Phase 4

Jorge Mejia, Leon Do, Ryan Hernandez

Group 5

CECS 458, Sec 3 12467, Fall 2025

Professor Moon

California State University, Long Beach

Dec. 1, 2025

1. Executive Summary

The AI-Powered Expression-to-Avatar System is a multimodal deep learning pipeline that aims to make virtual material more accessible to everyone. The system turns human facial reactions into real-time avatar animations by using computer vision to find facial landmarks and emotion recognition. It does this without needing expensive motion capture equipment or complicated manual rigging. The final version successfully combines MediaPipe Face Mesh, DeepFace, and Kalman filtering to control a Live2D avatar through VTube Studio. This makes a tool for streaming and working from home that protects privacy.

2. Problem Definition

In the changing world of digital communication, millions of people live stream, learn from home, and hold virtual meetings. But there is a big difference between privacy and freedom of speech.

1. High Barrier to Entry: To use professional avatar tools like Animaze, you usually need to pay for subscriptions, know a lot about technical rigging, or buy expensive gear.
2. Lack of Expression: Free options, like Zoom avatars, are static and don't let you change their faces or emotions.
3. Concerns about privacy: users want to show how they feel and who they are without giving away their location or name.

Our answer to these problems is an open-source, lightweight alternative that uses current webcam hardware to make avatars move and react in real time.

3. Proposed Solution

We made a desktop app that takes video input, runs it through an AI pipeline, and then connects the data to a virtual persona.

- Core Functions: The system takes 468 facial cues for geometry and figures out how someone is feeling (e.g., Happy, Sad, Surprise).
- Integration: It uses WebSocket to send data to VTube Studio, and OBS can then record them so they can be streamed on Twitch, YouTube, or Zoom.
- Accessibility: The software is meant to be free and watermark-free for individual users, and its flexible design lets the community create add-ons.

4. Deep Learning Techniques & System Architecture

Geometric analysis and Convolutional Neural Networks (CNNs) are both used in the system's multimodal workflow.

4.1. Real-Time Landmark Detection

We took out 468 3D face features using MediaPipe Face Mesh.

- Region of Interest: The forehead, cheeks, and chin are separated from the rest of the face by geometry.
- Signal Processing: We used Kalman Filters (KF1D) to reduce marker jitter, which is a common problem with webcam-based tracking. This makes the model's

interpretation of the raw data more even and normal, which makes sure that the avatar can move smoothly.

- Algorithmic Detail: Eye Aspect Ratio (EAR) We used the Eye Aspect Ratio (EAR) technique to convert the raw landmark data into the "Eye Open" parameter. For every eye, MediaPipe offers six coordinates. Based on the ratio of the vertical to horizontal distances between these landmarks, the EAR is computed to identify blinking and winking states. The following formula is applied:

  The horizontal corners of the eye are represented by p1 and p4, while the vertical eyelids are represented by the remaining points. The system records a "Blink" event and sets the avatar's EyeOpen parameter to 0.0 if the EAR drops below a calibrated threshold range defined in configuration. By default, it is set to (EAR_MIN) 0.08 as a fully closed blink while (EAR_MAX) 0.35 is defined as a fully open eye. This mathematical method guarantees that blinking is not the same as just glancing down. The eyes are also set as separate variables to accommodate winks and other expressions with one eye.

4.2. Emotion Recognition

"Happy," "neutral," "surprise," "angry," "sad," "fear," and "disgust" are the seven different emotional states that DeepFace helped us identify.

- Calibration: At starting, a "neutral calibration" step takes a picture of the user's face while it is at rest for two seconds. This baseline is taken away from new forecasts to make the classifier more sensitive to the features of each face.
- Smoothing: To stop emotions from changing quickly and unnaturally between states, they are run through an exponential moving average.

4.3. Parameter Fusion & Mapping

The system fuses two data streams:

- Signals that stay the same: Yaw, Pitch, Tongue position, Eye Openness (EAR), and Mouth Openness (MAR).
- Categorical Emotions: DeepFace identified the most prevalent emotion. These are transferred using the WebSocket API of VTube Studio and mapped to Live2D parameters (such as EyeOpenLeft, MouthOpen, and AngleX).

4.4. Producer-Consumer Camera Architecture

Before this improvement, our camera, emotion inference, MediaPipe processing, and GUI were all competing in the same execution loop. This created **toggling lag**, **frame stutters**, and occasional **race conditions**, especially when the user rapidly turned the camera on or off.

To solve this, we introduced a dedicated **Producer-Consumer pipeline** for frame acquisition:

- Producer (Camera Thread) :

A dedicated background thread continuously reads frames from the webcam and places them into a thread-safe queue capped at a maximum size of **1**. This guarantees the system always works with the most recent frame available and automatically discards older frames whenever the processing pipeline falls behind.

- Consumer (Main Thread) :
  The main application loop acts as the consumer by retrieving the most recent frame from this queue and performing face tracking, MediaPipe processing, and DeepFace inference on it.

- Benefit:
  This Producer-Consumer architecture decouples the potentially blocking camera I/O operations from the computationally heavy inference stages. As a result, the GUI remains responsive, animation updates stay smooth, and frame processing is no longer vulnerable to nondeterministic timing issues caused by camera read delays.

5. Model Evaluation (Quantitative & Qualitative)

5.1. Quantitative Performance

- Frame Rate Efficiency: A concurrent processing stream is successfully created by the pipeline. At about 30 frames per second, facial tracking (landmarks) produces smooth head and lip movements.
- Emotion Latency: In order to control computational load, the DeepFace emotion classifier runs at a reduced frequency (10-15 Hz). The exponential moving average effectively hides latency, preventing "flickering" emotions, even if it is slower than landmark tracking.
- System Resource Utilization: Testing was conducted on a standard mid-range laptop (Specs: NVIDIA RTX 3060, Intel i7).
    - GPU VRAM Usage: About 1.2 GB of VRAM are used by the DeepFace emotion model.
    - CPU Overhead: MediaPipe uses a lot of CPU power. CPU consumption averaged 18-22% during peak operation (tracking + streaming).
    - Latency Analysis: From physical movement to avatar reply, the average system latency is 150 milliseconds. This includes:
        - Camera Input Lag: ~30ms
        - Inference Time (DeepFace): ~70ms
        - WebSocket Transmission & Smoothing: ~50 ms Although competitive gaming scenarios may demand additional improvement, this latency is within acceptable bounds for casual broadcasting.

- Landmark Precision: Although severe head postures (angles > 45 degrees) occasionally caused landmark loss, using refine_landmarks=True in MediaPipe allowed for reliable iris tracking.

5.2. Qualitative Feedback & User Experience

- Stability: When compared to the initial proof-of-concept, the application of Kalman filters greatly enhanced the user experience. The "shaking" that is frequently observed in inexpensive trackers is absent from the avatar's idle state, which is stable.
- Calibration Sensitivity: The user must stay motionless for two seconds in order for the "Neutral Calibration" feature to successfully customize the experience. The emotion mapping becomes distorted if it is interrupted (e.g., resting face perceived as "Sad").
- Expression gaps: The MVP's current VTube Studio model does not have blend shapes for "Fear" and "Tongue Out." In those particular cases, there is a disconnect because although the AI model recognizes these, the avatar is unable to visually represent them.

6. Business Applicability & Market Impact

By 2029, the virtual avatar market is expected to reach $7.9 billion, up from $2.3 billion in 2024. Our technology uses a hybrid business approach to target the "Creator Economy":

- The "Colab of Virtual Avatars": The underlying tool is free for independent authors to utilize. Adoption and ecosystem expansion are fueled by this.
- Marketplace: By enabling developers to sell assets (costumes, props), a community store fosters a self-sufficient economy.
- Enterprise Cloud Tiers: For dedicated GPU performance and round-the-clock uptime, companies who oversee numerous virtual talent (such as agencies like VShojo) can sign up for cloud-hosted tiers.

7. Prototype Demo & User Guide

Core/landmarks.py, core/emotion.py, and vtube_studio/client.py are the modular components of the prototype, which is hosted on GitHub.

Setup Instructions:

1. Launch VTube Studio and enable the API server.
2. Run the Python client.
3. Calibration: Sit still for 2 seconds while the system captures the baseline.
4. Operation: The system will automatically pipe data to the avatar.
5. Streaming: Use OBS to capture the VTube Studio window via "Transparent Window Capture"

7.1. Troubleshooting & Common Issues

During the development and testing phases, several common setup issues were identified. Users should consult the following guide if the avatar behaves erratically:

- Jittering Avatar: Make sure the camera is positioned on a sturdy surface if the avatar trembles in spite of the Kalman filters. The AI perceives sensor noise in low-light conditions as micro-movements. Make the user's face more illuminated.
- Incorrect Emotion Locking: The Neutral Calibration was probably tainted if the avatar becomes "stuck" on a sad or angry look. During the first two seconds of initialization, restart the client and make sure the face is expressionless as well as not moving..
- Connection Refused: Verify that the VTube Studio settings' "Start API" button is turned on. By default, the program listens on port 8001; to permit local WebSocket traffic, firewall settings may need to be changed.

8. Roadmap & Future Work

The following actions are planned in order to move from an MVP to a commercial product:

1. Expanded Model Support: To match the AI's detecting skills, update the Live2D model to handle the missing "Fear" and "Tongue" attributes. Our code does support fear & tongue out detections, but the avatar doesn’t support it currently.
2. Dynamic Calibration: To overcome the restriction of requiring complete stillness, replace the static 2-second startup calibration with a dynamic background calibration that adapts as the user moves.
3. Cloud Integration: Create the backend infrastructure to enable the suggested enterprise business model by offloading DeepFace computation to the cloud for users with low-end hardware.

9. Ethical Considerations & Safety

Ethical deployment is crucial, just like with any AI solution that uses deep learning and facial recognition.

- Data privacy: Our system processes all video feeds locally, in contrast to cloud-based alternatives that upload user footage to distant servers. The client

computer is where the DeepFace and MediaPipe inference takes place. User anonymity is guaranteed since no biometric data is sent or kept.
- Puppeteering vs. Deepfakes: It's critical to differentiate this system from "Deepfake" technology. Instead of creating artificial human likenesses, our technology employs human signals to control a fictional, stylized character (puppeteering). This lowers the possibility of malevolent impersonation or identity theft.
- Inclusivity: We recognize that open-source datasets, such as those used to train DeepFace, can display bias with respect to age or skin tone. To make sure the emotion identification functions equally effectively for all users, future revisions of this project would benefit from testing across a more diverse audience.

10. Conclusion

Phases 3 and 4 effectively showed how a multimodal AI pipeline may make high-quality avatar animation more accessible. We developed a real-time, privacy-focused technology that easily integrates with current broadcasting software by combining MediaPipe's geometric precision with DeepFace's semantic comprehension. Although there are some small restrictions on model blend-shapes, the main architecture demonstrates that costly rigging is no longer necessary for digital expressiveness.

11. Relevant Links

- Github Repo
  https://github.com/RyanHernandezz/Vtuber

- Slides
  https://github.com/RyanHernandezz/Vtuber/blob/main/docs/Presentation_Slides.pdf

- Video Demo
  https://drive.google.com/file/d/1oqyNTWql2j57wjekeoORraIBYwqycN1V/view?usp=drive_link

References

MediaPipe Face Mesh
Google Research. "MediaPipe Face Mesh: High Fidelity Facial Landmark Detection."
Documentation: https://developers.google.com/mediapipe

DeepFace Framework
Sefik Ilkin Serengil and Alper Ozpinar. *"Lightweight Face Recognition and Facial Attribute Analysis (DeepFace Library)."* 2020.
GitHub: https://github.com/serengil/deepface

OpenCV
Bradski, G. "The OpenCV Library." *Dr. Dobb's Journal of Software Tools*, 2000.
https://opencv.org/

Kalman Filter Theory
Kalman, R. E. "A New Approach to Linear Filtering and Prediction Problems." *Transactions of the ASME-Journal of Basic Engineering*, 1960.

Kalman Filter Theory
DenchiSoft. "VTube Studio Public API."
https://github.com/DenchiSoft/VTubeStudio