

ĐẠI HỌC QUỐC GIA TP. HCM
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

HOÀNG MINH THANH

OPENHUMAN: HỆ THỐNG TỔNG HỢP
CỦ CHỈ HỘI THOẠI DỰA TRÊN CẢM
XÚC VÀ NGỮ NGHĨA

LUẬN VĂN THẠC SĨ

TP. Hồ Chí Minh – Năm 2024

ĐẠI HỌC QUỐC GIA TP. HCM
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

HOÀNG MINH THANH

**OPENHUMAN: HỆ THỐNG TỔNG HỢP CỦ CHỈ
HỘI THOẠI DỰA TRÊN CẢM XÚC VÀ NGỮ NGHĨA**

Ngành: Khoa học máy tính

Mã số Ngành: 8480101

NGƯỜI HƯỚNG DẪN KHOA HỌC
HDC: PSG. TS. LÝ QUỐC NGỌC

TP. Hồ Chí Minh – Năm 2024

LỜI CAM ĐOAN

Tôi cam đoan luận văn thạc sĩ ngành Khoa học máy tính, với đề tài OpenHuman: Hệ thống tổng hợp cử chỉ hội thoại dựa trên cảm xúc và ngữ nghĩa là công trình khoa học do tôi thực hiện dưới sự hướng dẫn của TS. Lý Quốc Ngọc.

Những kết quả nghiên cứu của luận văn hoàn toàn trung thực và chính xác.

TP. Hồ Chí Minh, ngày... tháng... năm...

HỌC VIÊN THỰC HIỆN

(Ký và ghi rõ họ tên)

LỜI CẢM ƠN

Tôi xin chân thành cảm ơn thầy Lý Quốc Ngọc đã tận tình hướng dẫn, truyền đạt kiến thức và kinh nghiệm, và đưa ra các giải pháp cho tôi trong suốt quá trình thực hiện đề tài luận văn tốt nghiệp này.

Xin gửi lời cảm ơn đến quý thầy cô Khoa Công Nghệ Thông Tin trường Đại Học Khoa Học Tự Nhiên - Đại Học Quốc Gia Thành Phố Hồ Chí Minh, những người đã truyền đạt kiến thức quý báu cho tôi trong thời gian học tập vừa qua.

Đồng thời cảm ơn các nhà khoa học đã nghiên cứu về đề tài mà tôi đã trích dẫn để có thể có những kiến thức hoàn thiện luận văn của tôi.

Sau cùng tôi xin gửi lời cảm ơn đến gia đình, bạn bè,.. những người luôn động viên, giúp đỡ tôi trong quá trình làm luận văn.

Một lần nữa, xin chân thành cảm ơn !

MỤC LỤC

Lời cam đoan	i
Lời cảm ơn	ii
Danh mục hình ảnh	vi
Danh mục các bảng số liệu	viii
Danh mục các thuật toán	viii
Danh mục các ký hiệu, các chữ viết tắt	ix
Bảng chú thích thuật ngữ	x
Trang thông tin luận văn tiếng Việt	xi
Trang thông tin luận văn tiếng Anh	xiii
 Chương 1 MỞ ĐẦU	 1
1.1 Bối cảnh chung	1
1.2 Động lực nghiên cứu	2
1.3 Dữ liệu bài toán	3
1.3.1 Kiến trúc khung xương của cursive	3
1.3.2 Kiến trúc chuyển động của cursive	4
1.4 Phát biểu bài toán	4
1.5 Các khó khăn cần giải quyết	5
1.6 Đóng góp dự kiến	6
 Chương 2 TỔNG QUAN	 8
2.1 Đặc điểm của cursive	8
2.2 Tổng quan các phương pháp cho bài toán sinh cursive	9
2.2.1 Phương pháp dựa trên luật	9
2.2.2 Phương pháp dựa trên thống kê	9
2.2.3 Phương pháp học sâu	10
2.2.4 So sánh các phương pháp sinh cursive	14
2.3 Công đoạn chung của phương pháp học sâu trong bài toán sinh cursive .	14
2.4 Diffusion-base Model	16

2.4.1	Nguyên lý chung của các phương pháp dựa trên Diffusion	16
2.4.2	Phương pháp tiêu biếu	16
2.4.3	Mô hình Diffusion phù hợp với bài toán sinh cử chỉ	18
Chương 3 PHƯƠNG PHÁP NGHIÊN CỨU		20
3.1	Mô hình diffusion cơ bản và các cải tiến	21
3.1.1	Quá trình gây nhiễu (forward diffusion process)	22
3.1.2	Quá trình khử nhiễu (denoising process)	24
3.1.3	Quá trình huấn luyện trong mô hình diffusion cơ bản	25
3.1.4	Quá trình lấy mẫu trong diffusion cơ bản	26
3.1.5	Cải tiến mô hình diffusion với dự đoán x_0 thay vì ϵ_t	28
3.1.6	Mô hình diffusion có điều kiện	30
3.2	Mô hình của đề xuất OHGesture	32
3.2.1	Công đoạn xử lý đặc trưng	33
3.2.2	Công đoạn trích xuất đặc trưng	34
3.2.3	Công đoạn mã hóa đặc trưng	36
3.2.4	Công đoạn kết hợp đặc trưng	36
3.2.5	Công đoạn giải mã đặc trưng	40
3.2.6	Điều khiển cảm xúc trong bài toán sinh cử chỉ	40
3.2.7	Quá trình huấn luyện	41
3.2.8	Quá trình lấy mẫu	42
Chương 4 THỰC NGHIỆM		45
4.1	Tập dữ liệu	45
4.2	Công đoạn tiền xử lý dữ liệu	45
4.3	Quá trình huấn luyện	47
4.4	Quá trình sử dụng Unity để kết xuất	48
Chương 5 KẾT QUẢ VÀ ĐÁNH GIÁ		49
5.1	Phương pháp đánh giá	49
5.1.1	Đánh giá dựa trên cảm nhận của con người	49
5.1.2	Đánh giá dựa trên các độ đo số học	50
5.2	Kết quả đánh giá	52
5.2.1	Kết quả đánh giá nghiên cứu người dùng	52
5.2.2	Kết quả đánh giá theo độ đo số học	53

5.3 Xây dựng và tiêu chuẩn hóa hệ thống đánh giá kết quả sinh cử chỉ	54
Chương 6 KẾT LUẬN	56
6.1 Kết quả đạt được	56
6.2 Ưu và nhược điểm của mô hình	56
6.3 Phương hướng phát triển và nghiên cứu trong tương lai	57
6.4 Đóng góp của luận văn	58
6.5 Lời kết	61
TÀI LIỆU THAM KHẢO	62
Chương A Các tham số trong Diffusion	70
A.1 Sự thay đổi của $\sqrt{\alpha}$ và $\sqrt{1 - \alpha}$	70
A.2 Sự thay đổi của $\sqrt{\bar{\alpha}}$ và $\sqrt{1 - \bar{\alpha}}$	71
A.3 Sự thay đổi của σ_t	72
Chương B Quá trình xử lý dữ liệu BVH	73
B.1 Cấu trúc khung xương của một nhân vật	73
B.2 Cấu trúc tệp dữ liệu BVH	74
B.3 Quá trình chuyển góc quay Euler sang Quaternion	75
Chương C Minh họa kết quả suy luận của cử chỉ	77
C.1 So sánh giữa cử chỉ ground truth và cử chỉ dự đoán	77
C.2 Minh họa các cảm xúc khác nhau của cử chỉ	78
C.3 Minh họa việc sinh cử chỉ với giọng nói nằm ngoài tập huấn luyện . .	79
C.4 Minh họa chuyển động của nhân vật	80

DANH MỤC HÌNH ẢNH

Hình 1.1: Minh họa kỹ thuật đồ họa máy tính trong việc xây dựng người kỹ thuật số	1
Hình 1.3: Minh họa một chuỗi cử chỉ, N khung hình đầu được lấy làm cử chỉ khởi tạo s (seed gesture) và M khung hình còn lại làm cử chỉ để học	5
Hình 2.1: Tổng quan về các mô hình tạo sinh khác nhau.	11
Hình 2.2: Các công đoạn trong mô hình sinh cử chỉ.	15
Hình 2.3: Minh họa nguyên lý chung của mô hình mô hình Diffusion. . .	16
Hình 3.1: Quá trình gây nhiễu (Diffuse) và khử nhiễu (Denoise) không huấn luyện	21
Hình 3.2: Thêm nhiễu trong quá trình huấn luyện	23
Hình 3.3: Khử nhiễu trong quá trình suy luận	24
Hình 3.4: Mô hình diffusion cơ bản	25
Hình 3.5: Quá trình Training và Sampling trong mô hình Diffusion tiêu chuẩn	27
Hình 3.6: So sánh ϵ objective (bên trên) và x_0 objective (bên dưới) . . .	29
Hình 3.7: Diffusion có điều kiện bằng vector điều hướng	30
Hình 3.8: Tổng quan về mô hình OHGesture	32
Hình 3.9: Mô hình trong OHGesture	35
Hình 3.10: Cơ chế Attention trong Transformer Encoder và Cross-Local Attention	38
Hình 3.12: Quá trình học Offline (Training) và Online (Inference)	42
Hình 4.1: Minh họa về 6 cử chỉ Happy, Sad, Neutral, Old, Relaxed và Angry	45

Hình 5.1: Hệ thống HEMVIP nhằm đánh giá kết quả sau khi kết xuất của hai mô hình	50
Hình A.1: Sự thay đổi của $\sqrt{\alpha}$ và $\sqrt{1 - \alpha}$	70
Hình A.2: Quá trình thay đổi của $\sqrt{\bar{\alpha}}$ và $\sqrt{1 - \bar{\alpha}}$	71
Hình A.3: Sự thay đổi của hệ số σ_t trong quá trình lấy mẫu	72
Hình B.1: Khung xương của một nhân vật	73

DANH MỤC CÁC BẢNG SỐ LIỆU

Bảng 2.1: Bảng so sánh ưu và nhược điểm của các phương pháp	14
Bảng 5.1: Kết quả đánh giá bằng MOS	53
Bảng 5.2: Kết quả đánh giá Mean Square Error theo 6 cảm xúc	53
Bảng 5.3: Kết quả đánh giá Fréchet Gesture Distance (FGD) trên $x^{1:M \times D}$ (từ khung hình 1 đến khung hình M, mỗi khung hình có D đặc trưng từng khung hình)	54

DANH MỤC CÁC THUẬT TOÁN

Thuật toán 1: Thuật toán training trong DDPM	26
Thuật toán 2: Thuật toán sampling trong DDPM	28
Thuật toán 3: Huấn luyện trong OHGesture	41
Thuật toán 4: Lấy mẫu (sampling) trong OHGesture	43

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

Từ viết tắt	Nội dung đầy đủ
θ	Trọng số huấn luyện cần để học
θ'	Trọng số huấn luyện sau khi học xong dùng để lấy mẫu
ϵ	Nhiều được thêm vào ảnh
$\hat{\epsilon}$	Nhiều dự đoán
ϵ_θ hay f_θ	Hàm dự đoán nhiễu DDPM với trọng số θ , f_θ và ϵ_θ là một.
N, M, D	Các số thể hiện số chiều của ma trận hoặc vector
$t : 1 \rightarrow T$	Bước gây nhiễu t từ bước $t = 1$ đến $t = T$
n_{joins}	Số khung xương
$\mathcal{N}(ax, b^2)$	Một hàm $f(x) = ax + b\epsilon$ với $\epsilon \in \mathcal{N}(0, 1)$ thì tương đương phân phối chuẩn $\mathcal{N}(ax, b^2)$
$\mathcal{N}(0, \mathbf{I})$	Phân phối chuẩn với mean là 0 và phương sai là 1
$\mathcal{N}(\mathbf{x}_t; \mu_\theta, \Sigma_\theta)$	Phân phối chuẩn với \mathbf{x}_t là output, μ trong bình của phân phối chuẩn đó với tham số θ , hàm dựa trên tham số θ có phương sai Σ
$\mathbf{x}_t^{1:M}$	Dữ liệu cử chỉ ở bước thứ t , bắt đầu từ khung hình thứ 1 đến khung hình thứ M
$\hat{\mathbf{x}}_0$	Dữ liệu dự đoán của mô hình
$f(x y)$	Hàm xác suất có điều kiện với y có trước tìm xác suất để có x
$q(\mathbf{x}_t \mathbf{x}_{t-1})$	Hàm xác suất có điều kiện của hàm gây nhiễu, khi biết trước \mathbf{x}_t và tìm \mathbf{x}_{t-1}
$p_\theta(\mathbf{x}_{t-1} \mathbf{x}_t)$	Hàm xác suất của quá trình khử nhiễu khi biết trước \mathbf{x}_t và tìm \mathbf{x}_{t-1} . Quá trình học để cập trọng số θ
$\mu_\theta(\mathbf{x}_t, t)$	Giá trung bình của \mathbf{x}_t ở bước thứ t sau khi đi qua mô hình với trọng số θ
\mathbb{E}	Kỳ vọng
\mathcal{L}^t	Hàm mất mát ở bước thứ t
G_θ	Mô hình OHGesture cần huấn luyện với trọng số θ
$\prod_{t=1}^T$	Hàm tích số học, $\prod_{t=1}^T a_t = a_1 \cdot a_2 \cdot a_3 \cdot \dots \cdot a_T$

BẢNG CHÚ THÍCH THUẬT NGỮ

Từ viết tắt	Nội dung đầy đủ
DDPM	Denoising Diffusion Probabilistic Models
OHGesture	Mô hình đề xuất của luận văn
Sampling/Inference	Quá trình lấy mẫu hoặc suy luận của mô hình
KL	Kullback–Leibler Divergence
MAE	Mean Absolute Error
MSE	Mean Squared Error
Forward Diffusion Process	Quá trình gây nhiễu không có huấn luyện bằng trọng số

TRANG THÔNG TIN LUẬN VĂN

Tên đề tài luận văn: OpenHuman: Hệ thống tổng hợp cử chỉ hội thoại dựa trên cảm xúc và ngữ nghĩa

Ngành: Khoa học máy tính

Mã số ngành: 8480101

Họ tên học viên cao học: Hoàng Minh Thanh

Khóa đào tạo: 31

Người hướng dẫn khoa học: PGS. TS. Lý Quốc Ngọc

Cơ sở đào tạo: Trường Đại học Khoa học Tự nhiên, ĐHQG.HCM

1. TÓM TẮT NỘI DUNG LUẬN VĂN:

Cùng với sự bùng nổ của phần cứng, các mô hình ngôn ngữ lớn, sự bùng nổ của các hệ thống trí tuệ nhân tạo dựa trên văn bản như ChatGPT, CharacterAI, Gemini,.. và sự phát triển của đồ họa máy tính thì nút nghẽn cổ chai hiện nay để phát triển người kỹ thuật số (digital human) chính là khả năng tạo ra chuyển động của nhân vật tương ứng với văn bản hoặc giọng nói. Một trong những khó khăn trong quá trình sinh cử chỉ là dữ liệu cử chỉ không đủ nhiều và chất lượng, cũng như sự thiếu thông nhất về ngữ cảnh trong cử chỉ là một trong những khó khăn trong việc xây dựng các hệ thống. Trong các phương pháp hiện nay, mô hình diffusion đạt kết quả tốt nhất do có khả năng tổng quát hóa và phủ được ở những vùng thiêu cân xứng giữa các đặc trưng, và các vùng có mật độ dữ liệu thấp.

Để có thể sinh cử chỉ tương ứng với giọng nói, và thông tin về cảm xúc cần học, luận văn sử dụng mô hình diffusion có điều kiện. Với điều kiện ở đây chính là cử chỉ khởi tạo, cảm xúc, giọng nói và văn bản tương ứng. Luận văn kế thừa từ mô hình DiffuseStyleGesture để xây dựng mô hình đề xuất **OHGesture**, trong luận văn giọng nói được chuyển thành văn bản, như một đặc trưng về ngữ nghĩa bổ sung trong quá trình học. Luận văn kế thừa mã nguồn Unity của mô hình DeepPhase để kết xuất, và trực quan hóa các chuyển động của nhân vật, cuối cùng các mã nguồn và mã nguồn chương trình được luận văn công khai để cộng đồng nghiên cứu về sinh cử chỉ tiếp tục phát triển các hệ thống sinh cử chỉ tối ưu hơn trong tương lai.

2. NHỮNG KẾT QUẢ MỚI CỦA LUẬN VĂN:

Thứ nhất, qua kết quả sinh cử chỉ thực tế, luận văn có thể chứng minh được mô hình đề xuất OHGesture đạt kết quả sinh cử chỉ tốt và thể hiện sự độ đồng bộ giữa

cử chỉ, giọng nói, và cảm xúc. Thứ hai, bằng việc tích hợp văn bản để bổ sung đặc trưng ngữ nghĩa, luận văn cung cấp thêm một hình học giúp mô hình có thể hiểu được từng ngữ cảnh cụ thể trong quá trình sinh cử chỉ. Thứ ba, nghiên cứu đóng góp một phương pháp mới trong việc xây dựng hệ thống đánh giá chuẩn hóa cho các mô hình sinh cử chỉ. Phương pháp kết hợp dữ liệu từ nhiều nguồn ngôn ngữ và sử dụng đánh giá của con người, tạo nên một nền tảng so sánh khách quan và toàn diện. Cuối cùng, luận văn sử dụng cộng minh họa chuyển động của nhân vật, mã nguồn trên github, mô hình pretrain trên Huggingface. Cũng như kết quả sinh cử mô hình đạt kết quả tốt với đầu vào nằm ngoài dữ liệu của quá trình huấn luyện.

3. CÁC ỨNG DỤNG/ KHẢ NĂNG ỨNG DỤNG TRONG THỰC TIỄN HAY NHỮNG VẤN ĐỀ CÒN BỎ NGỎ CẦN TIẾP TỤC NGHIÊN CỨU:

Mô hình OHGesture đã có thể sinh cử chỉ từ nhãn cảm xúc, giọng nói và văn bản tương ứng. Từ đây, chúng tôi có thể phát triển các hệ thống tương tác với người bằng cách kết hợp với các agent chuyên xử lý thông tin bằng văn bản khác như Character.AI, ChatGPT API,... Ngoài ra, chúng tôi cũng dự định làm một store tương tự như App Store nhưng chứa người ảo, với mỗi người ảo như một app được xây dựng từ nhiều người khác nhau có thể ứng dụng từ bạn gái ảo, giáo dục, trợ lý khách hàng...

Mô hình hiện tại đang sử dụng các chuỗi cử chỉ như một bức ảnh để khử nhiễu, có thể cải tiến bằng các sử dụng Fast Fourier Transform để trích xuất các đặc trưng về pha của chuyển động, từ đó xây dựng hệ thống sinh cử chỉ tốt hơn.

TẬP THỂ CÁN BỘ HƯỚNG DẪN HỌC VIÊN CAO HỌC

(Ký tên, họ tên)

(Ký tên, họ tên)

XÁC NHẬN CỦA CƠ SỞ ĐÀO TẠO HIỆU TRƯỞNG

THESIS INFORMATION

Thesis title: OpenHuman: A conversational gesture synthesis system based on emotions and semantics

Speciality: Computer Science

Code: 8480104

Name of Master Student: Hoàng Minh Thanh

Academic year: 31

Supervisor: Associate Professor Ly Quoc Ngoc

At: VNUHCM - University of Science

1. SUMMARY:

Along with the explosion of hardware advancements, large language models, text-based AI systems such as ChatGPT, CharacterAI, Gemini, and the development of computer graphics, the current bottleneck in creating digital humans lies in generating character movements corresponding to text or speech inputs.

One major challenge in gesture generation is the lack of sufficient and high-quality gesture data, as well as the lack of contextual consistency in gestures, making system development more difficult. Among current approaches, diffusion models achieve the best results due to their ability to generalize and handle imbalanced feature distributions and low-data-density regions effectively.

To generate gestures corresponding to speech and emotional information, this thesis employ a conditional diffusion model. The conditions include initial gestures, emotions, speech, and corresponding text. This thesis extends the DiffuseStyleGesture model to propose the **OHGesture** model. The thesis method converts speech into text as an additional semantic feature for training. We leverage the Unity codebase of the DeepPhase model for rendering and visualizing character movements. Lastly, this thesis make all our source code publicly available to encourage further development of gesture generation systems by the research community.

2. NOVELTY OF THESIS:

First, through actual gesture generation results, this thesis demonstrate that the proposed OHGesture model achieves high-quality gesture generation with synchronization between gestures, speech, and emotions.

Second, by integrating text to supplement semantic features, this thesis provide an additional mechanism to help the model understand specific contexts during gesture

generation.

Third, this research contributes a novel method for constructing standardized evaluation systems for gesture generation models. By combining data from multiple linguistic sources and utilizing human evaluations, this thesis establish an objective and comprehensive benchmarking platform.

Finally, this thesis utilize animated character visualization, source code hosted on GitHub, and pretrained models on Huggingface. The proposed model achieves favorable results even with inputs outside the training dataset.

3. APPLICATIONS/ APPLICABILITY/ PERSPECTIVE:

The OHGesture model is capable of generating gestures from emotion labels, speech, and corresponding text. This enables the development of human-interactive systems by integrating it with other text-based processing agents like Character.AI or the ChatGPT API. Additionally, we envision creating a store similar to the App Store but dedicated to virtual humans. Each virtual human, acting as an app, could be developed by various contributors and serve diverse applications such as virtual girlfriends, education, and customer assistance.

Currently, the model treats gesture sequences as images for denoising, which limits its performance. We plan to improve it by employing Fast Fourier Transform (FFT) to extract phase features of movements, aiming to develop a better gesture generation system.

SUPERVISOR Master STUDENT

(Signature, full name) (Signature, full name)

CERTIFICATION

CERTIFICATION UNIVERSITY OF SCIENCE

PRESIDENT

Chương 1. MỞ ĐẦU

1.1 Bối cảnh chung



(a) Công nghệ CGI với người kỹ thuật số siêu thật [23]



(b) Minh họa công trình tái tạo khuôn mặt tổng thống Obama [34]

Hình 1.1: Minh họa kỹ thuật đồ họa máy tính trong việc xây dựng người kỹ thuật số

Mỗi ngày, trên thế giới có hàng tỷ người nhìn vào màn hình RGB, kết quả hiển thị trên màn hình là đầu ra của mọi hệ thống phần mềm. Do đó, việc hiển thị từng pixel trên màn hình và cách để mô phỏng lại hình ảnh trên một cách chân thực nhất được các nhà khoa học về đồ họa máy tính (Computer Graphic) nghiên cứu từ những năm 1960s và đặc biệt là việc mô phỏng lại con người. Ngay từ năm 2014, các họa sĩ 3D đã có thể tạo nên một nhân vật người siêu thật như [Hình 1.1a](#) trong khi đó các phần cứng máy tính còn chưa phát triển như hiện nay.

Ngày nay, công nghệ đồ họa máy tính đã hoàn toàn có thể mô phỏng nhiều vật giống đến mức siêu thực (realistic), bao gồm các vật phức tạp như nước, đường xá, bánh mì,... và thậm chí là cả cơ thể và khuôn mặt con người với độ chi tiết đến từng lông tơ, nốt mụn và vân mắt. Vào năm 2015, bằng kỹ thuật quét 3 chiều ghi lại toàn bộ các góc của khuôn mặt, sự phản chiếu ánh sáng, trong công trình [34] các nhà

khoa học đồ họa máy tính đã có thể tái tạo toàn bộ khuôn mặt của tổng thống Obama trên máy tính với độ chính xác cao và gần như không thể phân biệt [Hình 1.1b](#).

Trí tuệ nhân tạo thể hiện kết quả vượt bậc những năm gần đây, không chỉ trong nghiên cứu mà còn trong ứng dụng thực tế, tiêu biểu như ứng dụng ChatGPT, Mid-Journey và sự phát triển cả theo chiều dọc và chiều ngang trong nhiều lĩnh vực ứng dụng trí tuệ nhân tạo. Mặc dù đồ họa máy tính đã có thể xây dựng khuôn mặt người siêu thật, việc sinh cử chỉ lại phụ thuộc vào việc chụp chuyển động (Motion Capture) từ các cảm biến và gặp khó khăn khi xây dựng hệ thống trí tuệ nhân tạo học từ dữ liệu.

Các hệ thống trí tuệ nhân tạo hiện nay đã có thể tạo văn bản và giọng nói tiêm cận như con người, nhưng một trong những trở ngại lớn nhất để xây dựng con người kỹ thuật số hiện nay chính là việc sinh cử chỉ. Chính vì vậy, mục tiêu của luận văn là xây dựng một hệ thống sinh cử chỉ hội thoại dựa trên cảm xúc và ngữ nghĩa với dữ liệu đầu vào gồm cả văn bản và giọng nói.

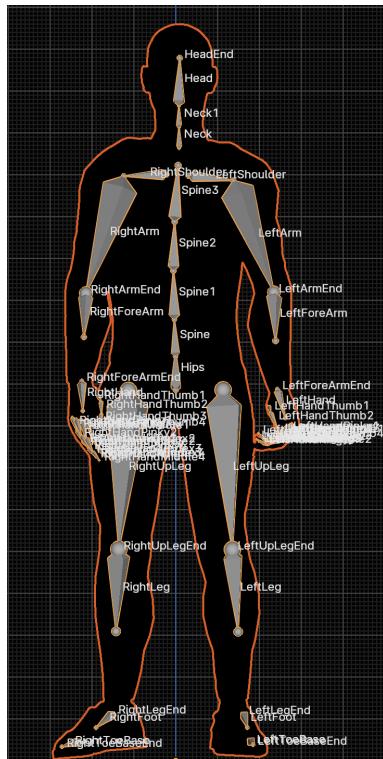
1.2 Động lực nghiên cứu

Sinh cử chỉ hội thoại giúp ích cho rất nhiều lĩnh vực như hoạt ảnh, dựng phim, trò chơi điện tử, giáo dục và những ứng dụng thực tế ảo. Việc sinh cử chỉ chuyển động được thực hiện theo cách truyền thống là thuê các diễn viên sử dụng thiết bị theo dõi chuyển động và bố trí các hệ thống cảm biến xung quanh thu nhận chuyển động để đạt được độ chính xác chân thực nhất. Tuy nhiên, các chuyển động thu được sau đó chỉ được phát lại và không có sự biến đổi linh hoạt giữa các hành động. Do đó, việc áp dụng trí tuệ nhân tạo để có thể học các chuyển động từ dữ liệu thu nhận và sau đó có thể sinh ra dữ liệu mới sẽ là một cuộc cách mạng trong ngành công nghiệp chụp chuyển động.

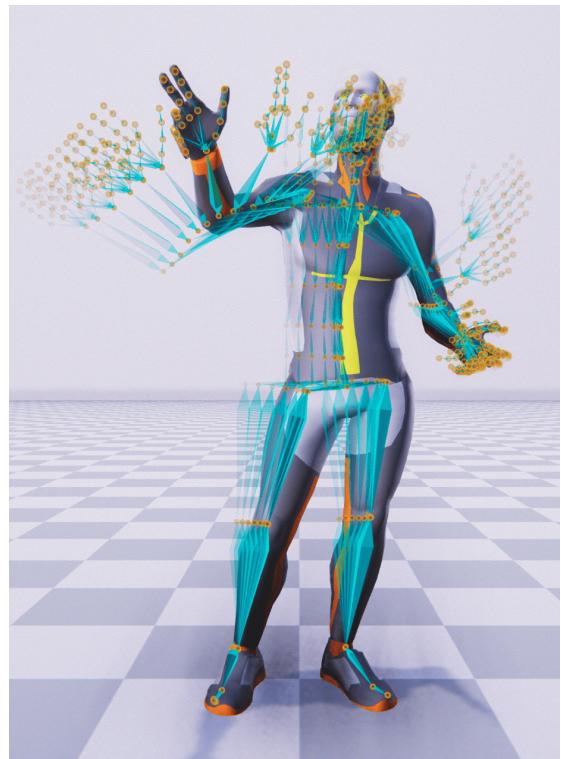
Vào năm 2011, một nhóm tác giả [\[5\]](#) đã chứng minh rằng có sự liên hệ giữa giọng nói và cử chỉ con người, đây chính là tiền đề cho thấy chúng ta có thể dùng dữ liệu giọng nói để có thể dùng để học và biểu diễn được cử chỉ con người. Với sự thành công của các mô hình ngôn ngữ tự nhiên trong xử lý văn bản và độ chính xác siêu thật trong việc mô phỏng gương mặt con người, ngành đồ họa máy tính đã đạt được những tiến bộ vượt bậc. Cùng với đó là sự dễ dàng và chính xác trong việc tổng hợp giọng nói con người. Việc ứng dụng trí tuệ nhân tạo để sinh cử chỉ con người là một trong những điểm nghẽn chính trong phát triển trợ lý ảo giao tiếp và tương tác với con

người.

1.3 Dữ liệu bài toán



(a) Khung xương và tên của các khớp của một khung xương trong mỗi khung hình.



(b) Chuỗi chuyển động của cử chỉ bao gồm 6 cử chỉ quá khứ và 6 cử chỉ tương lai.

1.3.1 Kiến trúc khung xương của cử chỉ

Cử chỉ (gesture) trong luận văn là sự chuyển động của toàn bộ cơ thể của một nhân vật như hình [Hình 1.2a](#) theo từng khung hình. Để có thể biểu diễn các chuyển động của một nhân vật trong đồ họa máy tính, các chuyển động sẽ được biểu diễn thành chuyển động của các xương (bone) riêng lẻ. Bao gồm cánh tay (hand), chân (leg), đầu (head), xương sống (spine),... Kiến trúc đầy đủ của một nhân vật được trình bày ở phụ lục [Mục B.1](#).

Dữ liệu một nhân vật theo thời gian trong thực tế sẽ được thu nhận bằng các hệ thống chụp chuyển động (motion capture) với các hệ thống camera và cảm biến chuyên biệt. Kết quả của quá trình motion capture là các tệp được định nghĩa dưới dạng tệp BVH (Biovision Hierarchy).

Các tệp BVH bao gồm hai thành phần chính: HIERARCHY và MOTION. HIERARCHY được thể hiện dưới dạng một cây bao gồm các thông tin về tên và vị trí khởi tạo các khung xương, MOTION là dữ liệu về chuyển động của toàn bộ khung xương theo từng khung hình (frame). Mỗi tệp BVH sẽ có thông tin về số khung hình trên giây (fps) và tổng số khung hình. Thông tin mỗi tệp BVH được trình bày ở phụ lục [Mục B.2](#).

1.3.2 Kiến trúc chuyển động của cử chỉ

Dữ liệu chuyển động của cử chỉ như [Hình 1.2b](#) hay phần MOTION của tệp BVH sẽ chứa thông tin về tọa độ và góc quay của một nhân vật theo từng khung hình. Dữ liệu mỗi khung hình là tập khung xương (skeleton) bao gồm 75 xương (bone), $\{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_{75}\}$, mỗi xương thể hiện vị trí (position) $\{p_x, p_y, p_z\}$ và góc quay (rotation) $\{r_x, r_y, r_z\}$ chuyển động của một nhân vật theo thời gian.

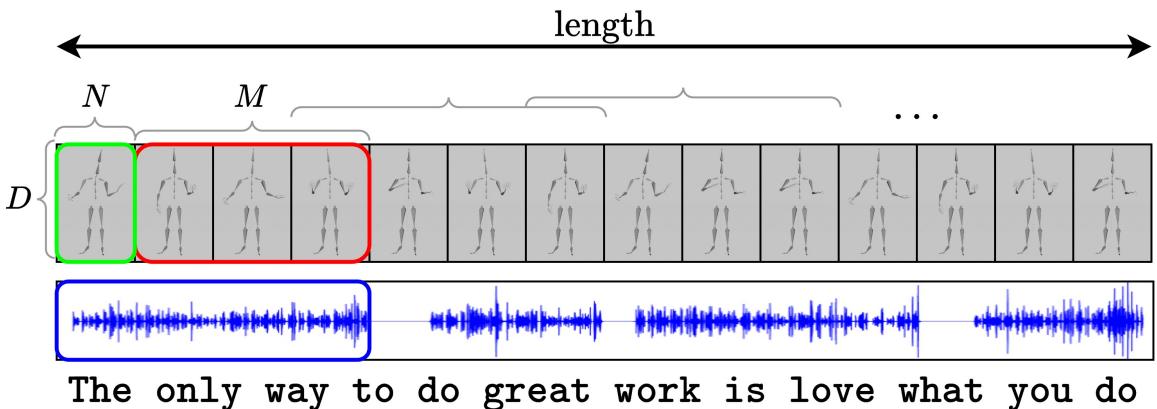
Kết quả của việc sinh cử chỉ (gesture generation) là sinh ra chuỗi chuyển động góc quay các xương của nhân vật theo từng khung hình (frame). Việc sinh cử chỉ (gesture generation) được đánh giá bằng việc tạo ra các cử chỉ tự nhiên, giống con người (human-likeness) và phù hợp với ngữ cảnh.

Trong luận văn, các dữ liệu về vị trí và góc quay của khung xương nhân vật được tiền xử lý để chuyển thành một vector đặc trưng $\mathbf{g} \in \mathbb{R}^D$ với $D = 1141$. Dữ liệu cần học khi đó sẽ là $\mathbf{x} \in \mathbb{R}^{M \times D}$.

Quá trình xử lý dữ liệu được trình bày đầy đủ ở [Phụ lục B](#).

1.4 Phát biểu bài toán

Với kết quả cuối cùng là chuỗi cử chỉ thể hiện sự chuyển động của các khung xương theo từng khung hình, có thể áp dụng nhiều phương pháp khác nhau như phương pháp học phân loại (classification), gom nhóm (clustering), hồi quy (regression), .. Trong luận văn này, bài toán sinh cử chỉ được xem xét như một bài toán hồi quy (regression), với đầu vào là một chuỗi cử chỉ cho trước và đầu ra là chuỗi cử chỉ tiếp theo cần dự đoán.



Hình 1.3: Minh họa một chuỗi cử chỉ, N khung hình đầu được lấy làm cử chỉ khởi tạo s (seed gesture) và M khung hình còn lại làm cử chỉ để học

Với một chuỗi cử chỉ có độ dài khung hình bất kỳ, cảm xúc sẽ được gán cho toàn bộ chuỗi cử chỉ. Một điểm cải tiến của luận văn là tương ứng với chuỗi cử chỉ là dữ liệu giọng nói và đoạn văn bản được dịch từ dữ liệu giọng nói tương ứng. Mục tiêu của luận văn là xây dựng mô hình để dự đoán từng đoạn nhỏ, với thông tin dữ liệu đầu vào bao gồm chuỗi N khung hình cử chỉ cho trước $s \in \mathbb{R}^{1:N \times D}$ (seed gesture), chuỗi giọng nói a (speech), văn bản v (text), cảm xúc e (emotion) tương ứng.

Kết quả dự đoán của mô hình là $\hat{x} \in \mathbb{R}^{1:M \times D}$ bao gồm từ khung hình 1 đến khung hình M chuỗi cử chỉ tiếp theo. Với dữ liệu gốc là chuỗi cử chỉ đã có $x \in \mathbb{R}^{1:M \times D}$.

Đầu vào

- Chuỗi cử chỉ khởi tạo: $s \in \mathbb{R}^{1:N \times D}$
- Chuỗi giọng nói: a
- Văn bản: v
- Cảm xúc: e
(Happy, Sad, Neutral, Angry,
Old, Funny)

Dữ liệu dự đoán

- Chuỗi cử chỉ dự đoán: $\hat{x} \in \mathbb{R}^{1:M \times D}$

Dữ liệu gốc

- Chuỗi cử chỉ gốc: $x \in \mathbb{R}^{1:M \times D}$

1.5 Các khó khăn cần giải quyết

Có rất nhiều khó khăn trong việc xây dựng một mô hình có thể học được các đặc trưng cử chỉ hội thoại như con người.

Thứ nhất, *dữ liệu không đủ nhiều và chất lượng*, chi phí để tạo ra một bộ dữ liệu trong ngành công nghiệp chụp chuyển động có chất lượng và quy mô lớn để ứng dụng thực tế là rất cao.

Thứ hai, *sự thiếu đồng nhất về ngữ cảnh của các loại dữ liệu*, các bộ dữ liệu về văn bản thường nhiều hơn so với giọng nói, và cũng không rõ văn bản đó được tạo ra bởi ai. Sự đồng bộ giữa giọng nói và cảm xúc khi nói cũng thường thiếu trong tập dữ liệu. Ngoài ra, dữ liệu văn bản trong tập dữ liệu huấn luyện lại thuộc nhiều chủ đề đa dạng.

Thứ ba, *sự phân bố không cân xứng về dữ liệu giữa các loại đặc trưng cần học*. Các dữ liệu dùng cho nghiên cứu cử chỉ hiện nay thường tập trung vào ngôn ngữ Tiếng Anh, các cử chỉ có sự phân bố không cân xứng giữa các trạng thái như nói, hỏi, hoặc im lặng.

Thứ tư, *chi phí tính toán với nhiều loại dữ liệu của mô hình là một thách thức lớn*. Với đầu vào của mô hình gồm nhiều loại dữ liệu như văn bản, tiếng nói và điểm 3D, cần nhiều lớp mã hóa cho từng loại dữ liệu, dẫn đến chi phí tính toán cao trong cả giai đoạn huấn luyện và suy luận. Nếu giảm thông tin dữ liệu đầu vào cũng sẽ giảm kết quả suy luận của mô hình khi sinh cử chỉ.

Cuối cùng, *các bước xử lý cần được thực hiện tuần tự*, cách hiệu quả nhất để con người tương tác với máy tính là thông qua giọng nói và nhập từ bàn phím, tuy nhiên việc xử lý được văn bản và giọng nói để làm đầu vào cho mô hình phải thực hiện tuần tự. Độ trễ trong quá trình suy luận của sản phẩm thực tế cũng là một vấn đề lớn, vì người dùng không thể chờ đợi quá lâu để nhận kết quả. Ngoài ra, việc hiển thị cử chỉ đó lên máy tính bằng kỹ thuật đồ họa cũng cần được tối ưu để giảm thời gian xử lý.

1.6 Đóng góp dự kiến

- Dựa trên tập dữ liệu có sẵn, luận văn chuyển giọng nói trong tập dữ liệu thành văn bản, và dùng văn bản đó để làm dữ liệu huấn luyện mới như là một dữ liệu ngữ nghĩa bổ sung trong quá trình học.
- Dựa trên mô hình cơ bản DiffuseStyleGesture, luận văn mở rộng thêm đặc trưng văn bản trong quá trình khử nhiễu có điều kiện.
- Luận văn sử dụng Unity để render, trích xuất dữ liệu và trực quan hóa kết quả sinh cử chỉ.

- Luận văn xây dựng hệ thống kết xuất, và minh họa chương trình bằng Unity

Chương 2. TỔNG QUAN

Bài toán sinh cử chỉ cũng tương tự như các bài toán khác, đều đã nghiên cứu và phát triển song hành với các phương pháp học máy truyền thống và hiện đại. Gồm các nhóm phương pháp dựa trên luật và các phương pháp dựa trên dữ liệu. Đầu tiên luận văn trình bày về các đặc điểm của cử chỉ [Mục 2.1](#), từ đó là tiền đề cho việc học mối quan hệ giữa cử chỉ, văn bản và giọng nói. Trong mục [Mục 2.3](#), luận văn sẽ trình bày về các công đoạn chung trong các phương pháp sinh cử chỉ. Ở phần [Mục 2.2](#), luận văn trình bày về các phương pháp đã được sử dụng trong quá trình sinh cử chỉ. Luận văn sẽ so sánh các phương pháp, từ đó nêu lý do luận văn sử dụng mô hình diffusion để áp dụng cho bài toán sinh cử chỉ. Trong phần [Mục 2.4](#), luận văn sẽ trình bày về cách các mô hình diffusion được áp dụng cho bài toán sinh cử chỉ gần đây.

2.1 Đặc điểm của cử chỉ

Theo ngôn ngữ học, cử chỉ có thể được phân thành 6 nhóm chính: cử chỉ thích nghi (adaptors), cử chỉ biểu tượng (emblems), cử chỉ chỉ định (deictics), cử chỉ biểu trưng (iconics), cử chỉ ẩn dụ (metaphorics), và cử chỉ nhấn mạnh (beat) [\[15\]](#), [\[41\]](#). Trong số đó, cử chỉ nhấn mạnh không mang ý nghĩa ngữ nghĩa trực tiếp nhưng đóng vai trò quan trọng trong việc đồng bộ nhịp điệu giữa giọng nói và cử chỉ [\[25\]](#), [\[41\]](#). Tuy nhiên, nhịp điệu giữa giọng nói và cử chỉ nhấn mạnh không hoàn toàn đồng bộ, khiến việc học mối quan hệ thời gian giữa chúng trở nên phức tạp [\[33\]](#), [\[6\]](#), [\[28\]](#), [\[55\]](#).

Cử chỉ tương tác với các cấp độ thông tin khác nhau trong giọng nói [\[41\]](#). Chẳng hạn, cử chỉ biểu tượng, như hành động giơ ngón cái, thường liên quan đến thông tin ngữ nghĩa cấp cao (ví dụ: "tốt" hoặc "tuyệt vời"), trong khi cử chỉ nhấn mạnh thường đi kèm với thông tin cấp thấp như nhấn mạnh trong âm thanh. Các nghiên cứu trước đây thường chỉ sử dụng đặc trưng từ lớp cuối cùng của bộ mã hóa giọng nói để tổng hợp cử chỉ [\[1\]](#), [\[6\]](#), [\[29\]](#), [\[38\]](#), [\[56\]](#). Tuy nhiên, cách tiếp cận này có thể làm trộn lẫn thông tin từ nhiều cấp độ, dẫn đến khó khăn trong việc phân tách rõ ràng nhịp điệu

và ngữ nghĩa.

Như các nghiên cứu ngôn ngữ học chỉ ra [25], [36], [46], cursive chỉ trong giao tiếp hàng ngày có thể được chia thành một số lượng giới hạn các đơn vị ngữ nghĩa với các biến thể chuyển động khác nhau. Dựa trên giả định này, được phân tách đặc trưng giọng nói thành hai loại: đặc trưng cấp cao đại diện cho các đơn vị ngữ nghĩa, và đặc trưng cấp thấp xác định các biến thể chuyển động. Từ đó, mối liên hệ giữa chúng được học thông qua các lớp khác nhau của bộ mã hóa giọng nói. Các thử nghiệm chứng minh rằng cơ chế này có khả năng tách biệt rõ ràng các đặc trưng ở nhiều cấp độ, đồng thời tổng hợp được cursive phù hợp về ngữ nghĩa và phong cách.

2.2 Tổng quan các phương pháp cho bài toán sinh cursive

2.2.1 Phương pháp dựa trên luật

2.2.1.1 Nguyên lý chung

Dựa trên việc xây dựng các quy tắc hoặc luật rõ ràng, các quy tắc được định nghĩa thủ công, xác định cách hệ thống xử lý đầu vào để tạo ra đầu ra.

2.2.1.2 Phương pháp

Tiêu biểu cho phương pháp dựa trên luật là *Robot behavior toolkit* [22] và *Animated conversation* [9]. Phương pháp dựa trên luật thường ánh xạ (mappings) từng giọng nói với từng đơn vị cursive, với các luật được tạo thủ công. Phương pháp dựa trên luật thì dễ dàng điều khiển kết quả của mô hình và có khả năng giải thích tốt kết quả dự đoán của mô hình. Tuy nhiên chi phí để tạo thủ công là không khả thi để xây dựng cho các ứng dụng phức tạp đòi hỏi phải xử lý một lượng dữ liệu rất lớn.

2.2.2 Phương pháp dựa trên thống kê

2.2.2.1 Nguyên lý chung

Dựa vào phân tích dữ liệu, học từ các mẫu trong tập dữ liệu và sử dụng các mô hình xác suất hoặc hàm toán học để đưa ra dự đoán. Phương pháp này dựa trên việc tối ưu hóa các tham số của mô hình để phù hợp với dữ liệu.

2.2.2.2 Phương pháp

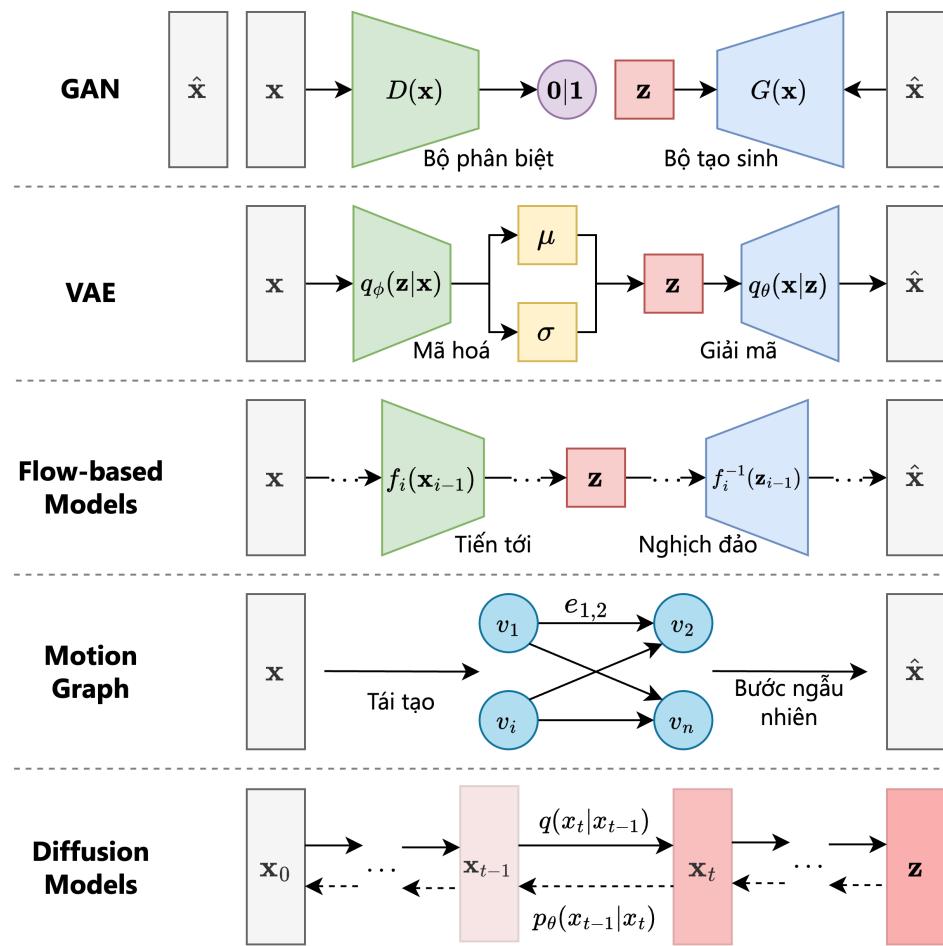
Tương tự như phương pháp dựa trên luật, phương pháp dựa trên dữ liệu cũng ánh xạ các đặc trưng của giọng nói tương ứng với cử chỉ nhưng thay vì làm thủ công thì được sử dụng học một cách tự động dựa trên các phân tích thống kê của dữ liệu.

Tiêu biểu của phương pháp thống kê là *Gesture controllers* [30], *Statistics-based* [54], các phương pháp này sử dụng phân phối xác suất để tìm sự tương đồng giữa các đặc trưng giọng nói và cử chỉ. *Gesture modeling* [36] xây dựng mô hình xác suất để học từng phong cách của từng người nói.

2.2.3 Phương pháp học sâu

Nguyên lý chung của phương pháp học sâu

Sử dụng mạng nơ-ron đa lớp (Multi-layer Perceptron) để tự động trích xuất đặc trưng từ dữ liệu thô và học các biểu diễn phức tạp của dữ liệu. Mô hình học thông qua tối ưu hóa các tham số.



Hình 2.1: Tổng quan về các mô hình tạo sinh khác nhau.

Phương pháp sinh cử chỉ bằng học sâu minh họa ở [Hình 2.1](#) được chia thành hai nhóm chính: học dựa trên ước lượng log likelihood (likelihood-based model) và phương pháp dựa vào các mô hình sinh ngầm định (implicit generative models) [42].

2.2.3.1 Mô hình dựa trên likelihood

Nguyên lý chung của mô hình dựa trên likelihood

Mô hình dựa trên Log Likelihood hoạt động bằng cách tối đa hóa xác suất hợp lý của dữ liệu quan sát được, dựa trên các tham số θ của mô hình. Mục tiêu là tìm được tham số tối ưu θ' dựa trên xác suất $p(x)$ của dữ liệu, trong đó x là dữ liệu chuỗi cử chỉ.

Phương pháp

Quá trình áp dụng các phương pháp học sâu vào bài toán sinh cử chỉ cũng tương đồng với sự phát triển của các phương pháp học sâu, từ RNN, LSTM và Transformer .v.v.. Các phương pháp tiêu biểu trong nhóm phương pháp dựa trên likelihood:

- Phương pháp *Gesticulator* [28] sử dụng kiến trúc Multilayer Perceptron (MLP) để mã hóa đặc trưng văn bản và âm thanh, sử dụng vector văn bản từ BERT làm vector đặc trưng cho văn bản.
- Trong *HA2G* [32] dựa trên kiến trúc Transformer xây dựng một mô hình phân cấp (hierarchical) để học các tương quan ở nhiều cấp độ giữa lời nói và cử chỉ, từ các đặc trưng cục bộ (local) đến toàn cục (global).
- *Gesture Generation from Trimodal Context* [55] dựa trên kiến trúc RNN, coi việc sinh cử chỉ là bài toán dịch trong xử lý ngôn ngữ.
- *DNN* [12] sử dụng LSTM kết hợp GRU để xây dựng mô hình phân loại (classifier neural network) từng đơn vị cử chỉ để chọn một đơn vị cử chỉ phù hợp dựa trên đầu vào là giọng nói.
- *Cascaded Motion Network (CaMN)* [31] đề xuất tập dữ liệu BEAT đồng thời để xuất mô hình giống như thác nước, các thông tin về người nói, nhãn cảm xúc, văn bản, giọng nói và cử chỉ được đi qua các lớp để trích xuất đặc trưng để được các vector tiềm ẩn. Trong giai đoạn kết hợp các đặc trưng, mô hình CaMN kết hợp các đặc trưng theo hình thác nước, lần lượt kết hợp các đặc trưng người nói và cảm xúc, và tiếp tục kết hợp với các vector tiềm ẩn của văn bản, giọng nói và cử chỉ.
- **Motion Graph** : Trong phương pháp *Gesturemaster* [58], mô hình xây dựng đồ thị ngữ nghĩa, từ trong câu được kết nối trong một đồ thị, nơi các liên kết biểu thị quan hệ ngữ nghĩa giữa chúng. Dựa trên thông tin đồ thị, mô hình chọn các đỉnh (node) và cạnh (edges) có sự tương đồng nhất để biểu diễn cử chỉ.

2.2.3.2 Mô hình sinh ngầm định

Nguyên lý

Mô hình sinh ngầm định học phân phối dữ liệu mà không cần biểu diễn tường minh hàm mật độ xác suất $p(\mathbf{x})$. Thay vì trực tiếp tính xác suất $p(\mathbf{x})$, mô hình sẽ học một ánh xạ $G_\theta : \mathcal{Z} \rightarrow \mathcal{X}$ thông qua so khớp phân phối giữa dữ liệu thật và dữ liệu sinh ra $\mathbf{x} = G_\theta(\mathbf{z}), \quad \mathbf{z} \sim p_z(\mathbf{z})$. Trong đó \mathcal{Z} là không gian nhiễu đầu vào, thường có phân phối đơn giản p_z (ví dụ: Gaussian, Uniform). \mathcal{X} là không gian dữ liệu thực, là chuỗi cử chỉ \mathbf{x} .

Trong bài toán sinh cử chỉ, để kết hợp thêm các điều kiện về giọng nói, nhãn văn bản, các mô hình sinh ngầm định thường kết hợp thêm điều kiện c , với điều kiện là ngữ cảnh của bài toán để có được kết quả sinh: $x = G_\theta(z, c)$. Ngữ cảnh của bài toán có thể bao gồm giọng nói, văn bản, mã định danh người nói, phong cách, cảm xúc hay cử chỉ khởi tạo.

Ví dụ tiêu biểu nhất là mô hình generative adversarial networks (GANs) và Diffusion. Khi dữ liệu được tổng hợp bằng cách chuyển phân phối dữ liệu ban đầu ở dạng phân phối chuẩn về phân phối của dữ liệu.

Phương pháp

Các phương pháp tiêu biểu cho mô hình sinh ngầm định là

- **MoGlow** [19] sử dụng Normalizing Flows được sử dụng để duy trì tính nhất quán và tính tương thích của chuyển động, đồng thời cung cấp khả năng điều khiển thông qua các thông số đầu vào. Điều này cho phép người dùng tạo ra các chuyển động mới hoặc thay đổi các chuyển động hiện có bằng cách thay đổi các thông số điều khiển.
- **GAN: GRU-based WGAN** [48] sử dụng Wasserstein GAN để đánh giá và cải thiện chất lượng của việc tổng hợp cử chỉ, mô hình tập trung vào việc tối ưu hóa hàm mất mát Wasserstein, giúp giảm tình trạng chảy đặc (mode collapse) thường gặp trong GAN truyền thống. GRU được sử dụng để xử lý dữ liệu lời nói và biến nó thành các đặc trưng cử chỉ có thể sử dụng được. Sau đó, những đặc trưng này được cung cấp vào mạng WGAN, nơi chúng sẽ được đánh giá và tinh chỉnh để tổng hợp cử chỉ.
- **VAE: FreeMo** [49] sử dụng VAE để phân rã (decomposing) các cử chỉ thành tư thế cử chỉ (pose modes) và giai điệu chuyển động (rhythmic motions). Tư thế cử chỉ được sinh ra ngẫu nhiên sử dụng quá trình lấy mẫu có điều kiện trong không gian ẩn của VAE. Trong khi các giai điệu chuyển động được sinh ra từ
- **VQ-VAE Rhythmic Gesticulator** [3] tiền xử lý các đoạn giọng nói theo nhịp điệu (beats) để tách giọng nói thành các đoạn nhỏ, và biểu diễn chúng thành các khối theo kích thước chuẩn hóa (normalization). Tương tự phương pháp VQ-VAE, các chuỗi cử chỉ đã chuẩn hóa sẽ được lượng tử hóa (quantization) thành các vùng không gian khác nhau của cử chỉ được gọi là các từ vựng cử chỉ (gesture lexicon). Mô hình học từ vựng cử chỉ (gesture lexicon) từ khối chuẩn

hóa trên với điều kiện là cursive chỉ trước đó, phong cách cursive chỉ và giọng nói. Sau đó mô hình tái tạo lại từ quá trình chuẩn hóa (denormalization) để có được chuỗi cursive chỉ ban đầu. Khác với mô hình sinh ngầm định như GANs hay Diffusion, VQ-VAE thường tập trung vào việc nén dữ liệu hơn là sinh dữ liệu trực tiếp.

- **Diffusion:** Mô hình Diffusion tập trung vào việc tạo ra dữ liệu mới từ trạng thái nhiễu, phục hồi ngược lại dữ liệu gốc. Các phương pháp dựa trên Diffusion sẽ được trình bày ở mục [Mục 2.4](#)

2.2.4 So sánh các phương pháp sinh cursive

Phương pháp tiêu biểu	Ưu điểm	Nhược điểm	Loại phương pháp
Robot behavior toolkit [22]	<ul style="list-style-type: none"> - Dễ hiểu và dễ triển khai. - Dễ giải thích và có thể kiểm soát được - Hiệu quả trong các trường hợp đơn giản hoặc dữ liệu nhỏ. 	<ul style="list-style-type: none"> - Không tổng quát hóa tốt với dữ liệu phức tạp. - Đòi hỏi nhiều công sức trong việc xây dựng quy tắc thủ công. 	Mô hình dựa trên luật
MLP [28], RNN [6], [32], [18], [55], CNN [17], Transformer [7]	<ul style="list-style-type: none"> - Khả năng ước lượng mật độ xác suất của dữ liệu - Có khả năng mở rộng và học từ dữ liệu lớn 	<ul style="list-style-type: none"> - Dễ bị ảnh hưởng bởi nhiễu - Kết quả thấp ở vùng dữ liệu hiếm - Khả năng sinh không đa dạng 	Mô hình dựa trên Likelihood
DiffusionStyle-Gesture [50], MDM [44], Motiondiffuse [57]	<ul style="list-style-type: none"> - Tạo ra dữ liệu chất lượng cao. - Linh hoạt và đa dạng - Phủ được vùng có mật độ dữ liệu thấp 	<ul style="list-style-type: none"> - Cần cấu hình phức tạp để đạt hiệu năng tốt. - Mỗi lần lấy nhiễu ngẫu nhiên là khác nhau. - Quá trình lấy mẫu chậm 	Mô hình sinh ngầm định

Bảng 2.1: Bảng so sánh ưu và nhược điểm của các phương pháp

2.3 Công đoạn chung của phương pháp học sâu trong bài toán sinh cursive

Như đã trình bày ở [Mục 1.3](#), cursive bao gồm chuỗi chuyển động của tọa độ điểm 3D. Với mỗi tập dữ liệu số lượng xương (bone) mỗi khung hình sẽ khác nhau.

Cách tiếp cận bằng học sâu với bài toán sinh cursive (gesture generation) được thực hiện với nhiều phương pháp khác nhau. Tuy nhiên luận văn tổng quát hóa lại thành

các công đoạn chính như [Hình 2.2](#) sau:



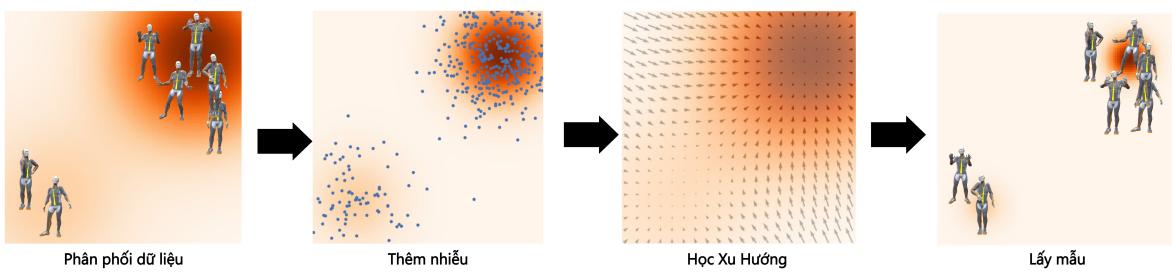
Hình 2.2: Các công đoạn trong mô hình sinh cử chỉ.

- 1. Tiền xử lý dữ liệu** (Preprocessing): Trong công đoạn tiền xử lý, các dữ liệu về đoạn giọng nói, dữ liệu về chuỗi cử chỉ, đoạn văn bản sẽ được đọc, số hóa để thu được các vector hoặc ma trận thể hiện thông tin thô của dữ liệu. Đối với mỗi phương pháp học khác nhau, các thông tin dữ liệu ban đầu sẽ được chọn để học cũng khác nhau.
- 2. Xử lý đặc trưng** (Feature Processing): Trong công đoạn xử lý đặc trưng, các dữ liệu thô như giọng nói và văn bản được nhúng (embedding) để biểu diễn thành các vector đặc trưng. Các phương pháp khác nhau sẽ chọn các mô hình nhúng khác nhau. Việc biểu diễn các cử chỉ của nhân vật thành các vector đặc trưng của mỗi phương pháp cũng khác nhau.
- 3. Trích xuất đặc trưng** (Feature Extraction): Công đoạn trích xuất đặc trưng sẽ dùng các lớp biến đổi tuyến tính (linear) hoặc các lớp CNN để trích xuất các đặc trưng của dữ liệu. Các dữ liệu về văn bản hoặc giọng nói sau khi xử lý đặc trưng có thể cũng được cho đi qua các lớp trích xuất đặc trưng để biểu diễn thành các vector đặc trưng để biểu diễn tương ứng với văn bản và giọng nói.
- 4. Mã hóa đặc trưng** (Feature Encoding): Trong công đoạn mã hóa đặc trưng, các vector về cử chỉ, cảm xúc, và giọng nói sẽ được biểu diễn lên không gian tiềm ẩn nhỏ hơn kích thước ban đầu nhằm thuận tiện cho việc tính sự tương quan giữa các đặc trưng ở công đoạn kết hợp đặc trưng.
- 5. Kết hợp đặc trưng** (Feature Fusion): Trong công đoạn kết hợp đặc trưng, các đặc trưng giọng nói, văn bản, cử chỉ cũng như các thông tin khác được kết hợp với nhau bằng việc concat, các lớp kết nối đầy đủ hoặc kết hợp các đặc trưng bằng cách cộng hoặc trừ các vector tiềm ẩn.
- 6. Giải mã đặc trưng** (Feature Decoding): Trong công đoạn giải mã đặc trưng, các vector tiềm ẩn sẽ được giải mã hay tăng chiều dữ liệu về kích thước ban đầu.

7. **Kết xuất** (Render): Sau khi có được vector ở kích thước ban đầu, các vector sẽ được biến đổi ngược trở về các tệp BVH để được kết xuất thông qua các phần mềm như Blender hoặc Unity để minh họa các chuyển động nhân vật.

2.4 Diffusion-base Model

2.4.1 Nguyên lý chung của các phương pháp dựa trên Diffusion



Hình 2.3: Minh họa nguyên lý chung của mô hình mô hình Diffusion.

Tương tự phương pháp trong nhóm phương pháp sinh ngầm định (Implicit Generative Models [Mục nhỏ 2.2.3.2](#)), nguyên lý của mô hình Diffusion là học một hàm f_θ biểu diễn xu hướng (drift) của quá trình từ phân phối chuẩn $\mathcal{N}(0, \mathbf{I})$ về phân phối của dữ liệu $\mathbf{x}_0 \sim p(\mathbf{x})$, từ đó lấy mẫu (sampling) lên từng bước của quá trình đảo ngược để sinh ra các dữ liệu mới như [Hình 2.3](#). Quá trình này được thực hiện từng T bước, với nhiễu được thêm trong quá trình lấy mẫu giảm dần để mô hình có thể điều hướng về các vùng có mật độ dữ liệu lớn.

2.4.2 Phương pháp tiêu biểu

2.4.2.1 Phương pháp tiêu biểu

- *MotionDiffuse* [57] sử dụng mô hình Diffusion có điều kiện, với điều kiện dựa trên là văn bản mà không bao gồm âm thanh. Ngoài ra mô hình dự đoán nhiều, chứ không dự đoán chuỗi cử chỉ gốc. Mô hình MotionDiffuse sử dụng các lớp Self-Attention và Cross Attention để tính tương quan giữa đặc trưng văn bản và đặc trưng cử chỉ trong công đoạn 5. *Kết hợp đặc trưng* ([Hình 2.2](#)).
- *Flame* [24] sử dụng mô hình diffusion với kiến trúc transformer. Trong công đoạn 2. *Xử lý đặc trưng* ([Hình 2.2](#)), mô hình sử dụng mô hình pre-train RoBERTa

để nhúng (embedding) văn bản để được các vector đặc trưng văn bản, và sử dụng chúng làm điều kiện cho mô hình. Trong công đoạn 5. *Kết hợp đặc trưng* ([Hình 2.2](#)), sử dụng văn bản làm token CLS đầu tiên của chuỗi cử chỉ trước khi đi qua lớp Transformer Decoder . Mô hình cũng dự đoán nhiều đã được thêm vào mô hình Diffusion chứ không dự đoán chuỗi cử chỉ gốc.

- *DiffWave* [27] sử dụng mô hình Diffusion với việc dự đoán nhiễu, các bước thời gian được đi qua nhiều lớp Fully Connected khác nhau và lớp kích hoạt Swish trước khi kết hợp các đặc trưng. Mô hình sử dụng kiến trúc dilated convolutions kế thừa từ WaveNet. Mô hình DiffWave giúp biểu diễn giọng nói được biểu diễn tốt hơn, tạo đà vào hiệu quả hơn cho mô hình Diffusion.
- *Listen, denoise, action* [2] kế thừa từ mô hình DiffWave [27], thay lớp dilated convolutions bằng Transformer , kết hợp với Conformers giúp cải thiện hiệu suất của mô hình.
- *DiffSHEG* [10] sử dụng mô hình diffusion, trong công đoạn 2. *Xử lý đặc trưng*, mô hình DiffSHEG sử dụng HuBERT để biểu diễn âm thanh. Coi biểu cảm là tín hiệu cho cử chỉ. Mô hình đạt được sự kết hợp thời gian thực của đồng thời cả biểu cảm khuôn mặt (facial expression) và cử chỉ (gesture).
- *GestureDiffuCLIP* [4] sử dụng mô hình Diffusion với điều kiện là văn bản, GestureDiffuCLIP sử dụng phương pháp học tương phản (Contrastive Learning) để kết hợp đặc trưng văn bản thông qua CLIP để điều khiển phong cách của cử chỉ. Tương tự các phương pháp trên, coi văn bản như là Prompt của các mô hình sinh văn bản như StableDiffusion, Midjourney giúp mô hình học được cử chỉ từ đoạn văn bản mô tả.
- *Freetalker* [52] sử dụng mô hình Diffusion để huấn luyện trên nhiều tập dữ liệu khác nhau, tạo cử chỉ cho người nói dựa trên lời nói và văn bản. Mô hình dự đoán cử chỉ gốc. Thay vì sử dụng Transformer, Freetalker sử dụng Attention-based Network để tính tương quan giữa các đặc trưng văn bản, âm thanh và cử chỉ trong công đoạn 5. *Kết hợp đặc trưng* ([Hình 2.2](#)).

2.4.2.2 Phương pháp được kế thừa trong luận văn

- *MDM* [44] áp dụng Diffusion có điều kiện cho bài toán sinh cử chỉ, với điều kiện là CLIP (Contrastive Language–Image Pre-training) của văn bản mô tả. MDM sử dụng kiến trúc transformer để tái tạo dữ liệu cử chỉ ban đầu, sử dụng Diffusion có điều kiện với điều kiện là đoạn văn bản mô tả chuyển động của cử chỉ. Tương tự các phương pháp diffusion sử dụng văn bản mô tả, trong công đoạn 3. *Trích xuất đặc trưng* ([Hình 2.2](#)), đoạn văn bản sẽ được mặt nạ ngẫu nhiên (Random Mask) để ẩn đi các đoạn văn bản, từ đó giúp mô hình xác định mức độ quan trọng của từng đoạn văn bản đối với các cử chỉ khác nhau. Trong công đoạn 5. *Kết hợp đặc trưng* ([Hình 2.2](#)), mô hình sử dụng văn bản làm token CLS đầu tiên của chuỗi cử chỉ trước khi đi qua lớp Transformer Encoder. Trong Transformer Encoder, cơ chế self-attention sẽ tính sự tương quan giữa văn bản với từng khung hình của cử chỉ. MDM dự đoán dữ liệu gốc thay vì dự đoán nhiễu.
- **DiffuseStyleGesture** [50] là mô hình Diffusion kế thừa từ *MDM* [44]. Mô hình kết hợp điều kiện bao gồm âm thanh, cử chỉ khởi tạo và phong cách. Trong công đoạn 1. *Tiền xử lý* ([Hình 2.2](#)) mô hình xử lý các tọa độ vector để thu được vectơ có số chiều $D = 1141$ ở mỗi khung hình. Trong công đoạn 2. *Xử lý đặc trưng* ([Hình 2.2](#)), DiffuseStyleGesture sử dụng WavLM để nhúng âm thanh. Trong công đoạn 5. *Kết hợp đặc trưng* ([Hình 2.2](#)), mô hình cải tiến MDM bằng việc sử dụng Cross-Local Attention trước khi đi qua lớp Transformer Encoder.

2.4.3 Mô hình Diffusion phù hợp với bài toán sinh cử chỉ

Với đặc điểm dữ liệu cử chỉ như giá trị của các góc quay và tọa độ điểm khớp, yêu cầu độ chi tiết cao đảm bảo độ chân thực cho chuyển động nhân vật. Tuy nhiên, dữ liệu trong các trường hợp cực trị của các tham số là rất hạn chế. Nhờ vào khả năng học chi tiết và bao quát dữ liệu trong các tình huống hiếm gặp. Mô hình Diffusion có thể giải quyết những khó khăn này, như được nêu trong [Mục 1.5](#). Mô hình Diffusion được đánh giá về ưu và nhược điểm so với các phương pháp hiện tại trong [Bảng 2.1](#). Luận văn chọn mô hình Diffusion để giải quyết bài toán sinh cử chỉ. Mô hình của luận văn kế thừa từ mô hình **DiffuseStyleGesture** [50], tích hợp văn bản như một đặc trưng ngữ nghĩa trong quá trình học trong công đoạn 5. *Kết hợp đặc trưng* ([Hình 2.2](#))

của quá trình sinh cử chỉ để xây dựng mô hình đề xuất **OHGesture**.

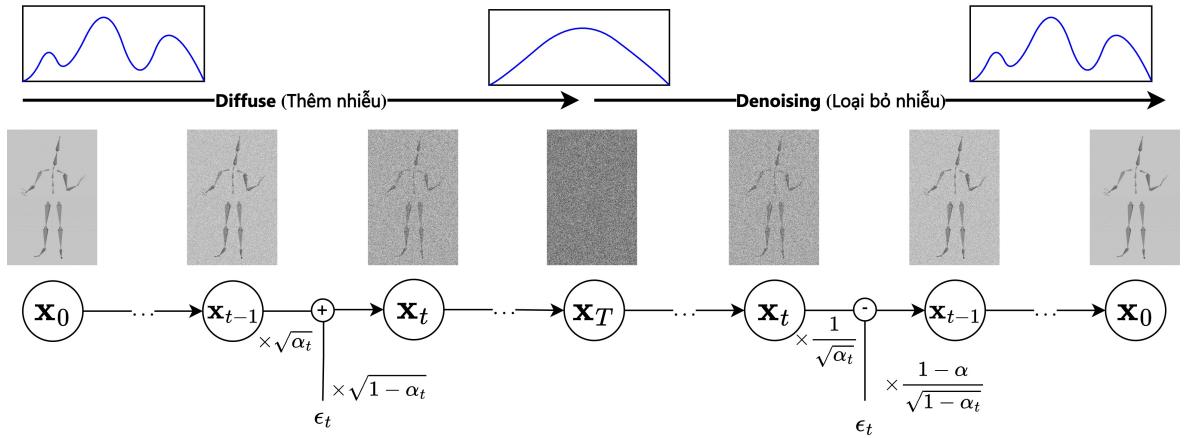
Chương 3. PHƯƠNG PHÁP NGHIÊN CỨU

Bản chất của các phương pháp dựa trên neural network là ước lượng xác suất của dữ liệu (probability density) nên cần chuẩn hóa (normalization), trong khi mô hình Diffusion sẽ học để có thể ước lượng đạo hàm của phân phối dữ liệu [42] (estimating gradients of data distribution) và không phải chuẩn hóa trên toàn bộ dữ liệu, nên có kết quả tốt hơn so với các phương pháp neural network không sử dụng diffusion. Mô hình diffusion có khả năng mô phỏng phân phối dữ liệu trong các trường hợp mật độ phân phối của dữ liệu thấp và có thể sinh được kết quả với độ chi tiết cao.

Mô hình của luận văn dựa trên mô hình DiffuseStyleGesture [53] với cải tiến chính là dựa trên dữ liệu giọng nói, luận văn chuyển giọng nói thành văn bản và nhúng văn bản để thu được các vector đặc trưng văn bản, và sử dụng như là một vector đặc trưng về ngữ nghĩa trong quá trình diffusion có điều kiện. Quá trình này được trình bày [Mục nhở 3.2.2](#). Ngoài ra ở công đoạn 7. *Kết xuất* ([Hình 2.2](#)), luận văn sử dụng Unity để trực quan hóa kết quả sinh cử chỉ. Các đóng góp chính của luận văn được trình bày đầy đủ ở [Mục 6.4](#).

Trước tiên luận văn trình bày về mô hình diffusion [Mục 3.1](#), và phần [Mục 3.2](#) sẽ trình bày về mô hình đề xuất OHGesture.

3.1 Mô hình diffusion cơ bản và các cải tiến



Hình 3.1: Quá trình gây nhiễu (Diffuse) và khử nhiễu (Denoise) **không** huấn luyện

Với các phương pháp sử dụng neural network như ResNet, InceptionNet,... kết quả là tìm được trọng số θ của hàm $f_\theta(x)$, mục tiêu là tối ưu hóa hàm lỗi $\mathcal{L}_{\text{loss}}$ giữa nhãn y và kết quả dự đoán là \hat{y} . Sau khi kết thúc quá trình huấn luyện và học được trọng số θ' ta dự đoán một mẫu dữ liệu mới x' bằng cách truyền qua hàm $f_{\theta'}$ để ra được kết quả dự đoán y' .

Tương tự phương pháp VAE, mã hóa (encode) ma trận đầu vào thành vector tiềm ẩn z và giải mã (decode) vector tiềm ẩn z trở lại ma trận kích thước ban đầu, tuy nhiên mô hình diffusion chia quá trình học thành từng T bước, ở bước thứ t quá trình gây nhiễu (forward diffusion process) $q(\mathbf{x})$ từ $1 \rightarrow T$ được thực hiện bằng cách thêm nhiễu $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$ phân phối chuẩn Gaussian vào dữ liệu, với \mathbf{I} là ma trận đơn vị của phân phối chuẩn Gaussian có giá trị trung bình (mean) bằng $\mathbb{E}[\epsilon] = 0$ và phương sai $\text{Var}(\epsilon) = 1$.

Quá trình thêm nhiễu từ trái qua phải gọi là hàm $q(\mathbf{x})$, còn quá trình khử nhiễu từ phải qua trái $p(\mathbf{x})$. Lưu ý rằng, ϵ_t được lấy ngẫu nhiên ở mọi bước t , và nhiễu ϵ_t được giữ cố định. Như trong [Hình 3.1](#), nếu ở bước thêm nhiễu (diffuse) ta **cộng** một lượng nhiễu ϵ_t và ở bước khử nhiễu (denoise) ta **cũng trừ** một lượng chính xác lượng nhiễu ϵ_t thì kết quả cuối cùng của bước khử nhiễu x_0 là **hoàn toàn giống** ảnh gốc ban đầu.

Tuy nhiên, ở bước khử nhiễu, thay vì **trừ** đi nhiễu thực tế ϵ_t đã được thêm vào ở bước gây nhiễu, luận văn sẽ dùng một hàm f_θ để dự đoán **nhiễu được thêm vào** ở quá trình diffuse, từ đó lần lượt lấy ảnh nhiễu x_T trừ đi nhiễu dự đoán để được ảnh dự

đoán $\hat{\mathbf{x}}_{T-1}$, thực hiện lần lượt cho đến khi đạt được $\hat{\mathbf{x}}_0$.

Trong mô hình Diffusion cơ bản hay Denoising Diffusion Probabilistic Models (DDPM [20]), quá trình khử nhiễu (denoising process) từ $T \rightarrow 1$, mục tiêu là học được trọng số θ của hàm dự đoán nhiễu f_θ hay còn được ký hiệu là hàm dự đoán lượng nhiễu (ϵ_θ) đã được thêm vào. Sau khi kết thúc quá trình học và thu được trọng số θ' của hàm dự đoán nhiễu ϵ , luận văn sẽ dùng hàm $f_{\theta'}$ để dự đoán nhiễu. Sau khi có nhiễu dự đoán $\hat{\epsilon}$, chúng ta trừ đi ảnh bị nhiễu \mathbf{x}_t để có được ảnh khử nhiễu \mathbf{x}_{t-1} , và cộng thêm nhiễu $\mathbf{z} \in \mathcal{N}(0, \mathbf{I})$ để tạo ra sự đa dạng cử chỉ. Thực hiện lần lượt từ $T \rightarrow 1$ để có được ảnh dự đoán $\hat{\mathbf{x}}_0$.

3.1.1 Quá trình gây nhiễu (forward diffusion process)

Cho dữ liệu \mathbf{x}_0 được lấy từ dữ liệu thật $\mathbf{x}_0 \sim q(x)$, với mỗi bước ta sẽ thêm nhiễu vào đầu vào \mathbf{x}_0 với tỷ lệ giữa nhiễu và ảnh gốc được kiểm soát bằng hệ số β :

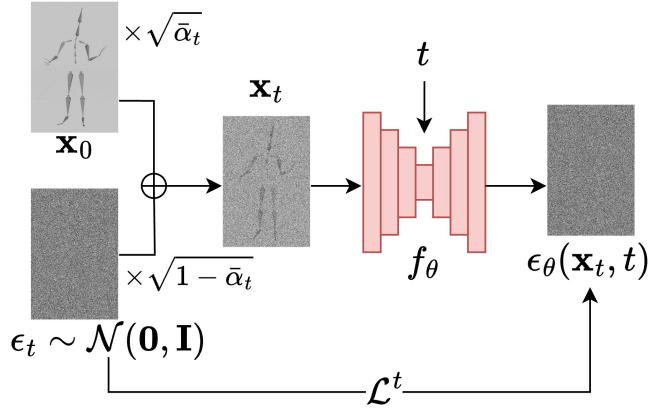
$$\mathbf{x}_t = \sqrt{1 - \beta_t} \mathbf{x}_{t-1} + \sqrt{\beta_t} \epsilon_{t-1} \quad (3.1)$$

Trong đó, quá trình gây nhiễu từ $1 \rightarrow T$, với mỗi bước t , quá trình thêm nhiễu ϵ được điều khiển bởi β_t thuộc tập hợp $\{\beta_t \in (0, 1)\}_{t=1}^T$.

Bởi vì một hàm $f(x)$ có dạng $f(x) = ax + b\epsilon$ với $\epsilon \in \mathcal{N}(0, \mathbf{I})$ sẽ tương đương $f(x) \sim \mathcal{N}(ax, b^2)$ (với ax là kỳ vọng hay trung bình, b^2 là phương sai), từ đó quá trình gây nhiễu có thể được viết gọn là ở từng bước nhiễu t , khi biết trước \mathbf{x}_{t-1} là:

$$\begin{aligned} q(\mathbf{x}_t | \mathbf{x}_{t-1}) &= \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}) \\ q(\mathbf{x}_{1:T} | \mathbf{x}_0) &= \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1}) \end{aligned} \quad (3.2)$$

Mục tiêu ở bước t của hệ số $\sqrt{1 - \beta_t}$ và β_t là để lần lượt giảm tỷ lệ của ảnh gốc \mathbf{x}_t và tăng dần nhiễu ϵ_{t-1} , vì vậy $\beta_1 < \beta_2 < \dots < \beta_T$. Khi $T \rightarrow \infty$ thì \mathbf{x}_T sẽ nhiễu hoàn toàn [47] (Isotropic Gaussian Distribution) $q(\mathbf{x}_T) = \mathcal{N}(0, \mathbf{I})$.



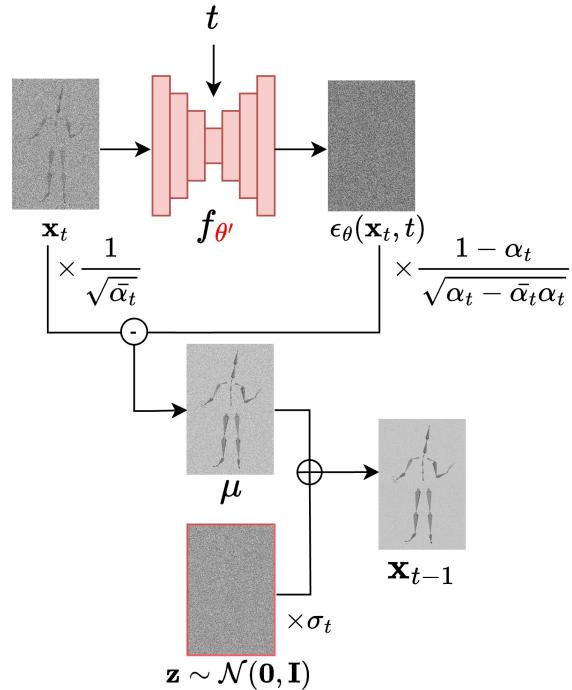
Hình 3.2: Thêm nhiễu trong quá trình huấn luyện

Vì nhiễu $\epsilon_{t-1}, \epsilon_{t-2}, \dots \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ luôn là các biến ngẫu nhiên tuân theo phân phối chuẩn Gaussian, và được xác định trước ở mỗi bước t , nên ta có thể dễ dàng tính \mathbf{x}_t từ \mathbf{x}_0 . Bằng cách đặt $\alpha_t = 1 - \beta_t$ và $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$, từ [Công thức 3.1](#), ta có hàm forward diffusion viết lại theo α như sau:

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \epsilon_{t-1} \\
&= \sqrt{\alpha_t} \left(\sqrt{\alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{1 - \alpha_{t-1}} \epsilon_{t-2} \right) + \sqrt{1 - \alpha_t} \epsilon_{t-1} \\
&= \sqrt{\alpha_t \alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{\alpha_t (1 - \alpha_{t-1})} \epsilon_{t-2} + \sqrt{1 - \alpha_t} \epsilon_{t-1} \\
&= \sqrt{\alpha_t \alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{\alpha_t (1 - \alpha_{t-1}) + (1 - \alpha_t)} \epsilon_{t-2} \\
&= \sqrt{\alpha_t \alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}} \epsilon_{t-2} \\
&= \dots \\
&= \sqrt{\prod_{i=1}^t \alpha_i} \mathbf{x}_0 + \sqrt{1 - \prod_{i=1}^t \alpha_i} \epsilon_0 \\
&= \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_0 \\
&\sim \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})
\end{aligned} \tag{3.3}$$

Sự thay đổi của $\sqrt{\alpha}$ và $\sqrt{1 - \alpha}$ trong quá trình gây nhiễu được thể hiện ở phụ lục [Phụ lục A](#)

3.1.2 Quá trình khử nhiễu (denoising process)



Hình 3.3: Khử nhiễu trong quá trình suy luận

Quá trình khử nhiễu $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ ở bước thứ t từ $T \rightarrow 1$ được bắt đầu từ \mathbf{x}_T là nhiễu hoàn toàn $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Luận văn sẽ dùng một neural network $f_\theta(\mathbf{x}_t, t)$ để dự đoán nhiễu $\hat{\epsilon}_t = f_\theta(\mathbf{x}_t, t)$ đã được thêm vào để được \mathbf{x}_{t-1} từ \mathbf{x}_t .

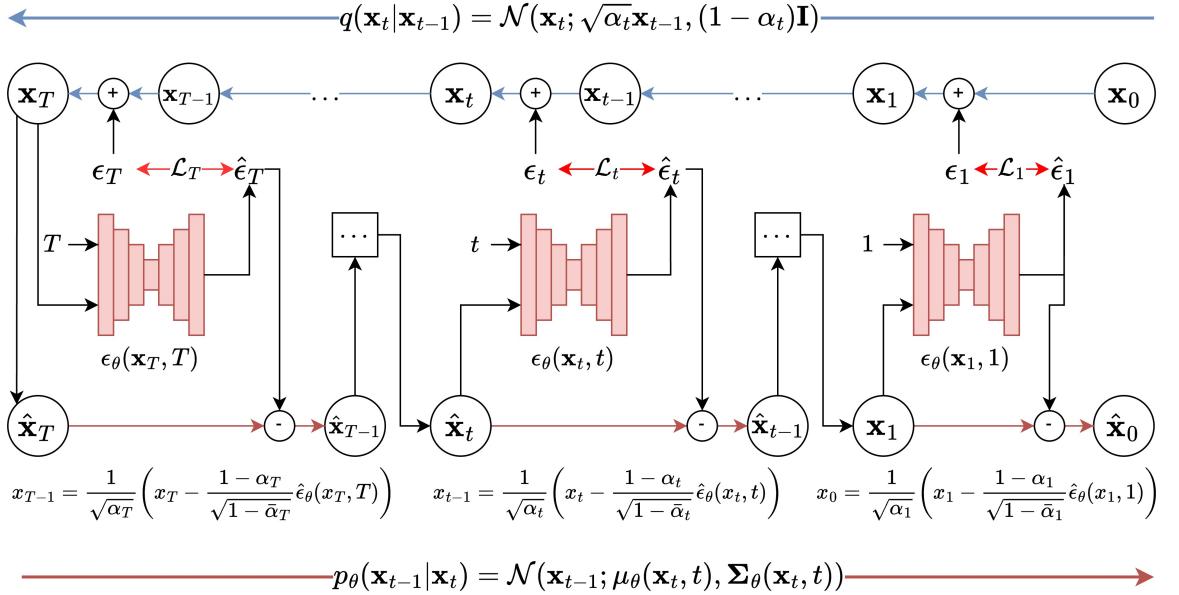
Quá trình khử nhiễu có trung bình $\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} f_\theta(\mathbf{x}_t, t) \right)$ và phương sai $\Sigma_\theta(\mathbf{x}_t, t)$ như sau:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \quad (3.4)$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

Quá trình khử nhiễu là quá trình bắt đầu từ nhiễu hoàn toàn, dùng một neural network f_θ để học được quá trình khử nhiễu.

3.1.3 Quá trình huấn luyện trong mô hình diffusion cơ bản



Hình 3.4: Mô hình diffusion cơ bản

Mô hình diffusion sẽ học trọng số θ của hàm dự đoán lỗi $f_\theta(x_t, t)$ hay còn ký hiệu là $\epsilon_\theta(x_t, t)$. Trong quá trình denoising, ta sẽ tối ưu độ lỗi giữa nhiễu dự đoán $\epsilon_\theta(\mathbf{x}_t, t)$ và nhiễu thực tế ϵ_t . Với mỗi bước thứ t ta sẽ tối ưu hàm loss \mathcal{L}_t để thu được trọng số θ .

$$\begin{aligned} \mathcal{L}^t &= \mathbb{E}_{t \sim [1, T], \mathbf{x}_0, \epsilon_t} \left[\|\epsilon_t - \epsilon_\theta(\mathbf{x}_t, t)\|^2 \right] \\ &= \mathbb{E}_{t \sim [1, T], \mathbf{x}_0, \epsilon_t} \left[\|\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_t, t)\|^2 \right] \end{aligned} \quad (3.5)$$

Trong đó, hàm măt măt \mathcal{L} tống sê là $\mathcal{L} = \sum_{t=1}^T \mathcal{L}^t$.

Với $f_\theta(x_{t-1}, t)$ hay ϵ_θ là một mô hình Unet dùng để mã hóa và giải mã dữ liệu để dự đoán nhiễu đã thêm vào dữ liệu. Quá trình tính toán hàm lỗi được trình bày minh họa trong [Hình 3.4](#).

Thuật toán 1 Thuật toán training trong DDPM

1. Tính sẵn các giá trị $\sqrt{\alpha_t}$, $\sqrt{1 - \alpha_t}$ và $\sqrt{\bar{\alpha}_t}$ cho mỗi bước $t : 1 \rightarrow T$. Xác định lịch trình nhiễu $\{\alpha_t \in (0, 1)\}_{t=1}^T$, với $\alpha_1 < \alpha_2 < \dots < \alpha_T$.
2. Lấy nhãn \mathbf{x}_0 từ phân phối của dữ liệu đã chuẩn hóa.
3. Tạo nhiễu ngẫu nhiên ϵ_t cho mỗi bước $t : 1 \rightarrow T$, với $\forall t : \epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
4. Gây nhiễu (forward) \mathbf{x}_0 để thu được \mathbf{x}_t ở mỗi bước $t : 1 \rightarrow T$:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_t$$

5. Với mỗi t , lấy t **ngẫu nhiên** từ $[1, T]$.
6. Cho \mathbf{x}_t và t vào mô hình để dự đoán nhiễu: $\hat{\epsilon} = \epsilon_\theta(\mathbf{x}_t, t)$.
7. Tính đạo hàm để cập nhật trọng số:

$$\nabla_{\theta_t} \|\epsilon_t - \epsilon_\theta(\mathbf{x}_t, t)\|^2$$

Tính loss:

$$\mathcal{L}^t = \mathbb{E}_{t \sim [1, T], \mathbf{x}_0, \epsilon_t} [\|\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_t, t)\|^2]$$

8. Quay lại bước 6 cho đến khi hội tụ để thu được trọng số tối ưu θ' .
-

3.1.4 Quá trình lấy mẫu trong diffusion cơ bản

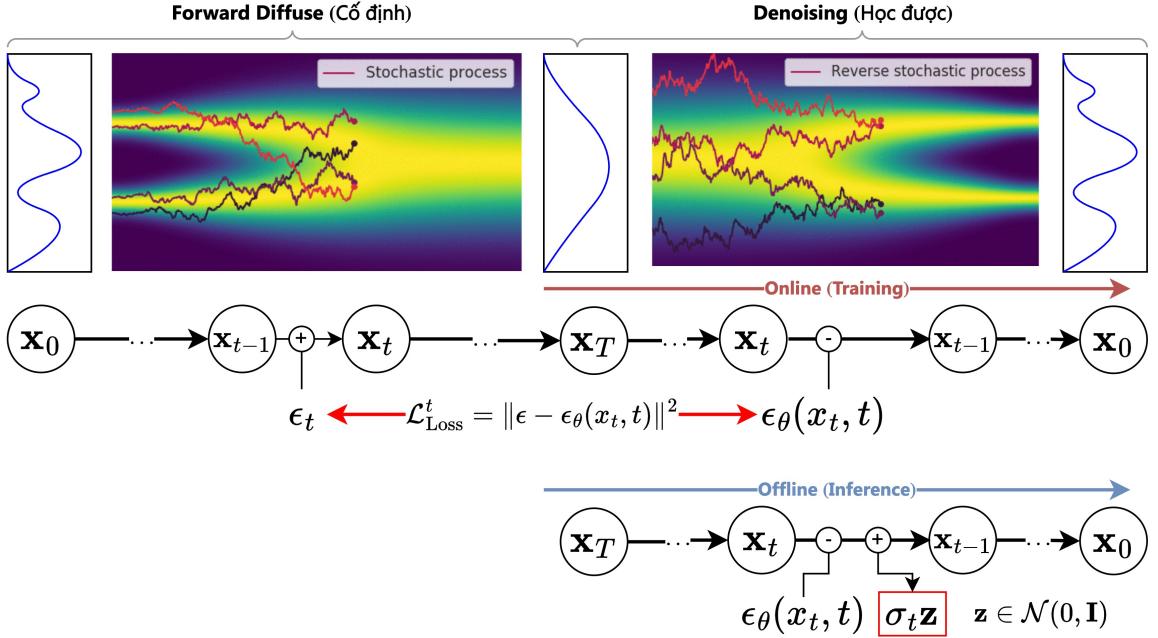
Sau khi thu được trọng số θ' , luận văn sẽ dùng hàm denoising để khử nhiễu từ nhiễu $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Quá trình biến đổi từ nhiễu hoàn toàn \mathbf{x}_T sang dự đoán $\hat{\mathbf{x}}_0$ như sau:

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} f_{\theta'}(\mathbf{x}_t, t) \right) + \sqrt{1 - \alpha_t} \tilde{\epsilon}_t \quad (3.6)$$

Lưu ý rằng ϵ_t là nhiễu cố định và nhiễu ngẫu nhiên được tạo ra trước quá trình huấn luyện, chỉ sử dụng lại kết quả ngẫu nhiên trong quá trình forward diffusion [Mục nhỏ 3.1.2](#) ở công thức [Công thức 3.1](#). Như [Hình 3.5](#), độ lỗi ϵ_t là độ nhiễu của từng bước t , và hàm loss \mathcal{L}^t sẽ tính theo từng bước t .

Tiếp theo quá trình huấn luyện, thuật toán lấy mẫu (sampling) trong DDPM bắt đầu từ bước tạo nhiễu hoàn toàn, tức là $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, nơi dữ liệu ban đầu hoàn toàn là nhiễu. Các giá trị $\sqrt{\alpha_t}$, $\sqrt{1 - \alpha_t}$ và $\sqrt{\bar{\alpha}_t}$, được tính từ quá trình huấn luyện, sẽ được sử dụng trong quá trình lấy mẫu để dự đoán lại dữ liệu gốc \mathbf{x}_0 . Bước tiếp theo là tính

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t} \mathbf{x}_t, (1 - \alpha_t) \mathbf{I}) \rightarrow p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)) \rightarrow$$



Hình 3.5: Quá trình Training và Sampling trong mô hình Diffusion tiêu chuẩn

toán hệ số điều chỉnh nhiễu σ_t , dựa vào lịch trình nhiễu α_t đã được thiết lập trong quá trình huấn luyện. Các giá trị này sẽ ảnh hưởng đến độ nhiễu được thêm vào trong quá trình lấy mẫu ngược.

Quá trình lấy mẫu (sampling) như [Thuật toán 2](#) sẽ được thực hiện từ bước T trở về 1, và trong mỗi bước, một nhiễu ngẫu nhiên $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ được tạo ra để cộng thêm vào kết quả dự đoán. Tại mỗi bước t , mô hình sẽ dự đoán nhiễu ϵ_θ dựa trên dữ liệu nhiễu \mathbf{x}_t và bước thời gian t , sau đó sử dụng dự đoán này để tính toán giá trị μ , là ước lượng của \mathbf{x}_0 . Cuối cùng, một lượng nhiễu $\sigma_t \mathbf{z}$ được cộng thêm vào μ để thu được $\hat{\mathbf{x}}_{t-1}$, dữ liệu nhiễu tại bước $t-1$. Quá trình này tiếp tục cho đến khi $t = 1$, và khi đó, chúng ta có được $\hat{\mathbf{x}}_0$ — dự đoán cuối cùng của dữ liệu gốc từ quá trình khử nhiễu.

Thuật toán 2 Thuật toán sampling trong DDPM

1. Bắt đầu với nhiễu: $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2. Các giá trị $\sqrt{\alpha_t}$, $\sqrt{1 - \alpha_t}$ và $\sqrt{\bar{\alpha}_t}$ được lấy từ quá trình huấn luyện.
3. Tính hệ số điều chỉnh nhiễu σ_t từ α_t ở mỗi bước $t : 1 \rightarrow T$:

$$\sigma_t = \sqrt{\frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} (1 - \alpha_t)}$$

4. Với mỗi t , lấy t **tuần tự** từ $[T, \dots, 1]$.
5. Tạo nhiễu ngẫu nhiên $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$.
6. Đưa \mathbf{x}_t vào mô hình để suy luận nhiễu: $\epsilon_{\theta'} = \epsilon_{\theta'}(\mathbf{x}_t, t)$.
7. Dùng nhiễu dự đoán để trừ đi \mathbf{x}_t ở bước t :

$$\mu = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_{\theta'}(\mathbf{x}_t, t) \right)$$

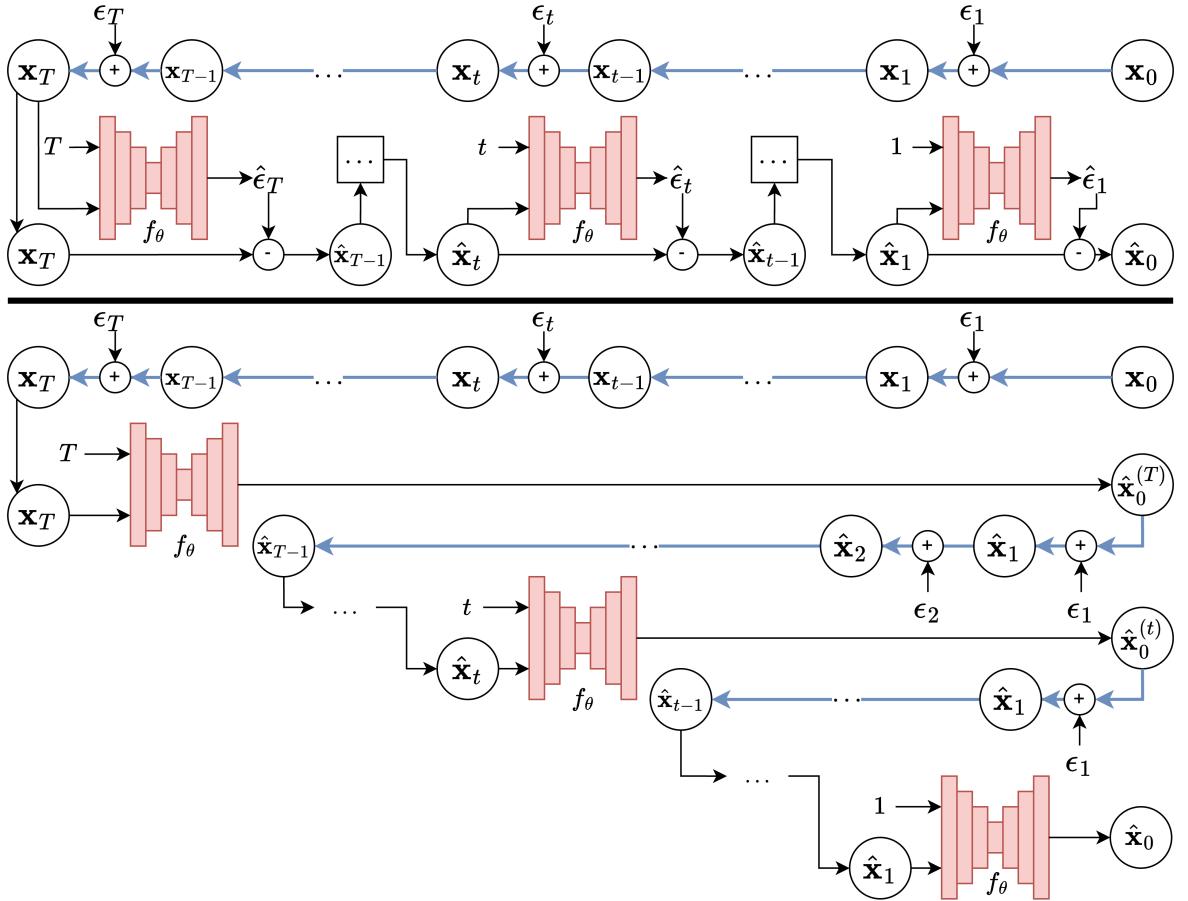
8. Cộng thêm một lượng nhiễu: $\hat{\mathbf{x}}_{t-1} = \mu + \sigma_t \mathbf{z}$.
 9. Khi $t = 1$, thu được $\hat{\mathbf{x}}_0$ từ quá trình khử nhiễu.
-

Điều quan trọng nhất trong quá trình lấy mẫu (denoising sampling) đó chính là phải cộng thêm một độ nhiễu $\mathbf{z} \in \mathcal{N}(0, \mathbf{I})$, với \mathbf{z} sẽ được lần lượt thêm ở từng bước t bằng hệ số điều khiển σ_t . Mục tiêu của nhiễu ϵ là để làm gờ phân phối (marginal noise distribution) cho mô hình f_θ (hay ϵ_θ) có thể học được nhiễu, còn với \mathbf{z} là để tăng tính đa dạng trong quá trình sinh và sự ổn định trong quá trình lấy mẫu. Quá trình giảm σ_t được trình bày ở phụ lục [Mục A.3](#)

3.1.5 Cải tiến mô hình diffusion với dự đoán \mathbf{x}_0 thay vì ϵ_t

Dựa vào [Công thức 3.3](#), có thể thấy nếu có \mathbf{x}_t thì sẽ suy ra được \mathbf{x}_0 . Và nếu có \mathbf{x}_0 thì cũng sẽ có thể suy ra được \mathbf{x}_t **một cách tường minh** với t bất kỳ bằng cách thêm nhiễu ϵ_t đã có từ quá trình thêm nhiễu thuận (forward diffusion).

Từ quan sát này, nhóm tác giả [\[37\]](#) đề xuất cải tiến DDPM, thay vì dùng hàm neural network $f_\theta(\mathbf{x}_t, t)$ để dự đoán ϵ_t như [Hình 3.4](#) thì $f_\theta(\mathbf{x}_t, t)$ sẽ được dùng để dự đoán trực tiếp \mathbf{x}_0 , và khi có \mathbf{x}_0 sẽ thêm nhiễu bằng [Công thức 3.3](#)



Hình 3.6: So sánh ϵ objective (bên trên) và x_0 objective (bên dưới)

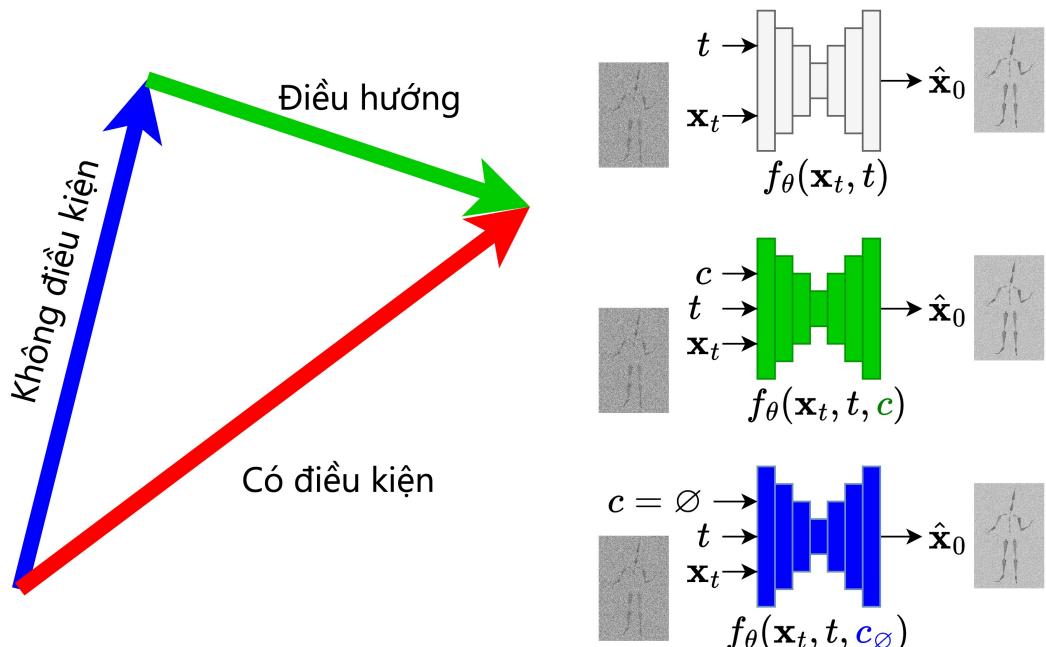
Như hình [Hình 3.6](#), ta bắt đầu bằng việc thêm nhiễu thuận (forward diffusion), từ $t : 1 \rightarrow T$ để được $x_T \sim \mathcal{N}(0, \mathbf{I})$. Sau khi có nhiễu x_T ta sẽ đưa vào hàm f_θ để dự đoán \hat{x}_0 , từ \hat{x}_0 sẽ thêm nhiễu để được \hat{x}_{t-1} . Thực hiện lần lượt đến khi $t = 1$, ta sẽ thu được \hat{x}_0 . Ta có thể so sánh hai phương pháp như sau:

- **ϵ objective** : mô hình sẽ dự đoán lỗi. Bắt đầu từ việc forward quá trình gây nhiễu để lấy được x_T , khi có được x_T ta sẽ sử dụng $x_T \in \mathcal{N}(0, \mathbf{I})$ để đưa vào quá trình denoise. Trong quá trình khử nhiễu, mô hình sẽ dự đoán nhiễu $\hat{\epsilon}_t$ đã được thêm vào từ quá trình forward, là nhiễu ϵ_t và tối ưu lỗi giữa nhiễu dự đoán và nhiễu thực tế từ quá trình forward.
- **x_0 objective** : tương tự mô hình sẽ gây nhiễu (forward diffusion) quá trình gây nhiễu để lấy được x_T , khi có được x_T ta sẽ sử dụng $x_T \in \mathcal{N}(0, \mathbf{I})$ để đưa vào quá trình denoise. Mô hình sẽ dự đoán trực tiếp x_0 , sau khi có x_0 mô hình sẽ tiếp

tục gây nhiễu đến bước thứ \mathbf{x}_{t-1} , và tiếp tục sử dụng \mathbf{x}_{t-1} để đưa vào mô hình dự đoán \mathbf{x}_0

3.1.6 Mô hình diffusion có điều kiện

Để có thể điều khiển quá trình sinh với các điều kiện khác nhau, thì cần phải sinh với điều kiện c , hay nói cách khác ta cần tìm xác suất $p(\mathbf{x}|c)$ khi biết trước c . Nhóm tác giả [14] đề xuất sử dụng một hàm f_ϕ riêng để huấn luyện cho điều kiện. Tuy nhiên phương pháp này gây khó khăn khi các điều kiện sinh thay đổi và việc kết hợp và cập nhật trọng số ở một mô hình riêng sẽ gây khó khăn khi mở rộng.



Hình 3.7: Diffusion có điều kiện bằng vector điều hướng

Như hình [Hình 3.5](#), để quá trình suy luận có thể sinh ra các kết quả \mathbf{x}_0 khác nhau và có thể điều khiển được dựa trên điều kiện c . Ở mỗi bước t , cần thêm một lượng điều hướng với một điều kiện cụ thể. Nhóm tác giả đề xuất Classifier-Free Diffusion Guidance [21] với việc kết quả \mathbf{x}_0 được cập nhật bằng cách cộng kết quả có điều kiện và không có điều kiện.

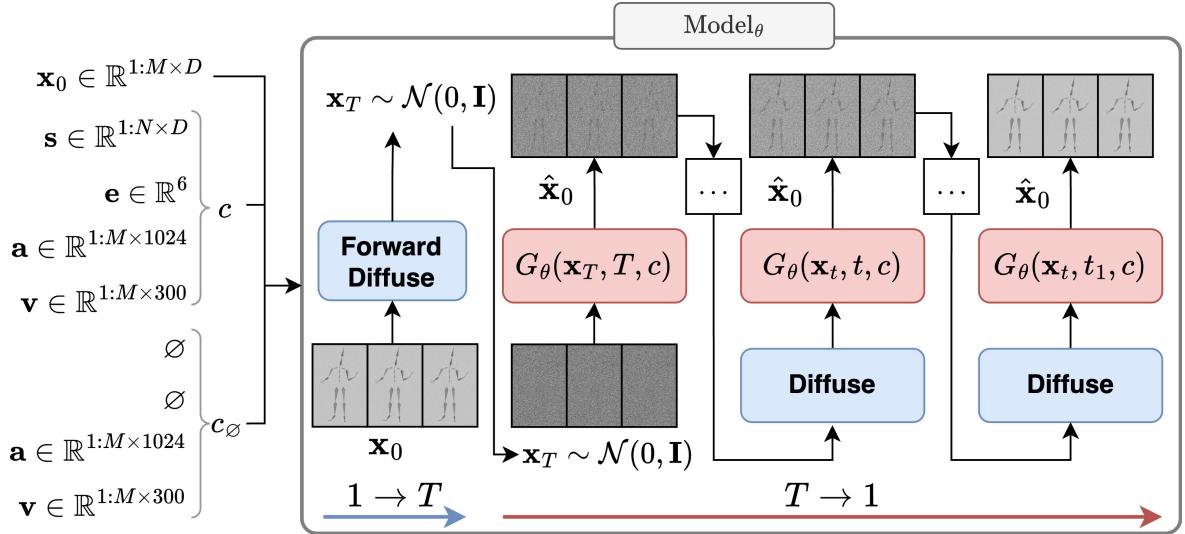
$$\hat{\mathbf{x}}_{0\gamma, \textcolor{red}{c}, \textcolor{blue}{c}_\emptyset} = \gamma \cdot f_\theta(\mathbf{x}_t, t, \textcolor{red}{c}) + (1 - \gamma) \cdot f_\theta(\mathbf{x}_t, t, \textcolor{blue}{c}_\emptyset) \quad (3.7)$$

Trong đó, c là điều kiện, c_\emptyset là điều kiện rỗng, kết quả sinh điều kiện được điều khiển bằng tham số γ , nếu γ càng lớn thì kết quả sinh ra càng sát với điều kiện c ,

ngược lại thì kết quả sẽ nghiêng về kết quả sinh không có điều kiện.

Như [Hình 3.7](#), hàm f_θ trên cùng là không có điều kiện, f_θ giữa là diffusion có điều kiện, f_θ (dưới cùng) là diffusion với điều kiện rỗng.

3.2 Mô hình của đề xuất OHGesture



Hình 3.8: Tổng quan về mô hình OHGesture

Mô hình đề xuất **OHGesture** (Open Human Gesture Generation) của luận văn được dựa trên mô hình **DiffuseStyleGesture** [50] áp dụng mô hình Diffusion [20] có điều kiện [21] (Classifier-Free Diffusion Guidance) để điều khiển các đặc trưng trong quá trình khử nhiễu.

Những điểm giống và khác của việc áp dụng mô hình diffusion cho bài toán sinh cử chỉ (gesture generation) so với mô hình Diffusion trong bài toán sinh ảnh:

Điểm giống

- Sử dụng mô hình Diffusion (Mục 3.1) trên cử chỉ $\mathbf{x}^{1:M \times D}$, với M khung hình theo thời gian, $D = 1141$ là các điểm tọa độ chuyển động của mỗi khung hình (tương tự width và height trong ảnh).
- Sử dụng Diffusion có điều kiện (Mục nhỏ 3.1.6) với \mathbf{x}_0 objective (Mục nhỏ 3.1.5).
- Ở công đoạn 4. *Mã hóa đặc trưng* và công đoạn 6. *Giải mã đặc trưng* Hình 2.2, mô hình sử dụng latent vector có số chiều là 256.

Điểm khác

- Sinh cử chỉ có điều kiện:

- Điều kiện cảm xúc: $c = [s, e, a, v]$ và $c_\emptyset = [\emptyset, \emptyset, a, v]$.
- Nội suy trạng thái giữa hai cảm xúc e_1, e_2 , sử dụng điều kiện: $c = [s, e_1, a, v]$ và $c_\emptyset = [s, e_2, a, v]$.
- Ở công đoạn 5. *Kết hợp đặc trưng* [Hình 2.2](#), mô hình sử dụng Self-Attention: Mỗi liên hệ giữa các cảm xúc, cử chỉ khởi tạo và từng khung hình (tương tự Dall-E 2 tính mối liên hệ giữa văn bản và ảnh).
- Ở công đoạn 5. *Kết hợp đặc trưng* [Hình 2.2](#), mô hình concat giọng nói và văn bản (Tương tự như ControlNet sử dụng Pixel-wise Condition)

Trong đó x_0 là chuỗi M khung hình cử chỉ $x \in \mathbb{R}^{1:M \times D}$ ($D = 1141$), với điều kiện $c = [s, e, a, v]$ bao gồm cử chỉ khởi tạo (seed gesture) s , cảm xúc (emotion) e , chuỗi giọng nói (audio) a tương ứng cử chỉ, và văn bản v .

Mục tiêu của mô hình là học được tham số θ của hàm G_θ (generative) với đầu vào là ma trận cử chỉ $x_t \in \mathbb{R}^{1:M \times D}$, bước thời gian t và điều kiện c . Tổng quan của mô hình đề xuất **OHGesture** được minh họa ở hình [Hình 3.8](#). Tương tự mô hình diffusion cơ bản bao gồm hai quá trình: quá trình tạo nhiễu (diffusion) q và quá trình khử nhiễu (denoising process) p_θ với trọng số θ . Công đoạn 1. *Tiền xử lý* sẽ được trình bày ở [Mục 4.2](#).

3.2.1 Công đoạn xử lý đặc trưng

Trong công đoạn 2. *Xử lý đặc trưng* ([Hình 2.2](#)), mục tiêu là chuyển dữ liệu thành các ma trận hoặc vector trước khi đưa vào mô hình.

- **Văn bản** (Text) $v \in \mathbb{R}^{1:M \times 300}$: Như đã trình bày trong [Mục nhở 2.4.2, MDM](#) [44], *DiffuseStyleGesture+* [53] trong công đoạn này sử dụng các đoạn văn bản mô tả cử chỉ như là Prompt của Midjourney làm đầu vào cho mô hình, tuy nhiên văn bản mô tả được dùng để phân cụm từng cử chỉ. Trong khi mục tiêu của luận văn là sử dụng văn bản như là một đặc trưng ngữ nghĩa, để căn chỉnh từng đoạn văn bản tương ứng với từng đoạn cử chỉ phục vụ cho việc xây dựng người kỹ thuật số.

Vì vậy, trong công đoạn này đóng góp của luận văn là sử dụng dữ liệu tiếng nói có sẵn, sau khi tiền xử lý ở [Mục 4.2](#) để thu được văn bản, luận văn sử dụng mô hình FastText [8] để nhúng (embedding) văn bản thành các vector căn

chỉnh tương ứng với số khung hình tương ứng với cử chỉ. Với các vùng không có văn bản sẽ được cho là ma trận 0, các vùng có từ vựng tương ứng sẽ được gán là giá trị nhúng theo từng khung hình của cử chỉ để được ma trận văn bản $v \in \mathbb{R}^{1:M \times 300}$.

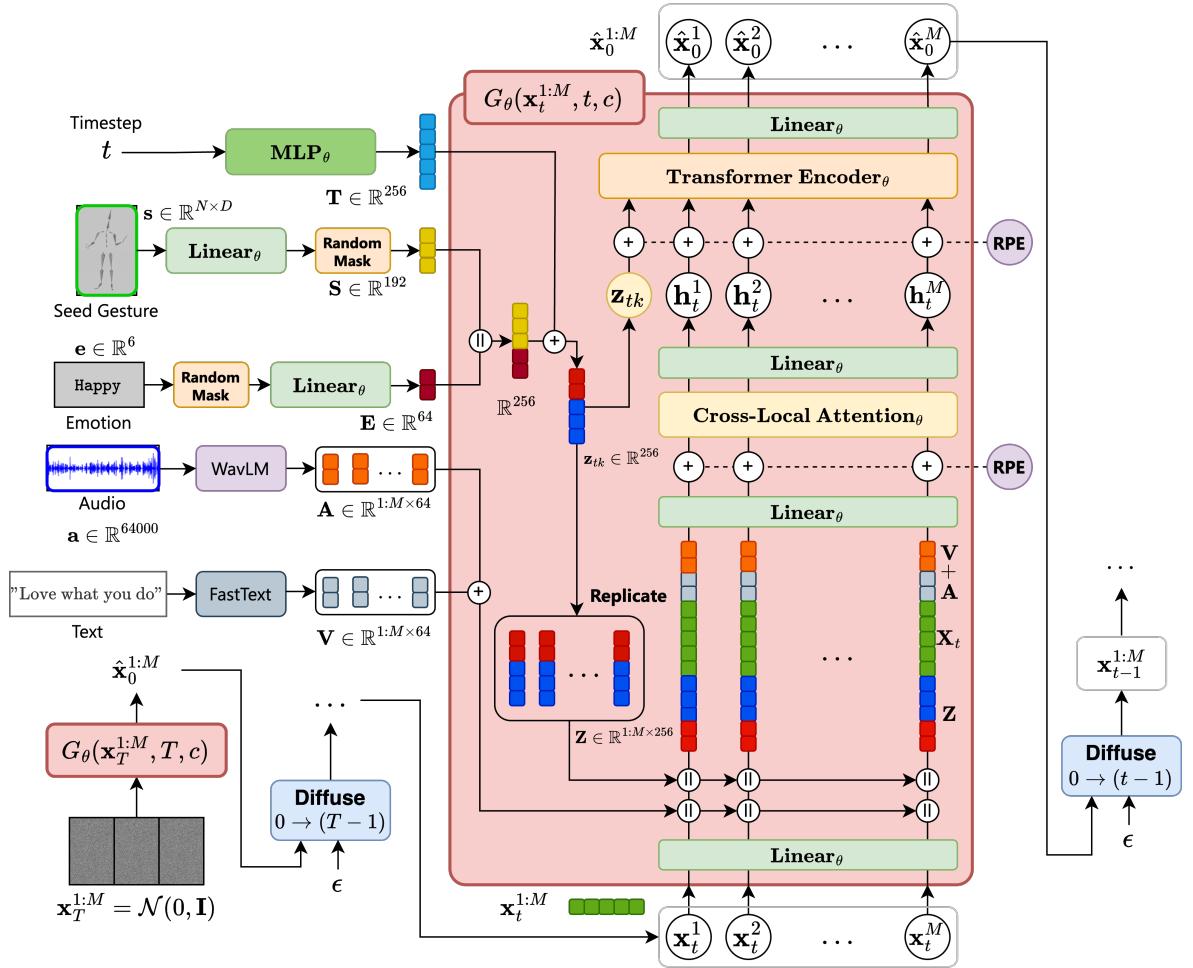
- **Giọng nói** (Speech) $a \in \mathbb{R}^{1:M \times 1024}$: Tất cả dữ liệu giọng nói ở dạng wav sẽ được giảm số sample rate (downsampled) xuống 16kHz, và giọng nói sẽ được lấy tương ứng với cử chỉ là 4s để được vector $a \in \mathbb{R}^{64000}$. Tương tự DiffuseStyleGesture, luận văn sử dụng mô hình pre-train WavLM Large [11] nhúng waveform thô vào để được vector thể hiện đặc trưng âm thanh. Sau đó sử dụng nội suy tuyến tính (interpolation) để chỉnh đặc trưng của vector tiềm ẩn trong WavLM theo chiều thời gian thành 20fps để được ma trận giọng nói $a \in \mathbb{R}^{1:M \times 1024}$
- **Cảm xúc** (Emotion) $e \in \mathbb{R}^6$: Cảm xúc là nhãn được chọn 1 trong 6 cảm xúc bao gồm: Happy, Sad, Neutral, Old, Relaxed và Angry.

Sau đó nhãn cảm xúc được đi qua lớp one-hot encoding để được vector $e \in \mathbb{R}^6$.

- **Cử chỉ khởi tạo** (Seed Gesture) $s \in \mathbb{R}^{1:N \times D}$: là chuỗi cử chỉ $N = 8$ khung hình bắt đầu, với mỗi khung hình là dữ liệu bao gồm 75 khớp xương, được xử lý theo công thức [Công thức 4.1](#) để được vector $D = 1141$ chiều.
- **Cử chỉ thật** (Ground Truth) $x_0 \in \mathbb{R}^{1:M \times D}$: Là chuỗi cử chỉ $M = 80$ khung hình (4 giây với 20fps), dữ liệu cử chỉ thật được tiền xử lý để được ma trận $x_0 \in \mathbb{R}^{1:M \times D}$.

3.2.2 Công đoạn trích xuất đặc trưng

Trong công đoạn 3. *Trích xuất đặc trưng* ([Hình 2.2](#)), mục tiêu là chuyển ma trận thành các vector tiềm ẩn (latent vector) biểu diễn thông tin của dữ liệu bằng cách cho các đặc trưng cần học đi qua các lớp biến đổi tuyến tính (linear) hoặc MLP (Multilayer Perceptron).



Hình 3.9: Mô hình trong OHGesture

- **Bước thời gian** $T \in \mathbb{R}^{256}$: Bước thời gian ở mỗi quá trình $t \in [0, T]$, mục tiêu của mô hình là để với bước thời gian t bất kỳ, mô hình có thể tổng quát hóa quá trình khử nhiễu, và có thể học được việc với từng bước t thì giá trị sẽ thay đổi thế nào đối với kết quả dự đoán x_0 . Giá trị timestep t được khởi tạo bằng việc mã hóa nhúng vị trí (Position Embedding) bằng hàm sinusoidal $PE(t) = [\sin\left(\frac{t}{10000^{2i/d}}\right), \cos\left(\frac{t}{10000^{2i/d}}\right)]$, sau đó được đưa vào một MLP (Multilayer perceptron) để biến đổi thành vector $T \in \mathbb{R}^{256}$.
- **Giọng nói** $A \in \mathbb{R}^{1:M \times 64}$: Từ giọng nói $a \in \mathbb{R}^{1:M \times 1024}$ sẽ đi qua một lớp tuyến tính (linear layer) để giảm kích thước của đặc trưng xuống còn vector đặc trưng 64 chiều để tạo thành ma trận cuối cùng $A \in \mathbb{R}^{1:M \times 64}$.
- **Văn bản** $V \in \mathbb{R}^{1:M \times 64}$: Văn bản sau quá trình tiền xử lý ở mục [Mục 4.2](#) sẽ được ma trận căn chỉnh tương ứng số khung hình M để được ma trận $v \in \mathbb{R}^{1:M \times 300}$,

văn bản sau đó được đi qua hàm biến đổi tuyến tính nhằm giảm chiều đặc trưng, và thu được ma trận $\mathbf{V} \in \mathbb{R}^{1:M \times 64}$ căn chỉnh tương ứng với ma trận âm thanh.

- **Cử chỉ khởi tạo $\mathbf{S} \in \mathbb{R}^{192}$:** Từ cử chỉ khởi tạo $\mathbf{s} \in \mathbb{R}^{1:N \times D}$ sẽ đi qua lớp biến đổi tuyến tính (linear layer) để được vector $\mathbf{S} \in \mathbb{R}^{192}$, vector \mathbf{S} sau đó được đi qua một lớp mặt nạ ngẫu nhiên (random mask) trong quá trình huấn luyện, mục tiêu là ẩn hoặc hiện từng đoạn cử chỉ trong N khung hình, và để mô hình có thể học được nếu từng khung hình bị thiếu thì sẽ ảnh hưởng như thế nào đến cử chỉ dự đoán.
- **Cảm xúc $\mathbf{E} \in \mathbb{R}^{64}$:** Từ vector cảm xúc $\mathbf{e} \in \mathbb{R}^6$ sẽ được đưa qua lớp tuyến tính (linear layer) để được vector đặc trưng $\mathbf{E} \in \mathbb{R}^{64}$. Mục tiêu là để có thể concat được với vector khởi tạo $\mathbf{S} \in \mathbb{R}^{192}$ để tạo thành vector tiềm ẩn 256 chiều.
- **Cử chỉ nhiễu $\mathbf{x}_T \in \mathbb{R}^{1:M \times D}$ (Noisy gesture):** khi huấn luyện, \mathbf{x}_t là cử chỉ nhiễu có cùng kích thước như cử chỉ gốc \mathbf{x}_0 sẽ được sinh ngẫu nhiên từ phân phối chuẩn $\mathcal{N}(0, \mathbf{I})$. Khi sinh ngẫu nhiên, cử chỉ nhiễu ban đầu \mathbf{x}_T được lấy mẫu từ phân phối Gaussian và các $\mathbf{x}_t, t < T$ khác là kết quả của bước làm nhiễu trước đó như [Hình 3.8](#).

3.2.3 Công đoạn mã hóa đặc trưng

Trong công đoạn 4. *Mã hóa đặc trưng* ([Hình 2.2](#)), mục tiêu là giảm số chiều của dữ liệu xuống kích thước thấp hơn, nhằm giảm việc bùng nổ khối lượng tính toán.

Với dữ liệu chính trong quá trình Diffusion chính là chuỗi cử chỉ $\mathbf{x}_t \in \mathbb{R}^{1:M \times D}$. Như minh họa ở [Hình 3.9](#), chuỗi cử chỉ với kích thước $M \times D$ sẽ đi qua lớp Linear_{θ} để được ma trận $\mathbf{X} \in \mathbb{R}^{1:M \times 256}$. Quá trình giảm chiều dữ liệu của \mathbf{x} sẽ được thực hiện trước khi chuỗi cử chỉ \mathbf{x} đi qua các lớp Cross-Local Attention và Transformer Encoder để tính sự tương quan giữa nhiều loại dữ liệu khác nhau.

3.2.4 Công đoạn kết hợp đặc trưng

Trong công đoạn 5. *Kết hợp đặc trưng* ([Hình 2.2](#)), mục tiêu là sử dụng các phép concat, cộng hoặc sử dụng các lớp attention để tính sự tương quan giữa các đặc trưng.

Đầu tiên cử chỉ khởi tạo $\mathbf{S} \in \mathbb{R}^{192}$ và vector cảm xúc $\mathbf{E} \in \mathbb{R}^{64}$ được ghép lại với nhau để tạo thành một vectơ có kích thước 256, bởi vì kích thước 256 là kích thước

vector tiềm ẩn được chọn để tính tương quan giữa các đặc trưng. Sau đó được cộng thêm vector timestep T để tạo thành vector $\mathbf{z}_{tk} \in \mathbb{R}^{256}$.

$$\mathbf{z}_{tk} = \text{concat}(\mathbf{E} \parallel \mathbf{S}) + \mathbf{T} \quad (3.8)$$

Quá trình kết hợp đặc trưng của \mathbf{z}_{tk} này được minh họa ở [Hình 3.11a](#).

3.2.4.1 Kết hợp đặc trưng theo khung hình

Sau đó, $\mathbf{z}_{tk} \xrightarrow{\text{replicate}} \mathbf{Z}$ sao chép (replicate) M lần để căn chỉnh kích thước tương đương với kích thước M khung hình để được ma trận $\mathbf{Z} \in \mathbb{R}^{1:M \times 256}$ như [Hình 3.9](#).

Ta sẽ cộng ma trận giọng nói \mathbf{A} và văn bản \mathbf{V} với nhau để được ma trận kết hợp thông tin của văn bản và giọng nói. Sau đó tiếp tục kết hợp concat với ma trận cù chỉ \mathbf{X} . Cuối cùng concat với ma trận \mathbf{Z} để được ma trận \mathbf{M} .

$$\mathbf{M} = \text{concat}(\mathbf{Z} \parallel \text{concat}(\mathbf{X} \parallel (\mathbf{V} + \mathbf{A}))) \quad (3.9)$$

Ma trận $\mathbf{M} \in \mathbb{R}^{1:M \times P}$ theo [Công thức 3.9](#) thể hiện đặc trưng theo khung hình từ khung hình 1 đến khung hình M , với mỗi khung hình có kích thước P là tổng của các đặc trưng đã concat. Với $\mathbf{X} \in \mathbb{R}^{1:M \times 256}$, $\mathbf{Z} \in \mathbb{R}^{1:M \times 256}$ và $\mathbf{A}, \mathbf{V} \in \mathbb{R}^{1:M \times 64}$, P sẽ có kích thước $P = 256 + 256 + 64$. Sau đó, ma trận \mathbf{m} sẽ được đi qua quá trình biến đổi tuyến tính để giảm chiều từ P chiều xuống 256 chiều để được ma trận $\mathbf{m}_t \in \mathbb{R}^{1:M \times 256}$.

$$\mathbf{m}_t = \text{Linear}_{\theta}(\mathbf{M}) \quad (3.10)$$

3.2.4.2 Cơ chế attention trong quá trình kết hợp các đặc trưng

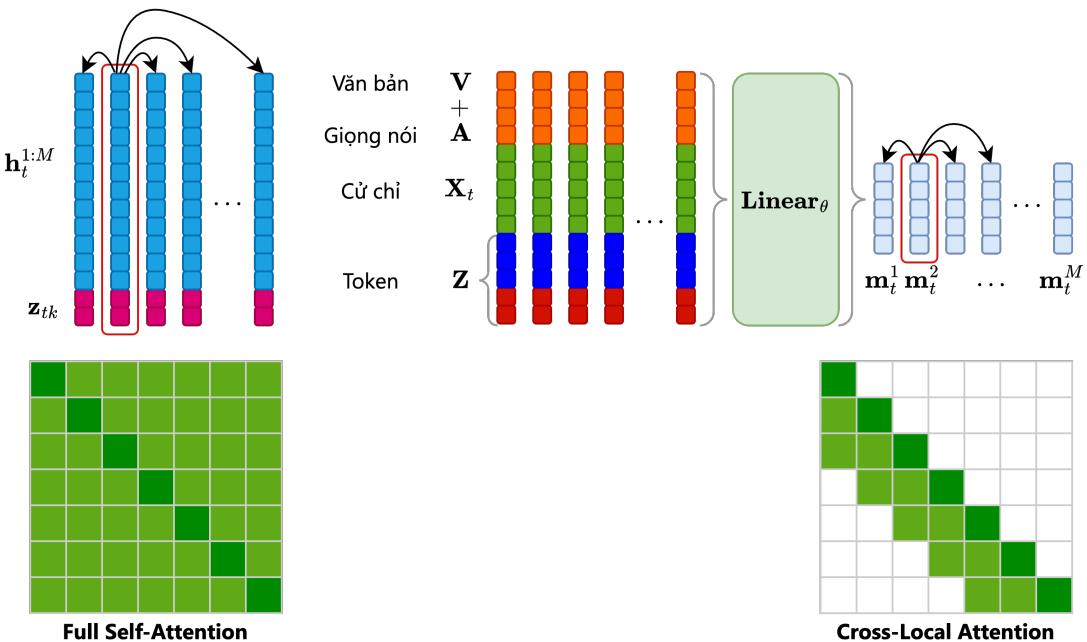
Trong mô hình của luận văn sử dụng cơ chế attention [45] để kết hợp các đặc trưng, mục tiêu áp dụng cơ chế attention là tìm được sự tương quan của từng khung hình với nhau. Cơ chế attention được áp dụng trong Cross-Local Attention và Self-Attention trong lớp Transformer Encoder.

Cơ chế attention có công thức như sau:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}, \text{Mask}) = \text{softmax} \left(\frac{\mathbf{Q}\mathbf{K}^T + \text{Mask}}{\sqrt{C}} \right) \mathbf{V} \quad (3.11)$$

Công thức Attention trên có \mathbf{Q} (Query), \mathbf{K} (Key), \mathbf{V} (Value) là các ma trận sau khi đi qua các ma trận biến đổi tuyến tính $\mathbf{Q} = X\mathbf{W}_Q$, $\mathbf{K} = X\mathbf{W}_K$, $\mathbf{V} = X\mathbf{W}_V$. Với đầu

vào là ma trận biểu thị chuỗi M khung hình, với mỗi khung hình là một vector được concat từ các vector đặc trưng bao gồm cả cử chỉ khởi tạo, văn bản, giọng nói, cảm xúc và cử chỉ x_t mà luận văn muốn thực hiện khử nhiễu. \sqrt{C} là hằng số (constant) chuẩn hóa kích thước ma trận. Quá trình Local-Cross Attention được điều khiển chỉ để tính các đặc trưng cục bộ của chuyển động của các cử chỉ và đặc trưng trong các khung hình lân cận.



Hình 3.10: Cơ chế Attention trong Transformer Encoder và Cross-Local Attention

Hàm Attention hoạt động như một bộ từ điển, với các thông tin cuối cùng tra được là ma trận V (value), còn Q (query) là từ khóa muốn tìm kiếm, K (key) là danh mục các từ khóa trong bộ từ điển tra cứu. Quá trình Attention sẽ tính toán mức độ tương đồng giữa Q và K để xác định trọng số cho các giá trị trong V .

Kết quả cuối cùng là tổ hợp các giá trị trong V , trong đó các giá trị tương ứng với các khóa giống truy vấn nhất sẽ có trọng số cao hơn. M là mặt nạ (mask) để thực hiện quá trình chú ý cục bộ. Cross-Local Attention sẽ được minh họa như [Hình 3.10](#) bên phải. Còn lớp Transformer Encoder sẽ sử dụng Self-Attention lớp bên trái.

3.2.4.3 Kết hợp đặc trưng cục bộ với Cross-Local Attention

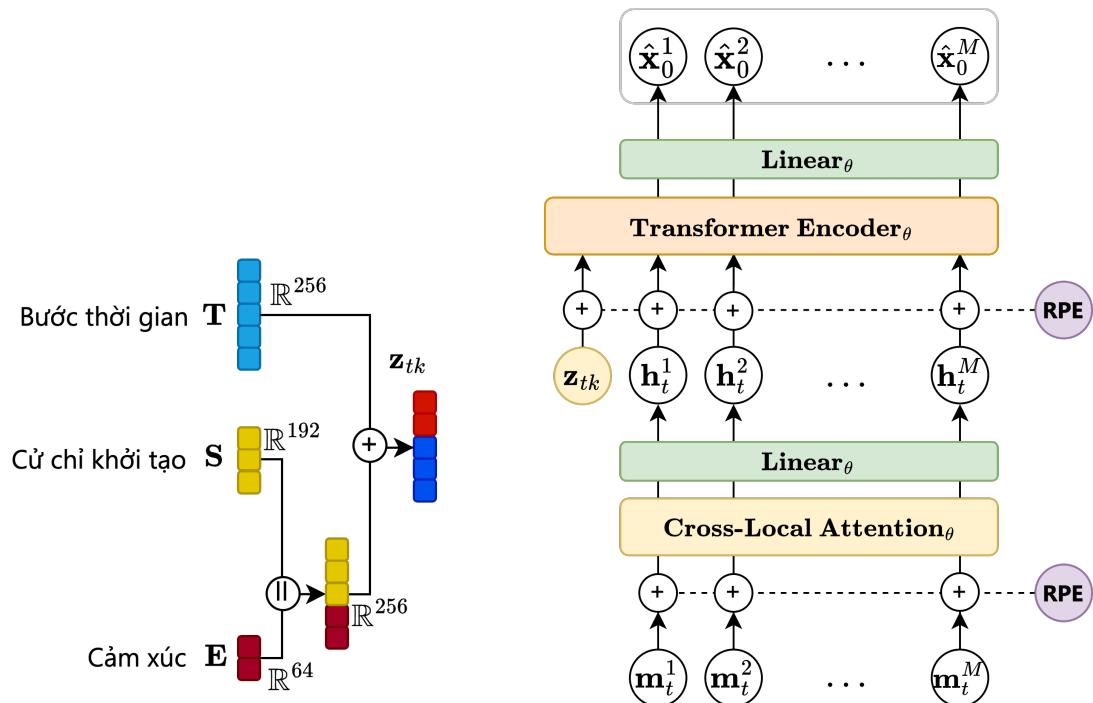
Ma trận $m_t \in \mathbb{R}^{1:M \times 256}$ sẽ tiếp tục đi qua lớp Cross-Local Attention để tính tương quan giữa các đặc trưng cục bộ.

$$\mathbf{h}_t = \text{Linear}_{\theta}(\text{Cross-Local Attention}(\mathbf{m}_t)) \quad (3.12)$$

Cross-local Attention sẽ thực hiện với $\mathbf{Q} = \mathbf{K} = \mathbf{V} = \mathbf{m}_t$. Tương tự ý tưởng từ phương pháp Routing Transformer [39], Cross-local Attention cho thấy rằng sự quan trọng trong việc xây dựng biểu diễn các vector đặc trưng trung gian trước khi đi qua lớp Transformer Encoder như [Hình 3.9](#). Các vector đặc trưng sẽ được cộng thêm một vector mã hóa vị trí tương đối RPE (Relative Position Encoding) để thể hiện được đặc trưng theo thứ tự thời gian trước khi đi qua lớp Cross-Local Attention.

Sau quá trình Cross-Local Attention, mô hình tiếp tục được đưa qua lớp tuyến tính theo [Công thức 3.12](#) để căn chỉnh tương ứng với M khung hình để được ma trận $\mathbf{h}_t \in \mathbb{R}^{1:M \times D}$.

3.2.4.4 Kết hợp đặc trưng toàn cục với Transformer Encoder



Kế thừa từ MDM [44], vector \mathbf{z}_{tk} là token đầu tiên thể hiện thông tin cho toàn bộ chuỗi khung hình, tương tự như \mathbf{z}_{tk} trong mô hình BERT [13], token CLS đầu tiên thể hiện thông tin của toàn bộ đoạn văn bản. Ở đây, luận văn sử dụng \mathbf{z}_{tk} là $\mathbf{z}_{tk} \in \mathbb{R}^{256}$

(Công thức 3.8) token đầu tiên biểu thị đặc trưng biểu thị cho toàn bộ chuỗi M khung hình.

$$\mathbf{X}_0 = \text{Transformer Encoder}(\text{concat}(\mathbf{z}_{tk} \parallel \mathbf{h}_t^{1:M})) \quad (3.13)$$

Các vector \mathbf{h}_t biểu thị cho chuỗi M khung hình, tương tự như phương pháp Reformer [26], trước khi đi vào lớp Self-Attention trong Transformer Encoder, mô hình sẽ sử dụng lớp mã hóa vị trí tương đối (Relative Position Encoding - RPE) để thay thế mã hóa vị trí tuyệt đối, giúp mô hình hiệu quả hơn trong việc xử lý chuỗi dài. Khi vào lớp Transformer Encoder [45] giúp tính toán được mối liên hệ giữa các chuỗi dữ liệu. Trong lớp Transformer Encoder mô hình sẽ áp dụng cơ chế tự chú ý tương tự như Công thức 3.11 trên nhưng không sử dụng mặt nạ Mask. Giúp tính sự tương quan giữa toàn bộ chuỗi.

3.2.5 Công đoạn giải mã đặc trưng

Trong công đoạn 6. *Giải mã đặc trưng* (Hình 2.2), sau khi tính được sự tương quan giữa các đặc trưng, mục tiêu là tăng kích thước dữ liệu trở về kích thước ban đầu.

Như minh họa ở Hình 3.9, ma trận tiềm ẩn \mathbf{X}_0 , sau khi đi qua lớp Transformer Encoder để tính sự tương quan giữa nhiều loại dữ liệu khác nhau. Kết quả sẽ đi qua lớp biến đổi tuyến tính $\hat{\mathbf{x}}_0 = \text{Linear}_\theta(\mathbf{X}_0)$ để tăng kích thước ma trận tiềm ẩn thành kích thước ban đầu $\hat{\mathbf{x}}_0 \in \mathbb{R}^{1:M \times D}$.

Công đoạn cuối cùng, kết xuất sẽ được trình bày ở Mục 4.4.

3.2.6 Điều khiển cảm xúc trong bài toán sinh cử chỉ

Ở các bước trên mô hình đã có thể học được cách sinh cử chỉ. Tuy nhiên, để mô hình học được các cảm xúc trong các tình huống khác nhau, luận văn tham số hóa và thay đổi lần lượt từng cảm xúc, sao cho kết quả dự đoán phản ánh đúng cảm xúc tương ứng.

Tương tự như các phương pháp sử dụng mô hình khử nhiễu có điều kiện [21], [44], luận văn sử dụng điều kiện $c = [\mathbf{s}, \mathbf{e}, \mathbf{a}, \mathbf{v}]$, bao gồm cử chỉ khởi tạo \mathbf{s} , cảm xúc \mathbf{e} , giọng nói tương ứng \mathbf{a} và văn bản \mathbf{v} . Mô hình diffusion có điều kiện c ở đây sẽ là tổng cả trường hợp ở từng bước t trong mô hình khử nhiễu $\mathbf{G}_\theta(\mathbf{x}_t, t, c)$, với $c_\emptyset = [\emptyset, \emptyset, \mathbf{a}, \mathbf{v}]$ không điều kiện và $c = [\mathbf{s}, \mathbf{e}, \mathbf{a}, \mathbf{v}]$ có điều kiện. Quá trình này có thể dễ dàng điều

khiến bằng một lớp mặt nạ ngẫu nhiên (random mask) trên các vector đặc trưng của cử chỉ khởi tạo s và cảm xúc e . Khi đó, mô hình chỉ việc thay đổi nhãn tương ứng với lớp mask đã lấy ngẫu nhiên để mô hình có thể tối ưu theo các điều kiện khác nhau.

$$\hat{\mathbf{x}}_{0c,c_\emptyset,\gamma} = \gamma \cdot G(\mathbf{x}_t, t, c) + (1 - \gamma) \cdot G(\mathbf{x}_t, t, c_\emptyset) \quad (3.14)$$

Điểm đặc biệt là có thể dựa trên việc học có điều kiện và không có điều kiện của classifier-free guidance [21], có thể nội suy giữa hai cảm xúc e_1 và cảm xúc e_2 khác nhau bằng cách cho điều kiện $c = [s, e_1, a, v]$ và $c_\emptyset = [s, e_2, a, v]$. Khi đó ta có thể viết lại $\hat{x}_{0\gamma,c_1,c_2} = \gamma \cdot G(x_t, t, c_1) + (1 - \gamma) \cdot G(x_t, t, c_2)$.

3.2.7 Quá trình huấn luyện

Thuật toán 3 Huấn luyện trong OHGesture

1. Tính sẵn các giá trị và siêu tham số: $\gamma, \sqrt{\alpha_t}, \sqrt{1 - \alpha_t}, \sqrt{\bar{\alpha}_t}$ và nhiễu ngẫu nhiên ϵ_t tại mỗi bước $t : 1 \rightarrow T$. Định nghĩa lịch nhiễu $\{\alpha_t \in (0, 1)\}_{t=1}^T$.
2. Lấy nhãn ban đầu x_0 từ phân phối dữ liệu đã chuẩn hóa.
3. Tạo ngẫu nhiên các mặt nạ Bernoulli $c_1 = [s, e_1, a, v], c_2 = [s, e_2, a, v]$, hoặc $c_2 = [\emptyset, \emptyset, a, v]$.
4. Thêm nhiễu để có cử chỉ nhiễu \mathbf{x}_t :

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}_t$$

5. Với mỗi bước t , lấy **ngẫu nhiên** t từ $[1, T]$.
6. Với \mathbf{x}_t, t và các điều kiện mặt nạ c_1, c_2 , dự đoán chuỗi cử chỉ:

$$\hat{\mathbf{x}}_{0\gamma,c_1,c_2} = \gamma \cdot G_\theta(\mathbf{x}_t, t, c_1) + (1 - \gamma) \cdot G_\theta(\mathbf{x}_t, t, c_2)$$

7. Tính loss và đạo hàm để cập nhật trọng số θ :

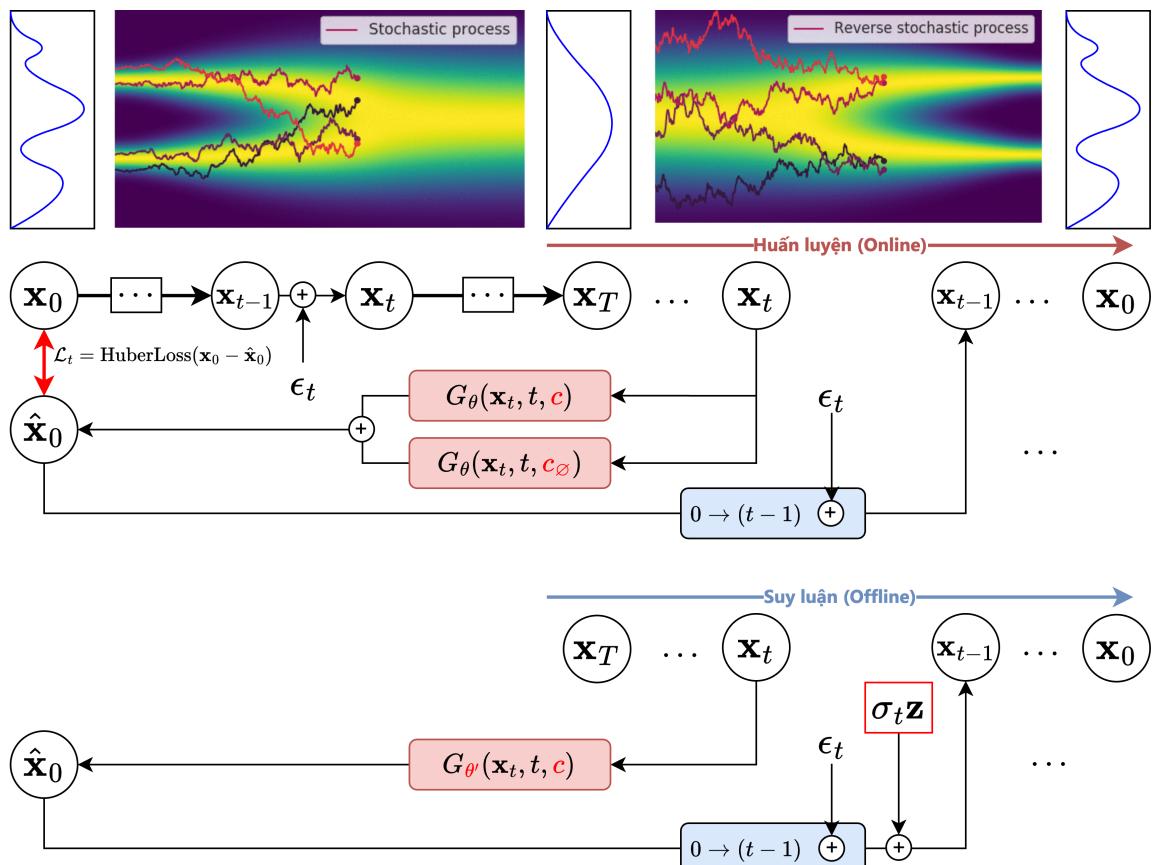
$$\mathcal{L}^t = \mathbb{E}_{t \sim [1, T], \mathbf{x}_0, \boldsymbol{\epsilon}_t} [\text{HuberLoss}(\mathbf{x}_0, \hat{\mathbf{x}}_0)]$$

8. Lặp lại từ bước 6 cho đến khi hội tụ, thu được các tham số tối ưu θ' .

Thuật toán 3 huấn luyện mô hình OHGesture bắt đầu bằng việc tính toán các giá trị và siêu tham số cần thiết như $\gamma, \sqrt{\alpha_t}, \sqrt{1 - \alpha_t}, \sqrt{\bar{\alpha}_t}$ và nhiễu ngẫu nhiên ϵ_t cho từng

bước thời gian t từ 1 đến T . Sau đó, nhãn ban đầu \mathbf{x}_0 , đại diện cho cờ chỉ gốc, được lấy từ phân phối dữ liệu đã chuẩn hóa. Tiếp theo, các mặt nạ Bernoulli c_1 và c_2 được tạo ngẫu nhiên, mô phỏng các điều kiện khác nhau như cờ chỉ, cảm xúc, giọng nói, hoặc văn bản, với một trong các mặt nạ có thể là không có thông tin cảm xúc. Sau khi có các mặt nạ, nhiều được thêm vào để tạo thành cờ chỉ nhiều \mathbf{x}_t , được tính bằng công thức kết hợp chuỗi cờ chỉ gốc và nhiều ngẫu nhiên. Quá trình tiếp theo là chọn ngẫu nhiên một bước thời gian t trong khoảng $[1, T]$ và sử dụng cờ chỉ nhiều \mathbf{x}_t cùng các mặt nạ để dự đoán lại chuỗi cờ chỉ gốc thông qua mô hình, trong đó dự đoán được tính bằng một sự kết hợp của các hàm tạo ra từ các điều kiện mặt nạ. Sau khi có dự đoán, loss được tính bằng cách sử dụng HuberLoss giữa chuỗi cờ chỉ gốc và dự đoán, từ đó đạo hàm loss để cập nhật các tham số trọng số θ . Quá trình huấn luyện này được lặp lại cho đến khi mô hình hội tụ và thu được các tham số tối ưu θ' .

3.2.8 Quá trình lấy mẫu



Hình 3.12: Quá trình học Offline (Training) và Online (Inference)

Để có thể sinh cử chỉ với chiều dài tùy ý, luận văn cắt chuỗi ban đầu thành các đoạn ngắn có chiều dài M . Trong quá trình huấn luyện, cử chỉ khởi tạo ban đầu có thể được tạo ra bằng cách lấy ngẫu nhiên một cử chỉ từ tập dữ liệu hoặc tính trung bình các đoạn đã cắt. Cụ thể ở đây, sẽ lấy góc quay trung bình trong các đoạn đã cắt được. Khi đó, chỉ cần lấy lần lượt các khung hình đã sinh ra và chọn $N = 8$ khung hình cuối cùng làm cử chỉ khởi tạo ở lượt tiếp theo. Đối với mỗi đoạn đã cắt ra, cử chỉ \mathbf{x}_t lần lượt sẽ được áp dụng hàm khử nhiễu $\hat{\mathbf{x}}_0 = G_{\theta'}(\mathbf{x}_t, t, c)$, sau khi có $\hat{\mathbf{x}}_0$ sẽ được thêm nhiễu (diffuse) cho đến khi được \mathbf{x}_{t-1} , và \mathbf{x}_{t-1} sẽ tiếp tục được thực hiện khử nhiễu cho đến bước $t = 1$ để được \mathbf{x}_0 .

Thuật toán 4 Lấy mẫu (sampling) trong OHGesture

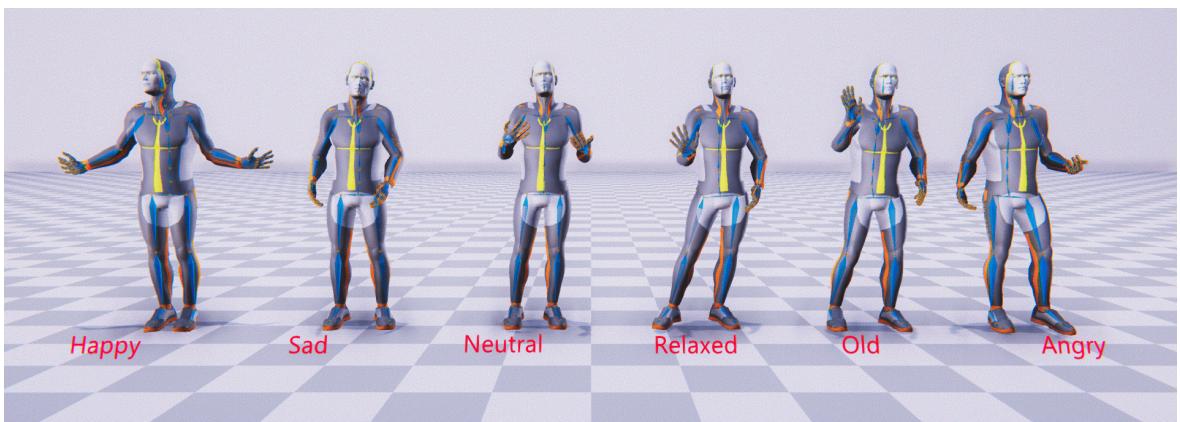
1. Khởi tạo với nhiễu: $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2. Các giá trị $\sqrt{\alpha_t}$, $\sqrt{1 - \alpha_t}$ và $\sqrt{\bar{\alpha}_t}$ lấy từ quá trình huấn luyện, tính sẵn giá trị σ_t từ α_t ở mỗi bước $t : 1 \rightarrow T$.
3. Chia mỗi đoạn giọng nói 4 giây thành: $\mathbf{a} \in \mathbb{R}^{64000}$. Cử chỉ khởi tạo s ban đầu là trung bình dữ liệu, sau đó được lấy từ đoạn cử chỉ đã suy luận. Chọn cảm xúc mong muốn, văn bản được phiên âm từ giọng nói a, tạo cặp điều kiện $c = [\mathbf{s}, \mathbf{e}, \mathbf{a}, \mathbf{v}]$.
4. Với mỗi t , lấy t **tuần tự** từ $[T, \dots, 1]$.
5. Tạo nhiễu ngẫu nhiên $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$.
6. Đưa \mathbf{x}_t vào để suy luận $\hat{\mathbf{x}}_0^{(t)} = G_{\theta'}(\mathbf{x}_t, t, c)$.
7. Thực hiện thêm nhiễu $\hat{\mathbf{x}}_0^{(t)}$ từ bước $0 \rightarrow t$ để nhận $\hat{\mathbf{x}}_{t-1}^{(t)}$.
8. Cộng thêm nhiễu: $\hat{\mathbf{x}}_{t-1} = \hat{\mathbf{x}}_{t-1}^{(t)} + \sigma_t \mathbf{z}$.
9. Quay lại bước 4. Khi $t = 1$, thu được $\hat{\mathbf{x}}_0$ từ quá trình khử nhiễu.

Thuật toán 4 lấy mẫu bắt đầu bằng việc khởi tạo cử chỉ nhiễu ban đầu \mathbf{x}_T từ phân phối chuẩn $\mathcal{N}(0, \mathbf{I})$. Sau đó, các giá trị $\sqrt{\alpha_t}$, $\sqrt{1 - \alpha_t}$ và $\sqrt{\bar{\alpha}_t}$ được lấy từ quá trình huấn luyện, cùng với giá trị σ_t được tính từ α_t tại mỗi bước thời gian t từ 1 đến T . Mỗi đoạn giọng nói 4 giây được chia thành các chuỗi dữ liệu a, và cử chỉ khởi tạo s được lấy từ trung bình dữ liệu hoặc từ đoạn cử chỉ đã suy luận. Cảm xúc mong muốn và văn bản được phiên âm từ giọng nói a, tạo thành một cặp điều kiện $c = [\mathbf{s}, \mathbf{e}, \mathbf{a}, \mathbf{v}]$. Thuật toán sau đó thực hiện các bước tuần tự, bắt đầu từ bước cuối cùng T và tiến ngược về 1. Ở mỗi bước, nhiễu ngẫu nhiên \mathbf{z} được tạo ra và mô hình dự đoán $\hat{\mathbf{x}}_0^{(t)}$ từ

củ chỉ nhiễu \mathbf{x}_t , thời gian t , và cặp điều kiện c . Dự đoán này được chuyển tiếp để tính toán $\hat{\mathbf{x}}_{t-1}^{(t)}$ cho bước tiếp theo, và nhiễu được cộng vào để cập nhật củ chỉ tại bước $t - 1$. Quá trình này lặp lại cho đến khi đạt đến bước 1, khi đó thuật toán thu được kết quả củ chỉ khử nhiễu $\hat{\mathbf{x}}_0$ như là kết quả dự đoán cuối cùng.

Chương 4. THỰC NGHIỆM

4.1 Tập dữ liệu



Hình 4.1: Minh họa về 6 cử chỉ Happy, Sad, Neutral, Old, Relaxed và Angry

Luận văn sử dụng tập dữ liệu ZeroEGGS [16] là một bộ dữ liệu motion capture được xây dựng để nghiên cứu và phát triển các mô hình tạo cử chỉ. Nó bao gồm 67 đoạn độc thoại do diễn viên motion capture nữ thực hiện, với tổng thời gian là 135 phút. Các đoạn hội thoại trong tập dữ liệu được biểu diễn với 6 cảm xúc khác nhau: Happy, Sad, Neutral, Old, Relaxed và Angry, giúp mô phỏng nhiều trạng thái cảm xúc khác nhau trong cử chỉ và chuyển động cơ thể. ZeroEGGS cung cấp một nền tảng phong phú để nghiên cứu khả năng kết hợp giữa bài nói và cử chỉ động, phục vụ cho việc tạo ra các mô hình có thể điều chỉnh cử chỉ tương ứng với cảm xúc và ngữ nghĩa của văn bản.

4.2 Công đoạn tiền xử lý dữ liệu

Trong công đoạn 1. Tiền xử lý dữ liệu ([Hình 2.2](#)), các dữ liệu về cử chỉ, giọng nói, văn bản được đọc và xử lý để biểu diễn thành các vector hoặc ma trận thể hiện thông

tin về dữ liệu thô.

Đối với dữ liệu văn bản: Luận văn sử dụng thư viện nltk tách các từ, sử dụng contractions để chuyển các từ viết tắt về dạng chuẩn hóa.

Một trong những đóng góp của luận văn là từ dữ liệu giọng nói có sẵn từ tập ZeroEGGS, luận văn đã sử dụng Adobe Speech To Text để chuyển giọng nói thành văn bản, dùng Montreal Forced Aligner [40] của từ điển âm vị Tiếng Anh để căn chỉnh được thời gian tương ứng với số khung hình của cử chỉ để thu được các TextGrid. Trong TextGrid sẽ có thời gian tương ứng với từng từ, dựa trên thông tin này, luận văn sử dụng gensim để nhúng vector word2vec.

Dữ liệu cử chỉ bao gồm các tệp BVH (BioVision Motion Capture) được thu nhận từ các cảm biến bằng các hệ thống Motion Capture. Trong các tệp BVH, có chứa hai thành phần gồm Hierarchy và Motion. Trong đó:

- **HIERARCHY:** là một cây định nghĩa thông tin khung xương bao gồm 75 xương $\{b_1, b_2 \dots b_{75}\}$, mỗi bone bao gồm thông tin về vị trí ban đầu (OFFSET), và các loại tham số CHANNELS thể hiện thông tin về loại, thứ tự của góc quay (Zrotation, Yrotation Xrotation) và vị trí (Xposition, Yposition, Zposition) sẽ định nghĩa trong phần MOTION. Bone (thường là Hips) đầu tiên sẽ là bone gốc b_{root} , là vị trí sau đó sẽ được forward kinematic (động học thuận) để được T-Shape là ví trí ban đầu của toàn bộ khung xương trước khi áp dụng chuyển động.
- **MOTION:** là chuỗi các khung hình. Mỗi khung hình là dữ liệu chuyển động thể hiện sự thay đổi của toàn bộ 75 khung xương đã định nghĩa trong CHANNELS của HIERARCHY.

Mô hình của luận văn chuyển dữ liệu từ góc quay Euler sang góc quay Quaternion, với góc quay Quaternion là một véc-tơ gồm 4 phần tử.

$$\mathbf{g} = \left[\mathbf{p}_{root}, \mathbf{r}_{root}, \mathbf{p}'_{root}, \mathbf{r}'_{root}, \mathbf{p}_{joins}, \mathbf{r}_{joins}, \mathbf{p}'_{joins}, \mathbf{r}'_{joins}, \mathbf{d}_{gaze} \right] \quad (4.1)$$

Trong đó với mỗi $\mathbf{g} \in \mathbb{R}^{1141}$ bao gồm:

- $\mathbf{p}_{root} \in \mathbb{R}^3$: tọa độ của điểm gốc
- $\mathbf{r}_{root} \in \mathbb{R}^4$: Góc quay của điểm gốc
- $\mathbf{p}'_{root} \in \mathbb{R}^3$: Vận tốc thay đổi của tọa độ gốc

- $\mathbf{r}'_{\text{root}} \in \mathbb{R}^3$: Vận tốc thay đổi của góc quay gốc
- $\mathbf{p}_{\text{joins}} \in \mathbb{R}^{3n_{\text{join}}}$: tọa độ của các khung xương
- $\mathbf{r}_{\text{joins}} \in \mathbb{R}^{6n_{\text{join}}}$: Góc quay của các khung xương theo mặt phẳng X và Y
- $\mathbf{p}'_{\text{joins}} \in \mathbb{R}^{3n_{\text{join}}}$: Vận tốc thay đổi của tọa độ các khung xương
- $\mathbf{r}'_{\text{joins}} \in \mathbb{R}^{3n_{\text{join}}}$: Vận tốc thay đổi của góc quay các khung xương
- $\mathbf{d}_{\text{gaze}} \in \mathbb{R}^3$: Là hướng nhìn

Chuỗi cử chỉ ban đầu có dữ liệu là góc quay sẽ được chuyển thành các góc quay radian, từ góc quay radian ở dạng Euler sẽ được chuyển sang góc quay Quaternion, với quá trình cụ thể được trình bày trong [Mục B.3](#).

Dữ liệu giọng nói: $\mathbf{a}_{\text{raw}} \in \mathbb{R}^{\text{length}}$ là chuỗi giọng nói thô được đọc ở sample rate 16000, sau đó được cắt thành $\mathbf{a} \in \mathbb{R}^{64000}$ tương ứng với 4 giây. Luận văn sử dụng thư viện `ffmpeg-normalize` để chuẩn hóa tiếng nói nhỏ hơn so với giọng nói gốc.

Cảm xúc: Dữ liệu cảm xúc sẽ được biểu diễn bằng một vector one-hot encoding sẵn. Trong quá trình lấy mẫu, tên tệp sẽ định nghĩa luôn thông tin về cảm xúc mong muốn.

Toàn bộ dữ liệu sẽ được lưu bằng `h5` được dùng để lưu trữ dữ liệu.

4.3 Quá trình huấn luyện

Toàn bộ quá trình huấn luyện mô hình được thực hiện trong gần 2 tuần với các tham số sau: số bước huấn luyện $T = 1000$, sử dụng GPU Nvidia 3090. Learning rate được thiết lập là 3×10^{-5} , với kích thước lô huấn luyện (batch size) là 640 và quá trình huấn luyện thực hiện tổng cộng 43,853 mẫu, tức ta sẽ lấy ngẫu nhiên t bước vào hàm f_θ để thực hiện dự đoán \mathbf{x}_0 . Tham số điều khiển cảm xúc là $\gamma = 0.1$. Tỷ lệ sử dụng mặt nạ ngẫu nhiên (random mask) cho cảm xúc và cử chỉ khởi tạo là 10%, quá trình sử dụng mặt nạ ngẫu nhiên với hàm Bernoulli nhằm ẩn hiện ma trận thông tin của cảm xúc và cử chỉ khởi tạo.

Giá trị của β bắt đầu từ 0.5 → 0.999.

Hàm HuberLoss($\mathbf{x}_0, \hat{\mathbf{x}}_0$) được tính như sau:

- Nếu $|\mathbf{x}_0 - \hat{\mathbf{x}}_0| \leq \delta$ thì $\mathcal{L}_{\delta, \mathbf{x}_0, \hat{\mathbf{x}}_0} = \frac{1}{2}(\mathbf{x}_0 - \hat{\mathbf{x}}_0)^2$: Nhỏ hơn ngưỡng thì bình phương khoảng cách. Lúc này HuberLoss hoạt động giống như MSE. MSE nhạy cảm hơn với các lỗi nhỏ giúp đạo hàm (gradient) mượt mà hơn.
- Nếu $|\mathbf{x}_0 - \hat{\mathbf{x}}_0| > \delta$ thì $\mathcal{L}_{\delta, \mathbf{x}_0, \hat{\mathbf{x}}_0} = \delta \cdot |\mathbf{x}_0 - \hat{\mathbf{x}}_0| - \frac{1}{2}\delta^2$: Lớn hơn ngưỡng thì được tính bằng khoảng cách tuyệt đối MAE. MAE nhằm giảm độ nhạy với lỗi lớn, không phạt nặng lỗi lớn để giúp mô hình ổn định hơn đối với trường hợp ngoại lai.

Quá trình huấn luyện được triển khai trên mã nguồn công khai tại: [Github/OHGesture](#)¹.

4.4 Quá trình sử dụng Unity để kết xuất

Để trực quan hóa quá trình sinh cử chỉ từ dữ liệu đầu ra của mô hình, trong công đoạn 7. *Kết xuất* ([Hình 2.2](#)) luận văn sử dụng Unity, kế thừa mã nguồn từ mô hình DeepPhase [43]. Dữ liệu sau khi sinh là tệp BVH (BioVision Motion Capture). Trong Unity, luận văn bổ sung mã nguồn C-Sharp để kết xuất theo vị trí tọa độ và nhãn tương ứng, với vị trí và góc quay của các xương được biểu diễn dưới dạng quaternion.

Chi tiết phần render cử chỉ được sinh ra, tôi trình bày ở [Phụ lục C](#).

Mã nguồn chương trình Unity được luận văn công khai ở [Github/DeepGesture-Unity](#)².

¹<https://github.com/hmthanh/OHGesture>

²<https://github.com/DeepGesture/deepgesture-unity>

Chương 5. KẾT QUẢ VÀ ĐÁNH GIÁ

5.1 Phương pháp đánh giá

Quá trình đánh giá được thực hiện qua hai độ đo chính là: Mean Opinion Scores (MOS) và Fréchet Inception Distance (FID).

5.1.1 Đánh giá dựa trên cảm nhận của con người

5.1.1.1 Mean Opinion Scores (MOS)

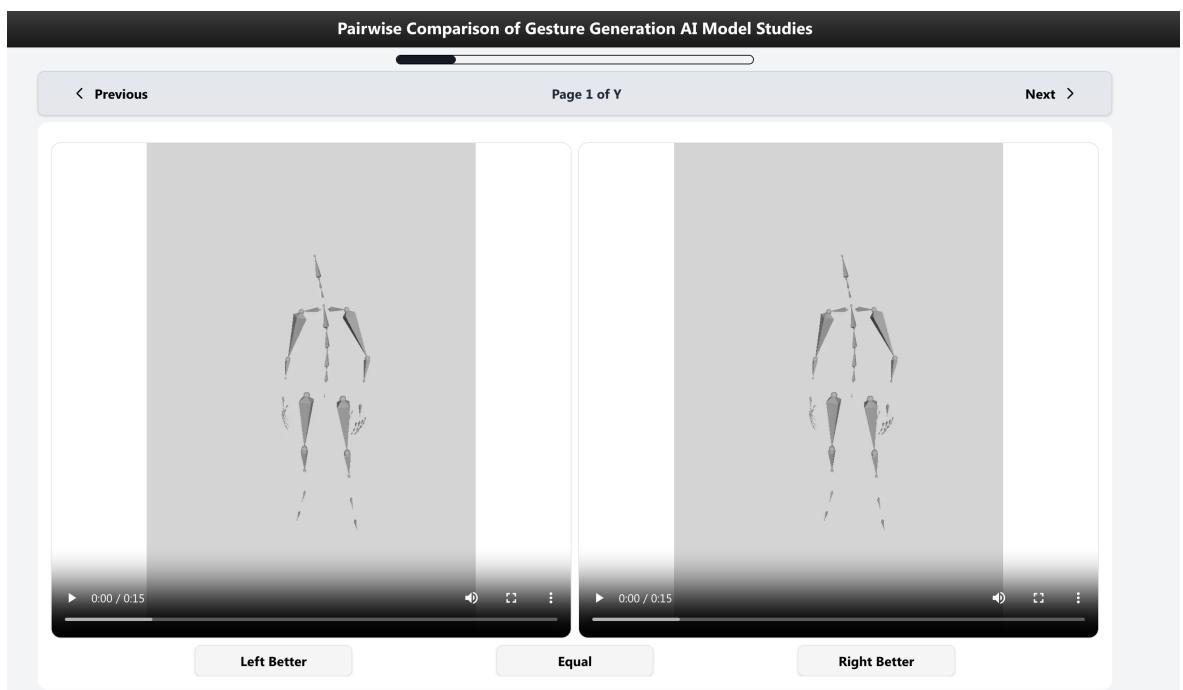
Hiện tại, chưa có một độ đo chung cho bài toán sinh cử chỉ, đặc biệt là sinh cử chỉ từ giọng nói, vì vậy luận văn dựa vào đánh giá chủ quan của con người để thực hiện các đánh giá thực nghiệm. Tương tự như các phương pháp trước đây [56], [29], các mô hình sinh cử chỉ điều khiển bằng giọng nói vẫn thiêu các chỉ số mục tiêu phản ánh một cách nhất quán với nhận thức chủ quan của con người [2].

MOS được đo lường thông qua ba tiêu chí:

- Human-likeness (Mức độ giống con người)
- Gesture-Speech Appropriateness (Sự phù hợp giữa cử chỉ và giọng nói)
- Gesture-style Appropriateness (Sự phù hợp giữa phong cách cử chỉ)

Một trong những đóng góp của luận văn là xây dựng **GENEA Leaderboard**¹ [35]. Hệ thống này bao gồm HEMVIP (**H**uman **E**valuation of **M**ultiple **V**ideos in **P**arallel) nhằm so sánh kết quả sinh trực quan giữa các video được tạo ra từ kết quả xuất.

¹<https://genea-workshop.github.io/leaderboard>



Hình 5.1: Hệ thống HEMVIP nhằm đánh giá kết quả sau khi kết xuất của hai mô hình

Trong nhóm GENEAE (Generation and Evaluation of Non-verbal Behaviour for Embodied Agents), chúng tôi sẽ thuê các người đánh giá trên Prolific và thực hiện nghiên cứu người dùng (User Study) theo kết quả kết xuất trên video, người tham gia sẽ đánh giá *Bên Trái Tốt Hơn* (Left Better), *Bằng Nhau* hoặc *Bên Phải Tốt Hơn* (Right Better). Điểm số sẽ cập nhật sẽ là $-1, 0$ và 1 cho mô hình cho các kết quả sinh bao gồm cả dữ liệu thật. Kết quả so sánh của toàn bộ mô hình sẽ được cập nhật bằng hệ thống đánh giá Elo (Elo rating system).

Mã nguồn của chương trình được công khai ở [github.com/hemvip.github.io](https://github.com/hemvip/github.io)².

5.1.2 Đánh giá dựa trên các độ đo số học

5.1.2.1 Sai số toàn phương trung bình (Mean Square Error)

Độ đo sai số toàn phương trung bình giữa chuỗi cử chỉ dự đoán $\hat{\mathbf{y}}_i^{1:M \times D}$ và chuỗi cử chỉ thật $\mathbf{y}_i^{1:M \times D}$ được thực hiện theo công thức sau:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n \left\| \mathbf{y}_i^{1:M \times D} - \hat{\mathbf{y}}_i^{1:M \times D} \right\|^2 \quad (5.1)$$

Trong đó:

²HEMVIP 2 <https://github.com/hemvip/hemvip.github.io>

- n là số lượng mẫu dữ liệu.
- $\mathbf{y}_i^{1:M \times D}$ là giá trị thực (ground truth) của mẫu thứ i , với M là số khung (frames) và D là số chiều dữ liệu.
- $\hat{\mathbf{y}}_i^{1:M \times D}$ là giá trị dự đoán của mẫu thứ i , có cùng kích thước $M \times D$.
- $\|\mathbf{y}_i^{1:M \times D} - \hat{\mathbf{y}}_i^{1:M \times D}\|^2$ là bình phương chuẩn của sự chênh lệch giữa ma trận thực và ma trận dự đoán.

MSE đo lường độ chênh lệch trung bình bình phương giữa các chuỗi cử chỉ thực và chuỗi cử chỉ dự đoán, càng nhỏ thì mô hình dự đoán càng chính xác. Kết quả đánh giá được cập nhật ở [Mục nhỏ 5.2.2.1](#).

5.1.2.2 Fréchet Gesture Distance (FGD)

Tương tự các phương pháp sinh ảnh sử dụng độ đo FID (Fréchet Inception Distance) nhằm đo khoảng cách phân phối của dữ liệu thật và dữ liệu dự đoán. Khoảng cách Fréchet trong cử chỉ hay FGD (Fréchet Gesture Distance) đo sự tương đồng về phân phối giữa chuỗi cử chỉ sinh ra $\hat{\mathbf{y}}_i^{1:M \times D}$ và cử chỉ thực tê $\mathbf{y}_i^{1:M \times D}$:

$$\text{FGD} = \|\hat{\mu} - \mu\|^2 + \text{Tr} \left(\Sigma + \hat{\Sigma} - 2\sqrt{\Sigma \hat{\Sigma}} \right) \quad (5.2)$$

Trong đó trong đó n là số lượng mẫu dữ liệu, các tham số:

- $\mu = \frac{1}{n} \sum_{i=1}^n \mathbf{y}_i^{1:M \times D}$ và $\hat{\mu} = \frac{1}{n} \sum_{i=1}^n \hat{\mathbf{y}}_i^{1:M \times D}$ lần lượt là vector trung bình của các đặc trưng (features) từ tập dữ liệu thực $\mathbf{y}_i^{1:M \times D}$ và tập dữ liệu sinh $\hat{\mathbf{y}}_i^{1:M \times D}$.
- $\Sigma = \frac{1}{n-1} \sum_{i=1}^n (\mathbf{y}_i^{1:M \times D} - \mu)(\mathbf{y}_i^{1:M \times D} - \mu)^T$ và
 $\hat{\Sigma} = \frac{1}{n-1} \sum_{i=1}^n (\hat{\mathbf{y}}_i^{1:M \times D} - \hat{\mu})(\hat{\mathbf{y}}_i^{1:M \times D} - \hat{\mu})^T$: là ma trận hiệp phương sai (covariance matrix) của các đặc trưng từ tập dữ liệu thực và sinh.
- $\text{Tr}(\cdot)$ là toán tử vết (trace) của ma trận, tính tổng các phần tử trên đường chéo chính.
- $\sqrt{\Sigma \hat{\Sigma}}$: là căn bậc hai ma trận (matrix square root) của tích hai ma trận hiệp phương sai.

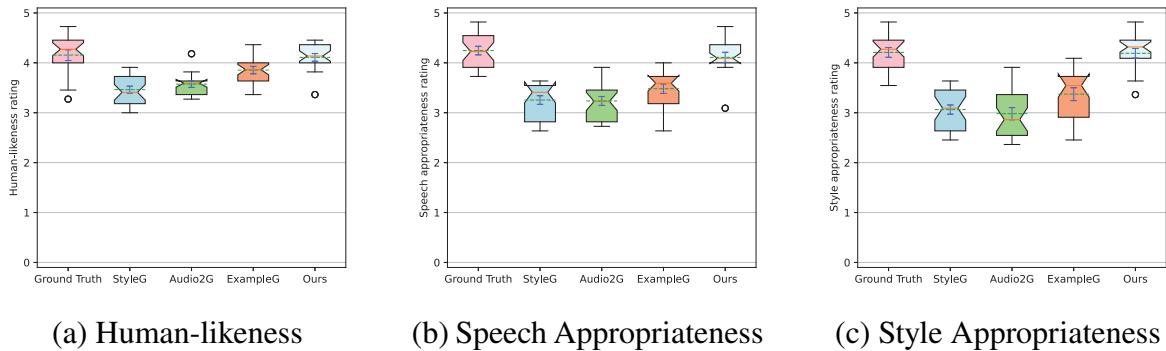
FGD thấp cho thấy phân phối của cursive chỉ sinh ra gần giống với cursive thực tế, trong khi FID cao gợi ý sự khác biệt lớn, cho thấy chất lượng cursive chỉ sinh ra kém hơn. Trong luận văn, quá trình đánh giá được thực hiện trên chuỗi cursive chỉ dự đoán $\hat{x}^0 \in \mathbb{R}^{1:M \times D}$ và chuỗi cursive thật $x_0 \in \mathbb{R}^{1:M \times D}$.

5.2 Kết quả đánh giá

5.2.1 Kết quả đánh giá nghiên cứu người dùng

5.2.1.1 Kết quả đánh giá bằng MOS

Ở đây luận văn sử dụng lại kết quả đánh giá của mô hình baseline **DiffuseStyleGesture** [50] trong độ đo về cảm nhận đánh giá của con người do lĩnh vực sinh cursive vẫn còn rất mới, chi phí để ước lượng các mô hình còn lớn nên luận văn không thể đánh giá được mô hình OHGesture. Vì vậy các kết quả này chưa bao gồm thông tin kết quả về mô hình OHGesture mà luận văn đề xuất.



Để hiểu hiệu suất thị giác thực tế của phương pháp của luận văn, luận văn tiến hành một nghiên cứu người dùng so sánh các cursive chỉ được tạo ra từ phương pháp của luận văn và dữ liệu chụp chuyển động thực tế. Độ dài của các đoạn clip đánh giá dao động từ 11 đến 51 giây, với độ dài trung bình là 31.6 giây, dài hơn so với các đoạn trong đánh giá GENEVA [56] (8-10 giây), vì thời gian dài hơn có thể tạo ra kết quả rõ ràng và thuyết phục hơn [51]. Người tham gia đánh giá trên thang điểm từ 5 đến 1, với các nhãn từ excellent, good, fair, poor, đến bad.

Name	Human ↑ likeness	Gesture-speech ↑ appropriateness
Ground Truth	4.15 ± 0.11	4.25 ± 0.09
Ours	4.11 ± 0.08	4.11 ± 0.10
– WavLM	4.05 ± 0.10	3.91 ± 0.11
– Cross-local attention	3.76 ± 0.09	3.51 ± 0.15
– Self-attention	3.55 ± 0.13	3.08 ± 0.10
– Attention + GRU	3.10 ± 0.11	2.98 ± 0.14
+ Forward attention	3.75 ± 0.15	3.23 ± 0.24

Bảng 5.1: Kết quả đánh giá bằng MOS

5.2.2 Kết quả đánh giá theo độ đo số học

5.2.2.1 Kết quả đánh giá bằng MSE

Trong luận văn, chuỗi cử chỉ dự đoán sẽ được đoạn trên M khung hình. Luận văn sử dụng Mean Square Error trên chuỗi cử chỉ $\mathbf{x}^{1:M \times D}$.

Cảm xúc	Tự Nhiên	Buồn Bã	Vui Vẻ	Thư Giãn	Lớn Tuổi	Giận Dữ
DiffuseStyleGesture	75.04	51.40	110.18	130.83	116.03	78.53
ZeroEGG	136.33	81.22	290.47	140.24	102.44	181.07
Mô hình đề xuất						
OHGesture	161.22	89.58	279.95	156.93	99.86	215.24

Bảng 5.2: Kết quả đánh giá Mean Square Error theo 6 cảm xúc

5.2.2.2 Kết quả đánh giá bằng FGD

Luận văn đề xuất Fréchet Gesture Distance (FGD), độ đo FID dựa trên cử chỉ (gesture), và xây dựng mã nguồn [GestureScore](#)³. Trong GestureScore, luận văn xây dựng một Inception V3 model, để mã hóa chuỗi khung hình $\mathbf{x}^{1:M \times D}$ thành vector tiềm ẩn kích thước 32×32 . Sử dụng vector này làm đầu vào cho [Công thức 5.2](#). Sau đây là [Bảng 5.3](#) đánh giá kết quả của mô hình OHGesture bằng GestureScore

³Github/GestureScore: <https://github.com/GestureScore/GestureScore>

Name	FGD trên vector đặc trưng	FGD dữ liệu gốc
Ground Truth	-	-
Ours		
OHGesture (Feature D=1141)	2.058	9465.546
OHGesture (Rotations)	3.513	9519.129

Bảng 5.3: Kết quả đánh giá Fréchet Gesture Distance (FGD) trên $\mathbf{x}^{1:M \times D}$ (từ khung hình 1 đến khung hình M, mỗi khung hình có D đặc trưng từng khung hình)

- **Vector đặc trưng** Sử dụng tệp BVH để chuyển toàn bộ skeleton của mỗi khung hình thành vector đặc trưng $D = 1141$ như công thức [Công thức 4.1](#)
- **Rotations:** Từ tệp BVH kết quả, luận văn trích xuất sự thay đổi của các góc quay (rotations), $D = 225$ ($225 = 75 \times 3$) của chuỗi cursive với chiều dài mỗi đoạn là M khung hình để đánh giá ở dòng OHGesture bên dưới.

5.3 Xây dựng và tiêu chuẩn hóa hệ thống đánh giá kết quả sinh cử chỉ

Hiện nay các mô hình sinh (gesture generation) cử chỉ được quan tâm nghiên cứu với rất nhiều mô hình khác nhau, tuy nhiên do không có độ đo chung, vì các độ đo truyền thống như FID (Fréchet Inception Distance) hay IS (Inception Score),.. không thể hiện được hết các tính chất Giống người (Human-likeness), tính phù hợp với giọng nói (Speech Appropriateness), và tính phù hợp với phong cách (Style Appropriateness) của cử chỉ. Các mô hình cũng được thực hiện và huấn luyện trên một tập dữ liệu khác nhau, vì vậy rất khó để biết được mô hình nào đã đạt kết quả tốt hơn, và mô hình nào là mô hình state-of-the-art, từ đó khó có thể đạt được sự tiến bộ trong lĩnh vực sinh cử chỉ. Việc thiếu tiêu chuẩn chung để đánh giá trong cộng đồng nghiên cứu khiến luận văn mong muốn xây dựng một hệ thống xếp hạng trực tuyến [35] [GENEA Leaderboard](#)⁴, là một bảng xếp hạng các mô hình sinh cử chỉ. Luận văn thu thập và xử lý một tập dữ liệu cử chỉ từ nhiều ngôn ngữ và từ nhiều tập dữ liệu khác nhau và tiêu chuẩn hóa để tạo ra một tập dữ liệu duy nhất. Sau đó, luận văn sẽ mời các tác giả của các mô hình để học và dự đoán trên tập dữ liệu đã được tiêu chuẩn hóa, sau

⁴GENEA Leaderboard: <https://genea-workshop.github.io/leaderboard/>

khi có kết quả sinh cử chỉ, luận văn sẽ thuê những người tham gia nghiên cứu trên Prolific để đánh giá và xếp hạng kết quả sinh cử chỉ của các mô hình. Hiện tại luận văn đang xây dựng hệ thống trực tuyến hemvip/hemvip.github.io⁵ với mục tiêu đánh giá kết quả sinh của cử chỉ thông qua việc thuê các người tham gia đánh giá kết quả thông qua Prolific. Luận văn sẽ bổ sung các đánh giá về mô hình OHGesture dựa trên các điểm số trên.

Thông qua hệ thống đánh giá này, nghiên cứu kỳ vọng sẽ thiết lập một tiêu chuẩn chung, từ đó tạo động lực cho sự tiến bộ trong lĩnh vực sinh cử chỉ.

⁵HEMVIP2 <https://github.com/hemvip/hemvip.github.io>

Chương 6. KẾT LUẬN

6.1 Kết quả đạt được

Luận văn đã phát triển và thử nghiệm thành công mô hình sinh cử chỉ OHGesture, một hệ thống có khả năng tạo ra cử chỉ rất tự nhiên, mang lại cảm giác giống người. Điểm nổi bật của mô hình là khả năng đồng bộ chính xác giữa cử chỉ và cảm xúc, nội dung của giọng nói đầu vào, đồng thời có khả năng suy luận vượt ra khỏi các dữ liệu được huấn luyện ban đầu. Điều này có nghĩa là mô hình dựa trên Diffusion không chỉ phụ thuộc vào các mẫu dữ liệu cử chỉ đã được học mà còn có thể có tính khái quát hóa cao, và có thể sinh cử chỉ cho các giọng nói và ngữ cảnh có xác suất dữ liệu thấp.

Một điểm đáng chú ý khác trong nghiên cứu này là việc mở rộng dữ liệu đầu vào. Luận văn không chỉ giới hạn ở cử chỉ, giọng nói và nhãn cảm xúc mà còn tích hợp thêm các công cụ chuyển đổi văn bản thành giọng nói để chuyển giọng nói thành văn bản. Nhờ có thêm một đặc trưng về văn bản, mô hình có thể bổ sung thêm đặc trưng về ngữ nghĩa của cử chỉ và giúp mô hình có thêm nguồn thông tin để đạt kết quả sinh tốt hơn, đồng thời giúp hệ thống hiểu rõ hơn ngữ cảnh và tạo ra cử chỉ phù hợp với từng nội dung cụ thể.

6.2 Ưu và nhược điểm của mô hình

Mô hình sinh cử chỉ OHGesture có nhiều ưu điểm đáng kể, góp phần quan trọng trong việc phát triển các hệ thống tương tác người-máy tự nhiên và linh hoạt hơn. Tuy nhiên, vẫn còn một số nhược điểm để cải thiện hiệu quả trong tương lai.

Ưu điểm:

- Độ chân thực cao: Dựa vào kết quả sinh cử chỉ, có thể thấy mô hình OHGesture đã đạt được kết quả sinh cử chỉ có độ giống người cao. Các cử chỉ sinh ra phản

ánh được sắc thái và nhịp điệu của giọng nói, giúp hệ thống có khả năng phản hồi đồng bộ với thông tin cảm xúc và nội dung của giọng nói.

- **Khả năng tổng quát tốt:** Nhờ mô hình khử nhiễu có thể phủ được các điểm dữ liệu có xác suất thấp, mô hình có thể suy luận cử chỉ cho các tình huống và trạng thái cảm xúc chưa từng xuất hiện trong tập huấn luyện, cho thấy tiềm năng ứng dụng trong nhiều bối cảnh thực tế.
- **Có khả năng kiểm soát** được nhiều đặc trưng khác nhau: Mô hình diffusion có khả năng điều khiển được các cảm xúc khác nhau, có khả năng nội suy trạng thái giữa các cảm xúc khác nhau.

Nhược điểm:

- Mô hình hiện chưa có khả năng suy luận theo thời gian thực (real time) và cần nhiều bước để ra kết quả cuối cùng.
- Thông tin đặc trưng theo $D = 1141$ được xử lý như một tấm ảnh, không thể hiện đúng đặc trưng chuyển động.
- Phụ thuộc vào dữ liệu đầu vào chất lượng cao: Mô hình yêu cầu dữ liệu giọng nói đầu vào có chất lượng tốt và rõ ràng để đảm bảo độ chính xác của cử chỉ sinh ra. Khi giọng nói đầu vào bị nhiễu hoặc chứa nhiều biến thể cảm xúc khó phân biệt, độ chính xác của cử chỉ sinh ra có thể giảm sút.

6.3 Phương hướng phát triển và nghiên cứu trong tương lai

Trong tương lai, có nhiều hướng phát triển tiềm năng để cải thiện và mở rộng mô hình sinh cử chỉ OHGesture, nhằm đáp ứng tốt hơn các yêu cầu thực tế và nâng cao tính ứng dụng của hệ thống. Một số hướng nghiên cứu và phát triển chính bao gồm:

- **Tối ưu hóa mô hình** để có thể suy luận theo thời gian thực, hiện tại mô hình phải chia chuỗi giọng nói, và kết quả sinh phải được import vào Unity mới có thể render. Trong tương lai, luận văn mong muốn có thể xây dựng các hệ thống với thời gian thực, để có thể tương tác và tăng tính ứng dụng của mô hình

- Tối ưu hóa quá trình sampling và giảm số bước lấy mẫu: Hiện tại, quá trình sinh cử chỉ đòi hỏi một số bước lấy mẫu (sampling steps) tương đối lớn, ảnh hưởng đến tốc độ và hiệu quả của hệ thống. Việc tối ưu hóa để giảm số bước sampling mà không làm suy giảm chất lượng cử chỉ sinh ra sẽ giúp mô hình đáp ứng nhanh hơn và phù hợp cho các ứng dụng thời gian thực.
- Tích hợp và thử nghiệm các kỹ thuật embedding mới: Sử dụng các phương pháp embedding mới, đa dạng hóa thông tin đầu vào có thể giúp mô hình hiểu và phản ánh tốt hơn ngữ cảnh và cảm xúc của giọng nói. Đây cũng là một hướng phát triển để nâng cao khả năng của hệ thống trong việc sinh cử chỉ phù hợp với ngữ nghĩa của các ngôn ngữ khác nhau.
- Mở rộng sang các ngôn ngữ mới: Hiện tại, mô hình chủ yếu hoạt động với các dữ liệu giọng nói tiếng Anh. Nghiên cứu mở rộng mô hình để sinh cử chỉ cho các ngôn ngữ và nền văn hóa khác nhau sẽ là một bước tiến quan trọng, giúp hệ thống trở nên đa dạng và ứng dụng rộng rãi hơn.
- Kết hợp với mô hình DeepPhase [43] để đạt hiệu quả sinh cử chỉ thời gian thực: Mục tiêu của luận văn là tích hợp OHGesture với mô hình DeepPhase để phát triển hệ thống có khả năng phản hồi cử chỉ trong thời gian thực, ứng dụng trong các tình huống giao tiếp tự nhiên như hội thoại người-máy và các hệ thống điều khiển bằng giọng nói. Với mục tiêu là học các đặc trưng về pha của các chuyển động để có thể trích xuất được các đặc trưng chuyển động, thay vì sử dụng đặc trưng như một tấm ảnh của mô hình.
- Cải thiện đánh giá khách quan bằng các độ đo tự động: Để giảm sự phụ thuộc vào các đánh giá chủ quan, cần phát triển và tích hợp các phương pháp đánh giá tự động đáng tin cậy, cho phép mô hình có thể tự đánh giá và điều chỉnh theo các chỉ số khách quan.

6.4 Đóng góp của luận văn

Trong luận văn này, luận văn đã phát triển hệ thống sinh cử chỉ OHGesture, với các đóng góp quan trọng như sau:

- Phát triển mô hình sinh cử chỉ dựa trên Diffusion: Hệ thống OHGesture được thiết kế để tạo ra các cử chỉ đồng bộ với giọng nói đầu vào và phản ánh đúng

cảm xúc. Mô hình này còn có khả năng tổng quát hóa, cho phép sinh cử chỉ ngay cả với các mẫu giọng nói ngoài dữ liệu huấn luyện, mang lại độ chân thực cao.

- Công khai mã nguồn và mô hình trên các nền tảng mở: Để thúc đẩy ứng dụng và cải tiến từ cộng đồng, luận văn đã cung cấp mã nguồn trên GitHub, luận văn mở rộng mã nguồn, và được công khai ở [Github/OHGesture](https://github.com/hmthanh/OHGesture)¹ và một phiên bản pre-trained trên Huggingface huggingface.co/openhuman/openhuman², giúp các nhà nghiên cứu khác dễ dàng tiếp cận, tái hiện và mở rộng hệ thống.
- Tích hợp thêm văn bản và giọng nói chuyển ngữ trong quá trình sinh cử chỉ: Với tập dữ liệu ZeroEGGS chỉ bao gồm giọng nói, cử chỉ và nhãn cảm xúc, luận văn sử dụng các API của Azure và Google để phiên âm (transcribe) các tệp giọng nói thành văn bản. Giúp mở rộng dữ liệu đầu vào của mô hình sinh cử chỉ thêm đặc trưng văn bản, giúp hệ thống có thêm ngữ cảnh để sinh ra các cử chỉ phù hợp hơn với nội dung cụ thể.
- Đóng góp vào việc xây dựng hệ thống đánh giá chuẩn hóa: Chúng tôi phát triển một hệ thống xếp hạng trực tuyến [GENEA Leaderboard](https://genea-workshop.github.io/leaderboard)³ [35] cho các mô hình sinh cử chỉ. GENEAL Leaderboard sẽ thu thập và xử lý các dữ liệu cử chỉ từ nhiều nguồn ngôn ngữ và tập dữ liệu khác nhau, tạo thành một tập dữ liệu chuẩn hóa duy nhất. Xây dựng một bản xếp hạng để so sánh nhiều mô hình khác nhau. Và sử dụng người để đánh giá các mô hình sinh, giúp đánh giá chính xác kết quả sinh của cử chỉ so với các độ đo trước đây, vốn không thể đo được sự phức tạp và đa dạng trong chuyển động của cử chỉ tương ứng với giọng nói. Điều này giúp tạo nên một nền tảng dữ liệu thống nhất, hỗ trợ việc đánh giá đồng nhất giữa các mô hình. Từ đó thúc đẩy tính thống nhất trong lĩnh vực sinh cử chỉ trong cộng đồng nghiên cứu.
- Xây dựng công cụ trực quan hóa bằng Unity: Các hệ thống trực quan hóa cử chỉ hiện nay đều dựa trên Blender, và không đạt kết quả hiển thị tốt khi sinh cử chỉ, bằng việc kế thừa mã nguồn từ mô hình DeepPhase [43] để xây dựng hệ thống kết xuất bằng Unity [Github/DeepGesture-Unity](https://github.com/DeepGesture/deepgesture-unity)⁴.

¹Mã nguồn Github <https://github.com/hmthanh/OHGesture>

²HuggingFace <https://huggingface.co/openhuman/openhuman>

³GENEA Leaderboard: <https://genea-workshop.github.io/leaderboard/>

⁴Hệ thống render quá trình sinh cử chỉ bằng Unity <https://github.com/DeepGesture/deepgesture-unity>

- Xây dựng hệ thống đánh giá cử chỉ bằng FGD (Fréchet Gesture Distance): Dựa trên mã nguồn FGD [55], luận văn xây dựng GestureScore và huấn luyện một mô hình mới để đánh giá sự khác nhau về phân bố dữ liệu về góc quay giữa dữ liệu dự đoán và dữ liệu thật. Mã nguồn của GestureScore⁵ và mô hình pretrain của mô hình đánh giá được công khai ở Huggingface⁶
- Xây dựng hướng phát triển trong tương lai: Dựa trên nền tảng của mô hình diffusion và hiểu rõ bản chất của quá trình sinh cử chỉ, luận văn có thể kết hợp các mô hình sinh cử chỉ dựa trên pha với các thuật toán xử lý và trích xuất thông tin, nhằm tối ưu hóa chất lượng và tính phù hợp của các cử chỉ được sinh ra. Hướng phát triển này mở ra triển vọng cho các cải tiến sâu rộng trong việc tương tác giữa cử chỉ và các yếu tố ngữ cảnh phức tạp như biểu cảm khuôn mặt, ngữ điệu và động lực cảm xúc, tạo nền tảng cho những bước tiến trong các ứng dụng giao tiếp người - máy và các lĩnh vực liên quan khác.

⁵Github/GestureScore: <https://github.com/GestureScore/GestureScore>

⁶Huggingface/GestureScore: <https://huggingface.co/GestureScore/GestureScore>

6.5 Lời kết

Qua quá trình thử nghiệm và phân tích kết quả sinh cử chỉ thực tế, mô hình OHGesture của luận văn đã kế thừa và phát triển từ mô hình DiffuseStyleGesture, có khả năng sinh cử chỉ chân thực, không chỉ trên các mẫu dữ liệu trong tập huấn luyện mà còn mở rộng được với những giọng nói không có trong dữ liệu huấn luyện, ví dụ như giọng nói của Steve Jobs. Điều này minh chứng cho tiềm năng của mô hình Diffusion trong việc sinh cử chỉ cho các dữ liệu hiếm, nơi xác suất dữ liệu thấp.

Bên cạnh đó, luận văn đã đóng góp các mã nguồn chỉnh sửa trên Github, bao gồm quá trình kết xuất (render) và xử lý dữ liệu trên nền tảng Unity, tạo nền tảng thuận lợi cho các nghiên cứu và cải tiến tiếp theo đối với mô hình OHGesture. Việc kết hợp thêm văn bản vào quá trình sinh cử chỉ cũng là một bước đột phá, mở ra nhiều hướng phát triển ứng dụng trong các lĩnh vực yêu cầu tương tác tự nhiên và hiệu quả hơn.

TÀI LIỆU THAM KHẢO

- [1] Simon Alexanderson, Gustav Eje Henter, Taras Kucherenko, and Jonas Beskow. Style-controllable speech-driven gesture synthesis using normalising flows. *Computer Graphics Forum*, 39(2):487–496, 2020.
- [2] Simon Alexanderson, Rajmund Nagy, Jonas Beskow, and Gustav Eje Henter. Listen, denoise, action! audio-driven motion synthesis with diffusion models. *CoRR*, abs/2211.09707, 2022.
- [3] Tenglong Ao, Qingzhe Gao, Yuke Lou, Baoquan Chen, and Libin Liu. Rhythmic gesticulator: Rhythm-aware co-speech gesture synthesis with hierarchical neural embeddings. *ACM Trans. Graph.*, 41(6):209:1–209:19, 2022.
- [4] Tenglong Ao, Zeyi Zhang, and Libin Liu. Gesturediffuclip: Gesture diffusion model with clip latents. *ACM Transactions on Graphics (TOG)*, 42(4):1–18, 2023.
- [5] Kirsten Bergmann, Volkan Aksu, and Stefan Kopp. The relation of speech and gestures: Temporal synchrony follows semantic synchrony. In *Proceedings of the 2nd Workshop on Gesture and Speech in Interaction (GeSpIn 2011)*, 2011.
- [6] Uttaran Bhattacharya, Elizabeth Childs, Nicholas Rewkowski, and Dinesh Manocha. Speech2affectivestrajectories: Synthesizing co-speech gestures with generative adversarial affective expression learning. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 2027–2036, 2021.
- [7] Uttaran Bhattacharya, Nicholas Rewkowski, Abhishek Banerjee, Pooja Guhan, Aniket Bera, and Dinesh Manocha. Text2gestures: A transformer-based network for generating emotive body gestures for virtual agents. In *2021 IEEE virtual reality and 3D user interfaces (VR)*, pages 1–10. IEEE, 2021.

- [8] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.
- [9] Justine Cassell, Catherine Pelachaud, Norman Badler, Mark Steedman, Brett Achorn, Tripp Becket, Brett Douville, Scott Prevost, and Matthew Stone. Animated conversation: rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 413–420, 1994.
- [10] Junming Chen, Yunfei Liu, Jianan Wang, Ailing Zeng, Yu Li, and Qifeng Chen. Diffsheg: A diffusion-based approach for real-time speech-driven holistic 3d expression and gesture generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7352–7361, 2024.
- [11] 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. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, October 2022.
- [12] Chung-Cheng Chiu, Louis-Philippe Morency, and Stacy Marsella. Predicting co-verbal gestures: A deep and temporal modeling approach. In *Intelligent Virtual Agents: 15th International Conference, IVA 2015, Delft, The Netherlands, August 26-28, 2015, Proceedings 15*, pages 152–166. Springer, 2015.
- [13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [14] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.
- [15] Paul Ekman and Wallace V Friesen. The repertoire of nonverbal behavior: Categories, origins, usage, and coding. *semiotica*, 1(1):49–98, 1969.

- [16] Saeed Ghorbani, Ylva Ferstl, Daniel Holden, Nikolaus F. Troje, and Marc-André Carbonneau. Zeroeggs: Zero-shot example-based gesture generation from speech, 2022.
- [17] Ikhsanul Habibie, Weipeng Xu, Dushyant Mehta, Lingjie Liu, Hans-Peter Seidel, Gerard Pons-Moll, Mohamed Elgharib, and Christian Theobalt. Learning speech-driven 3d conversational gestures from video. In *Proceedings of the 21st ACM International Conference on Intelligent Virtual Agents*, pages 101–108, 2021.
- [18] Dai Hasegawa, Naoshi Kaneko, Shinichi Shirakawa, Hiroshi Sakuta, and Kazuhiko Sumi. Evaluation of speech-to-gesture generation using bi-directional lstm network. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents*, pages 79–86, 2018.
- [19] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Transactions on Graphics (TOG)*, 39(6):1–14, 2020.
- [20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- [21] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [22] Chien-Ming Huang and Bilge Mutlu. Robot behavior toolkit: generating effective social behaviors for robots. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 25–32, 2012.
- [23] Chris Jones. Ed - realistic digital human, 2014. Accessed: 2024-11-18.
- [24] Jihoon Kim, Jiseob Kim, and Sungjoon Choi. Flame: Free-form language-based motion synthesis & editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8255–8263, 2023.
- [25] Michael Kipp. *Gesture generation by imitation: From human behavior to computer character animation*. Universal-Publishers, 2005.

- [26] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.
- [27] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.
- [28] Taras Kucherenko, Patrik Jonell, Sanne Van Waveren, Gustav Eje Henter, Simon Alexandersson, Iolanda Leite, and Hedvig Kjellström. Gesticulator: A framework for semantically-aware speech-driven gesture generation. In *Proceedings of the 2020 international conference on multimodal interaction*, pages 242–250, 2020.
- [29] Taras Kucherenko, Patrik Jonell, Youngwoo Yoon, Pieter Wolfert, and Gustav Eje Henter. A large, crowdsourced evaluation of gesture generation systems on common data: The genea challenge 2020. In *26th international conference on intelligent user interfaces*, pages 11–21, 2021.
- [30] Sergey Levine, Philipp Krähenbühl, Sebastian Thrun, and Vladlen Koltun. Gesture controllers. In *Acm siggraph 2010 papers*, pages 1–11. ACM, 2010.
- [31] Haiyang Liu, Zihao Zhu, Naoya Iwamoto, Yichen Peng, Zhengqing Li, You Zhou, Elif Bozkurt, and Bo Zheng. Beat: A large-scale semantic and emotional multi-modal dataset for conversational gestures synthesis. In *European conference on computer vision*, pages 612–630. Springer, 2022.
- [32] Xian Liu, Qianyi Wu, Hang Zhou, Yinghao Xu, Rui Qian, Xinyi Lin, Xiaowei Zhou, Wayne Wu, Bo Dai, and Bolei Zhou. Learning hierarchical cross-modal association for co-speech gesture generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10462–10472, 2022.
- [33] Evelyn McClave. Gestural beats: The rhythm hypothesis. *Journal of psycholinguistic research*, 23(1):45–66, 1994.
- [34] Adam Metallo, Vincent Rossi, Jonathan Blundell, Günter Waibel, Paul Graham, Graham Fyffe, Xueming Yu, and Paul Debevec. Scanning and printing a 3d

- portrait of president barack obama. In *SIGGRAPH 2015: Studio*, pages 1–1. ACM, 2015.
- [35] Rajmund Nagy, Hendric Voss, Youngwoo Yoon, Taras Kucherenko, Teodor Nikolov, Thanh Hoang-Minh, Rachel McDonnell, Stefan Kopp, Michael Neff, and Gustav Eje Henter. Towards a genea leaderboard—an extended, living benchmark for evaluating and advancing conversational motion synthesis. *arXiv preprint arXiv:2410.06327*, 2024.
- [36] Michael Neff, Michael Kipp, Irene Albrecht, and Hans-Peter Seidel. Gesture modeling and animation based on a probabilistic re-creation of speaker style. *ACM Transactions On Graphics (TOG)*, 27(1):1–24, 2008.
- [37] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.
- [38] Shenhan Qian, Zhi Tu, Yihao Zhi, Wen Liu, and Shenghua Gao. Speech drives templates: Co-speech gesture synthesis with learned templates. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11077–11086, 2021.
- [39] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.
- [40] Michael Saxon, Ayush Tripathi, Yishan Jiao, Julie M Liss, and Visar Berisha. Robust estimation of hypernasality in dysarthria with acoustic model likelihood features. *IEEE/ACM transactions on audio, speech, and language processing*, 28:2511–2522, 2020.
- [41] Thomas A Sebeok and Jean Umiker-Sebeok. *Advances in visual semiotics: The semiotic web 1992-93*, volume 118. Walter de Gruyter, 2011.
- [42] Yang Song. Score-based generative modeling: A brief introduction, 2021. Accessed: 2024-11-18.

- [43] Sebastian Starke, Ian Mason, and Taku Komura. Deepphase: Periodic autoencoders for learning motion phase manifolds. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022.
- [44] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.
- [45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [46] Rebecca A Webb. *Linguistic features of metaphoric gestures*. University of Rochester, 1997.
- [47] Lilian Weng. What are diffusion models? *lilianweng.github.io*, Jul 2021.
- [48] Bowen Wu, Chaoran Liu, Carlos T Ishi, and Hiroshi Ishiguro. Probabilistic human-like gesture synthesis from speech using gru-based wgan. In *Companion Publication of the 2021 International Conference on Multimodal Interaction*, pages 194–201, 2021.
- [49] Jing Xu, Wei Zhang, Yalong Bai, Qibin Sun, and Tao Mei. Freeform body motion generation from speech. *arXiv preprint arXiv:2203.02291*, 2022.
- [50] Sicheng Yang, Zhiyong Wu, Minglei Li, Zhensong Zhang, Lei Hao, Weihong Bao, Ming Cheng, and Long Xiao. Diffusestylegesture: Stylized audio-driven co-speech gesture generation with diffusion models. *arXiv preprint arXiv:2305.04919*, 2023.
- [51] Sicheng Yang, Zhiyong Wu, Minglei Li, Mengchen Zhao, Jiuxin Lin, Liyang Chen, and Weihong Bao. The reprgesture entry to the genea challenge 2022. In *Proceedings of the 2022 International Conference on Multimodal Interaction*, pages 758–763, 2022.
- [52] Sicheng Yang, Zunnan Xu, Haiwei Xue, Yongkang Cheng, Shaoli Huang, Mingming Gong, and Zhiyong Wu. Freetalker: Controllable speech and text-driven gesture generation based on diffusion models for enhanced speaker naturalness.

In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7945–7949. IEEE, 2024.

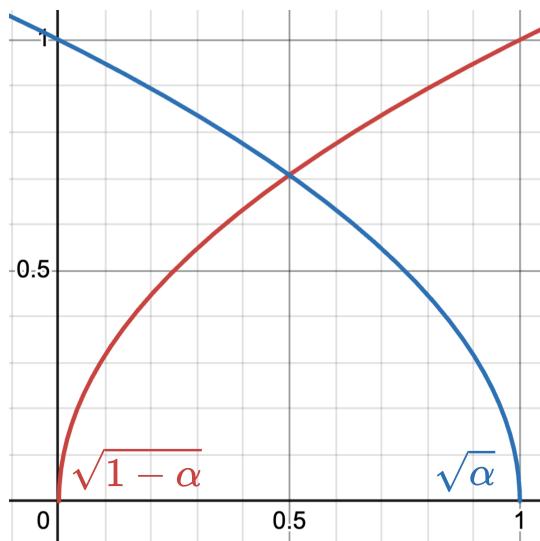
- [53] Sicheng Yang, Haiwei Xue, Zhensong Zhang, Minglei Li, Zhiyong Wu, Xiaofei Wu, Songcen Xu, and Zonghong Dai. The diffusestylegesture+ entry to the genea challenge 2023. In *Proceedings of the 2023 International Conference on Multimodal Interaction*, 2023.
- [54] Yanzhe Yang, Jimei Yang, and Jessica Hodgins. Statistics-based motion synthesis for social conversations. In *Computer Graphics Forum*, volume 39, pages 201–212. Wiley Online Library, 2020.
- [55] Youngwoo Yoon, Bok Cha, Joo-Haeng Lee, Minsu Jang, Jaeyeon Lee, Jaehong Kim, and Geehyuk Lee. Speech gesture generation from the trimodal context of text, audio, and speaker identity. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020.
- [56] Youngwoo Yoon, Pieter Wolfert, Taras Kucherenko, Carla Viegas, Teodor Nikolov, Mihail Tsakov, and Gustav Eje Henter. The genea challenge 2022: A large evaluation of data-driven co-speech gesture generation. In *Proceedings of the 2022 International Conference on Multimodal Interaction*, pages 736–747, 2022.
- [57] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001*, 2022.
- [58] Chi Zhou, Tengyue Bian, and Kang Chen. Gesturermaster: Graph-based speech-driven gesture generation. In *Proceedings of the 2022 International Conference on Multimodal Interaction*, pages 764–770, 2022.

DANH MỤC CÔNG TRÌNH CỦA HVCH

1. Towards a GENEVA Leaderboard – an Extended, Living Benchmark for Evaluating and Advancing Conversational Motion Synthesis [35].

Phụ lục A. CÁC THAM SỐ TRONG DIFFUSION

A.1 Sự thay đổi của $\sqrt{\alpha}$ và $\sqrt{1 - \alpha}$



Hình A.1: Sự thay đổi của $\sqrt{\alpha}$ và $\sqrt{1 - \alpha}$

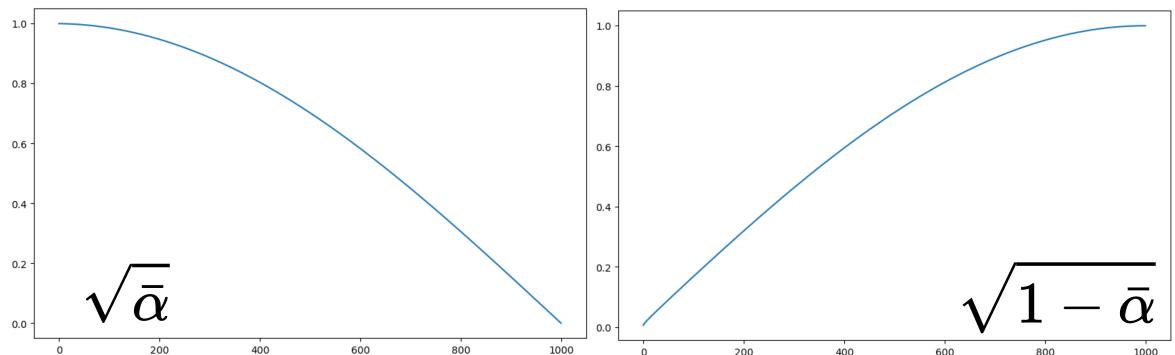
Trong quá trình diffusion, hệ thống muốn giảm dần sự hiện diện của x và tăng dần sự hiện diện của nhiễu ϵ_t . Với $x_t = \sqrt{\alpha_t}x_{t-1} + \sqrt{1 - \alpha_t}\epsilon_t$. Các hệ số $\sqrt{\alpha_t}$ và $\sqrt{1 - \alpha_t}$ giúp điều khiển quá trình này như sau:

- $\sqrt{\alpha_t}$: Ban đầu giá trị lớn nhưng giảm dần để giảm sự ảnh hưởng vào kết quả tổng giữa nhiễu và x_t
- $\sqrt{1 - \alpha_t}$: Ban đầu giá trị nhỏ nhưng tăng dần theo từng bước. Với mục tiêu là đến cuối cùng thì sự ảnh hưởng của nhiễu càng lớn và trở thành nhiễu hoàn toàn.

Cả hai đại lượng này thay đổi theo thời gian trong quá trình diffusion, từ đó quyết định mức độ nhiễu được thêm vào từng bước t và mức độ giữ lại trạng thái trước đó qua từng bước.

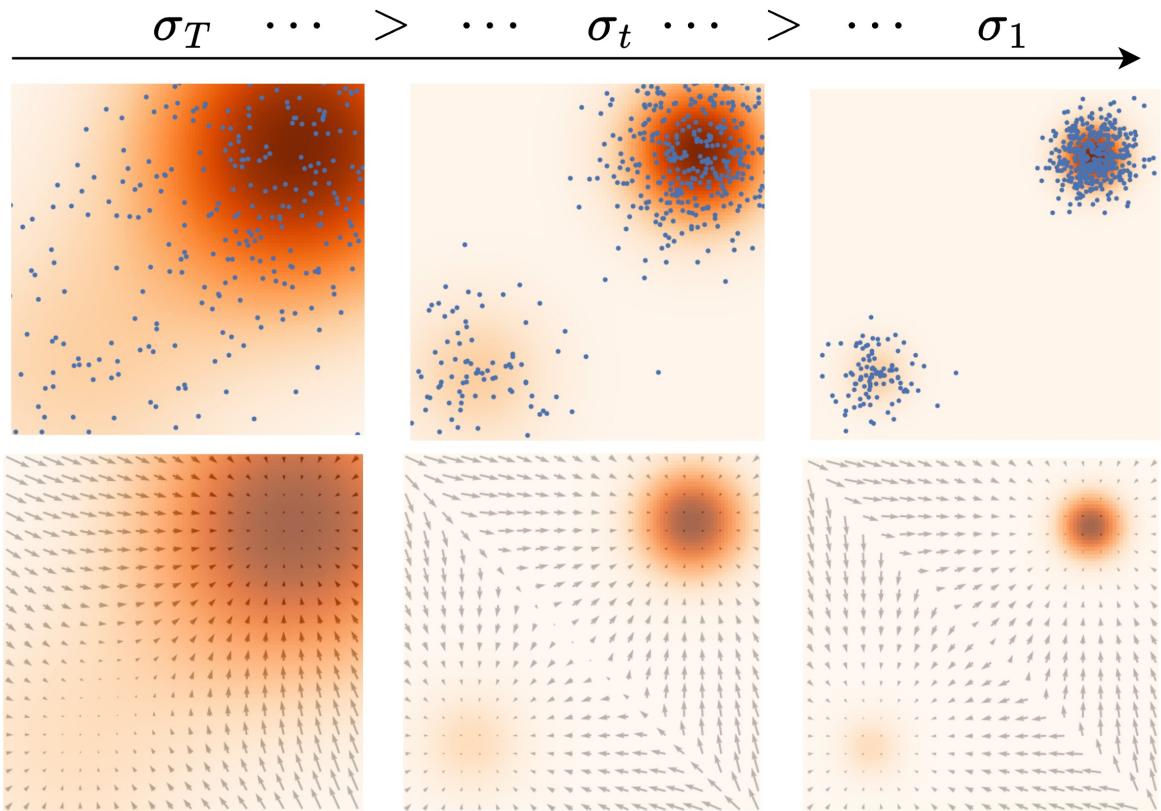
A.2 Sự thay đổi của $\sqrt{\bar{\alpha}}$ và $\sqrt{1 - \bar{\alpha}}$

$\bar{\alpha}_t = \prod_{i=1}^t \alpha_i \cdot \sqrt{1 - \bar{\alpha}_t} = \sqrt{1 - \prod_{i=1}^t \alpha_i}$. Cả $\sqrt{\bar{\alpha}}$ và $\sqrt{1 - \bar{\alpha}}$ đóng một vai trò quan trọng trong quá trình huấn luyện và sinh cử chỉ. Giúp kiểm soát mức độ nhiễu được thêm vào và cử chỉ, ta có thể thấy $\sqrt{\bar{\alpha}}$ giảm dần còn $\sqrt{1 - \bar{\alpha}}$ thì tăng dần..



Hình A.2: Quá trình thay đổi của $\sqrt{\bar{\alpha}}$ và $\sqrt{1 - \bar{\alpha}}$

A.3 Sự thay đổi của σ_t



Hình A.3: Sự thay đổi của hệ số σ_t trong quá trình lấy mẫu

Trong quá trình lấy mẫu, mục tiêu của việc giảm dần σ_t , mô hình có thể điều hướng về các vùng có mật độ dữ liệu cao ở từng bước t .

Phụ lục B. QUÁ TRÌNH XỬ LÝ DỮ LIỆU BVH

B.1 Cấu trúc khung xương của một nhân vật

Một số tên khung xương nhân vật trong 75 khung xương chuyển động bao gồm:

Hips, Spine, Neck, Head, RightShoulder, RightArm, RightForeArm, RightHand, LeftShoulder, LeftArm, LeftForeArm, LeftHand, RightUpLeg, RightLeg, RightFoot, RightToeBase, LeftUpLeg, LeftLeg, LeftFoot, LeftToeBase, ...



Hình B.1: Khung xương của một nhân vật

B.2 Cấu trúc tệp dữ liệu BVH

File BVH (Biovision Hierarchy) là một định dạng file dữ liệu chứa thông tin về cấu trúc xương và dữ liệu về chuyển động của xương trong một hệ thống xương. File BVH bao gồm hai phần chính: phần khai báo cấu trúc xương và phần dữ liệu chuyển động của xương.

- **HIERARCHY:**

- Định nghĩa thành phần và tên các khung xương, vị trí khởi tạo của các khung xương ở vị trí T-pose (Tay diễn viên motion capture sẽ được duỗi thẳng thành hình chữ T).
- Định nghĩa quan hệ cha-con từ nút gốc đến các nút lá của một khung xương. Với nút gốc thường là cột sống (Spine).
- Định nghĩa các dữ liệu sẽ được ghi nhận như vị trí hoặc góc quay theo chiều X, Y, Z của mỗi khung xương theo thời gian.

- **MOTION:** Là chuỗi chuyển động theo từng khung hình, với mỗi khung hình là dữ liệu của từng xương về chuyển động theo thông tin về góc quay hoặc vị trí đã định nghĩa ở HIERARCHY.

```

HIERARCHY
ROOT Hips
{
    OFFSET 0.00352400006 86.606987 0
    CHANNELS 6 Xposition Yposition Zposition Zrotation Yrotation Xrotation
    JOINT Spine
    {
        OFFSET 3.72965547e-17 8.81424618 -2.0801697
        CHANNELS 6 Xposition Yposition Zposition Zrotation Yrotation Xrotation
        JOINT Spine1
        {
            OFFSET -1.01898064e-17 8.84414101 -1.06827795e-17
            CHANNELS 6 Xposition Yposition Zposition Zrotation Yrotation Xrotation
            JOINT Spine2
            {
                OFFSET -7.98921476e-17 11.6822681 -3.720187e-17
                CHANNELS 6 Xposition Yposition Zposition Zrotation Yrotation Xrotation
                JOINT Spine3
                {
                    OFFSET -1.73472348e-17 11.7467003 -3.81639165e-16
                    CHANNELS 6 Xposition Yposition Zposition Zrotation Yrotation Xrotation
                    JOINT Neck
}

```

(a) HIERARCHY trong tệp BVH

```

MOTION
Frames: 7286
Frame Time: 0.016667
0.00352400006 86.606987 0 0.490480989 0.0279319994 -6.49757195
0.00352400006 86.606987 -0.0283540003 0.48171699 -0.0194809996
0.00352400006 86.606987 -0.0546680018 0.47310701 -0.0689229965
0.00352400006 86.6056061 -0.0783089995 0.464415014 -0.12521399
0.00352400006 86.6043854 -0.100622997 0.455056995 -0.185530007
0.00352400006 86.6031876 -0.122184001 0.444750011 -0.249752 -6
0.00352400006 86.6019974 -0.142823994 0.433441013 -0.317882001
0.00352400006 86.6007996 -0.163385004 0.420841008 -0.38962099
0.00352400006 86.5995941 -0.185066 0.405855 -0.464527011 -6.83
0.00352400006 86.5983505 -0.207151994 0.389432997 -0.541544974
0.00352400006 86.5970688 -0.229480997 0.372554004 -0.614641011
0.00352400006 86.5951233 -0.25286001 0.353704005 -0.684701025
0.00352400006 86.5928802 -0.277058005 0.333912998 -0.756413996
0.00352400006 86.5905991 -0.30133 0.314761996 -0.830637991 -7.
0.00352400006 86.588295 -0.325839996 0.296337992 -0.905332029
0.00352400006 86.5859909 -0.350551009 0.278358996 -0.9790055992
0.00352400006 86.5836182 -0.375133991 0.262282014 -1.05174804
5.09999991e-05 86.5807953 -0.39993 0.249116004 -1.12400305 -7.

```

(b) MOTION trong tệp BVH

$\text{rotation}_i^{\text{local}} = \{\alpha, \beta, \gamma\}$ lần lượt là góc quay quanh các trục Z, Y , và X , góc quay tổng hợp ba trục trong không gian Euler là

$$R = R_Z(\alpha)R_Y(\beta)R_X(\gamma) \quad (\text{B.1})$$

Trong đó:

1. Ma trận quay quanh trục Z :

$$R_Z(\alpha) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2. Ma trận quay quanh trục Y :

$$R_Y(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix}$$

3. Ma trận quay quanh trục X :

$$R_X(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{bmatrix}$$

Để tính tọa độ chuyển động của một nhân vật, thực hiện phép toán sau:

$$\text{position}_{\text{global}} = R \cdot \text{position}_{\text{local}} + \mathbf{t} \quad (\text{B.2})$$

B.3 Quá trình chuyển góc quay Euler sang Quaternion

Để tránh Gimbal lock, dữ liệu ở dạng góc quay Euler phải được chuyển sang dạng góc quay Quaternion. Trong đó góc quay từng Bone dạng Euler ZYX sang dạng Quaternion, mỗi Bone biểu diễn bằng 4 phần tử $q = (q_w, q_x, q_y, q_z)$, với các giá trị được tính như sau:

Đầu tiên, tính các giá trị cos và sin của một nửa góc quay cho mỗi trục:

- $c_\alpha = \cos\left(\frac{\alpha}{2}\right), \quad s_\alpha = \sin\left(\frac{\alpha}{2}\right)$
- $c_\beta = \cos\left(\frac{\beta}{2}\right), \quad s_\beta = \sin\left(\frac{\beta}{2}\right)$
- $c_\gamma = \cos\left(\frac{\gamma}{2}\right), \quad s_\gamma = \sin\left(\frac{\gamma}{2}\right)$

Dựa trên các giá trị tính được ở trên, các thành phần của quaternion được tính như sau:

- $q_w = c_\alpha c_\beta c_\gamma + s_\alpha s_\beta s_\gamma$
- $q_x = c_\alpha c_\beta s_\gamma - s_\alpha s_\beta c_\gamma$
- $q_y = c_\alpha s_\beta c_\gamma + s_\alpha c_\beta s_\gamma$
- $q_z = s_\alpha c_\beta c_\gamma - c_\alpha s_\beta s_\gamma$

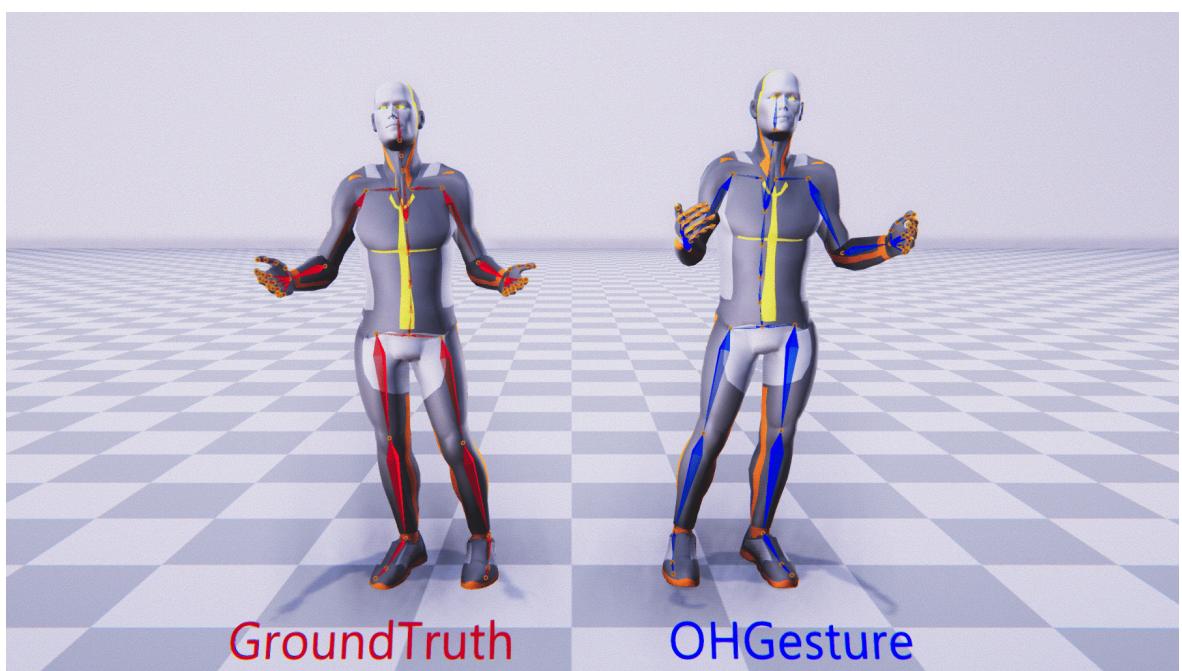
Với quaternion q đã tính, vị trí toàn cục của bone $\mathbf{p}_{\text{global}}$ được xác định bằng cách quay vị trí cục bộ $\mathbf{p}_{\text{local}}$ thông qua công thức:

$$\mathbf{p}_{\text{global}} = q \cdot \mathbf{p}_{\text{local}} \cdot q^{-1} + \mathbf{t} \quad (\text{B.3})$$

với \mathbf{t} là vị trí gốc của bone trong không gian toàn cục.

Phụ lục C. MINH HỌA KẾT QUẢ SUY LUẬN CỦA CỦ CHỈ

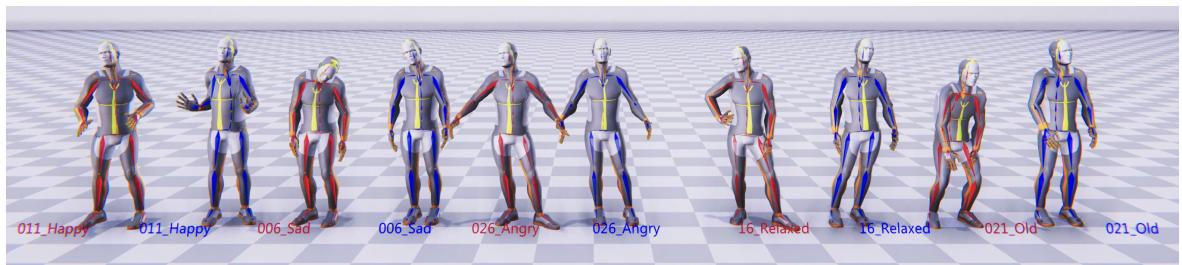
C.1 So sánh giữa cử chỉ ground truth và cử chỉ dự đoán



Click vào ảnh để xem video

Kết quả cử chỉ ground truth và cử chỉ dự đoán của mô hình ở khung hình 3821 dự đoán trên mẫu giọng nói 003_Neutral_2_x_1_0

C.2 Minh họa các cảm xúc khác nhau của cử chỉ



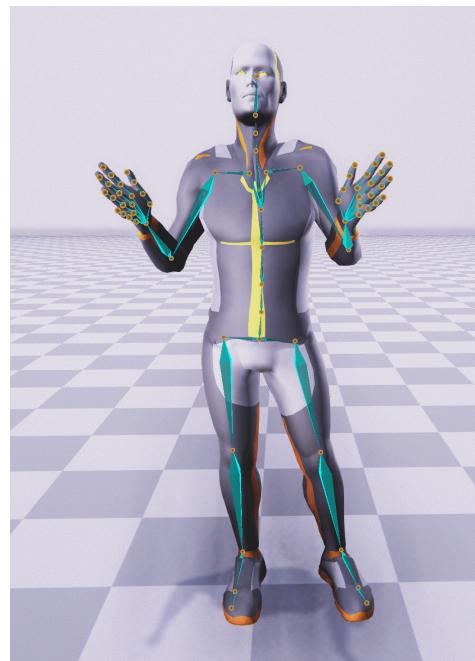
Click vào ảnh để xem video

Kết quả sinh hay suy luận của mô hình OHGesture với các cảm xúc khác nhau. Màu đỏ là ground truth trong tập dữ liệu ZeroEGGS, màu xanh là kết quả sinh của mô hình OHGesture.



Click vào ảnh để xem video

C.3 Minh họa việc sinh cử chỉ với giọng nói nằm ngoài tập huấn luyện



Click vào ảnh để xem video

Minh họa sinh cử chỉ tương ứng với giọng nói của Steve Job



Click vào ảnh để xem video

Minh họa sinh cử chỉ với giọng nói được tạo từ Microsoft Azure giới thiệu về đề tài.

C.4 Minh họa chuyển động của nhân vật



Click vào ảnh để xem video

Minh họa cử chỉ được lấy 6 khung hình phía trước và 6 khung hình phía sau của cử chỉ.