Generative Interpolation of Sign Language Poses using RVQ-VAE

Fidel Omar Tito Cruz
Universidad Nacional de Ingenieria
Lima, Perú
ftitoc@uni.pe

Gissella Bejarano
Marist College
Poughkeepsie, NY - USA
gissella.bejarano@marist.edu

Abstract

In Sign Language Production (SLP) tasks, a common approach is to have individual sign language words and then concatenate their motion representation to form complete sentences. However, this process poses challenges due to missing frames in the middle, which lead to abrupt transitions and reduced smoothness, making the resulting sequences difficult to interpret. To address this issue, this paper presents a Residual Vector Quantized Variational Autoencoder (RVQVAE) model for interpolating 2D keypoint motion in videos. Our experiments simulate individual sign transitions by randomly hiding groups of frames within a sequence of video keypoints. The proposed model is evaluated by comparing its performance to a baseline method on frames hidden. Improvements in matrix distance errors and dynamic time-warping metrics demonstrate that the RVQVAE model produces promising results for generating intermediate frames. These findings highlight the potential for developing applications that enhance sign language production to benefit the deaf community.

1 Introduction

Research in Sign Language has gained significant attention in recent years, particularly in applications such as Sign Language Recognition and Transcription. However, progress in Sign Language Production (SLP) approaches remains limited. Current techniques include generating full skeleton sequences from a textual vocabulary (text-to-pose) Fang et al. [2024, 2023], Saunders et al. [2020], Zelinka and Kanis [2020] or incorporating intermediate steps like glosses. With over 100 different sign languages worldwide, these techniques face challenges due to the specificity of large datasets for each language Fang et al. [2024], Camgoz et al. [2018], Duarte et al. [2021], making it difficult to generalize motion generation across languages (e.g., using a model trained on American Sign Language to generate motions for German Sign Language). Another common approach involves constructing a vocabulary of words with their corresponding frame sequences, which are then stitched together to form complete sentences Saunders et al. [2022], Walsh et al. [2024]. This method leverages pre-prepared, co-articulated motion, requiring only the generation of missing intermediate frames between stitched segments.

In this context, motion interpolation plays a crucial role in generating transition frames between two discrete movements. Traditional methods for producing sparse keypoints, such as linear interpolation or spherical linear interpolation Iyer et al. [2021], Naert et al. [2021], treat each joint as an independent variable, overlooking the relationships between connected joints and resulting in unnatural movements. To address this limitation, deep learning-based approaches consider the relationships among keypoints and the complex patterns of motion to enhance the expressiveness of the sequences Zeng et al. [2020], Gil-Martín et al. [2023].

Latinx in AI @ NeurIPS-24 (LXAI 2024).

In this paper, we present a Residual Vector Quantized Variational Autoencoder (RVQ-VAE) model to effectively generate intermediate frames in motion. This RVQ technique encodes data with missing frames into discrete representations and employs iterative rounds of residual quantization to progressively reduce quantization errors Guo et al. [2024], Lee et al. [2022]. Subsequently, a decoder is then used to reconstruct the data, completing the missing frames. To evaluate the robustness of the method, we conduct experiments on videos with randomly hidden frames, introducing challenging scenarios where the missing frames may occur in the middle of an action rather than just during transition movements. Additionally, we train and test our model using the How2Sign dataset Duarte et al. [2021]. However, since the method focuses on motion interpolation, it is not limited to any specific sign language and can be applied across different sign languages.

2 Related Work

Sign language has been a subject of study for many years Bauer et al. [2000], and advancements in AI, particularly with encoder-decoder architectures Vaswani [2017], Kingma et al. [2019] and generative models Goodfellow et al. [2020], Ho et al. [2020], have driven significant progress in sign language production (SLP) research.

2.1 Sign Language Production

The state of the art has approached this problem in diverse ways. Some methods focus on generating continuous skeleton pose sequences or photorealistic videos from text Fang et al. [2024, 2023], Saunders et al. [2020], Zelinka and Kanis [2020], while others use sign language videos as input Dong et al. [2024], Suo et al. [2024]. Most of these approaches adopt an end-to-end pipeline, although some work introduce intermediate steps such as translating text to glosses before producing final outputs Saunders et al. [2022], Walsh et al. [2024]. Alongside these novel techniques, new datasets have been developed to support SLP research Fang et al. [2024], Camgoz et al. [2018], Duarte et al. [2021].

2.2 Motion Generation

Whether starting from an initial pose Wang et al. [2020] or a text description Ahuja and Morency [2019], Ghosh et al. [2021], generating 2D or 3D motion sequences has long been a challenging task due to the diversity of dynamic movement patterns. In recent years, generative models, such as VAE, have demonstrated impressive results in the field of 3D human motion generation Guo et al. [2022], Petrovich et al. [2021], Guo et al. [2024]. In particular, VAEs have also been utilized for SLP applications, enabling the generation of complete movement sequences Xie et al. [2023]. Although generating an entire motion sequence from scratch is difficult, it is not always necessary. Motion interpolation, which creates sequences by iteratively predicting intermediate frames based on previously observed ones, presents a practical alternative. This technique can be applied using mathematical approaches Iyer et al. [2021], Naert et al. [2021] or deep learning-based methods Zeng et al. [2020], Gil-Martín et al. [2023], making it a valuable tool to produce accurate motion sequences.

3 Methodology

3.1 Residual Quantized VAE Model

A Variational Autoencoder (VAE) consists of a neural network architecture where the encoder approximates the posterior distribution over latent random variables, which is then used by the decoder to reconstruct the input Kingma et al. [2019]. In contrast, the Vector Quantization (VQ) approach replaces the continuous latent space with discrete latent variables, converting the posterior and prior distributions into discrete categories Van Den Oord et al. [2017]. This transformation is presented in MoMask Guo et al. [2024], which enables the conversion of motion sequences into discrete motion tokens.

First, given a sequence of motion keypoints $m_{1:N} \in \mathbb{R}^{N \times D}$, where N is the number of frames and D is the dimension of the vector representation containing the keypoints, the sequence is encoded with encoder E into a latent vector z(x) . This vector is then replaced with the nearest point from a finite

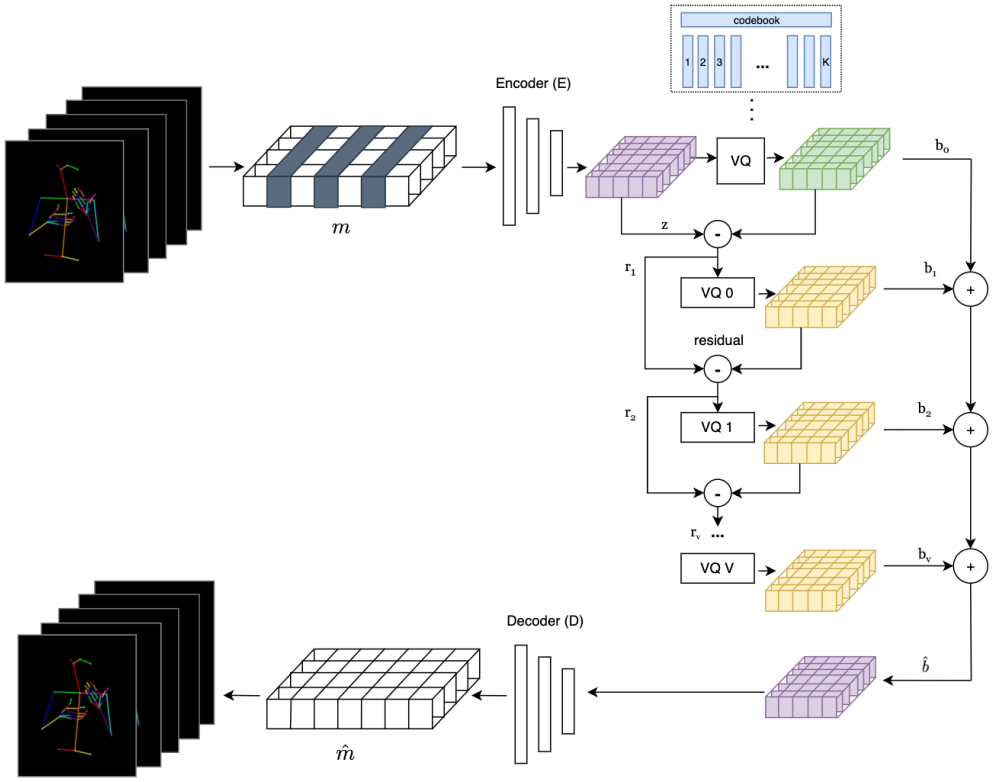

Figure 1: RVQ-VAE architecture

codebook $C = \{c_k\}_{k=1}^{K}$ and code embedding e(k) , resulting in a discrete latent representation $\hat{z}$ of the input, where:

$$b = Q(z; C) = \arg\min_{k \in [K]} \|z - e(k)\|_2^2$$

However, quantization introduces lossy compression during the transformation to discrete space. To address this, residual quantization (RQ) is used to approximate a more precise latent vector without expanding the embedding space. RQ employs V layers of depth, each iteratively calculating an approximation of the residual r_v as follows:

For the $v = 0$:

$$r_0 = z$$

For $v = 1, ..., V$:

$$b_v = Q(r_v), \quad r_v = r_{v-1} - b_v$$

After computing across all V layers, the final discrete representation $\hat{b}$ is obtained by summing the quantized residuals, resulting in a discrete code map that provides a finer approximation. The codebook can be shared across layers, though it's also possible to construct separate embedding spaces for each depth v .

Finally, the decoder D reconstructs the motion input, filling the missing intermediate frames, yielding the motion vector $\hat{m}_{1:N} \in \mathbb{R}^{N \times D}$. During training, the weighted sum of reconstruction loss and commitment loss is used in the gradient descent process, defined as:

$$Loss_{RVQ} = \|m - \hat{m}\|_1 + \beta \sum_{v=1}^{V} \|r_v - sg[b_v]\|_2^2$$

where $sg[\cdot]$ represents the stop-gradient operator, and β is a weighting factor that constrains the embedding.

3.2 Preprocessing Strategy

During training Each video in the training set, containing T frames, is processed using sliding windows of size N , resulting in $T - N$ motion samples of constant length N per video. To simulate missing intermediate frames, we randomly select $g \in [2, 4]$ groups, where each group corresponds to h intermediate frames to be hidden within the motion sequence, which are used to prevent the model from learning to predict a fixed range of missing frames. This results in a total number of hidden frames f_{hidden} ranging from 2h to 4h, constrained by the motion length. We experimented with different values of h from the set [5, 10, 15, 20] to evaluate interpolation performance for both short-term and long-term missing frames. This analysis helps identify the most suitable h values for real-world interpolation scenarios, offering insights for future methods. To simulate a hidden frame, all keypoint values for that frame are replaced with a constant. Before the motion data are input into the model, the data is normalized using the mean and standard deviation computed from the entire training set.

During testing The full sequence length of each video in the test set is used without applying sliding windows, i.e., $T = N$. For model evaluation, the same frame-hiding strategy and normalization are applied to the input sequences to ensure consistency between training and testing phases.

3.3 Baseline Model

To enable the comparison of our evaluation results, we construct a baseline model. This rule-based model processes the motion input with grouped hidden frames by replacing all hidden values with the corresponding values from the last visible keypoint vector. This approach serves as a useful benchmark, as it maintains continuity in the motion sequence while simplifying the reconstruction of hidden frames by using the last observed keypoint values and mimicking a motion flow

4 Experiments

4.1 Experimental Setup

Dataset Our training and testing were conducted using the widely adopted How2Sign dataset Duarte et al. [2021], an American Sign Language dataset released in 2021. We utilized 2,343 video recordings, with 1,897 allocated to the training set, 93 for validation, and 353 for testing. The How2Sign dataset was generated by applying the OpenPose framework Cao et al. [2017] to extract keypoints from sign language videos, providing 2D joint positions for the body, hands, and face. These keypoint estimations, released by the dataset authors, serve as ground-truth labels in our study. Since the How2Sign dataset does not contain individual words to concatenate into sentences, but rather complete phrases, we simulate sign transitions by applying the preprocessing hide strategy described in Section 3.2. In our experiments, we specifically focused on the body and hands, resulting in 67 keypoints (25 for the body and 21 for each hand), which were reshaped into a tensor of size (N, 134). Furthermore, to ensure data is prepared for sliding windows, only videos with more than 32 frames were included in both the training and validation sets.

Implementation details The RVQ-VAE model is implemented using PyTorch. Both the encoder and decoder employ 1D convolutional layers, ReLU activations between layers Agarap [2018], and a downsampling rate of 2. The codebook consists of $K \in \mathbb{R}^{512\times512}$. We use 3 quantizers layers, each trained without shared codebooks, allowing for more precise latent encoding. A quantization dropout ratio of 0.2 is applied to prevent overfitting. For training, a sliding window of size 32 frames is used. Optimization is performed using the AdamW optimizer with a batch size of 256, and the learning rate reach 2×10^{-4} after 1,500 iterations with a linear warm-up schedule. Training runs for a maximum of 60 epochs on a single NVIDIA A100 GPU.

4.2 Quantitative results

Error metrics L_1, L_2, and L_3 are employed to evaluate the discrepancies between the original videos and the motion sequences with predicted intermediate frames. Given that the predicted motion may not perfectly align with the original, we also utilize the Dynamic Time Warping (DTW) metric

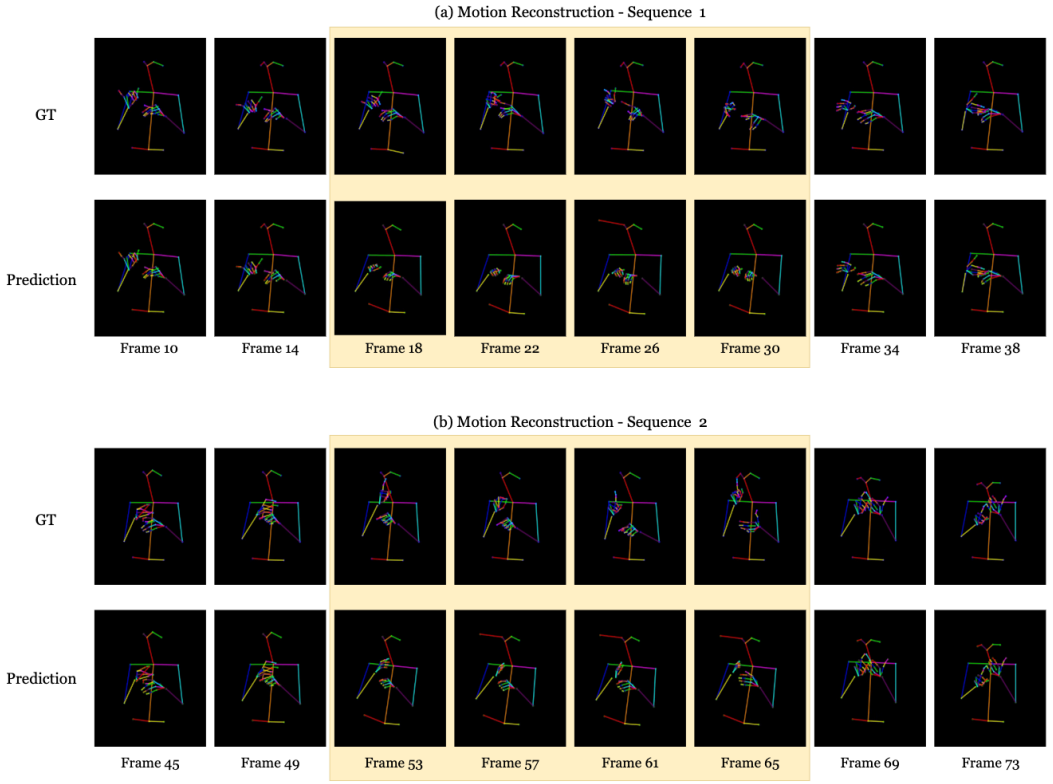

Figure 2: Qualitative comparison of (a) Example Sequence 1, and (b) Example Sequence 2. Highlighted frames illustrate the model's interpolation of the hidden frames.

Müller [2007] to determine the optimal alignment between the two sequences, allowing for a more accurate computation of the distance error L_2 after alignment. The results of these comparisons are summarized in Table 1, which includes a test set consisting of 353 videos. Since no slicing of frames is applied to these videos, the input and output shapes during inference remain consistent. The results presented include both the mean and standard deviation for each distance metric. Additionally, we experiment with the impact of varying the number of hidden frames, denoted by h, by training separate models for each configuration. This approach allows us to assess how the increasing extent of frame masking affects the quality of motion generation and to identify a range of values that can be applied without significantly increasing the error.

Table 1: Comparison of error metrics (values scaled to 10^{-3})

h	Model	L1 ↓	L2 ↓	L3 ↓	DTW ↓
5	Baseline	82.95 ± 118.17	3.80 ± 4.37	1.63 ± 1.48	47.25 ± 82.06
	VQ	38.75 ± 18.18	2.23 ± 1.23	1.20 ± 0.68	24.76 ± 10.96
10	Baseline	187.25 ± 245.10	5.86 ± 6.07	2.21 ± 1.80	106.41 ± 174.43
	VQ	88.9 ± 39.00	3.27 ± 1.54	1.51 ± 0.75	55.07 ± 23.47
15	Baseline	289.70 ± 393.38	7.29 ± 7.35	2.57 ± 1.99	163.16 ± 281.07
	VQ	144.80 ± 80.17	4.29 ± 2.07	1.83 ± 0.86	88.21 ± 49.03
20	Baseline	354.81 ± 416.02	8.05 ± 7.49	2.75 ± 1.99	196.00 ± 292.99
	VQ	207.70 ± 104.89	5.36 ± 2.30	2.13 ± 0.92	124.83 ± 63.06

Table 1 demonstrates that the VQ model outperforms the baseline model across all error metrics, as indicated by lower mean values. Additionally, the VQ model displays lower standard deviations in most cases, indicating more consistent performance compared to the baseline model.

4.3 Qualitative results

Figure 2 provides a qualitative comparison of motion interpolation with hidden frames. The results demonstrate effective interpolation for body keypoints, with smooth transitions capturing the overall motion trajectory. In Example Sequence 1, the model interpolates between two distinct signs, highlighting its potential for generating complete sign language sequences. This capability allows using ground truth from individual signs and interpolating missing frames, without requiring a large dataset for training on a specific language.

For hand keypoints, while transitions between discrete movements are smooth, the model may not accurately predict specific finger movements, particularly when hidden frames fall in the middle of a sign gesture, as happens in Example Sequence 2. This is likely due to the complexity of finger articulation during sign language. Additionally, we observe occasional anomalies, such as in Example Sequence 2, where some head keypoints deviate from the expected alignment. This can be attributed to the inherent limitations of the training dataset, which was generated using pose estimation inference from OpenPose, resulting in some ground-truth data containing errors. Errors in the ground-truth data caused by pose estimation inaccuracies may propagate during training, affecting the model's motion interpolation performance and highlighting the importance of high-quality individual signs for effective interpolation.

5 Conclusion

This research introduced an RVQ-VAE model for sign language motion interpolation and evaluated its performance using distance error metrics. Our model demonstrated strong qualitative and quantitative results, outperforming the baseline rule-based model. Additionally, we tested the model's ability to handle varying numbers of hidden frames, showing that it can predict motion effectively, even when larger sequence groups of frames are missing.

The impact of this experimentation highlights the model's ability to interpolate smooth transitions between frames, providing valuable insights for potential gloss-to-pose sign language production (SLP) applications. By leveraging a dictionary of expressive gloss-poses, the model could facilitate more natural and accurate sign language generation without needing large dataset per specific language. In future work, we aim to further develop this approach by incorporating 3D predictions to animate realistic human characters to build a fully SLP system that would benefit the deaf community. Future evaluation metrics will also include back-translation of the generated poses, which will be assessed to demonstrate that the generated poses preserve the meaning of individual signs within complete sentences.

References

A. Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

C. Ahuja and L.-P. Morency. Language2pose: Natural language grounded pose forecasting. In *2019 International Conference on 3D Vision (3DV)*, pages 719–728. IEEE, 2019.

B. Bauer, H. Hienz, and K.-F. Kraiss. Video-based continuous sign language recognition using statistical methods. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 2, pages 463–466. IEEE, 2000.

N. C. Camgoz, S. Hadfield, O. Koller, H. Ney, and R. Bowden. Neural sign language translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7784–7793, 2018.

Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017.

L. Dong, L. Chaudhary, F. Xu, X. Wang, M. Lary, and I. Nwogu. Signavatar: Sign language 3d motion reconstruction and generation. *arXiv preprint arXiv:2405.07974*, 2024.

A. Duarte, S. Palaskar, L. Ventura, D. Ghadiyaram, K. DeHaan, F. Metze, J. Torres, and X. Giro-i Nieto. How2sign: a large-scale multimodal dataset for continuous american sign language. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2735–2744, 2021.

S. Fang, C. Sui, X. Zhang, and Y. Tian. Signdiff: Learning diffusion models for american sign language production. *arXiv preprint arXiv:2308.16082*, 2023.

S. Fang, L. Wang, C. Zheng, Y. Tian, and C. Chen. Signllm: Sign languages production large language models. *arXiv preprint arXiv:2405.10718*, 2024.

A. Ghosh, N. Cheema, C. Oguz, C. Theobalt, and P. Slusallek. Synthesis of compositional animations from textual descriptions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1396–1406, 2021.

M. Gil-Martín, M. Villa-Monedero, A. Pomirski, D. Sáez-Trigueros, and R. San-Segundo. Sign language motion generation from sign characteristics. *Sensors*, 23(23):9365, 2023.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

C. Guo, X. Zuo, S. Wang, X. Liu, S. Zou, M. Gong, and L. Cheng. Action2video: Generating videos of human 3d actions. *International Journal of Computer Vision*, 130(2):285–315, 2022.

C. Guo, Y. Mu, M. G. Javed, S. Wang, and L. Cheng. Momask: Generative masked modeling of 3d human motions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1900–1910, 2024.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

S. J. Iyer, P. Saranya, and M. Sivaram. Human pose-estimation and low-cost interpolation for text to indian sign language. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 130–135. IEEE, 2021.

D. P. Kingma, M. Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

D. Lee, C. Kim, S. Kim, M. Cho, and W.-S. Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532, 2022.

M. Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

L. Naert, C. Larboulette, and S. Gibet. Motion synthesis and editing for the generation of new sign language content: Building new signs with phonological recombination. *Machine Translation*, 35(3):405–430, 2021.

M. Petrovich, M. J. Black, and G. Varol. Action-conditioned 3d human motion synthesis with transformer vae. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10985–10995, 2021.

B. Saunders, N. C. Camgoz, and R. Bowden. Everybody sign now: Translating spoken language to photo realistic sign language video. *arXiv preprint arXiv:2011.09846*, 2020.

B. Saunders, N. C. Camgoz, and R. Bowden. Signing at scale: Learning to co-articulate signs for large-scale photo-realistic sign language production. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5141–5151, 2022.

Y. Suo, Z. Zheng, X. Wang, B. Zhang, and Y. Yang. Jointly harnessing prior structures and temporal consistency for sign language video generation. *ACM Transactions on Multimedia Computing, Communications and Applications*, 20(6):1–18, 2024.

A. Van Den Oord, O. Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

A. Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

H. Walsh, B. Saunders, and R. Bowden. Sign stitching: A novel approach to sign language production. *arXiv preprint arXiv:2405.07663*, 2024.

Z. Wang, P. Yu, Y. Zhao, R. Zhang, Y. Zhou, J. Yuan, and C. Chen. Learning diverse stochastic human-action generators by learning smooth latent transitions. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 12281–12288, 2020.

P. Xie, Q. Zhang, T. Peng, H. Tang, Y. Du, and Z. Li. G2p-ddm: Generating sign pose sequence from gloss sequence with discrete diffusion model, 2023. URL `https://arxiv.org/abs/2208.09141`.

J. Zelinka and J. Kanis. Neural sign language synthesis: Words are our glosses. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3395–3403, 2020.

N. Zeng, Y. Chen, Y. Gu, D. Liu, and Y. Xing. Highly fluent sign language synthesis based on variable motion frame interpolation. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1772–1777. IEEE, 2020.