GenStream: Semantic Streaming Framework for Generative Reconstruction of Human-centric Media

ABSTRACT

Video streaming dominates global internet traffic, yet conventional pipelines remain inefficient for structured, human-centric content such as sports, performance, or interactive media. Standard codecs re-encode entire frames, foreground and background alike, treating all pixels uniformly and ignoring the semantic structure of the scene. This leads to significant bandwidth waste, particularly in scenarios where backgrounds are static and motion is constrained to a few salient actors. We introduce GenStream, a semantic streaming framework that replaces dense video frames with compact, structured metadata. Instead of transmitting pixels, GenStream encodes each scene as a combination of skeletal keypoints, camera viewpoint parameters, and a static 3D background model. These elements are transmitted to the client, where a generative model reconstructs photorealistic human figures and composites them into the 3D scene from the original viewpoint. This paradigm enables extreme compression, achieving over 99.9% bandwidth reduction compared to HEVC. We partially validate GenStream on Olympic figure skating footage and demonstrate potential high perceptual fidelity under minimal data. Looking forward, GenStream opens new directions in volumetric avatar synthesis, canonical 3D actor fusion across views, personalized and immersive viewing experiences at arbitrary viewpoints, and lightweight scene reconstruction, laying the groundwork for scalable, intelligent streaming in the post-codec era.

1 INTRODUCTION

Video streaming plays a pivotal role in digital media consumption and accounts for the majority of global Internet data traffic [1]. Specifically, major live international events such as the Olympic Games attract millions of viewers worldwide, generating massive volumes of data as high-definition video streams are delivered simultaneously to heterogeneous devices [2]. With the growing appetite for immersive and interactive content, the limitations of existing video streaming paradigms are becoming increasingly evident.

HTTP Adaptive Streaming (HAS) has become the de facto standard for video delivery over the Internet, dynamically adjusting video quality based on real-time network conditions [3, 4]. In HAS, video content is partitioned into segments of fixed duration (*e.g.*, 2 to 10 seconds [5]), each encoded independently using a video codec such as Advanced Video Coding (AVC) [6] or its successor, High Efficiency Video Coding (HEVC) [7]. To support adaptive switching, each segment is encoded into several quality representations, where each representation is defined by a specific bitrate and resolution pair, collectively referred to as the bitrate ladder.

While modern codecs are effective at reducing redundancy through intra-frame (spatial) and inter-frame (temporal) compression [6, 7], they operate under strict constraints: each segment is independently encoded and, hence, can only reference frames within its temporal window, limiting inter-frame prediction to short-term dependencies [8]. This means that when long-term repetitive patterns occur, the codec repeatedly re-encodes similar content, failing to capitalize on the inherent structural redundancy [9].

A compelling example of such inefficiency can be observed in sporting events, where athletes move against a generally static or slowly evolving background, such as a stadium, lighting fixtures, and surrounding audience. Despite the structural consistency of such scenes, conventional codecs repeatedly re-encode the same portions of background every time the camera passes over them. This redundancy issue is further exacerbated in multi-camera productions, where identical scenes captured from different viewpoints are encoded independently, squandering opportunities to exploit cross-view redundancy [10, 11].

To address these inefficiencies, we propose GenStream, a visionary system designed for live streaming that redefines how such events can be streamed and experienced. Instead of transmitting full video frames, GenStream captures skeleton-based keypoints (*e.g.*, body landmarks) of the performers with their relative bounding box and sends this compact representation to the client. The server then tackles a critical challenge: estimating the original camera viewpoint for each frame, which is essential for replicating the broadcast viewpoint. This pose, along with the performer keypoints, is sent to the client. At the client side, a second core challenge arises: generating a photorealistic rendering of the performer from sparse input using a generative model such as a conditional Generative Adversarial Network (cGAN) [12] or diffusion model [13]. The synthesized figure is then composited into a pre-scanned 3D model of the venue, and rendered from the provided viewpoint to faithfully recreate the original scene.

This paradigm shift offers dramatically reduced bandwidth usage, and support for personalized viewing experiences, such as a stylistic customization of the performers or points of view that differ from what the real cameras are capturing.

According to our preliminary results, GenStream can reduce the required live streaming bitrate to less than 1% that of a baseline HEVC-encoded video, assuming the 3D representation of the venue and generative model weights are pre-distributed to clients. This impressive reduction in bandwidth enables a stall-free streaming experience at the massive scale required for worldwide sporting events, even under constrained or fluctuating network conditions. At the same time, GenStream preserves high visual fidelity through client-side synthesis, albeit at the expense of increased computational complexity.

2 RELATED WORK

2.1 Traditional Video Streaming and Codecs

Video streaming today is predominantly driven by standard protocols such as HTTP Adaptive Streaming (HAS) [14], which underlie systems like HLS [15] and MPEG-DASH [16]. These methods divide content into short segments and offer multiple quality tiers to allow the client to adapt playback to current network conditions.

Combined with client-side adaptive bitrate (ABR) logic, these systems ensure playback continuity and have been highly successful in scaling video delivery across heterogeneous networks and devices.

Compression in these pipelines is handled by increasingly sophisticated codecs, such as AV1 [17] and VVC (H.266) [18], which improve upon earlier standards in terms of compression efficiency, often achieving up to 50% savings. These advances are achieved through better motion prediction, transform coding, and entropy models. However, such gains come at a significant computational cost, especially on the encoder side, with high-resolution or real-time use cases facing increased energy and latency burdens [17, 18].

Despite decades of progress, traditional codecs treat all pixels equally and operate at the block level, agnostic to scene semantics. As a result, even small improvements in compression now require disproportionate increases in complexity, yielding diminishing returns for practical deployment.

2.2 Emerging Architectures for Video Encoding

To address some of the inefficiencies of standard codecs, recent research has explored novel encoding paradigms that move beyond traditional block-based compression. One approach augments existing codecs with machine learning components to improve rate control, motion estimation, or mode decisions [19, 20]. These techniques often integrate seamlessly with legacy pipelines, but remain bound by the structure of the original codec.

A more radical line of work proposes fully neural video compression systems, in which residuals and motion fields are learned end-to-end by deep networks [20, 21]. While promising in terms of visual quality, these models are often too computationally intensive for real-time use or deployment on resource-limited clients.

Complementary to these efforts, other works focus on client-side enhancement techniques that improve the subjective quality of the received video. Methods such as frame interpolation [22, 23] and super-resolution [24–26] can increase perceived temporal or spatial fidelity without increasing the transmitted bitrate.

More recently, data-driven representations like Neural Radiance Fields (NeRF) [27] and 3D Gaussian Splatting (3DGS) [28] have opened new possibilities for compact and continuous scene representations. While typically used for static scene modeling, they provide new pathways for transmitting visual content as a structured function of space and viewpoint.

However, these emerging techniques all share a common trait: they treat video as a dense signal composed of low-level features such as pixels, motion vectors, or residuals. They do not leverage higher-level understanding of scene composition, such as objects, or actions, and thus miss a key opportunity for more structured and efficient encoding.

2.3 Semantic Streaming and Object-Centric Representations

Semantic video streaming shifts the encoding focus from raw pixels to meaningful scene components [29]. This perspective recognizes that not all regions of a video are equally important: agents (such as humans, vehicles, or other active foreground elements) are typically of greater perceptual and narrative significance than background elements. By decoupling foreground and background and encoding them differently, streaming systems can better match perceptual salience to bitrate allocation.

To enable semantic encoding, a variety of tools have emerged. Object detection and tracking algorithms allow for persistent identification of foreground elements across time [30, 31]. Instance segmentation networks can further delineate objects from their surroundings, enabling region-specific treatment during encoding [32]. These components form the foundation for object-centric video understanding.

While these techniques have seen some integration in video conferencing systems, where avatar-based transmission and background substitution are common, their use remains tightly scoped to face-to-face communication scenarios. In broader video streaming, particularly for entertainment, sports, or interactive media, semantic approaches are largely absent. The vast majority of streaming systems continue to transmit raw pixels rather than structured representations, leaving the potential of object-level encoding underutilized.

2.4 Generative Reconstruction

Generative models have recently demonstrated remarkable capabilities in synthesizing photorealistic video content from abstract inputs such as pose skeletons, semantic maps, or depth [33, 34]. This progress has been fueled by advances in both image and video generation, ranging from GAN-based architectures to diffusion models, and has enabled applications in human reanimation, style transfer, and controllable video generation [12, 13, 25, 35].

In the context of streaming, these models offer an opportunity to decouple the representation of motion and appearance. Instead of transmitting dense pixel data, a system may instead send a sparse representation, such as 2D keypoints or depth maps, and rely on a generative model at the client to reconstruct the full image content. This approach opens the door to radically more efficient pipelines.

Early explorations into this idea include ELVIS [36], which uses semantic segmentation to identify low-priority regions of the video, removes them prior to network transmission, and performs generative inpainting at the client side to recreate them. While effective at avoiding the transmission of unimportant areas, ELVIS still relies on traditional encoding for important foreground objects and does not offer a fully object-based or motion-driven reconstruction strategy.

To overcome these limitations and develop a truly semantic streaming system, future efforts can draw from advances in camera motion estimation and 3D scene reconstruction. Accurate tracking of camera viewpoint [37–39] can enable dynamic viewpoint control on the client. Meanwhile, neural scene representations allow the static background to be precomputed and encoded efficiently as a view-consistent 3D asset. When combined with semantic foreground representations, these modalities enable a modular and compositional approach to streaming where agents, environments, and motion can each be transmitted and reconstructed independently.

3 PROBLEM DESCRIPTION AND MOTIVATION

To evaluate the effectiveness of our approach in terms of video quality and bandwidth reduction, we first establish a baseline by

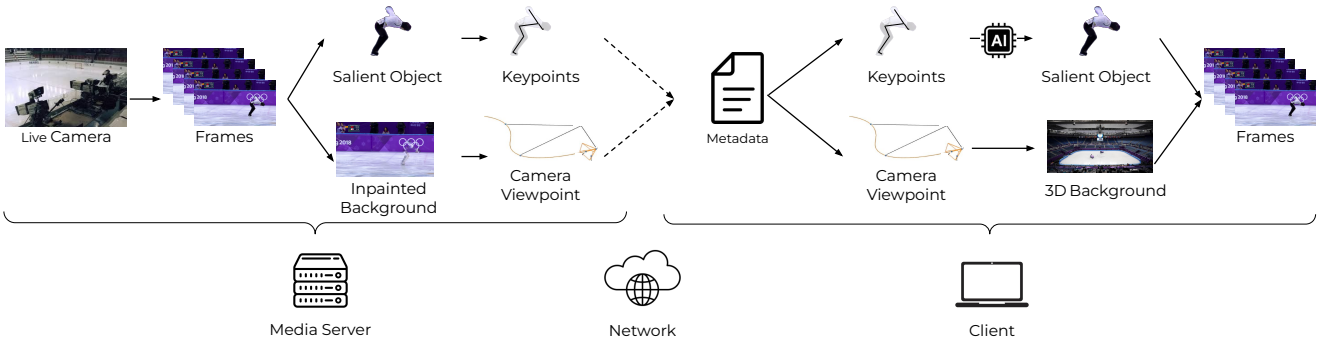

Figure 1: Conceptual architecture of GenStream.

measuring the performance of current state-of-the-art codecs under constrained bandwidth conditions. We apply this comparison to a representative 1080p video scene from an artistic ice skating competition, recorded at a frame rate of 30 fps and 50 s in duration. As our reference codec, we select the High-Efficiency Video Codec (HEVC) [7] due to its widespread adoption [40] and high compression efficiency [7]. The target video [41] is encoded at a resolution of 1920×1080 with a bitrate of 5.8 Mbps.

Artistic ice skating competitions present a compelling opportunity for compression via semantic and structural decomposition. A typical broadcast sequence features: *(i)* a static or slowly evolving background (the ice rink, boards, and stadium interior), and *(ii)* one or two foreground objects of interest (an individual skater or a couple of skaters).

Rather than encoding full video frames pixel-by-pixel, the scene can be decomposed into a persistent 3D background and a set of dynamic pose-driven elements. Therefore, assuming we possess the 3D point cloud of the stadium, to reconstruct the scene, we must first estimate the camera viewpoint T from the input video scene [42]:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \in \mathbb{R}^{4\times4}$$

Here, $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the rotation matrix, and $\mathbf{t} \in \mathbb{R}^3$ is the camera center in world coordinates. The bottom row $[\mathbf{0}^\top 1]$ makes it a 4×4 homogeneous transformation matrix.

This 3D scene needs to be rendered at the client side within the Unity [43] framework. Since Unity represents any rotation via a quaternion $\mathbf{q} \in \mathbb{R}^4$, which represents a mathematically convenient alternative to the euler angle representation, we can convert $\mathbf{R}$ to $\mathbf{q}$ using quaternion algebra [44]. This way, we reduce the required values to represent $\mathbf{T}$ from 12 (3 for $\mathbf{t}$ and 9 for $\mathbf{R}$) to 7 (3 for $\mathbf{t}$ and 4 for $\mathbf{q}$). With the camera viewpoint $\mathbf{T}$ and the 3D point cloud of the stadium, the background can be easily reconstructed.

The foreground, instead, includes dynamic objects, *i.e.*, the skaters, that can be cropped out of the captured 2D video, delivered as separate streams to the clients, and overlaid in the camera view based on their original position. Although it requires less bandwidth, this approach presents several technical challenges. Since the bounding box used to extract each skater varies over time, the resulting cropped figures would have non-uniform spatial resolutions, complicating real-time rendering and synchronization on the client side. Padding cropped frames with black pixels to a fixed resolution is bandwidth-inefficient, similar to codec overhead from aligning with fixed-size coding blocks [45]. Transmitting each crop as an independent image sequence is even worse, as it forgoes inter-frame compression gains [46]. These inefficiencies led us to replace pixel-based representations with a minimal structured alternative.

Rather than transmitting full pixel-based representations of the skaters, we transmit a compact set of skeletal keypoints $\mathcal{S} = \{(x; y) \mid x \leq W; y \leq H\}$, where W and H denote the width and height of the video frame, respectively. These keypoints efficiently encode the motion and structural pose of the skaters over time. The total number V of values necessary to reconstruct the scene is:

$$V = (2 \times |\mathcal{S}| + |\mathcal{B}| + |\mathbf{T}|) \times F \times L \tag{1}$$

where $\mathcal{B}$ is the per-frame set of values representing the bounding box of the skater, $|\mathbf{T}|$ is the minimum number of parameters to represent the camera viewpoint $\mathbf{T}$, F is the frame rate of the video in frames per second, and L is the length of the video in seconds.

It is worth noting that the generative model can be transmitted to the client before the stream takes place and, therefore, does not contribute to the streaming bandwidth.

Depending on the resolution of the input video, values in $\mathcal{B}$ and $\mathcal{S}$ can be efficiently represented using 11 bit (1920×1080) or 12 bit (3840×2160) integers. For the camera viewpoint T, we consider 32 bit floating-point values.

The motion of each ice skater throughout the whole video, encoded as 17 keypoints [34], requires 64 kB (plus less than 4 kB for the bounding boxes), while camera viewpoints require additional 42 kB. The total data transmitted is, therefore, approximately 110 kB, corresponding to a 99.9% reduction in bandwidth compared to the original 134 MB HEVC-encoded video.

Despite this impressive compression, perceptual coherence is preserved. The reconstructed stadium maintains spatial realism, while the skaters' poses and trajectories are accurately rendered in appearance and timing. Although conventional quality metrics like VMAF may decline due to the lack of pixel-level fidelity, the semantic content and visual plausibility remain intact, especially for downstream use cases such as highlight replays, performance analysis, or live streaming in low-bandwidth environments.

These insights form the basis of our system, GenStream, a perceptually aware, bandwidth-efficient streaming architecture tailored for structured, domain-specific content like artistic ice skating competitions.

4 SYSTEM DESIGN

GenStream proposes a radical departure from conventional video pipelines by eliminating pixel-level redundancy at the server and instead transmitting only a sparse, generative codebook of the scene. The system rethinks the encoding of human-centric motion and background information as two disjoint but complementary generative tasks: appearance reconstruction and spatiotemporal placement.

Unlike conventional codecs or even object-aware compression pipelines, GenStream treats video as a generative instruction set rather than a sequence of images. At its core is a hybrid pipeline that combines object understanding, 3D scene modeling, and generative synthesis. Figure 1 illustrates the architecture of GenStream, broken down into its server-side and client-side pipelines.

4.1 Server-side GenStream

The server-side component transforms raw videos into a set of semantically meaningful layers: dynamic object keypoints, background images, and camera points of view. We describe four key stages below:

Salient Actor Extraction. The pipeline begins by identifying and segmenting salient actors from each frame. This is accomplished via Artificial Intelligence (AI) through object detection and instance segmentation models.

To lower the chance of misclassifying a spectator for the actor, we segment only the largest detected human in each frame. This produces an alpha mask that is used to construct two complementary Red Green Blue Alpha (RGBA) layers per frame: *(i)* A foreground image containing only the actor, and *(ii)* a background image with the actor masked out.

The segmentation step acts as an attention filter that reduces each frame to its most perceptually critical content, discarding clutter, crowds, and occlusions.

Keypoint-based Pose Encoding. For each segmented actor, GenStream extracts high-resolution keypoint trajectories across time. These form the structured motion encoding that guides generative synthesis on the client side.

Our system uses state-of-the-art pose estimation models to track anatomical keypoints of the actor, and encodes them into a temporal stream. The keypoints serve as lightweight motion blueprints for actor synthesis downstream.

Unlike traditional video codecs that preserve every pixel, GenStream transmits only these skeletal pose traces, implicitly entrusting the client to fill in visual details through generative inference.

Background Inpainting and 3D Reconstruction. The server assembles a clean, actor-free background model from the masked RGBA background layers. Regions previously occluded by actors are inpainted using context-aware algorithms to restore full-frame realism and aid the Structure-from-Motion (SfM) algorithm to infer camera viewpoint. These poses allow the client to re-render dynamic actors within a coherent 3D scene.

This use of background-only frames to drive global camera understanding is a key innovation of GenStream: it decouples the spatiotemporal understanding of scene layout from the actors themselves, enabling more robust reconstructions.

It is important to note that this step becomes essential when the 3D model of the venue and camera viewpoints are not provided by the competition organizers apriori.

Encoding for Transmission. Finally, the server packages actor keypoints and camera viewpoints coordinates into a single metadata file. The structured nature of these coordinates allows for a highly optimized compression logic, resulting in massive bitrate gains when compared to the traditional block-based video encoding.

4.2 Client-side GenStream

The client-side of GenStream is responsible for converting the structured transmission into a visually plausible, temporally coherent video. Instead of decoding pixels, it interprets abstract control representations (keypoints + poses) and re-renders the video using a generative prior. Three key modules are involved:

Skeleton-Driven Actor Reconstruction. The received keypoint stream is decoded into a full-body actor reconstruction via a conditional generative model.

This step synthesizes full-frame RGB actors at arbitrary resolutions and quality levels, depending on the client's capabilities and time constraints.

Viewpoint-Aware Scene Recomposition. Using the 3D camera viewpoints and original spatial metadata, the client reinserts each synthesized actor into the 3D background, which can be a custom-made representation sent to the client prior to the streaming event, or extracted by the same algorithm that inferred the camera viewpoints. The composition is performed frame-by-frame, guided by the temporal evolution of the keypoints and viewpoint metadata.

Frame Synthesis and Playback. The final video is reconstructed by compositing synthesized actors into the 3D background. This reconstructed sequence is passed to the video player for smooth, low-latency playback.

By relying on learned priors and sparse representations, GenStream is robust to bandwidth drops, jitter, or loss of intermediate frames. It behaves more like a generative animation system than a classic video streaming system.

4.3 Design Summary

In summary, GenStream replaces blocks of pixels with poses, shifting the core payload from image data to motion semantics, separates actor and background processing, enabling independent treatment and optimization of each layer, and uses client-side generative models to recreate known figures, dramatically reducing the required bandwidth.

This design rethinks the video stack from first principles, offering a blueprint for streaming systems where generative intelligence, not bitrate, defines visual quality.

5 IMPLEMENTATION

Content. To validate our approach, we applied the system to a dataset comprising more than 40 000 frames grouped in 48 scenes with duration ranging 9 s to 55 s extracted from four figure skating performances at 30 fps and 1080p resolution [47]. All videos belong to the 2018 Pyongchang Olympic Games, and thus share key characteristics, including: *(i)* the same ice rink background with predominantly stationary objects and spectators, *(ii)* one figure skater actively moving in the center of the ice rink, and *(iii)* a set of cameras, all following the skater from different points of the ice rink's lower ring.

Setup. All processing is performed on an Ubuntu 20.04 LTS server equipped with an Intel Xeon Gold 5218 CPU (64 cores, 2.30 GHz) and an NVIDIA Quadro RTX 8000 GPU with 48 GB of memory. The full system is packaged into two Docker [48] containers: one for the server-side pipeline and another for Unity-based client rendering. The pipeline is implemented in Python using OpenCV, PyTorch, and external tools like COLMAP [49], with Unity handling the rendering pipeline on the client side.

Several components of GenStream are fully implemented and tested independently, including actor segmentation, keypoint extraction, inpainting, and data export to 3D scenes. Others, such as camera viewpoint recovery, pose novel challenges and are currently under scrutiny.

Actor Segmentation and Frame Decomposition. Each frame captured by the camera is parsed using YOLOv11 [32], which detects all individuals present. Among them, we retain only the largest bounding box to discard bystanders or referees. This bounding box is passed to Segment Anything Model 2 (SAM 2) [50], which produces a fine segmentation mask, as shown in Figure 2 (Original Skater). SAM 2 is a high precision and reliability segmentation model. However, its complexity does not permit real time performance. Alternatively, the detection and segmentation can be performed in one pass by YOLOv11. This approach is much faster, and compatible with real-time live processing, but its output quality is not as robust.

We use the mask to generate, for each frame: *(i)* actor.png: the actor extracted from the scene, and *(ii)* background.png: the frame with the actor region masked. The mask is used to add transparency, resulting in RGBA images that clearly differentiate the portions of the frame that belong to the actor and the background, enabling actor-specific compression and background reconstruction.

Keypoint extraction. After detection and segmentation, GenStream proceeds to a pose estimation stage. This step aims to convert each segmented actor into a structured representation of keypoints that compactly captures their motion and posture over time. Since the bounding box of the primary actor has already been computed in the segmentation phase, keypoint extraction is performed in parallel, enabling efficient processing.

To extract keypoints from each actor, we compare three methods with decreasing spatial density: a Canny edge-based filter [51], a grid sampling filter, and a YOLOv11 Pose-based skeleton detector [52]. The Canny filter highlights contours by detecting high-gradient edges, while the grid filter overlays a regular grid on the edge map and selects representative points from high-detail regions, marking a square's center when it contains sufficient edge pixels. The YOLOv11 Pose model estimates 17 anatomical landmarks directly and connects them into a human skeleton using predefined edges. All three methods, shown in Figure 2, are applied to the segmented RGBA image of the actor, ensuring no background interference. Since all approaches yield comparable perceptual quality in downstream tasks, we select the YOLO-based method due to its compact representation and lower bandwidth requirements.

Background inpainting. To enable a clearer view synthesis, image inpainting is used to dynamically fill regions of the background images where actors have been segmented out. Tools such as OpenCV's Telea [53] or Stable Diffusion [13] are used depending on available computational resources. The output is a sequence of image files without foreground actors, to aid the camera viewpoint estimation phase. While inpainting may introduce some inconsistency across frames, leaving dynamic foreground actors in place typically leads to more severe errors in COLMAP's reconstruction, as it assumes a static scene. Removing the actors and filling their regions, even imperfectly, helps reduce false feature matches and improve the accuracy of the background structure estimation.

3D Ice Rink Modeling. We manually reconstructed an Olympic ice rink in Unity to serve as the scene's geometric and visual reference. This includes accurately placed lines, lighting, and reflection properties to match typical broadcast footage (see Fig. 3).

Camera Viewpoint Recovery: Challenges and Approaches (Partially Implemented). Correctly recovering the camera parameters for each video remains one of the most critical and technically demanding components of GenStream. We explore two alternative approaches:

- *(i)* **COLMAP-based 3D Reconstruction (Partially Implemented)** We use COLMAP to generate a sparse point cloud of the static scene and estimate the per-frame camera intrinsics and extrinsics. Figure 4 shows a 2D projection of a reconstruction result. While COLMAP succeeds in generating plausible camera trajectories, integration with Unity remains incomplete. Converting COLMAP's coordinate system and focal length parameters into Unity's camera model introduces geometric inconsistencies that still need to be resolved. Furthermore, to maintain the significant bandwidth reduction that GenStream proposes, COLMAP features extracted from different video scenes need to be fused together into a complete and realistic point cloud model, and sent to the client before the live even takes place. The origin of such video scenes may be the owner of the background stadium, or previous scenes that have been transmitted via regular video streaming.
- *(ii)* **Differentiable Camera Viewpoint Optimization (Partially Implemented)** Another explored approach involves directly optimizing camera parameters by rendering the 3D scene from a randomly initialized camera angle, capturing the resulting view, and comparing it to the inpainted background from the original video frame. The difference between the rendered and inpainted images serves as a loss function, which is minimized through iterative optimization to estimate the correct camera viewpoint. Efforts in this direction have raised many challenges, both in terms of implementation structure and resulting quality. Furthermore,

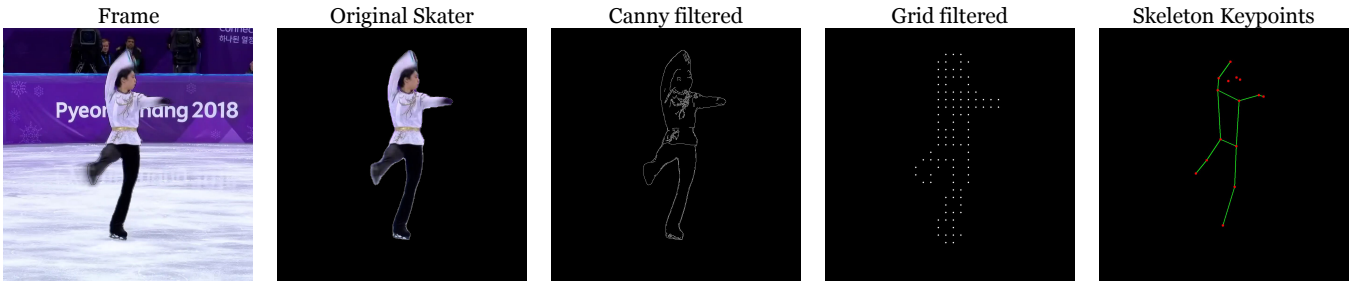

Figure 2: Visual comparison between original frame and various input maps and methods. All images are 1024×1024.

Figure 3: Qualitative comparison between original frame (left) and rendering of the 3D model with applied skater (right).

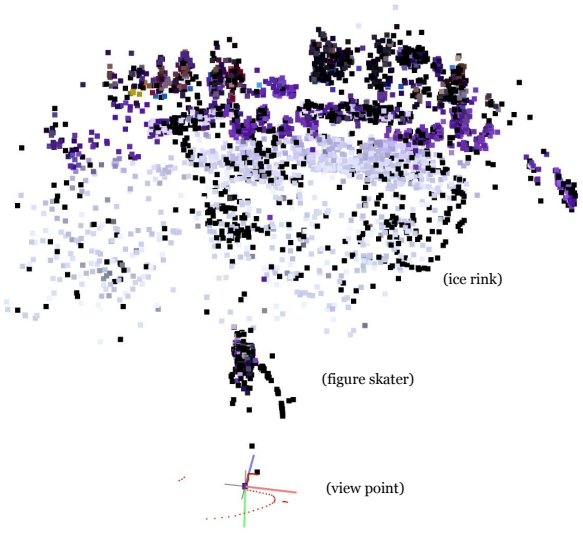

Figure 4: COLMAP generated point cloud of one scene, complete of camera view intrinsics and extrinsics, figure skater in foreground, and ice rink in the background.

the computation required for such an optimization is not compatible with live streaming. To address this last challenge, we implemented a more efficient solution: we perform the full optimization only for the first frame of each scene, then use a SIFT-based relative pose estimation [54] to propagate camera viewpoints to subsequent frames by comparing each one to its immediate predecessor. This significantly reduces the overall computational burden while maintaining adequate tracking accuracy.

Data export (Partially Implemented. Finally, the extracted data, comprising actor bounding boxes and keypoints, and camera viewpoints (or optionally COLMAP's complete scene information), is saved to a CSV file and then encoded for transmission.

With regards to keypoints transmission, ach coordinate is encoded as a 11-bit unsigned integer, sufficient to represent pixel positions within a 1080p resolution frame (up to 1920 pixels). Crucially, because the number and ordering of keypoints are fixed and known in advance, each segment of the bitstream corresponds to a specific coordinate of a specific object in a deterministic way. For example, the first 11 bits always encode the X-coordinate of the first keypoint, the next 11 bits the corresponding Y-coordinate, and so on. This layout removes the need for additional metadata, enabling both tight compression and extremely fast parsing on the client side. If an actor is not detected in a particular frame, a sentinel control value (2000, greater than 1920 and thus outside the possible coordinate range) is inserted in place of the missing coordinates. This allows the client to gracefully handle missed detections while maintaining alignment with the expected bitstream format. However, to develop such a tight compression scheme for COLMAP's features, it is required to solve the related challenges mentioned previously.

It is worth noting that the compressed data is delivered to the client at intervals, aligned with the target latency requirements of live video streaming applications.

Generative Model Training Setup. To generate realistic figure skater images from skeleton keypoints, we train a cGAN which receives in input paired image data, where each sample consists of two components: a reconstructed image of the person based on their keypoints, and the corresponding ground truth image. These two images, originally sized at either 128×128, 256×256, or 1024×1024, are concatenated to form paired input images of size 256×128, 512×256, and 2048×1024, respectively. The input images, split into 80 % for training and 20 % for testing, are augmented using random resizing, cropping, and mirroring to enhance robustness.

Client-Side Reconstruction. GenStream's client-side rendering is implemented in Unity and currently uses the manually constructed 3D model of the Olympic ice rink shown in Figure 3 as its background rendering approach. To complete the synthetic view, the generated figure skater is placed at the correct spatial location along the camera's optical axis, ensuring that their size and position match the original video.

In our current implementation, camera parameters are inferred manually to approximate the original view. Using these values, we can overlay the generated figure skater and achieve a visually plausible frame composition on top of the 3D background. However, this manual solution is clearly not scalable, and useful only for qualitative presentations. Once this challenge is solved, the Unity renderer is already set up to accept per-frame camera values.

6 CONCLUSION

GenStream introduces a radically different paradigm for video streaming, one that replaces dense pixel transmission with high-level semantic instructions that guide client-side generative reconstruction. By transmitting only sparse skeleton-based keypoints and camera viewpoint metadata, GenStream achieves a bandwidth reduction of over 99.9% compared to conventional HEVC encoding, while preserving essential perceptual cues for immersive viewing. This system provides a blueprint for a new generation of streaming architectures that prioritize semantic content over pixel fidelity. However, several open challenges remain before GenStream can reach its full potential.

A key future direction is the evolution from 2D sprite-like renderings to fully volumetric, viewpoint-consistent representations. Instead of compositing a 2D actor into a 3D scene, we envision modeling the actor as a moving 3D skeleton driving a volumetric avatar. This would allow the system to support free-viewpoint video synthesis, enabling clients to explore scenes from arbitrary perspectives, even those not originally captured by any camera. Realizing this vision introduces two critical challenges. First, it requires the fusion of multiple partial point clouds captured from different video sequences into a canonical 3D representation of the actor and background. Second, it demands lifting 2D skeletons to reliable 3D joint configurations in single-camera scenarios. These challenges intersect fields such as multi-scene registration, 3D human reconstruction, and sparse generative modeling.

Furthermore, while our current implementation uses Unity for client-side rendering, future deployments might benefit from alternative engines or custom viewers that better align with emerging hardware and web standards. Importantly, GenStream remains renderer-agnostic; any tool capable of interpreting the structured metadata and synthesizing scenes from it could serve as the playback environment.

Another important avenue lies in extending GenStream beyond the controlled setting of figure skating to more complex, multi-agent scenes such as football or gymnastics. These scenarios involve richer interactions, dynamic camera work, and greater occlusion, each posing new demands on segmentation, viewpoint estimation, and generative synthesis.

Finally, we note that background modeling using COLMAP or similar SfM tools requires careful fusion of features from multiple scenes to build a unified point cloud representation. These features can originate from the event organizer's asset library or be extracted from previously streamed footage. Solving this fusion problem efficiently and compactly is crucial for pre-distributing 3D venue models to clients.

GenStream is a first step toward a future where semantic understanding and generative capabilities replace brute-force video transmission. By offloading reconstruction to powerful client-side models and transmitting only what truly matters, it offers a scalable, perceptually meaningful approach to media delivery for the post-codec era.

REFERENCES

[1] Sandvine Holdings UK Limited, "2022 Global Internet Phenomena Report." https://www.sandvine.com/global-internet-phenomena-report-2022, 2022.

[2] Warner Bros. Discovery Sports Europe, "Record Olympics Streaming and TV Audiences for Warner Bros. Discovery as MAX Boosts Viewership and Engagement Across Europe." https://media.wbdsports.com/post/record-olympics-streaming-and-tv-audiences-for-warner-bros-disco, 2024.

[3] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 562–585, 2019.

[4] M. Nguyen, D. Lorenzi, F. Tashtarian, H. Hellwagner, and C. Timmerer, "DoFP+: An HTTP/3-Based Adaptive Bitrate Approach Using Retransmission Techniques," *IEEE Access*, vol. 10, pp. 109565–109579, 2022.

[5] T. Zhang, F. Ren, W. Cheng, X. Luo, R. Shu, and X. Liu, "Towards influence of chunk size variation on video streaming in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 7, pp. 1715–1730, 2019.

[6] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[7] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[8] Y. Zhang, C. Zhang, R. Fan, S. Ma, Z. Chen, and C.-C. J. Kuo, "Recent Advances on HEVC Inter-Frame Coding: From Optimization to Implementation and Beyond," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, p. 4321–4339, Nov. 2020.

[9] R. Skupin, C. Bartnik, A. Wieckowski, Y. Sanchez, B. Bross, C. Hellge, and T. Schierl, "Open GOP resolution switching in HTTP adaptive streaming with VVC," in *2021 Picture coding symposium (PCS)*, pp. 1–5, IEEE, 2021.

[10] X. Zhang, L. Toni, P. Frossard, Y. Zhao, and C. Lin, "Adaptive streaming in interactive multiview video systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 4, pp. 1130–1144, 2018.

[11] C. Zhu, G. Lu, B. He, R. Xie, and L. Song, "Implicit-explicit Integrated Representations for Multi-view Video Compression," 2023.

[12] T. Chen, Z. He, X. Xu, and T. Liu, "Joint Rate-Distortion Optimization for Video Compression With Conditional GAN," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[13] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," 2023.

[14] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP – Standards and Design Principles," in *ACM Conference on Multimedia Systems (MMSys)*, 2011.

[15] Apple, "HTTP Live Streaming (HLS) Authoring Specification for Apple Devices." https://bit.ly/apple_hls, 2015.

[16] Dash Industry Forum, "Client Implementation for the Playback of MPEG-DASH via Javascript." https://github.com/Dash-Industry-Forum/dash.js, 2025.

[17] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi, C.-H. Chiang, Y. Wang, P. Wilkins, J. Bankoski, L. Trudeau, N. Egge, J.-M. Valin, T. Davies, S. Midtskogen, A. Norkin, and P. de Rivaz, "An Overview of Core Coding Tools in the AV1 Video Codec," in *Picture Coding Symposium (PCS)*, pp. 41–45, 2018.

[18] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the Versatile Video Coding (VVC) Standard and its Applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3736–3764, 2021.

[19] D. Liu, Y. Li, J. Lin, H. Li, and F. Wu, "Deep Learning-Based Video Coding: A Review and a Case Study," *ACM Computing Surveys*, vol. 53, no. 1, pp. 1–35, 2020.

[20] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, , and Z. Gao, "DVC: An End-to-End Deep Video Compression Framework," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[21] H. Chen, B. He, H. Wang, Y. Ren, S. N. Lim, and A. Shrivastava, "NeRV: Neural Representations for Videos," in *International Conference on Neural Information Processing Systems (NIPS)*, 2021.

[22] S. Niklaus and F. Liu, "Softmax Splatting for Video Frame Interpolation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[23] X. Xu, L. Siyao, W. Sun, Q. Yin, and M.-H. Yang, "Quadratic Video Interpolation," in *International Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[24] H. Park, "Semantic Super-Resolution via Self-Distillation and Adversarial Learning," *IEEE Access*, vol. 12, pp. 2361–2370, 2024.

[25] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[26] J. W. Choi, J. Kim, N. I. Kim, and D. Jeon, "Faster and Better: A Machine Learning Approach to Accelerated Video Super-Resolution," in *IEEE Transactions on Multimedia*, vol. 22, pp. 2547–2559, 2020.

[27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[28] S. Zhou, H. Chang, S. Jiang, Z. Fan, Z. Zhu, D. Xu, P. Chari, S. You, Z. Wang, and A. Kadambi, "Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21676–21685, 2024.

[29] J. Li, Y. Zhang, L. Pu, T. Lin, and J. Yan, "Bimodal Semantic-Driven 3D Immersive Telepresence System," in *Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, 2025.

[30] T. Meinhardt, A. Kirillov, L. Leal-Taixé, and C. Feichtenhofer, "TrackFormer: Multi-Object Tracking with Transformers," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[31] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," in *European conference on computer vision*, pp. 1–21, Springer, 2022.

[32] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[33] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-Device Real-Time Body Pose Tracking," *arXiv:2006.10204*, 2020.

[34] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2021.

[35] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," in *International Conference on Neural Information Processing Systems (NIPS)*, 2020.

[36] E. Artioli, F. Tashtarian, and C. Timmerer, "End-to-End Learning-based Video Streaming Enhancement Pipeline: A Generative AI Approach," in *Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, 2025.

[37] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: A versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[38] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[39] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki, "Sfm-net: Learning of structure and motion from video," *arXiv preprint arXiv:1704.07804*, 2017.

[40] Bitmovin Inc., "The Annual Bitmovin Video Developer Report." https://bitmovin.com/video-developer-report/, 2025.

[41] Olympics, "Yuzuru Hanyu (JPN) - Gold Medal | Men's Figure Skating | Free Programme | PyeongChang 2018." https://www.youtube.com/watch?v=23EfsN7vEOA, 2018.

[42] R. Hartley, *Multiple view geometry in computer vision*, vol. 665. Cambridge university press, 2003.

[43] Unity Technologies, "Unity Real-Time Development Platform | 3D, 2D, VR ŏ026 AR Engine." https://unity.com/, 2025.

[44] J. Voight, *Quaternion algebras*. Springer Nature, 2021.

[45] M. Li, Y. Chang, F. Yang, and S. Wan, "Rate-distortion criterion based picture padding for arbitrary resolution video coding using H. 264/MPEG-4 AVC," *IEEE transactions on circuits and systems for video technology*, vol. 20, no. 9, pp. 1233–1241, 2010.

[46] K. A. Salih, I. A. Ali, and R. J. Mstafa, "Impact of Video Motion Content on HEVC Coding Efficiency," *Computers*, vol. 13, no. 8, p. 204, 2024.

[47] Olympics, "Olympics - YouTube." https://www.youtube.com/@Olympics, 2018.

[48] D. Merkel, "Docker: Lightweight Linux Containers for Consistent Development and Deployment," *Linux Journal, https://www.linuxjournal.com/content/docker-lightweight-linux-containers-consistent-development-and-deployment*, vol. 239, no. 2, p. 2, 2014.

[49] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion Revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[50] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, "Sam 2: Segment anything in images and videos," 2024.

[51] W. McIlhagga, "The Canny Edge Detector Revisited," *International Journal of Computer Vision*, vol. 91, pp. 251–261, 2011.

[52] D. Maji, S. Nagori, M. Mathew, and D. Poddar, "Yolo-pose: Enhancing yolo for multi person pose estimation using object keypoint similarity loss," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2637–2646, 2022.

[53] P. Chatterjee, S. Jana, and S. Ghosh, "Comparative study of opencv inpainting algorithms," *Global Journal of Computer Science and Technology*, vol. 21, no. 2, pp. 27–37, 2021.

[54] G. Lowe, "Sift-the scale invariant feature transform," *Int. J*, vol. 2, no. 91-110, p. 2, 2004.