Comparison of Low Bitrate Quantizers for Encoding Swedish Sign Language

Anna Klezovich[1], **Johanna Mesch**[1,2], **Gustav Eje Henter**[1], **Jonas Beskow**[1]
[1]Speech, Music and Hearing, KTH Royal Institute of Technology, [2]Stockholm University
Stockholm, Sweden
{annkle, ghe, beskow}@kth.se, johanna.mesch@su.se

Abstract

This paper investigates the bitrate–distortion trade-off of different discrete representations for Swedish Sign Language (STS) using the STS Mocap v1 motion capture dataset. We compare the K-Means algorithm with the Residual Vector Quantized Variational Autoencoder (RQ-VAE) to determine how efficiently each method preserves salient motion information at low bitrates. The results show that RQ-VAE consistently achieves lower reconstruction error than K-Means at matching bitrates, particularly for body motion, and better preserves the signing space volume. We further demonstrate that quantized representations can serve as conditioning for a flow-matching generative model, producing plausible but still imperfect sign sequences at low bitrates. These findings highlight the advantages of vector quantized models for efficient sign language motion encoding.

Keywords: compression quality trade-off, input bitrate, Swedish Sign Language, motion capture data, 3D sign language generation

1. Introduction

Having a strong representation learning model as a step before a generative model can significantly improve the quality of the generative model output. This has been shown by many recent works in the domains of sign language (Cruz and Bejarano, 2024, Xie et al., 2024, Malmberg et al., 2024), co-speech gesture (Guichoux et al., 2025), dance (Alexanderson et al., 2023, Siyao et al., 2023), and human motion generation (Jiang et al., 2023, Zhang et al., 2024), as well as in spoken speech and audio generation papers (SoundStream by Zeghidour et al. (2021), WavLM by Chen et al. (2022)). One possible explanation for this is that those learned representations are able to capture small discrete patterns relevant for each domain, for example, dance moves for dance generation or in the case of speech and sign language — phonological information. Representation learning models should ideally be trained on large amounts of data, which is why finding an optimal compression quality trade-off can potentially offer faster and more efficient training that preserves all the relevant information.

For sign language translation, promising attempts have been made to apply large language models for sign language translation (Gong et al., 2024), which requires sign language to be represented with an information rate (i.e. bitrate) of a similar order of magnitude as that of the text. This motivates our investigation in bitrate vs quality for quantized sign language representations.

Quantization implies mapping the data from a continuous space to a smaller discrete space, while preserving meaningful information. For example, Vector Quantized Variational Autoencoder (VQ-VAE) type models (van den Oord et al., 2017, (Zeghidour et al., 2021)) comprise an encoder, a codebook that learns quantized representations, and a decoder. A VQVAE model is jointly trained to condense and quantize the data into the codebook and then reconstruct it back into the original space with the decoder, optimized through a reconstruction loss. These quantized representations from the codebook are then used as inputs to generative models.

Determining the optimal size of the codebook and the lowest bitrate at which the most salient information is preserved (Malmberg et al., 2024), especially when changing between motion domains, remains a challenging task.

In this paper, we investigate two types of representation in terms of their bitrate-distortion tradeoff in the sign language domain based on the Swedish Sign Language (STS) motion capture dataset, STS Mocap v1 (Klezovich et al., 2025). We compare K-Means representations with Residual VQVAE (RQ-VAE) codebook representations (Zeghidour et al., 2021) in order to find out whether a more advanced and complex RQ-VAE outperforms K-means at the same bitrates or not.

2. Background

Multiple papers have already investigated different types of representation learning methods for the domain of sign language generation and understanding. For example, Cruz and Bejarano (2024) investigated the use of the RVQ-VAE quantizer for generative interpolation of sign language poses in

Proceedings of the 12th Workshop on the Representation and Processing of Sign Languages, pages 256–261
16 May 2026.

2D. Xie et al. (2024) proposed a VQ-VAE model called Pose-VQVAE as a step in translation tasks from sign language gloss sequences to sign language 2D poses. The authors fed the learnt sign language representations from a codebook as an input into a diffusion-type model. Another representation learning approach for sign languages is the SHuBERT model introduced in (Gueuwou et al., 2025). This is a self-supervised framework for sign language representation that adapts masked token prediction to multi-stream visual input — jointly modeling hand, face, and body pose streams via cluster prediction following a HuBERT paper idea from the audio synthesis domain (Hsu et al., 2021). In (Malmberg et al., 2024), the authors applied a representation learning model from the generic human motion domain to 2D sign language data, namely (Jiang et al., 2023), which is also a VQVAE-type model.

Beyond sign language, similar representation learning strategies have proven effective in other human motion domains. (Alexanderson et al., 2023) applied diffusion models to audio-driven motion synthesis for co-speech gesture, dance, and locomotion. Guichoux et al. (2025) also used RQ-VAE as a motion tokenizer, but in a unified co-speech gesture and audio synthesis domain.

The RQ-VAE model (Zeghidour et al., 2021) that we use in this study builds on the idea introduced in the original VQVAE model paper by van den Oord et al. (2017). RQ-VAE is designed for audio processing codecs, but has also proven effective in other domains. The difference between RQ-VAE and VQVAE is that the former has multiple subsequent quantizers, with each of them quantizing the residual error of the previous quantizers. As a result, this method performs joint compression and enhances the audio at a lower bitrates than other similar methods.

K-Means clustering (Bradley and Fayyad, 1998) offers a baseline for investigating bitrate-distortion trade-offs, as its bitrate is determined directly by the number of clusters. Even though the K-Means algorithm is a classical machine learning based quantization method that is entirely different from VQ-VAE, the CVQ-VAE Zheng and Vedaldi (2023) paper showed that the classic VQ-VAE can even benefit from merging with K-Means. The CVQ-VAE model uses online K-Means-style anchor updates to remedy a codebook collapse problem and reinitialize dead codes.

3. Methods

3.1. Motion Capture Dataset

The STS Mocap v1 dataset (Klezovich et al., 2025) used in this study comprises of approximately 4 hours of high quality motion capture data (body, hands, and face) for Swedish Sign Language (STS). The data consist of Swedish sentence translations from *8sidor* (MTM and Hillblom, 2025), a newspaper written in simplified Swedish (around 3.5 hours), and signs and sentences featuring these signs from STS dictionary (Svenskt teckenspråkslexikon, 2025) (around 30 minutes). All of the data was recorded with a single CODA (aka. Child of Deaf adult(s)) signer in consultation with a second, deaf L1 signer.

The mocap data was transformed into both 3D coordinates and joint angles (as exponential maps) with the help of the *pymo* module (Alemi, 2019). For K-means clustering and RQ-VAE quantization experiments, we used joint angles, while 3D coordinates were used in evaluation to get metrics in centimeters. We did not use 3D positions in the training/fitting stage and used 3D angles instead, because 3D positions do not convey a head rotation.

The data were split into three body parts: body, right hand, and left hand (face data was not considered for the purposes of this study). The hands and the body have different motion dynamics in sign language, the hands have more joints, move faster, and produce more complex poses. We wanted to model them separately to take into account that one of the methods could be better for the hands or for the body than the other.

3.2. RQ-VAE and K-Means Implementation

RQ-VAE in this paper is based on Python library *vector-quantize-pytorch* (Wang, 2021) implementation of residual vector quantizer. The encoder and decoder have a very simple structure: each consists of three linear layers with ReLU activations between them.

K-Means algorithm is implemented with *scikit-learn* Python library.

3.3. Bitrates Calculation

To calculate bits per second for K-Means algorithm, we employed the formula

$$bitrate = log_2(k) * fps$$

where k is the number of K-means clusters, and fps stands for frames per second.

In RQ-VAE the number of quantizers signify the number of stages of quantization and each subsequent quantizer encodes a residual error of the previous one, adding more codes and hence more bits of information per frame. So the formula for bitrate for RQ-VAE case requires multiplication by the

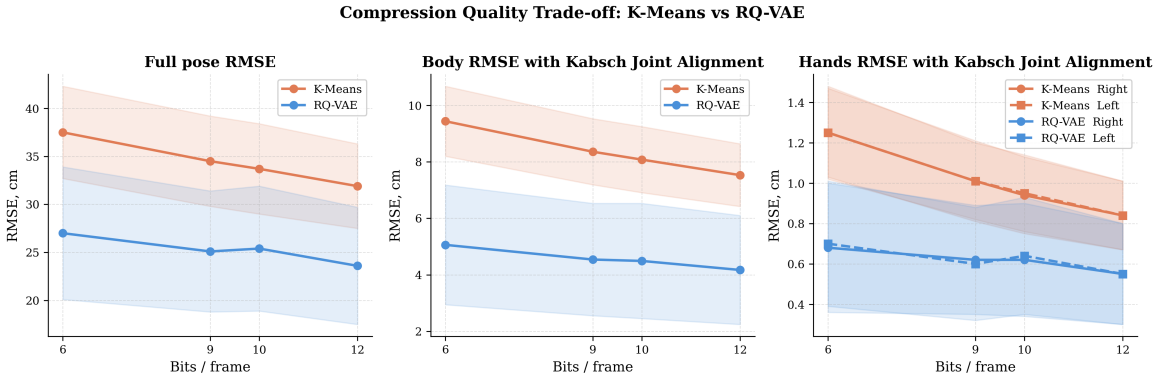

Figure 1: Bitrate–RMSE for K-Means and RQ-VAE.

Algorithm	K		Bits/frame	Bitrate
K-Means	64		6	180
K-Means	512		9	270
K-Means	1 024		10	300
K-Means	4 096		12	360
	D	**Q**		
RQ-VAE	8	2	6	180
RQ-VAE	8	3	9	270
RQ-VAE	32	2	10	300
RQ-VAE	8	4	12	360

Table 1: Bits per frame in each experiment. All experiments have the same frame rate of 30 frames per second.

number of quantizers Q (D stands for the codebook size):

$$bitrate = log_2(D) * Q * fps$$

Table 1 lists the experiments that we carried out with their hyperparameters. The lower values of codebook size D are chosen so that it is possible to match RQ-VAE experiments with K-Means in terms of bitrate. For example, bitrate of 16 would requires 65 536 clusters already, while the number of frames in the whole train set is 626 664, which is only 9.6 times higher than the number of clusters in this case. Selecting that many clusters brings this method closer to a temporal subsampling rather than clusterisation.

3.4. Evaluation Metrics

As for the reconstruction quality, we calculate the root mean square error (RMSE) on 3D positions of the original pose vs. the reconstructed pose. The RMSE values for full poses are calculated this way on the raw 3D positions and essentially show the averaged distances between joints in centimeters.

In order to have a closer look at different body parts, since the quantization was performed on them separately, we first align translation and rotation of frames for each body part with the Kabsch algorithm (Kabsch, 1976). This is conceptually similar to the Procrustes alignment, but does not do the scaling, so that the measures stay in real units, in our case centimeters. It is important to do the alignment first, because otherwise the RMSE error would say more about the global positions of body parts than the differences between the hand shapes or arms positions in the body pose. After alignment is done, we calculate the RMSE in the same way for the aligned body poses and the aligned hand poses.

Last but not least, we use an expressivity-related metric. For that we calculate the convex hull of the wrist joints point cloud. The intuition is the bigger the volume of this space, the more expressive the hands are. Convex hull volume was calculated with the help of *scipy* Python library. We first extract wrists 3D coordinates both from reconstructed sequences and original. Then convex hull volume is calculated over a point cloud of these coordinates for each animation sequence. For better interpretability, we present these values as a percentage: $(V_{reconstructed} \div V_{original}) * 100$. This value would answer how much of the original signing space the hands are taking up in the reconstructions.

4. Results

The comparison of K-Means with RQ-VAE in Figure 1 shows that RQ-VAE outperforms K-Means in most scenarios in terms of compression quality trade-off. For the full pose RMSE calculated on raw 3D coordinates, RQ-VAE performed slightly better than K-Means for all bitrates, but more so for lower bitrates. If we look at body RMSEs calculated with Kabsch joint alignment, RQ-VAE showed consistently lower RMSEs with no overlap of standard errors, on average 1.8 times lower than for K-means at the same bitrates. The picture is more interest-

Bits/frame	K-Means: Convex Hull Volume	RQ-VAE: Convex Hull Volume
6	55.3%	68.2%
9	76.6%	91.3%
10	81.6%	86.8%
12	87.9%	109.3%

Table 2: Percentage of used space out of total space of motion. For reference 0.81 m^3 is a ground truth average convex hull volume of the wrists.

ing for the hands RMSEs (also done with Kabsch joint alignment) — the results overlapped more at higher bitrates, while RQ-VAe was still on average better than K-Means. Given that we used very low bitrates in our experiments, and according to our visual inspection of the reconstructions, the most likely explanation is that both methods fall equally short in reconstructing hand poses. The error for the hands was in general smaller than for the body for both K-Means and RQ-VAE at all bitrates, but this is because the hands perform smaller motions than the body.

We rendered[1] two sequences into a video for a visual comparison with the original sequence. This video compares side-by-side sequences reconstructed for 10 bits per frame, namely one sequence created with K-Means 1 024 clusters, and another one created with RQ-VAE with 32 codebook size and 2 quantizers. In a still frame from this visualization in figure 2, aside from obvious errors due to the generally low bitrate, the signing space is visually smaller in both of the reconstructions compared to the original. To compare the signing space size across our methods, we calculated the convex hull volume of the hands point clouds based on 3D coordinates of the wrists and the percentages of space volume used by the hands in the reconstruction compared to the original space volume. The results are given in Table 2. They show that the percentage of the convex hull volume used increases as the bitrate increases. For K-Means, this process was relatively linear, while for RQ-VAE, it was not, and it might also correlate with the choice of codebook size and other hyperparameters. The Table also shows that the percentage of used convex hull volume in RQ-VAE reconstructions was higher than with K-Means at the same bitrates.

To demonstrate that these quantized sequences do in fact produce reasonable generated animations after passing them as input to diffusion-type models, we trained a flow-matching model. To be more precise, we employed an OT-CFM model introduced by (Lipman et al., 2022) and adaped in Mehta et al. (2024)'s Matcha-TTSG model. As the input, we took angular (exponential maps) cluster centroid for each frame from K-means with $k = 1\,024$. It showed promising results with a smooth motion[2]. Even though the synthesized motion resembled the input sequence from a test set,

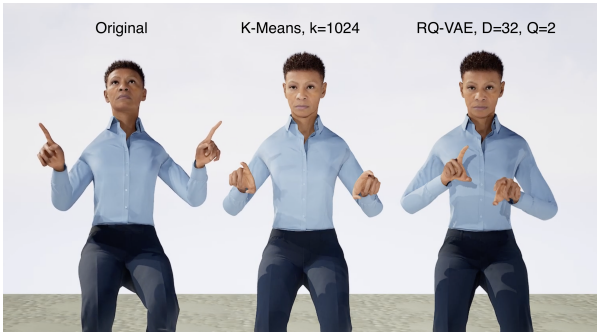

Figure 2: Still frame out of a video render. Left to right: original mocap sequence; reconstruction of that sequence with K-means, $k = 1\,024$; reconstruction of that sequence with RQ-VAE, $D = 32$, $Q = 2$

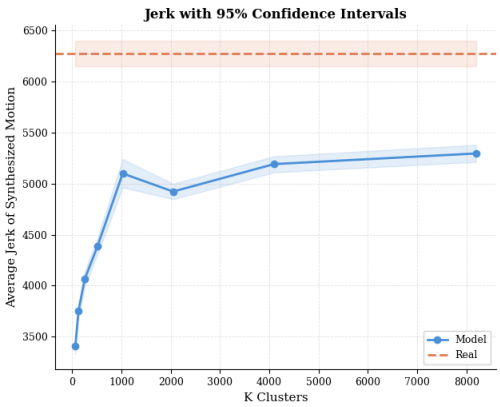

Figure 3: Jerk, CI Vs. number of K-Means clusters, trained and clustered on 2D positions. For comparison the jerk on the original dataset is 6 275.

[1] Link to reconstructions example render. The data is rendered out in Unreal Engine 5.4.4 with the help of Metahuman avatars (Epic Games, 2025). Left to right: original mocap sequence; reconstruction of that sequence with K-means, $k = 1\,024$; reconstruction of that sequence with RQ-VAE, $D = 32$, $Q = 2$

[2] Link - Original mocap sequence Vs. sequence synthesized with the model trained on K-means, $k = 1\,024$ angular cluster centroids.

it had too many errors in reconstructing the handshapes and overall motion to be semantically and phonologically correct. For a better result, it is important to have a stronger representation learning model both in terms of complexity and bitrate.

Since we were testing a very low number of clusters and codebook sizes to match the bitrates for the main compression–quality trade-off experiment, we also examined whether the metrics on the model synthesis Vs. original test data grow linearly or not at these low bitrates. Figure 3 shows that K-Means started producing results closer to the real data distribution in terms of the average jerk only at 1 024 clusters (the "knee" on the graph). The average jerk over the whole dataset in 2D was 6 275, CI [6 150, 6 401]. OT-CFM models conditioned on values lower than 1 024 did not capture enough information and their synthesized sequences were closer to the mean pose than to the data. Given that K-Means plateaus after 1 024 clusters in a 2D conditioning setting and also does not produce entirely accurate poses in either 2D conditioned or exponential maps conditioned settings, it is possible that this method reached its limitations when applied to this complex data. However, VQ-VAE type models, such as RQ-VAE of sufficient size is a promising future research direction.

5. Conclusions

In this paper, we compared two quantization methods — the K-Means algorithm and the RQ-VAE qunatizer — in terms of how well they encode Swedish Sign Language motion capture sentences at different bitrates. We compared them at four bitrates between 6 and 12 bits per frame. The results showed that RQ-VAE quantization outperforms the K-Means type quantization at matching bitrates in terms of RMSE in 3D joint space; it did so more strongly for the body than for the hands joints. When comparing the wrist motion expressivity of reconstructions Vs. the original wrist motions, we found that RQ-VAE used a higher percentage of signing space than K-Means at matching bitrates. For K-Means, the percentage of used signing space is with the growing bitrate, while the same is not true for RQ-VAE. This means other factors must be affecting this metric, such as the codebook size of other hyperparametres.

We also demonstrated that these sign language representations can be used for the sign language generation downstream task. To do so, we trained a flow-matching generative model, Matcha-TTSG (Mehta et al., 2024), on K-Means cluster centroids in exponential maps 3D space, at $k = 1\,024$. This model synthesized reasonable sign language sequences that resembled the original sequences. However, it still made errors in hand shape and body pose, because the bitrates used were too low for this task.

6. Discussion and Limitations

Even though the sign language generation task possibly requires a stronger quantizer at a higher bitrate, the lack of expressivity highlighted in this study is not an issue for diffusion and flow-matching models as long as the quantized motion does not have erroneous poses in it. Diffusion and flow-matching models can be used together with the classifier-free guidance method (Lipman et al., 2022), which helps to extrapolate the synthesis towards a stronger conditioning using unconditional model outputs.

In the future perspective, it would also be interesting to finetune an LLM to predict these discrete representation from text and use them for machine translation downstream task, from text to 3D signing. This text-to-sign experiment could help evaluate the quality of representations from a perspective of linguistic usefulness. In a current experiment setup it is possible to compare bitrate only against reconstruction quality of the 3D motion, which does not necessarily mean that these representations preserve semantically and phonologically relevant information.

7. References

Omid Alemi. 2019. PyMO: Motion capture library. Accessed: 2025-10-14.

Simon Alexanderson, Rajmund Nagy, Jonas Beskow, and Gustav Eje Henter. 2023. Listen, Denoise, Action! Audio-Driven Motion Synthesis with Diffusion Models. *ACM Trans. Graph.*, 42(4).

Paul S. Bradley and Usama M. Fayyad. 1998. Refining Initial Points for K-Means Clustering. In *International Conference on Machine Learning*.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. 2022. WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518.

Fidel Omar Tito Cruz and Gissella Bejarano. 2024. Generative Interpolation of Sign Language Poses using RVQ-VAE. In *Latinx in AI @ NeurIPS 2024*.

Epic Games. 2025. Metahuman creator overview. Accessed: 2025-06-05.

Jia Gong, Lin Geng Foo, Yixuan He, Hossein Rahmani, and Jun Liu. 2024. LLMs are Good Sign Language Translators. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18362–18372, Los Alamitos, CA, USA. IEEE Computer Society.

Shester Gueuwou, Xiaodan Du, Greg Shakhnarovich, Karen Livescu, and Alexander H Liu. 2025. SHuBERT: Self-Supervised Sign Language Representation Learning via Multi-Stream Cluster Prediction. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28792–28810.

Téo Guichoux, Théodor Lemerle, Shivam Mehta, Jonas Beskow, Gustav Eje Henter, Laure Soulier, Catherine Pelachaud, and Nicolas Obin. 2025. Gelina: Unified Speech and Gesture Synthesis via Interleaved Token Prediction. *arXiv:2510.12834*. Working paper or preprint.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 29:3451–3460.

Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. 2023. MotionGPT: Human Motion as a Foreign Language. *Advances in Neural Information Processing Systems*, 36:20067–20079.

Wolfgang Kabsch. 1976. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923.

Anna Klezovich, Johanna Mesch, and Jonas Beskow. 2025. Motion Capture Driven Avatars for Swedish Sign language. In *Adjunct Proceedings of the 25th ACM International Conference on Intelligent Virtual Agents*, IVA Adjunct '25, New York, NY, USA. Association for Computing Machinery.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. 2022. Flow Matching for Generative Modeling. *arXiv:2210.02747*.

Fredrik Malmberg, Anna Klezovich, Johanna Mesch, and Jonas Beskow. 2024. Exploring Latent Sign Language Representations with Isolated Signs, Sentences and In-the-Wild Data. In *Proceedings of the LREC-COLING 2024 11th Workshop on the Representation and Processing of Sign Languages: Evaluation of Sign Language Resources*, pages 219–224, Torino, Italia. ELRA and ICCL.

Shivam Mehta, Ruibo Tu, Simon Alexanderson, Jonas Beskow, Eva Szekely, and Gustav Henter. 2024. Unified Speech and Gesture Synthesis Using Flow Matching. In *2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2024)*, pages 8220–8224.

Myndigheten MTM and Marie Hillblom. 2025. 8 Sidor: Nyheter på lätt svenska. Accessed: 2025-10-14.

Li Siyao, Weijiang Yu, Tianpei Gu, Chunze Lin, Quan Wang, Chen Qian, Chen Change Loy, and Ziwei Liu. 2023. Bailando ++: 3D Dance GPT with Choreographic Memory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–15.

Svenskt teckenspråkslexikon. 2025. Swedish Sign Language Dictionary online. Accessed: 2025-10-14.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural Discrete Representation Learning. *Advances in Neural Information Processing Systems*, 30.

Phil Wang. 2021. Vector Quantize PyTorch. `https://github.com/lucidrains/vector-quantize-pytorch`. Accessed: 2026-02-20.

Pan Xie, Qipeng Zhang, Peng Taiying, Hao Tang, Yao Du, and Zexian Li. 2024. G2P-DDM: Generating Sign Pose Sequence from Gloss Sequence with Discrete Diffusion Model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 6234–6242.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2021. SoundStream: An End-to-End Neural Audio Codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507.

Yaqi Zhang, Di Huang, Bin Liu, Shixiang Tang, Yan Lu, Lu Chen, Lei Bai, Qi Chu, Nenghai Yu, and Wanli Ouyang. 2024. MotionGPT: Finetuned LLMs Are General-Purpose Motion Generators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7368–7376.

Chuanxia Zheng and Andrea Vedaldi. 2023. Online Clustered Codebook. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22798–22807.