FLEXA: A Modular and Flexible Framework for AI-Assisted Video Communication

Elton Vivot
eltonvivot@discente.ufg.br
Universidade Federal de Goiás
Goiânia, Goiás, Brazil

Isabela Teixeira
isabela_teixeira@discente.ufg.br
Universidade Federal de Goiás
Goiânia, Goiás, Brazil

Kleber Vieira Cardoso
kleber@ufg.br
Universidade Federal de Goiás
Goiânia, Goiás, Brazil

ABSTRACT

Semantic communication offers a promising paradigm for efficient video transmission. However, existing implementations often rely on specialized, tightly coupled architectures that hinder the integration and comparative evaluation of new Artificial Intelligence (AI) models. In this paper, we present FLEXA, a modular framework for AI-assisted video communication that decouples AI processing from transport logic, enabling simplified integration of new models within a standard Web Real-Time Communication (WebRTC) environment. The architecture introduces policy-driven semantic orchestrators that dynamically manage and chain AI models during transmission. To demonstrate its capabilities, we integrated a Super-Resolution Generative Adversarial Network (SRGAN) and evaluated seven upscaling configurations under different operating policies. Results indicate that one configuration achieves the best balance between perceptual quality and efficiency, with LPIPS < 0.1 and SSIM > 0.7 under moderate processing cost. These findings demonstrate that FLEXA effectively manages different AI-driven policies, revealing an optimal operating point for resolution enhancement and quantifying the distinct performance trade-offs between different operating policies.

KEYWORDS

semantic communication, neural networks, webRTC framework, video transmission

1 INTRODUCTION

The growing demand for high-resolution video streaming across applications like telemedicine, remote collaboration, and immersive entertainment has intensified pressure on network infrastructure [4]. Semantic communication has emerged as a promising approach to optimize resource utilization by transmitting contextual meaning [14]. This paradigm enables significant efficiency gains, as demonstrated by recent innovations: generative reconstruction reduces bandwidth requirements while maintaining perceptual quality, semantic-guided super-resolution enhances visual fidelity under constrained bitrates, and neural error concealment maintains continuity during network disruptions [12, 20, 21].

Specific implementations demonstrate these advantages: WiserVR [21] achieves 36% bandwidth reduction in VR streaming through semantic-region prioritization, SAW [20] enhances streams using diffusion-based semantic super-resolution, and Grace [5] improves loss resilience in real-time neural video coding by jointly training the encoder and decoder under simulated packet losses. Each system achieves high performance within its target domain by integrating specialized Artificial Intelligence (AI) technologies–WiserVR employs attention-guided transmission, SAW combines scalable coding with semantic enhancement, and Grace implements end-to-end neural compression.

However, these implementations face integration constraints due to their specialized architectures. WiserVR builds custom wireless protocols for Virtual Reality (VR), SAW modifies Web Real-Time Communication (WebRTC) [19] with diffusion-specific enhancements, and Grace develops a complete neural codec replacement. This specialization introduces three challenges for semantic communication research: 1) AI techniques become tightly coupled with transport implementations, 2) cross-approach comparison requires reimplementation across disparate systems, and 3) evaluating new models demands significant engineering effort to rebuild transmission infrastructure.

To address these constraints, we propose FLEXA, a modular and flexible framework for AI-assisted video communication that decouples AI processing from transport logic. Unlike domain-specific implementations, our solution introduces policy-driven orchestrators that integrate diverse AI models as configurable components within standard WebRTC [19]. This enables integration and evaluation of semantic techniques without modifying underlying transmission infrastructure. Our contributions include:

(1) Modular Orchestration Architecture: A WebRTC-based framework featuring semantic encoder/decoder orchestrators that dynamically chain AI models through policy-managed pipelines.
(2) System Implementation: implementation of core components including model registration, runtime adaptation, pipeline sequencing, and metrics collection. The source code is available at GitHub Repository. [1]
(3) Case Study Evaluation: Demonstration of the framework's capabilities through integration of Super-Resolution Generative Adversarial Network (SRGAN) for resolution enhancement, evaluated across 7 resolution mappings in a web conferencing scenario.
(4) Quality-Performance Tradeoff Analysis: Assessment of policy-driven quality-latency tradeoffs and bitstream efficiency correlations derived from experimental results.

In: Proceedings of the Brazilian Symposium on Multimedia and the Web (WebMedia'2025). Rio de Janeiro, Brazil. Porto Alegre: Brazilian Computer Society, 2025.

ISSN 2966-2753

[1] https://github.com/LABORA-INF-UFG/paper-EIK-2025

This work is structured as follows. Section 2 reviews existing literature on semantic video communication and positions our framework in contrast to prior work. Section 3 details the proposed system, presenting its conceptual overview, modular architecture, and the interaction between its core components. Section 4 describes the experimental methodology, including the AI model used, its integration into the framework, the simulation setup, and the performance metrics. Section 5 presents and analyzes the results of our evaluation, focusing on quality-performance trade-offs under different operating conditions. Finally, Section 6 discusses the current limitations of the framework, and Section 7 concludes the paper by summarizing our contributions and outlining future research directions.

2 RELATED WORK

Semantic communication has emerged as a strategy to optimize video transmission under bandwidth and reliability constraints. It enables systems to transmit essential or compressed representations of visual data while maintaining intelligibility and usability [14]. Recent works can be grouped by their main technical contributions, as summarized in Table 1.

Multiple studies propose methods for bandwidth reduction by minimizing the amount of data transmitted [8, 9, 12, 13, 16, 17, 21]. [16] and [17] use generative models to reconstruct high-quality images or video from low-resolution or audio inputs, reducing bandwidth demands. The approach in [17] avoids video transmission entirely by generating talking-head video from audio and a reference image. [12] reconstructs corrupted frames without retransmission, indirectly lowering bandwidth use. [13] transmits semantic keypoints and layouts instead of full frames. [21] and [9] transmit semantic segments rather than full data. [8] sends only essential information based on channel conditions.

Some approaches incorporate resolution enhancement. [20] improves visual quality by combining scalable video layers with semantic segmentation for super-resolution reconstruction. [16] performs frame reconstruction by applying super-resolution to a low-resolution base frame, guided by semantic keypoints extracted from the person in the video.

Robustness to network conditions is addressed by [12], [9], and [17]. [5] uses a transmission model that maintains decodability under low-bitrate and adverse network conditions. [9] applies hybrid analog-digital transmission and semantic prioritization to preserve perceptual information when the channel is unstable. [12] introduces a neural error concealment method that uses spatiotemporal priors to reconstruct and repair video frames.

Frame interpolation and generation are addressed by [13], [17], [8], which produce intermediate or full frames using keypoints, semantic layouts, or generative models. Frame segmentation is used in [20], [13], [21], and [9], enabling transmission of semantic regions instead of entire frames.

Some frameworks integrate channel adaptivity to adjust transmission strategies based on network conditions. [13], [21], and [17] implement such adaptation by modifying which content to transmit and by jointly training encoders and decoders to work under multiple conditions.

Semantic communication frameworks have enabled improvements in video transmission, as shown in Table 1, but they often integrate AI models directly with transport-layer components. This design requires modifying codecs or developing new systems to evaluate different AI techniques, which complicates comparative analysis.

To address this, we propose a framework that decouples AI model execution from the communication pipeline. This architecture allows to integrate and evaluate various AI models—for tasks such as frame interpolation, generation, segmentation, repair, or channel adaptivity—without altering the underlying codec infrastructure.

We demonstrate the framework's function by integrating a Generative Adversarial Network (GAN) for bandwidth reduction and resolution enhancement, as indicated by the black circles in Table 1. The gray circles denote capabilities supported by the framework but not implemented in this work.

3 SYSTEM DESIGN

This work proposes a modular video transmission framework designed to simplify the integration of AI models into the communication pipeline. The goal is to enable semantic communication by allowing AI-based processing at different stages of the transmission, without altering the underlying transport logic. The framework supports real-time operation and can incorporate models for tasks such as generation, compression, enhancement, or adaptation.

Figure 1 illustrates the conceptual operation of the framework. On the sender side, the media source is processed by the Semantic Encoder, which uses AI models to identify frame segments or detect new features. Resolution-sensitive segments are transmitted at high resolution, while less relevant regions are sent at lower quality. When new features are detected, a full-resolution reference frame is sent. On the receiver side, the Semantic Decoder reconstructs the final frame by combining received segments, applying upscaling models to enhance resolution, and optionally using interpolation models to increase the frame rate.

3.1 Architecture

The framework is built on WebRTC [19], an open-source library for real-time media transmission over the Real-Time Transport Protocol (RTP) [15]. To support AI-based frame processing and integrate with PyTorch [7], we adopt *aiortc* [1], a Python implementation of WebRTC.

The framework implements two main orchestrators: the Semantic Encoder Orchestrator and the Semantic Decoder Orchestrator. Figure 2 illustrates the FLEXA's internal structure. Each orchestrator contains four core modules:

- Encoder/Decoder Manager: Registers encoders, decoders, and their associated AI models. Multiple encoders or decoders can be available simultaneously, each assigned to specific tasks such as segmentation, reference handling, enhancement, reconstruction, or interpolation. The manager provides these components to the Pipeline Orchestrator for dynamic chaining.
- Policy Manager: Manages policies associated with encoders and decoders, defining the actions to be executed under specific runtime conditions. Policies are implemented as sets of

Feature	[16]	[5]	[12]	[20]	[13]	[21]	[9]	[17]	[8]	FLEXA
Open Source	●	●	○	●	●	○	○	○	○	●
Bandwidth Reduction	●	○	●	○	●	●	●	●	●	●
Resolution Enhancement	○	○	○	●	○	○	○	○	○	●
Robustness to Network Conditions	○	●	●	○	○	○	●	○	○	◐
Frame Interpolation	○	○	○	○	●	○	○	○	●	◐
Frame Generation	●	○	○	○	○	○	○	●	●	◐
Frame Segmentation	○	○	○	●	●	●	●	○	○	◐
Frame Repair	○	○	●	○	○	○	○	○	○	◐
Channel Adaptivity	○	○	○	○	●	●	○	●	○	◐

Table 1: Feature comparison of recent semantic and neural communication frameworks for video and audio transmission.

rules that use feedback from the Monitor Module, such as pipeline execution time, to decide processing steps. For example, if a delay threshold is exceeded, a policy may skip certain model stages, process the frame without neural enhancement, or discard the frame and display the last processed one. Policies can be general, applying to the entire pipeline, or specific to individual models. All policies are defined before transmission starts. They are created programmatically using functions provided by the FLEXA codebase, which allow developers to define and register new runtime behaviors [2]. During transmission, the Policy Manager checks the predefined policies for every processed frame and applies the corresponding actions dynamically according to current runtime conditions. New policies can be created as needed, enabling adaptive and extensible control of data processing during transmission.

- Pipeline Orchestrator: Determines the sequence of encoders or decoders to be applied for frame processing, based on the components registered in the system and the currently active policies. On the sender side, for example, the pipeline may include a Preparation Encoder, a Segmentation Encoder, and a Reference Encoder. On the receiver side, it may include a Reconstruction Decoder, an Enhancer Decoder, and an Interpolation Decoder. The Pipeline Orchestrator ensures that these components interoperate correctly, managing data flow between them, adjusting execution order, and activating or deactivating modules in response to policy decisions.
- Monitor Module: Collects runtime statistics and feedback from the transmission process. It tracks metrics such as execution time per pipeline stage, triggered policies, active encoders/decoders, media stream size, per-model processing time, and generated frames. Metric collection is fully customizable to include only data relevant for evaluation or adaptation.

3.2 Module Interaction in Transmission

Figure 3 illustrates the end-to-end module interactions during transmission. On the sender side, the Media Source provides video frames to the Pipeline Orchestrator, which coordinates the encoding process. The Encoder Manager loads the required encoders and associated AI models, while the Policy Manager applies attached policies by evaluating runtime conditions. The encoders then process the frame according to the configured sequence and policy constraints. The processed output is forwarded to WebRTC for RTP packetization and network transmission.

On the receiver side, WebRTC reconstructs the transmitted frame from packetized data and delivers it to the Pipeline Orchestrator. The Decoder Manager loads appropriate decoders and AI models, while the Policy Manager evaluates receiver-side conditions and enforces corresponding policies. The decoders process the frame according to the defined pipeline configuration, after which the final output is rendered by the Media Renderer. This architecture maintains WebRTC compatibility while enabling dynamic reconfiguration of AI processing components through policy-driven adaptation, without modifications to the underlying transport layer.

4 EVALUATION

We evaluate our framework in a web conference simulation using the SRGAN [11] as the resolution enhancement model within the proposed semantic video transmission architecture. This section is organized as follows: Section 4.1 describes the SRGAN model; Section 4.2 details its integration into the Encoder/Decoder Managers and policies; Section 4.3 presents the simulation environment and experimental setup; and Section 4.4 defines the metrics used for quality and performance assessment.

4.1 SRGAN

SRGAN is a deep learning model designed to reconstruct high-resolution images from low-resolution inputs. Its generator consists of 16 residual blocks with upsampling layers, trained using adversarial loss and perceptual loss computed from VGG19 features.

The model was trained on the Flickr-Faces-HQ (FFHQ) dataset [10], which contains human face images, using alternating optimization between the generator and discriminator. Inputs were normalized to $[-1, 1]$, and the loss combined perceptual and adversarial components.

Training was performed for 10,000 epochs across seven resolution upscaling mappings: i) 64×64 to 128×128, ii) 64×64 to 256×256,

[2] Additional details in the FLEXA's documentation at GitHub repository.

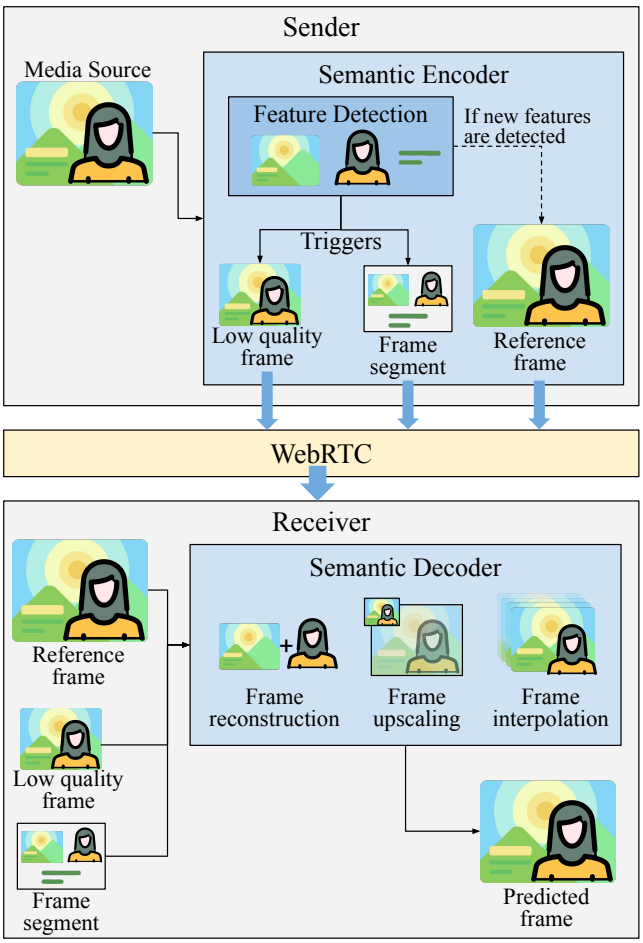

Figure 1: FLEXA's conceptual workflow. The sender encodes low-quality video frames using a Semantic Encoder that detects visual features and, based on trigger conditions, generates either reference frames or frame segments. At the receiver side, a Semantic Decoder reconstructs, interpolates, and upscales the frames using the received reference data, enabling adaptive AI-driven enhancement within a standard WebRTC transmission pipeline.

iii) 64×64 to 512×512, iv) 64×64 to 1024×1024, v) 128×128 to 1024×1024, vi) 256×256 to 1024×1024, and vii) 512×512 to 1024×1024.

4.2 Semantic Integration

On the sender side, the Semantic Encoder Orchestrator was configured using the framework's modular architecture. The Encoder Manager registered a Preparation Encoder responsible for downscaling frames to specified transmission resolutions (64×64 to 512×512). The Policy Manager enforced input policies that defined the target resolution parameter, which was set during transmission initialization. During operation, the Pipeline Orchestrator first validated the

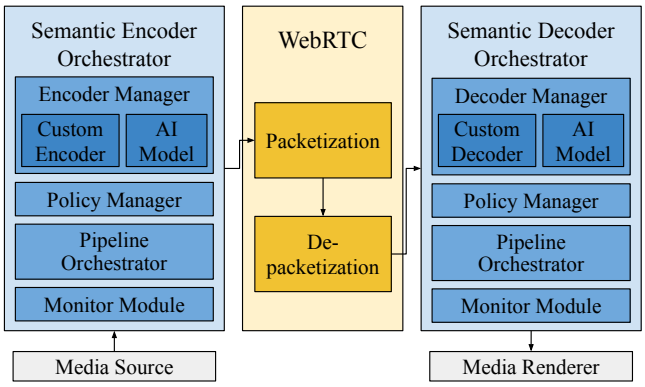

Figure 2: FLEXA's internal architecture integrated with the WebRTC layer. Each orchestrator includes managers, policy controllers, and AI model interfaces coordinated by a pipeline orchestrator and a monitoring module. The data flow passes through encoder or decoder managers that dynamically select AI models according to predefined policies, enabling modular and adaptive semantic processing. Transmission occurs through WebRTC packetization and depacketization stages.

active resolution policy, then directed frames through the Preparation Encoder for resizing before packetization via WebRTC. The Monitor Module tracked the encoded VP8 [3] bitstream values.

On the receiver side, the Decoder Manager instantiated a specialized SRGAN Decoder capable of upscaling frames to various target resolutions, dynamically loading appropriate models and weights based on the active output policy. The Policy Manager implemented four interconnected policies: 1) An Output Resolution policy defining the target upscale resolution, 2) A Threshold policy monitoring pipeline execution delay, 3) A Skipping policy that bypassed SRGAN processing to directly output low-resolution frames when the delay exceeded 300ms, and 4) A Freezing policy that discarded current frames and repeated the last upscaled frame. The Pipeline Orchestrator coordinated these policies to maintain real-time constraints–frame dropping or skipping gradually decreases processing delay until the Threshold policy permitted AI upscaling again.

4.3 Experimental Setup

Evaluation was performed in a simulated environment using a 9-second video of a woman in a typical web conferencing context. The simulation was repeated for all seven resolution mappings and for both Skipping and Freezing policies. Additionally, we included an Unconstrained scenario, in which it is assumed that the SRGAN always has enough time to enhance every received frame, regardless of processing delay.

Sender and receiver components were executed on the same physical machine to eliminate network variability. The system used an AMD Ryzen 7 5700X3D CPU, an NVIDIA GeForce RTX 4070 Super GPU, and 32 GB of RAM. Output videos were saved by the receiver for offline metric computation.

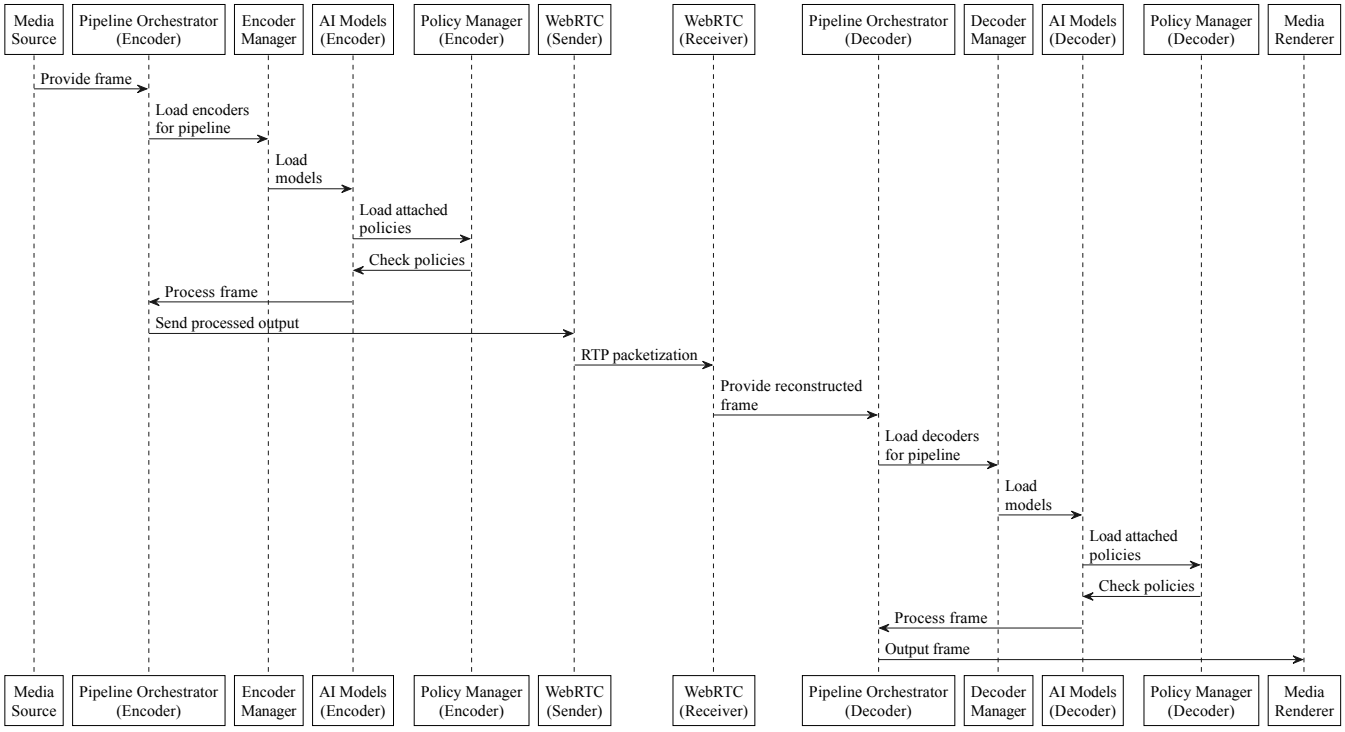

Figure 3: Sequence diagram of the encoder and decoder pipelines. The figure illustrates the end-to-end data flow between sender and receiver. On the sender side, the Pipeline Orchestrator coordinates the Encoder Manager, AI models, and Policy Manager to process each frame according to predefined policies before transmission via WebRTC. On the receiver side, a corresponding Decoder Orchestrator loads the relevant models and policies to reconstruct, enhance, and render the video stream.

4.4 Metrics

Quality metrics included Structural Similarity Index Measure (SSIM) [18], Peak Signal-to-Noise Ratio (PSNR) [6], and Learned Perceptual Image Patch Similarity (LPIPS) [22], computed by comparing the processed video with the original. Performance metrics included bitstream and frame enhancement time.

These metrics were collected in real time by the Monitor Module. Through its configurable data collection feature, only the relevant metrics for each experiment were recorded.

5 RESULTS

This section presents experimental results evaluating the semantic video transmission framework with SRGAN enhancement. We analyze two primary configurations: 1) upscaling fixed-resolution inputs (64×64) to varying target resolutions, and 2) enhancing variable input resolutions to a fixed 1024×1024 output. Quality is assessed using three metrics:

- PSNR: Higher values indicate better quality (ideal: ∞)
- SSIM: Higher values indicate better quality (ideal: 1.0)
- LPIPS: Lower values indicate better perceptual quality (ideal: 0)

These metrics are compared across three operating modes: Unconstrained, Freezing, and Skipping. Results reveal trade-offs between visual quality, processing time, and transmission efficiency.

5.1 Upscaling from 64x64 to varying resolutions

Figure 4 demonstrates a consistent degradation in quality metrics as the upscaling factor increases from 128×128 to 1024×1024. The Unconstrained condition maintains superior performance across all resolutions, while the Skipping policy outperforms Freezing at higher target resolutions (512×512 and above). This advantage stems from Skipping's preservation of temporal consistency - though lower in resolution, each frame reflects current content, whereas Freezing repeats outdated frames during overload, introducing noticeable motion artifacts. Notably, LPIPS (perceptual similarity) exhibits greater variance than pixel-based metrics at extreme upscaling factors, highlighting SRGAN's instability when generating high-frequency details from minimal input data.

Figure 5 relates quality to processing time. The 64×64→256×256 configuration yields the highest PSNR-per-time, suggesting a practical limit for real-time use. Skipping improves LPIPS-per-time over Freezing at 1024×1024 because it avoids additional latency accumulation in the enhancement stage. Beyond 512×512, normalized quality tends to plateau while processing time continues to increase, which indicates diminishing returns for larger targets.

5.2 Upscaling to 1024x1024 from varying inputs

In Figure 6, Skipping sometimes achieves higher PSNR and SSIM than the Unconstrained mode. For 512×512→1024×1024, the median PSNR is 34.2 for Skipping versus 32.5 for Unconstrained. This

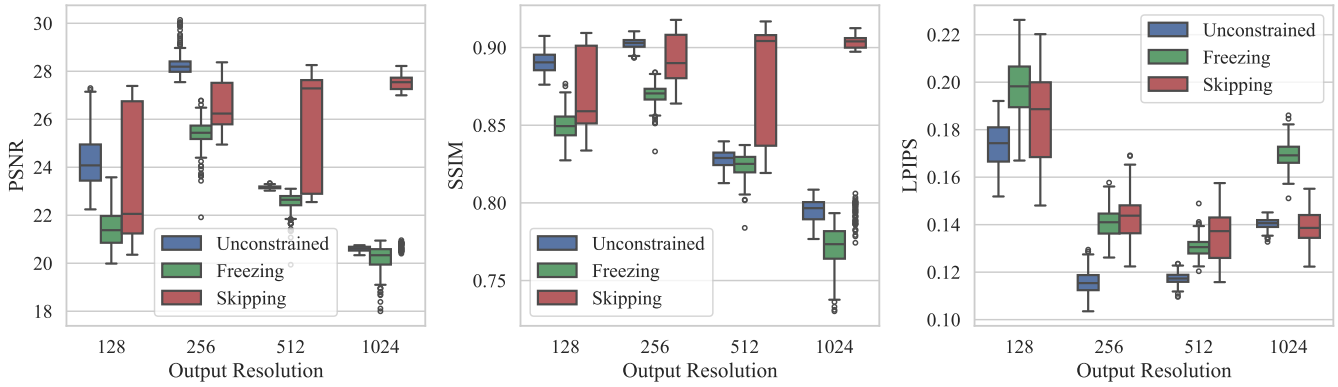

Figure 4: Quality metrics for 64×64 inputs upscaled to target resolutions.

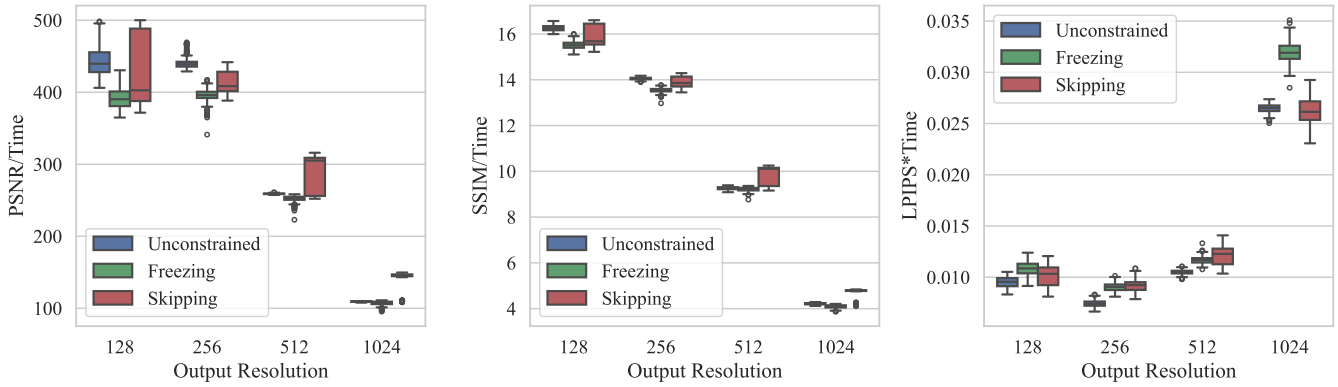

Figure 5: Quality–time efficiency for 64×64 inputs upscaled to target resolutions.

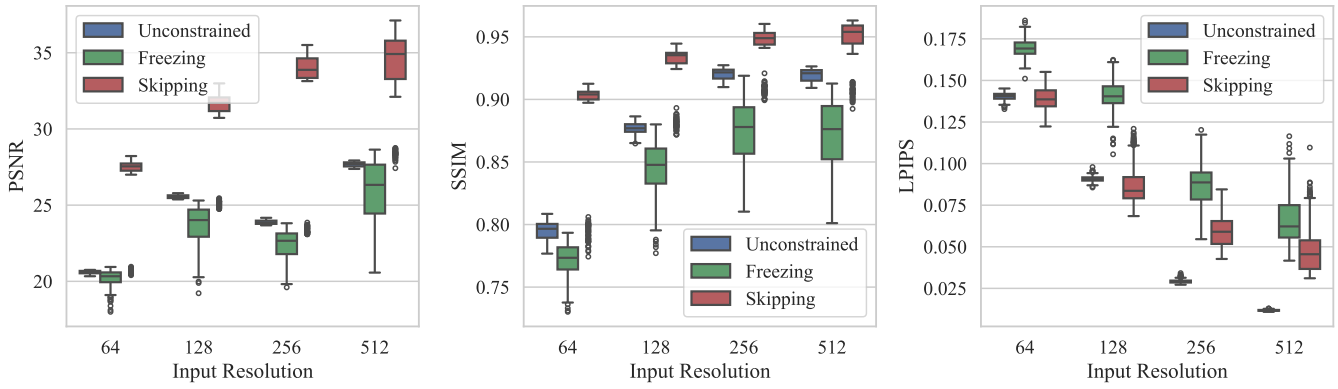

Figure 6: Quality metrics for fixed 1024×1024 output from different input resolutions.

arises because the generative component in SRGAN introduces pixel-level deviations that reduce full-reference scores even when the perceptual impression improves. LPIPS captures this behavior: Unconstrained reaches lower LPIPS (0.042) than Skipping (0.068), indicating better perceptual similarity despite lower PSNR/SSIM. Increasing the source resolution narrows the gap between modes; for instance, the SSIM difference for 256×256→1024×1024 is smaller than for 64×64→1024×1024, which indicates that SRGAN benefits from richer input content.

Figure 7 analyzes quality relative to bitstream size. Inputs of 256×256 and 512×512 produce similar LPIPS-per-size because VP8 compresses both with comparable efficiency. The upscaling of

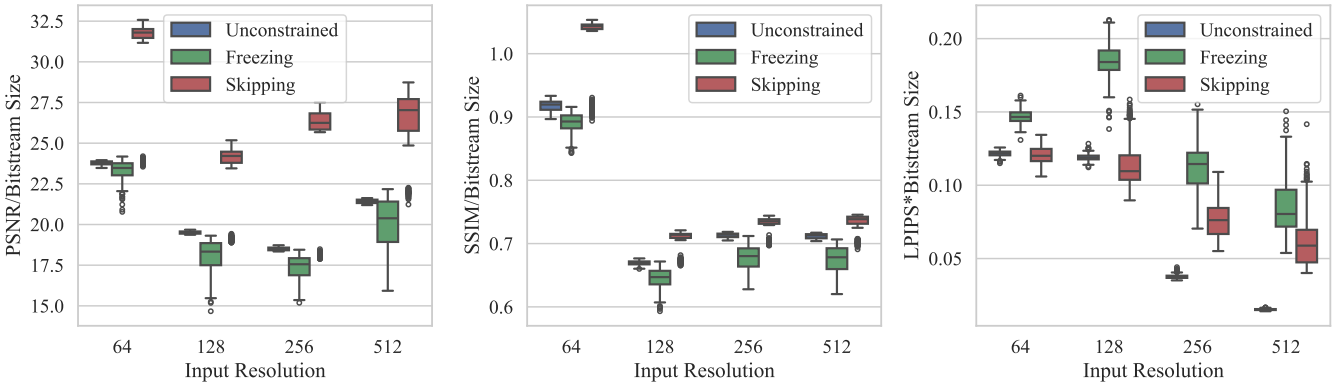

Figure 7: Quality–bitstream efficiency. Metrics are normalized by transmitted frame size.

128×128→1024×1024 path gives the best SSIM-per-size trade-off (about 0.87 per MB), surpassing 512×512→1024×1024 by roughly 23% even though its absolute SSIM is lower. At very low inputs (64×64→1024×1024), Skipping yields about 2.1× higher PSNR-per-size than Freezing, which makes it preferable when bandwidth is constrained.

5.3 Key Insights

Across both configurations, the 256×256→1024×1024 setting offers a balanced operating point: LPIPS remains below 0.1 for most sequences, SSIM normalized by size exceeds 0.7, and dispersion is low. Policy selection should depend on the operating goal. Skipping is appropriate when bandwidth or pixel-accurate metrics dominate the objective; Freezing is useful to mask short overloads when maintaining a continuous stream is more important than enhancing every frame; Unconstrained operation suits cases where latency is not critical and the goal is the best perceptual reconstruction. Finally, PSNR and SSIM do not always track perceptual gains from GAN-based enhancement, so LPIPS should be considered the primary criterion for perceptual evaluation, with PSNR and SSIM reported for completeness.

6 LIMITATIONS

While the proposed framework enables flexible integration of AI models into real-time video transmission, two primary limitations merit consideration for practical deployment. First, the architecture assumes AI models process data in standard video frame formats (e.g., RGB pixel arrays). Models generating non-visual outputs—such as autoencoders [2] producing latent representations, feature maps, or semantic embeddings—require custom packetization schemes to interface with WebRTC's transport layer. Implementing such schemes would necessitate modifications to the WebRTC integration layer, including developing specialized serialization/deserialization handlers, which falls outside the current implementation scope.

Second, the framework lacks native support for temporal models requiring multi-frame contexts (e.g., optical flow networks, 3D convolutional networks, or recurrent video processors). Although the modular design could accommodate such models through custom components, this would demand per-model implementation of: 1) Frame buffering and synchronization logic, 2) Inter-frame dependency management, and 3) Delay compensation mechanisms without centralized coordination. Future extensions could address this through a pipeline buffer module at the orchestrator level, which would manage frame histories and temporal dependencies holistically while maintaining policy-controlled latency bounds. This extension would enable temporal processing without requiring fragmented per-component solutions.

7 CONCLUSIONS

This work explored the application of semantic communication to video transmission systems, for which we proposed a modular WebRTC framework. Policy-driven orchestrators with configurable encoder/decoder managers were implemented in both sender and receiver components, which enable dynamic chaining of diverse AI models without modifying the underlying transmission infrastructure. Furthermore, system implementation and experimental validation were conducted, demonstrating quantifiable quality-performance tradeoffs. We anticipate this framework will serve as a foundation for flexible evaluation of semantic techniques in resource-constrained video communication environments.

Future work will focus on three key extensions: First, we will develop packetization schemes for non-visual semantic representations (e.g., latent embeddings) and implement temporal buffering subsystems to support multi-frame AI models. Second, we will expand the framework's applicability by integrating advanced semantic techniques – including neural compression and context-aware transmission – and evaluate them under diverse network scenarios. Third, we will establish joint encoder-decoder coordination protocols that leverage bidirectional metadata exchange to dynamically optimize processing pipelines based on end-to-end semantic requirements and resource constraints.

Acronym	Definition
AI	Artificial Intelligence
FFHQ	Flickr-Faces-HQ
GAN	Generative Adversarial Network
LPIPS	Learned Perceptual Image Patch Similarity
PSNR	Peak Signal-to-Noise Ratio
RTP	Real-Time Transport Protocol
SSIM	Structural Similarity Index Measure
SRGAN	Super-Resolution Generative Adversarial Network
VR	Virtual Reality
WebRTC	Web Real-Time Communication

Table 2: Summary of acronyms.

ACKNOWLEDGMENTS

This work was supported by FAPEG (Fundação de Amparo à Pesquisa do Estado de Goiás), by RNP (Rede Nacional de Ensino e Pesquisa)/ MCTIC (Ministério da Ciência, Tecnologia e Inovações) under grant no. 01245.020548/2021-07 (Brasil 6G project), by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) under grant no. 306283/2025-5, and by MCTIC/CGI.br/FAPESP (São Paulo Research Foundation) under grant no. 2020/05127-2 (SAMURAI project).

REFERENCES

[1] aiortc. 2018. aiortc. https://github.com/aiortc/aiortc. Accessed: 2025-08-11.
[2] Dor Bank, Noam Koenigstein, and Raja Giryes. 2023. *Autoencoders.* Springer International Publishing, Cham, 353–374. doi:10.1007/978-3-031-24628-9_16
[3] James Bankoski, John Koleszar, Lou Quillio, Janne Salonen, Paul Wilkins, and Yaowu Xu. 2011. *Vp8 data format and decoding guide.* Technical Report. Google Inc.
[4] Niklas Blum, Serge Lachapelle, and Harald Alvestrand. 2021. Webrtc: Real-time communication for the open web platform. *Commun. ACM* 64, 8 (2021), 50–54.
[5] Yihua Cheng, Ziyi Zhang, Hanchen Li, Anton Arapin, Yue Zhang, Qizheng Zhang, Yuhan Liu, Kuntai Du, Xu Zhang, Francis Y. Yan, Amrita Mazumdar, Nick Feamster, and Junchen Jiang. 2024. GRACE: loss-resilient real-time video through neural codecs. In *Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation (NSDI'24).* USENIX Association, USA, 509–531.
[6] Alain Horé and Djemel Ziou. 2010. Image Quality Metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition.* IEEE, Turkey, 2366–2369.
[7] Sagar Imambi, Kolla Bhanu Prakash, and G. R. Kanagachidambaresan. 2021. *PyTorch.* Springer International Publishing, Cham, 87–104. doi:10.1007/978-3-030-57077-4_10
[8] Peiwen Jiang, Chao-Kai Wen, and Shi Jin. 2022. Adaptive Semantic Video Conferencing for OFDM Systems. In *2022 IEEE 32nd International Workshop on Machine Learning for Signal Processing (MLSP).* IEEE, China, 1–6.
[9] Peiwen Jiang, Chao-Kai Wen, Shi Jin, and Geoffrey Ye Li. 2022. Wireless semantic communications for video conferencing. *IEEE Journal on Selected Areas in Communications* 41, 1 (2022), 230–244.
[10] Tero Karras, Samuli Laine, and Timo Aila. 2019. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, USA, 4401–4410.
[11] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* IEEE, USA, 4681–4690.
[12] Tianhong Li, Vibhaalakshmi Sivaraman, Pantea Karimi, Lijie Fan, Mohammad Alizadeh, and Dina Katabi. 2024. Reparo: Loss-Resilient Generative Codec for Video Conferencing. arXiv:2305.14135 [cs.NI] https://arxiv.org/abs/2305.14135
[13] Chengsi Liang, Xiangyi Deng, Yao Sun, Runze Cheng, Le Xia, Dusit Niyato, and Muhammad Ali Imran. 2023. VISTA: Video Transmission over A Semantic Communication Approach. In *2023 IEEE International Conference on Communications Workshops (ICC Workshops).* IEEE, Italy, 1777–1782.
[14] Xuewen Luo, Hsiao-Hwa Chen, and Qing Guo. 2022. Semantic communications: Overview, open issues, and future research directions. *IEEE Wireless communications* 29, 1 (2022), 210–219.
[15] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. 2003. *RFC3550: RTP: A Transport Protocol for Real-Time Applications.* Technical Report. Blue Coat Systems Inc., USA.
[16] Vibhaalakshmi Sivaraman, Pantea Karimi, Vedantha Venkatapathy, Mehrdad Khani, Sadjad Fouladi, Mohammad Alizadeh, Frédo Durand, and Vivienne Sze. 2024. Gemino: Practical and Robust Neural Compression for Video Conferencing. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24).* USENIX Association, Santa Clara, CA, 569–590.
[17] Haonan Tong, Haopeng Li, Hongyang Du, Zhaohui Yang, Changchuan Yin, and Dusit Niyato. 2025. Multimodal Semantic Communication for Generative Audio-Driven Video Conferencing. *IEEE Wireless Communications Letters* 14, 1 (2025), 93–97.
[18] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.
[19] WebRTC. 2025. WebRTC. https://webrtc.org/. Accessed: 2025-08-11.
[20] Yihan Wen, Zheng Zhang, Jiayi Sun, Jinglei Li, Chung Shue Chen, and Guanchong Niu. 2025. SAW: Semantic-Aware WebRTC Transmission Using Diffusion-Based Scalable Video Coding. *IEEE Internet of Things Journal* 12, 5 (2025), 5346–5359.
[21] Le Xia, Yao Sun, Chengsi Liang, Daquan Feng, Runze Cheng, Yang Yang, and Muhammad Ali Imran. 2023. WiserVR: Semantic communication enabled wireless virtual reality delivery. *IEEE Wireless Communications* 30, 2 (2023), 32–39.
[22] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* IEEE, USA, 586–595.