a signal varies slowly with time, this signal will be smaller than the original signal.
- **Linear extrapolation** Current sample is estimated as a linear extrapolation from previous samples. Also known as first order polynomial regression.
- **2nd to 5th order polynomial regression** This technique assumes that the signal is a polynomial which is estimated from a least squares regression of past samples. This polynomial is then extended to get a prediction of the current sample.
- **Spline extrapolation** A spline is the minimum curvature piece-wise polynomial which connects a set of points. It is commonly used for interpolation, namely computer graphics smoothing. This technique was selected as splines are known to avoid Runge's phenomenon which is witnessed when extrapolating higher order polynomials. Results from the cubic spline with natural boundary conditions are presented in this paper.
- **Linearized rotating gravity** A Newtonian physics based model in which a constant magnitude acceleration vector rotates according to gyroscope readings.
- **Trained auto-correlation linear prediction** The optimal linear predictor of training data utilizing each signal's history.
- **Trained cross-correlation linear prediction** The optimal linear predictor of training data utilizing all six co-located signal histories.

C. Optimal Model

To complete the context for our proposed modeling techniques, we numerically compute the optimal linear model for our data. To do so, we will formulate our model as a linear prediction with the error, or residual signal defined as

$$\tilde{e} = Ax - b$$

TODO: handle negatives! Where each row of A is the past history of our signal, each element of b is the current sample, and x is our model. For our application, we are interested in the residual signal which can be encoded into the minimum number of bits. To compute this, we consider the effect of Rice-Golomb coding on our residual signal. A Rice-Golomb encoding of order $M \in \mathbb{N}_0$, splits each \tilde{e}_i into a quotient and remainder:

$$q_i = \lfloor \frac{\tilde{e}_i}{2^M} \rfloor, r_i = \tilde{e}_i - q_i * 2^M$$

The remainder r_i is truncated binary encoded at a fixed size of M bytes, while the quotient q_i is unary encoded. The size in bits of each Rice-Golomb encoded element of the residual signal is thus:

$$M + \lfloor \frac{\tilde{e}_i}{2^M} \rfloor + 1$$

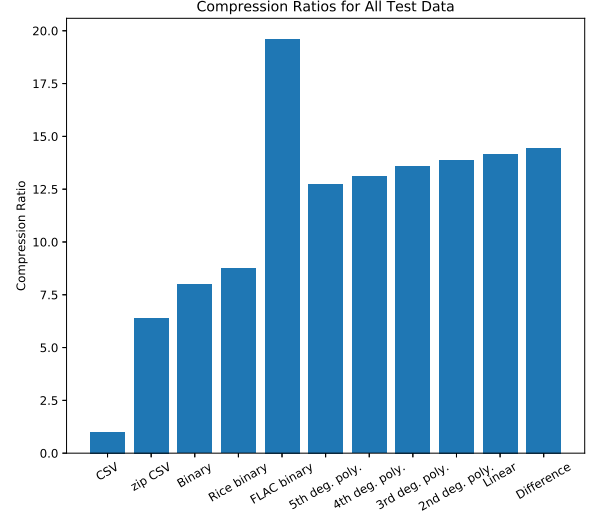


Fig. 2. Compression ratio (original size / compressed size) of each encoding technique on all test data. Larger is better.

If we relax our rounding operation, the size of each element can be approximated as a linear function of \tilde{e}_i which leads to an approximate compressed size of:

$$k + kM + 2^{-M} \sum_{i=0}^k \tilde{e}_i$$

Minimizing this value with $l-1$ normalization is equivalent to the optimization problem:

$$\underset{x}{\text{minimize}} \quad \|Ax - y\|_1 + \lambda \|x\|_1$$

$l-1$ normalization is used since it encourages sparsity in the model. Sparsity is desirable in this application as it reduces quantization error of the model coefficients and reduces the fixed point arithmetic error during execution. Both of these sources of error are significant when performing fixed point arithmetic. Since this problem is convex, it can readily be solved with a variety of numerical solvers. For this study, python bindings for the Splitting Conic Solver were used [5] [6] [3].

D. Implementation

To show algorithm performance in a realistic scenario, as well as to provide tools of benefit to the community, the algorithms in this study have been implemented in the C programming language and released as open-source code found at https://github.com/dchiasson/kinetic_codec. TODO[change the name of the repo, clean up docs, and release]. The choice of programming language as well as the restriction to integer arithmetic allow this code to be easily incorporated into programs executing on either a personal computer or embedded computation platforms, even those without a floating-point unit.

TABLE I

	Activity						Body Segment					
	All	Walk	Run	Sit	Stand	Bike	Foot		Shin		Thigh	
							R	L	R	L	R	L
CSV	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
zip CSV	3.97	3.06	3.26	6.07	6.18	3.09	3.99	4.10	3.95	3.98	3.98	3.93
Binary	8.32	8.32	8.32	8.32	8.32	8.32	8.32	8.32	8.32	8.32	8.32	8.32
Rice Binary	9.91	9.08	9.01	10.93	11.04	9.40	9.74	9.75	9.89	9.95	10.08	10.06
Spline	12.20	9.63	9.23	16.61	17.24	10.74	12.04	11.99	11.98	12.12	12.60	12.48
5th deg. poly.	11.25	8.92	8.59	15.28	15.78	9.87	11.24	11.16	11.10	11.20	11.49	11.34
4th deg. poly.	11.57	9.09	8.73	15.93	16.39	10.15	11.54	11.47	11.42	11.54	11.81	11.65
3rd deg. poly.	11.80	9.22	8.84	16.44	16.80	10.30	11.74	11.67	11.66	11.78	12.05	11.88
2nd deg. poly.	12.00	9.35	8.97	16.77	17.29	10.44	11.95	11.87	11.85	12.00	12.25	12.07
Linear	12.22	9.52	9.11	17.19	17.81	10.52	12.21	12.11	12.07	12.20	12.45	12.26
Difference	12.75	9.95	9.52	17.87	18.69	10.96	12.68	12.61	12.54	12.71	13.07	12.87
Auto-FIR	12.73											
Cross-FIR												

	Subject																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
CSV	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
zip CSV	4.0	8.4	4.0	3.5	3.3	3.5	4.1	5.3	4.3	4.6	4.1	3.5	4.5	3.3	3.3	3.8	4.3	4.8
Binary	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3	8.3
Rice Binary	10.1	10.9	9.8	9.2	9.3	10.0	9.9	10.6	9.9	10.5	10.2	9.3	10.0	9.1	9.1	9.6	9.9	10.3
Spline	12.8	20.0	11.9	10.4	10.0	11.3	12.3	15.0	12.6	13.6	12.5	10.7	13.0	10.2	10.1	11.5	12.5	13.9
5th deg. poly.	11.8	18.6	11.0	9.7	9.3	10.3	11.3	14.0	11.6	12.5	11.5	9.9	12.2	9.4	9.4	10.7	11.6	12.7
4th deg. poly.	12.2	19.8	11.3	9.9	9.4	10.6	11.6	14.5	11.9	12.9	11.8	10.1	12.5	9.5	9.6	11.0	11.9	13.0
3rd deg. poly.	12.4	20.5	11.5	10.1	9.6	10.8	11.8	14.9	12.1	13.2	12.0	10.3	12.7	9.7	9.8	11.2	12.1	13.2
2nd deg. poly.	12.6	21.3	11.7	10.2	9.7	10.9	12.0	15.2	12.4	13.5	12.3	10.5	13.0	9.8	9.9	11.4	12.4	13.5
Linear	12.8	22.4	12.0	10.4	9.9	11.1	12.3	15.5	12.6	13.7	12.5	10.6	13.3	10.0	10.1	11.6	12.6	13.8
Difference	13.3	22.6	12.5	10.8	10.3	11.6	12.8	16.1	13.2	14.4	13.1	11.1	13.8	10.6	10.6	12.1	13.2	14.6
Auto-FIR		23.6	12.4	10.3	10.2													
Cross-FIR		21.8	10.8	9.9	10.1													

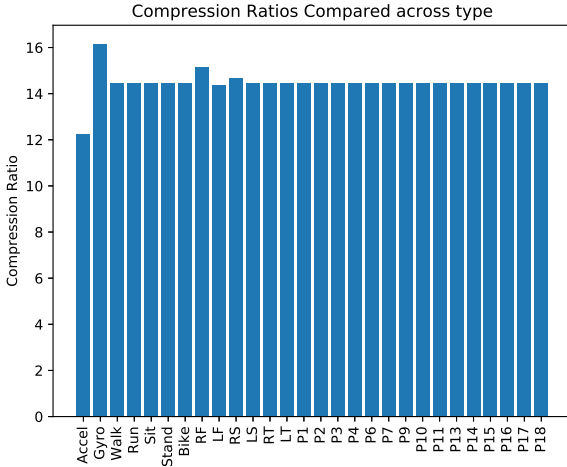
TABLE II
SUBJECT DATA

Fig. 3. Compression ratio (original size / compressed size) of each encoding technique on all test data. Larger is better.

III. RESULTS

Figure 2 Table ?? shows the best compression ratio of all test data for each encoding type. Delta encoding saw the highest compression of the polynomial regression techniques

(CR=14.457), and each higher degree polynomial performed progressively worse with 5th degree polynomial at the bottom (CR=12.740).

Table TODO shows the breakdown of CR for each modality, activity, body segment, and subject. No significant difference was found with any algorithm for compression between subject or activity. There are minor differences across body segment and major differences between sensor modality.

TODO: quantify statistical significance

Tables of detailed results k, order: total, accel/gryo, activity, segment, subject

TODO: Discussion of cross stream FIR coefficients, pole zero plots, precision effect

IV. DISCUSSION

These results show that even simple techniques result in a significant compression over the traditional encoding of CSV. If bandwidth or storage space is any concern in a kinematic data application, using a binary format would have very few disadvantages other than the loss of human-readability. Even more significant gains can be achieved with the trivial technique of delta encoding.

We were surprised by the small variation in CR across body segments, and the complete lack of variation across activity

and subject. The small differences of body segments may even be due not to the differences in movement, but due to noise and bias differences between sensor hardware, as each trial had the same sensor on the same body segment (can I confirm this?). The activity and subject results average across sensor hardware, but body segment would expose any differences.

This observation, along with the surprisingly poor performance of our physics based model suggests that the information in our signal has less to do with the human body and it's movement than previously assumed. If the movement of the human body made up the vast majority of the signal content, we would expect to see significant differences in compression ratios between running and sitting. It is possible that the bulk of the entropy in our signal is due to noise, sculling and/or coning. If this is the case, it presents a significant opportunity for lossy compression algorithms to achieve significantly more compression than shown in this study.

An alternate explanation for this result may be the aliasing of the kinematic signal. 60Hz is common for consumer grade IMUs, but it is generally agreed to be well below what is required to fully capture human movement [TO CITE]. Fast movements such as running are often sampled at 1000Hz. If the signals are significantly undersampled, we may be losing much of the high frequency information that would differentiate fast movements from slow movements. [TODO: research effect of aliasing on information content]

expected redundancy between accelerometer and gyroscope information. If such a linear redundancy exists, it is too small to be usable in this practical application.

Was poor performance of complex algorithms due to precision issues?

Surprising performance of difference encoding

Lack of correlation between accelerometer and gyroscope

Compression rate compared to audio

A. limitations

The compression ratios presented in this paper are intended to demonstrate the relative difference between compression techniques and may not be representative of the absolute CR experienced in other applications. There are many other factors which can affect the compression ratio which are not explored in this paper. Namely, sensor hardware differences of precision, noise, bias, and sampling rate are expected to have a large impact on the CR achieved. Additionally, CSV encoding which is used as the reference in this study, does not have a well defined size per sample. While in this study the samples were formatted to a fixed text length to preserve precision and cause deterministic size, the real size of a CSV file could vary dramatically with no upper limit depending on the formatting chosen. (should I leave that part out? I basically admitted that all my CR values are arbitrary...)

In the future, these techniques will be applied to various sensing hardware, sampling rates, and body segments. improvements: dynamic K, blocking for error recovery

V. CONCLUSION

Kabam! Splat! optional

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (51875347).

REFERENCES

- [1] Roman Chereshevnev and Attila Kertész-Farkas. HuGaDB: Human gait database for activity recognition from wearable inertial sensor networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10716 LNCS:131–141, 2018.
- [2] P. Deutsch. DEFLATE Compressed Data Format Specification version 1.3, 1996.
- [3] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.
- [4] David A. Huffman. A Method for the Construction of Minimum-Redundancy Codes. *Proceedings of the IRE*, 1952.
- [5] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, June 2016.
- [6] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. SCS: Splitting conic solver, version 2.1.2. <https://github.com/cvxgrp/scs>, November 2019.
- [7] C. E. Shannon. A Mathematical Theory of Communication. *Bell Syst. Tech. J.*, 27:379–423, 1948.

SCRATCH SECTION

TODO: rewrite paragraphs below, maybe drop or move to method

The goal of the model is to produce an accurate PMF with minimum entropy. Intuitively this means that compression is higher the more "sure" the model is about some occurrences over other ones. If an optimal encoder is used, the compression ratio can be regarded as a quantitative measurement of the model's understanding of the underlying signal. Because of this, we expect the performance of various signal models to give us insight into the content of the kinematic signal.

The same coder and decoder is employed by all compression techniques explored in this work, and techniques are thus identified by the model used. Models are presented as predictive algorithms. Using this paradigm, the residual is the difference between each input symbol and what the model predicted the symbol to be. The compressed size is the first order entropy of the residual signal.

Consider deleting below:

TODO: why do we assume our signal is geometrically distributed about the mean!? TODO: relationship for natural signals of residual signal and compression ratio

In this work, the maximum likelihood occurrence according to the model will be referred to as the prediction. The difference between the prediction and the actual sample will be referred to as the residual. For practical implementation, it is advantageous to convert the original signal into the residual signal before coding so that the coder can be computationally optimized to compute encodings for zero mean random variables (or same distribution? not recalculating).

Define stream? node? anything else?