Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 10 , October - 2026

Morph-Talk: Avatar-Based Video Conversation

A Browser-Based System Mapping Facial and Upper-Body Movements to VRM Avatars

Keerthana Bk
Department of Computer Science and Engineering
G. Madegowda Institute of Technology (GMIT)
Mandya, Karnataka, India

Rakshitha RS
Department of Computer Science and Engineering
G. Madegowda Institute of Technology (GMIT)
Mandya, Karnataka, India

Yashaswini SR
Department of Computer Science and Engineering
G. Madegowda Institute of Technology (GMIT)
Mandya, Karnataka, India

Yogitha MK
Department of Computer Science and Engineering
G. Madegowda Institute of Technology (GMIT)
Mandya, Karnataka, India

Divyarani SY [GUIDE]
Department of Computer Science and Engineering
G. Madegowda Institute of Technology (GMIT)
Mandya, Karnataka, India

Pallavi R [GUIDE]
Department of CST-CY
ATME, Mysore

***Abstract* - Video calls normally show the camera image of each participant. This paper presents Morph-Talk, a full-stack web application in which two users hold a video call while each appears to the other as an animated three-dimensional anime-style VRM avatar instead of the real camera feed. Each user's webcam video is used only locally: MediaPipe Holistic extracts face and body tracking information from it inside the browser, and this tracking data, not the video, is relayed to the other participant through Socket.IO. The receiving browser animates the corresponding avatar live with Three.js and three-vrm in a React.js interface, while the participants hear each other's voices during the call. Facial expressions are derived from geometric metrics measured against a calibrated neutral baseline and mapped to avatar expressions, while head orientation and arm movement are estimated from landmarks. A Node.js server with Socket.IO handles call signalling, with the call connection set up using WebRTC, and MongoDB is used for data persistence. We describe the architecture, the processing pipeline and the engineering problems met during implementation: differences between VRM versions, per-user calibration, secure-context browser requirements and audio device routing. The implemented features and the observations made during development are summarised. No quantitative accuracy, latency or frame-rate evaluation and no user study has been carried out; these are identified as future work.**

***Keywords* - Avatar-based communication; VRM avatar; WebRTC; MediaPipe Holistic; facial expression mapping; Three.js; Socket.IO**

I. Introduction

Video calling over the web is routine, and its building blocks are standardised: the W3C WebRTC (Web Real-Time Communication) interfaces and the accompanying IETF protocols let browsers exchange audio, video and data directly [1], [2]. In a conventional call, however, the visual channel is the camera image itself, so each participant's appearance and surroundings are shown to the other party. Research on collaborative virtual environments has long discussed separating a person's rendered representation from their actual appearance while still conveying nonverbal behaviour [3].

Morph-Talk is a full-stack web application that lets two users video call each other while appearing as animated 3D anime avatars instead of their real camera feed. The user's facial expressions and body and head movements are tracked live and mirrored in real time on an avatar in the VRM (Virtual Reality Model) format [4], so that the other person sees an animated character that moves and emotes like the real user.

The problem addressed is an engineering one: to combine browser-side landmark tracking, VRM avatar rendering, live tracking-data exchange, voice communication and a signalling back end in a single web application that copes with differences between users and between avatar models. The objectives are:

- to support a two-user video call with call setup, acceptance and rejection;
- to map facial expressions to VRM expressions relative to a neutral baseline calibrated for each user;
- to estimate head orientation and arm movement from landmarks and apply them to the avatar;
- to support both VRM 0.x and VRM 1.0 avatar models;
- to relay the extracted tracking data through Socket.IO so that the remote participant sees the avatar moving live, without the webcam video being shown to them; and
- to support voice communication between the participants during the call.

The scope is a browser-based, two-user prototype. The contributions are the architecture and processing pipeline of the prototype, an account of the engineering problems met while building it, and a summary of its implemented features with an explicit statement of what has not yet been measured.

The remainder of the paper reviews related work (Section II), presents the proposed system (III), describes the methodology and implementation (IV), reports functional observations and discussion (V), and closes with limitations and future scope (VI) and the conclusion (VII).

II. Literature Review

A. Browser-Based Real-Time Communication

WebRTC is defined as a set of browser interfaces by the W3C [1] and as a set of network protocols by the IETF, summarised in RFC 8825 [2]. Peers behind network address translators find usable network paths with Interactive Connectivity Establishment (ICE), which builds on the STUN and TURN protocols [5]. The specifications leave signalling to the application. Morph-Talk follows this pattern and uses Socket.IO for signalling. Socket.IO provides low-latency, bidirectional, event-based

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 10 , October - 2026

communication between a client and a server; it uses the WebSocket protocol [6] for transport when possible but is not itself a WebSocket implementation [7].

B. Landmark-Based Face and Body Tracking

MediaPipe is a framework for assembling perception pipelines from modular components [8]. Its face model predicts a dense mesh of 468 vertices from a single camera image [9], and its pose model, BlazePose, predicts 33 body keypoints using a detector–tracker design intended for real-time, on-device use [10]. MediaPipe Holistic combines the pose, face and hand components and outputs 543 landmarks in total: 33 pose, 468 face and 21 per hand [11]. Morph-Talk uses the face and pose outputs.

C. 3D Avatars, VRM and Avatar-Mediated Communication

Transformed social interaction (TSI) describes collaborative virtual environments in which the rendered behaviour of a participant can be deliberately decoupled from the actual behaviour [3]. In one TSI study, a presenter's actual head movements were transformed by an algorithm to alter perceived gaze, showing that avatar rendering can be driven by tracked head motion [12]. Denser reconstruction approaches also exist: Face2Face tracks the expressions of a webcam-captured source actor with a dense photometric measure and transfers them to a target video in real time [13]. Morph-Talk differs in its target: sparse landmarks drive a stylised avatar instead of re-rendering a photorealistic face.

Expression names such as happiness, sadness, anger and surprise come from the literature on discrete basic emotions [14]. Morph-Talk uses this vocabulary only as a set of avatar expressions; it does not claim to recognise a user's emotional state. VRM is an open file format for humanoid 3D avatars based on glTF 2.0, providing a humanoid bone structure, expressions and metadata [4]. The @pixiv/three-vrm library loads VRM models into Three.js [15], [16] and supports both VRM 1.0 and the earlier version of the format [15].

D. Positioning of This Work

The works above address individual components. Morph-Talk addresses their integration in one browser application, with per-user calibration of facial expressions and handling of both VRM generations. No systematic comparison with existing avatar-based calling products has been carried out, so no comparative claims are made.

III. Proposed System

A. System Overview

Morph-Talk consists of a React.js front end, a Node.js back end using Socket.IO with MongoDB for persistence, and a call connection between the participants' browsers set up with WebRTC and Socket.IO signalling (Fig. 1). Each browser captures its user's webcam and microphone. The webcam video is used only locally: MediaPipe Holistic extracts face and body landmarks from it, and expression, head and arm tracking data are obtained from them. The tracking data, not the webcam video, are sent through Socket.IO to the other participant, whose browser animates the corresponding VRM avatar with Three.js and three-vrm. The participants also hear each other's voices during the call.

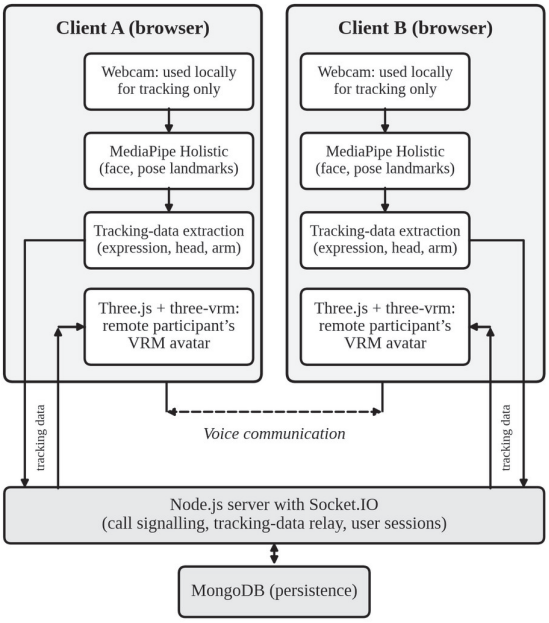


Fig. 1. Architecture of Morph-Talk (illustrative diagram). The webcam is used locally for tracking only; tracking data and signalling pass through the Socket.IO server; the receiving browser renders the avatar; voice communication takes place between the participants.

B. Data Flows and Overall Workflow

Four kinds of information are kept distinct. First, the *local webcam video* is used only inside the user's own browser as input to MediaPipe Holistic; it is neither displayed to nor transmitted to the remote participant. Second, the *tracking data* extracted from the webcam input are relayed to the other participant through Socket.IO. Third, the *animated avatar*: the receiving participant sees the corresponding 3D VRM avatar moving live according to the received tracking data. Fourth, *voice communication*: the participants hear each other's voices during the call. In addition, *signalling* messages (call requests, acceptance or rejection, offer, answer and ICE candidates) pass through the Socket.IO server. The overall workflow, from call request to an active avatar call, is shown in Fig. 2.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 10 , October - 2026

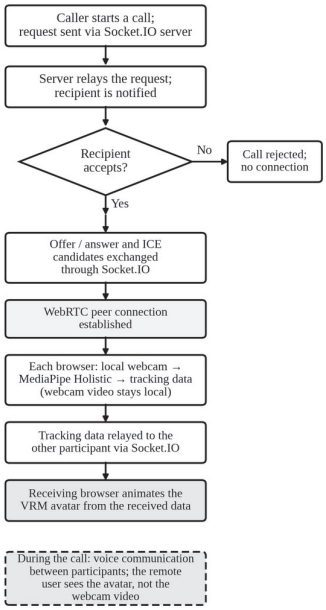


Fig. 2. Overall call workflow of Morph-Talk (illustrative diagram). The remote participant sees the animated avatar, not the webcam video.

C. Hardware and Software Requirements

Table I lists the requirements that follow from the implemented design.

TABLE I. REQUIREMENTS OF THE SYSTEM

Item	Role
Webcam	Local video input for face and body landmark extraction; not shown to the remote participant
Microphone	Audio captured through getUserMedia for voice communication between participants
Web browser	Runs the React.js interface, MediaPipe Holistic, Three.js avatar rendering and the WebRTC call connection; needs a secure origin (HTTPS or localhost)
VRM models	Gender-specific VRM 0.x or 1.0 avatars, loaded directly
Server	Node.js with Socket.IO; MongoDB database
Operating system	Windows audio-device settings can affect microphone routing

D. Technology Stack

Table II lists the technologies used and their responsibilities.

TABLE II. TECHNOLOGY STACK AND RESPONSIBILITIES

Layer	Technology	Responsibility
Front end	React.js	Component-based user interface; call and expression controls
3D rendering	Three.js [16]	Scene and rendering of the avatar
Avatar loading	@pixiv/three-vrm [15]	Loads VRM models; humanoid bone rigging
Tracking	MediaPipe Holistic [11]	Face and pose landmarks from the local webcam input, in the browser
Call	WebRTC [1]	Peer connection between the participants, set up through Socket.IO signalling
Signalling	Socket.IO [7]	Offer, answer and ICE exchange; call setup; relay of tracking data to the other participant
Back end	Node.js	Real-time events; user session management
Database	MongoDB	Data persistence

IV. METHODOLOGY AND SYSTEM IMPLEMENTATION

Fig. 3 shows the processing pipeline from the local webcam input to the animated avatar on the receiving side. The stages are described in the order of their data dependence. Execution rates of the stages were not measured and are not claimed.

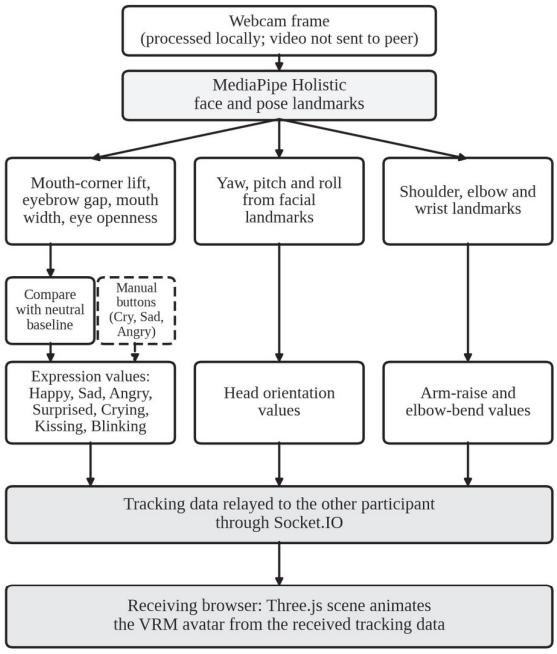


Fig. 3. Facial and body tracking pipeline, from local webcam input to the avatar animated on the receiving side (illustrative diagram). Dashed element: manual user control.

A. Media Capture

The application requests the webcam and microphone with getUserMedia, which requires user permission and is available only in secure contexts, that is, pages served over HTTPS or from localhost [17].

B. Landmark Extraction

Webcam frames are analysed in the browser with MediaPipe Holistic. The face output provides 468 facial landmarks and the pose output provides body landmarks; Morph-Talk uses the face landmarks and the shoulder, elbow and wrist pose landmarks. All extraction is client-side. The webcam video is used only for this local analysis and is not displayed to the remote participant.

C. Facial Expression Tracking and Mapping

Custom logic derives four metrics from the facial landmarks: mouth-corner lift, eyebrow gap, mouth width and eye openness. Each is interpreted relative to a calibrated neutral-face baseline of the individual user. The relative measurements are mapped to the VRM blendshape expressions Happy, Sad, Angry, Surprised, Crying, Kissing and Blinking. Because detection is relative to each user's own neutral face rather than fixed absolute geometry, a resting mouth shape is less likely to be misread as an

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 10 , October - 2026

expression. Manual Cry, Sad and Angry override buttons are also provided, allowing the user to set these expressions directly.

D. Head Movement Tracking

Head yaw, pitch and roll are derived from the facial landmarks and applied to the head bone of the avatar's humanoid skeleton, so that the avatar's head follows the user's head.

E. Upper-Body and Arm Tracking

The shoulder, elbow and wrist pose landmarks drive live arm-raise and elbow-bend animation of the avatar. Table III summarises the three tracked channels. Hand and finger tracking is not part of the documented feature set.

TABLE III. TRACKED CHANNELS AND AVATAR OUTPUT

Channel	Input landmarks	Derived quantity	Avatar output
Face	468 face landmarks	Mouth-corner lift, eyebrow gap, mouth width, eye openness (vs. neutral baseline)	Seven VRM expressions
Head	Facial landmarks	Yaw, pitch, roll	Head-bone rotation
Arms	Shoulder, elbow, wrist	Arm raise, elbow bend	Arm animation

F. VRM Avatar Rendering

A Three.js scene [16] renders the VRM avatar, with gender-specific models, in place of the camera feed: the receiving participant's browser animates the corresponding avatar from the tracking data received through Socket.IO; three-vrm [15] loads the model and exposes its humanoid bones and expressions. VRM models are loaded directly rather than converted to generic 3D formats, because conversion removes the bone and skeleton data needed for animation. Facial tracking results are applied as VRM expressions, and head and arm results as bone rotations. VRM 0.x and 1.0 models face opposite directions by default and have different rest poses; the application therefore detects the specification version of each model at load time and adjusts its rotation and orientation logic.

G. Call Setup and Real-Time Communication

A call begins with a request routed through the server, which the other user accepts or rejects. Socket.IO manages call setup, accept and reject, and ICE negotiation. After acceptance, the offer, the answer and the ICE candidates [5] are exchanged through Socket.IO until the WebRTC peer connection [1] is established. During the call, each browser sends the tracking data extracted from its local webcam input to the other participant through Socket.IO, and the receiving browser animates the corresponding avatar accordingly. The participants also hear each other's voices. The webcam video itself is not displayed to the remote participant.

H. Back End and Database

The Node.js server uses Socket.IO for real-time events and for user session management, and relays tracking data between users. MongoDB provides data persistence. The database schema and any user-authentication mechanism are outside the scope of this paper.

I. Implementation Challenges

Four engineering problems were met and are summarised in Table IV.

TABLE IV. IMPLEMENTATION CHALLENGES

Challenge	Cause	Resolution
VRM version compatibility	VRM 0.x and 1.0 use opposite default facing directions and different rest poses	Specification version detected at load time; rotation and orientation logic adjusted
False expression detection	A resting mouth shape can be misread as "kissing"	Detection calibrated against each user's neutral-face baseline
Secure-context requirement	Browsers allow camera and microphone only on HTTPS or localhost; a LAN IP address silently disabled media access	Application run from localhost
Audio device routing	Windows keeps separate Default Device and Default Communication Device; browsers may use the latter, e.g. a Bluetooth headset instead of the built-in microphone	Cause identified as operating-system device settings

V. RESULTS AND DISCUSSION

A. Functional Testing and Observations

The prototype's features were examined functionally during development. Table V summarises each feature area and the observation recorded for it. Development testing was carried out on localhost and, at one stage, across a local-network IP address, which exposed the secure-context restriction described in Section IV-I. The observations are qualitative; formal test cases, hardware and browser configurations are not reported, and they are not performance or accuracy measurements.

TABLE V. FUNCTIONAL AREAS AND RECORDED OBSERVATIONS

Area	Feature	Recorded observation
Call workflow	Call setup, accept/reject, ICE negotiation; tracking-data relay	Implemented with WebRTC and Socket.IO
Facial tracking	Metrics, calibration, seven expressions, manual overrides	Implemented; calibration adopted to reduce false positives (effect not quantified)
Head tracking	Yaw, pitch, roll on head bone	Implemented
Body tracking	Arm raise and elbow bend	Implemented
Voice	Voice communication between participants	Implemented; Windows device-routing issue identified
VRM rendering	Gender-specific models; VRM 0.x and 1.0	Implemented; version-specific orientation handled
Device access	Camera and microphone permissions	Silent failure over LAN IP; works on localhost

B. What Has Not Been Measured

No quantitative evaluation has been carried out. Frame rate, end-to-end latency, expression-mapping accuracy, pose accuracy, processor and memory use, bandwidth consumption, behaviour under different network conditions

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 10 , October - 2026

and any user study remain to be done. This paper therefore makes no performance, accuracy or comparative claims.

C. Discussion

The prototype shows that an avatar-mediated call can be assembled in a web browser from open components, with landmark extraction performed on the client and therefore without a server-side vision service. Per-user calibration and version-aware VRM handling were the main measures that made the same pipeline usable across users and avatar models. Replacing the camera image with an avatar changes what the remote participant sees, since the webcam video is not shown to them, but it is not a guarantee of anonymity or privacy: the speaker's voice is transmitted and tracking data are relayed. This paper claims neither.

VI. Limitations and Future Scope

Morph-Talk currently supports calls between two users. Its expression set is limited to the seven presets in Section IV-C, and its documented body movement to arm raise, elbow bend and head orientation. Tracking quality under different lighting, camera placement and occlusion, and connectivity across restrictive networks, have not been evaluated. Microphone selection depends on operating-system settings, and use beyond one machine requires a secure origin. User authentication is not part of the documented feature set, and the paper does not describe the database schema. The following items are proposed and are not implemented:

- quantitative evaluation of frame rate, latency and resource use on defined hardware, browsers and networks;
- accuracy evaluation of expression mapping against labelled recordings or independent observers;
- a user study of usability and perceived presence;
- extension of tracking to the hands, whose landmarks MediaPipe Holistic already provides, and to further expressions;
- in-application audio device selection; and
- secure deployment with HTTPS and relay servers for NAT traversal.

VII. Conclusion

Morph-Talk is a browser-based prototype in which two users converse through VRM avatars driven by locally computed facial, head and arm tracking, with Socket.IO relaying the tracking data and handling call signalling, a WebRTC call connection between the participants, and voice communication during the call. The remote participant sees the animated avatar rather than the webcam video. Its features are implemented and the engineering problems met during development are documented, but no quantitative measurements have been made. The work is therefore best read as an implemented prototype and a base for the measurements and user study proposed above.

Acknowledgment

The authors thank the Department of Computer Science and Engineering, GMIT Mandya, for support and guidance.

References

[1] World Wide Web Consortium, "WebRTC 1.0: Real-time communication between browsers," W3C Recommendation, Jan. 2021. [Online]. Available: https://www.w3.org/TR/2021/REC-webrtc-20210126/

[2] H. Alvestrand, "Overview: Real-time protocols for browser-based applications," Internet Engineering Task Force, RFC 8825, Jan. 2021, doi: 10.17487/RFC8825.

[3] J. N. Bailenson, A. C. Beall, J. Loomis, J. Blascovich, and M. Turk, "Transformed social interaction: Decoupling representation from behavior and form in collaborative virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 13, no. 4, pp. 428–441, 2004, doi: 10.1162/1054746041944803.

[4] VRM Consortium, "VRM specification," GitHub repository. [Online]. Available: https://github.com/vrm-c/vrm-specification

[5] A. Keränen, C. Holmberg, and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A protocol for Network Address Translator (NAT) traversal," Internet Engineering Task Force, RFC 8445, Jul. 2018, doi: 10.17487/RFC8445.

[6] I. Fette and A. Melnikov, "The WebSocket protocol," Internet Engineering Task Force, RFC 6455, Dec. 2011, doi: 10.17487/RFC6455.

[7] Socket.IO, "Introduction," Socket.IO documentation, version 4.x. [Online]. Available: https://socket.io/docs/v4/

[8] C. Lugaresi et al., "MediaPipe: A framework for building perception pipelines," arXiv:1906.08172, Jun. 2019.

[9] Y. Kartynnik, A. Ablavatski, I. Grishchenko, and M. Grundmann, "Real-time facial surface geometry from monocular video on mobile GPUs," in *Proc. CVPR Workshop on Computer Vision for Augmented and Virtual Reality*, Long Beach, CA, USA, 2019, arXiv:1907.06724.

[10] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device real-time body pose tracking," in *Proc. CVPR Workshop on Computer Vision for Augmented and Virtual Reality*, 2020, arXiv:2006.10204.

[11] Google, "Holistic landmarks detection guide," MediaPipe Solutions documentation. [Online]. Available: https://developers.google.com/edge/mediapipe/solutions/vision/holistic_landmarker

[12] J. N. Bailenson, A. C. Beall, J. Loomis, J. Blascovich, and M. Turk, "Transformed social interaction, augmented gaze, and social influence in immersive virtual environments," *Human Communication Research*, vol. 31, no. 4, pp. 511–537, 2005, doi: 10.1111/j.1468-2958.2005.tb00881.x.

[13] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2Face: Real-time face capture and reenactment of RGB videos," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 2387–2395, doi: 10.1109/CVPR.2016.262.

[14] P. Ekman, "An argument for basic emotions," *Cognition and Emotion*, vol. 6, no. 3–4, pp. 169–200, 1992, doi: 10.1080/02699939208411068.

[15] pixiv Inc., "@pixiv/three-vrm: Use VRM on three.js," GitHub repository. [Online]. Available: https://github.com/pixiv/three-vrm

[16] three.js, "three.js: JavaScript 3D library." [Online]. Available: https://threejs.org/

[17] Mozilla, "MediaDevices: getUserMedia() method," MDN Web Docs. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices/getUserMedia

(This work is licensed under a Creative Commons Attribution 4.0 International License.)