



# A Framework for Semantic and Musical Hyperlapses

Raphael Carmo Silva Nepomuceno   [ Universidade Federal de Viçosa | [raphael.nepomuceno@ufv.br](mailto:raphael.nepomuceno@ufv.br) ]

Luísa de Souza Ferreira  [ Universidade Federal de Viçosa | [luisa.ferreira@ufv.br](mailto:luisa.ferreira@ufv.br) ]

Michel Melo da Silva  [ Universidade Federal de Viçosa | [michel.m.silva@ufv.br](mailto:michel.m.silva@ufv.br) ]

 Department of Informatics, Universidade Federal de Viçosa, Av. P H Rolfs, s/n - Campus Universitário, Viçosa - MG, 36570-900, Brazil.

**Received:** 04 April 2025 • **Accepted:** 17 July 2025 • **Published:** 03 October 2025

**Abstract** With the growing prevalence of portable cameras—such as smartphones, action cameras, and smart glasses—recording first-person videos of daily activities has become increasingly common. However, these recordings often suffer from shaky footage caused by the wearer’s continuous movements, making them physically uncomfortable to watch, and include repetitive or irrelevant segments that make them tedious to watch. To address these challenges, hyperlapse methods fast-forward first-person videos while stabilizing camera motion, and semantic hyperlapse methods additionally preserve the most important segments. Although audio is an important part of watching videos, it is often overlooked in hyperlapse creation, leaving the choice of soundtrack to the user. In this work, we introduce a multimodal hyperlapse algorithm that jointly optimizes semantic content retention, visual stability, and playback alignment with a user-chosen song’s loudness. Specifically, the hyperlapse slows down during quiet parts of the song to highlight important frames and speeds up during louder segments to de-emphasize less critical content. We also propose strategies to select songs that best complement the hyperlapse. Our experiments show that this approach outperforms existing methods in semantic retention and loudness–speed correlation, while maintaining comparable camera stability and temporal continuity.

**Keywords:** Video Summarization, Semantic Fast-Forward, First-Person Videos, Hyperlapse, Loudness

## 1 Introduction

“The best camera is the one you have with you” is a popular saying among photographers: carrying large cameras can be cumbersome, and opportunities to capture significant moments are often missed when no device is readily available. Thus, having a more convenient, albeit lower-quality, option at hand is often the better strategy. The ubiquity of smartphones illustrates this perspective: with increasingly better cameras, large storage capacity, and the social expectation of carrying one at all times, recording videos at a moment’s notice has become accessible to the average person.

On the other hand, dedicated cameras are far from obsolete, and we draw special attention to personal action cameras—such as the GoPro™—which are designed for hands-free, continuous recording from the wearer’s perspective during sports or everyday activities.

A third type of camera—although not yet widespread—is smart glasses, which benefit from the miniaturization of processors, batteries, and cameras. Their compact form factor and multifunctionality make them easy to integrate into daily life, aligning with the “best camera is the one you have with you” narrative for capturing lifelogging content.

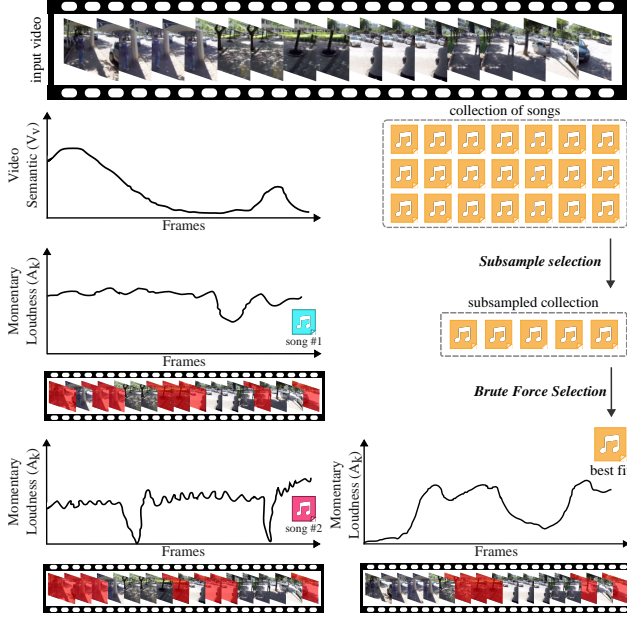
Sharing such videos on social media has become increasingly common, competing for our time and attention. Because the wearer is focused on the activity itself rather than managing the camera, hands-free recordings of daily activities often contain repetitive or irrelevant content, which can make such videos tedious to watch. First-Person Video summarization seeks to understand the intent of the wearer, reduce

irrelevant content, and create a more enjoyable viewing experience [del Molino *et al.*, 2017]. In particular, dynamic fast-forward methods assign semantic importance scores to the video according to domain-specific criteria, such as route guidance [Okamoto and Yanai, 2014] or presence of people [Ramos *et al.*, 2016]. These scores are used to decrease playback speed during important moments and increase it during less relevant segments, resulting in a continuous and representative summary video without gaps between scenes.

Videos recorded with a head-mounted, body-mounted, or handheld camera are often shaky due to body movements. When these videos are accelerated using dynamic fast-forward techniques, the motion becomes more exaggerated, which can lead to an unpleasant or even nauseating viewing experience [Silva *et al.*, 2016]. To address this, research on creating accelerated first-person videos with stable camera motion, referred to as hyperlapses, includes methods for selecting video frames that minimize shakiness [Joshi *et al.*, 2015] and applying video stabilization with fine-tuning for such videos [Silva *et al.*, 2016]. An extension called *semantic hyperlapse* [Ramos *et al.*, 2016], integrates semantic criteria from video summarization with hyperlapse techniques to produce summaries of first-person video minimizing camera instability.

Sound plays an important role in the video-watching experience, yet most hyperlapse methods require users to manually add an audio track. A user study revealed that although most participants disliked videos without any sound, they also did not want high-pitched audio resulting from fast-forwarding [Cheng *et al.*, 2009]. Aligning an accelerated video with background music is challenging due to the subjectivity of

what constitutes a good match. Matos *et al.* [2021] propose speeding up the video to match the emotions induced by the video with those induced by the song. However, this approach relies on subjective emotional responses rather than objective semantic elements of the video to determine the speed, making it difficult to rationalize the changes in video pacing.



**Figure 1. Overview of our method.** We extract semantic information from the video and the loudness curve from the song as inputs for optimal frame selection, generating a fast-forward video. To automate song selection, we first narrow candidates with an approximate algorithm, then refine the choice using brute force over the subset.

## 1.1 Contributions

In this work, we introduce a multimodal hyperlapse algorithm that selects the most important frames of a video while adjusting the playback speed to align it with the loudness—*i.e.*, the perceived intensity of a sound—of a chosen song: the hyperlapse slows down during quiet parts of the song to highlight important frames and speeds up during louder segments to de-emphasize less critical content. A high-level overview of this process is shown in Figure 1.

For instance, in a recording of a walk, interactions with other people can be considered important. These segments are played at a slower speed and aligned with quieter parts of the song, emphasizing the video. Conversely, unimportant segments are played faster and aligned with louder portions of the song, shifting attention away from the video.

We adopt the semantic definition proposed by Ramos *et al.* [2016], which considers the presence and proximity of people in the video as indicators of higher importance. The loudness curve of the song is calculated according to the guidelines of EBU Tech 3341 [European Broadcasting Union, 2023].

To choose the optimal set of frames that will form the hyperlapse, our method uses a dynamic programming formulation that minimizes a cost function combining four components: the video semantic cost, which prioritizes important frames; the audio alignment cost, which matches playback speed to

loudness; the acceleration cost, which smooths speed transitions; and the frame matching cost, which reduces camera shakiness. Rank normalization [Tsodikov *et al.*, 2002] is applied to the semantic scores and the loudness curve to make the optimization robust to outliers and unknown distributions.

In addition to describing the theoretical behavior of our algorithm, we introduce optimizations related to CPU, memory, and parallelism to improve the efficiency and viability of its implementation on real hardware.

Compared to prior work, our method achieves superior semantic retention and loudness–speed correlation while maintaining comparable performance in stability and temporal continuity. It improves upon existing hyperlapse techniques by introducing a multimodal approach that simultaneously integrates semantic and audio-driven playback speed adjustments.

Furthermore, we propose three methods for selecting a suitable song from a library by balancing the trade-off between semantic retention and audio alignment, ensuring the best match for a given video: (i) the exact, yet expensive brute-force method; (ii) an approximation of the brute-force method through subsampling of the video and song; and (iii) a hybrid that first narrows candidates with the approximate method, then refines the choice using brute force on the subset, as shown in Figure 1.

**Differences to our previous work.** Compared to Nepomuceno *et al.* [2024], our main contributions to the frame sampling algorithm include a revised cost function that improves speed-up behavior and a parallelized implementation. We evaluate these changes through an ablation study, along with a new comparison against a baseline greedy algorithm. Additionally, we introduce a song selection framework that formalizes the trade-off between semantic retention and loudness–speed correlation, proposing three methods to identify the best pairing.

## 2 Related Work

Video summarization techniques, designed to reduce video length, can be divided into three main categories based on the type of summary they produce: *storyboards*, which select a set of representative static frames; *video skimming*, which retains a discontinuous set of the most relevant segments; and *fast-forwarding*, where the video is accelerated at a constant or variable playback speed [del Molino *et al.*, 2017]. Our work focuses on fast-forwarding, as the other methods introduce temporal gaps between scenes, potentially causing viewers to lose track of the recorder’s path [Okamoto and Yanai, 2014] or the overall context of the video [Silva *et al.*, 2021].

In this section, we review related works focusing on the following areas:

- (i) **hyperlapse**, which involves fast-forwarding first-person videos while minimizing camera shakiness;
- (ii) **semantic fast-forward**, which adjusts playback speed based on the importance of scenes;
- (iii) **semantic hyperlapse**, which combines the principles of hyperlapse and semantic fast-forwarding;

- (iv) **audio-driven semantic hyperlapse**, a specific case of semantic hyperlapse that incorporates features derived from an audio track.

## 2.1 Hyperlapse

Methods to address the accentuated camera motion in fast-forward first-person videos, termed hyperlapses, were pioneered by Kopf *et al.* [2014]. Their approach uses structure-from-motion to reconstruct the scene in 3D and then renders an accelerated video through a smooth virtual camera trajectory in 2D. While this method is notable, it has a high computational cost, on the order of minutes per frame, and requires sufficient camera motion in a scene to perform the reconstruction step.

Karpenko [2014] proposed a method to create hyperlapses using gyroscope metadata to estimate camera orientations to stabilize the video and to calculate the optimal amount of zooming needed to avoid introducing empty regions into the output video. Although this approach requires additional metadata, it is notable for being the first hyperlapse method designed for real-time usage and for powering the Instagram Hyperlapse app.

Poleg *et al.* [2015] formulated the hyperlapse as a graph problem, modeling video frames as nodes and frame-to-frame transitions as edges weighted with shakiness, velocity, and appearance cost functions. The frame sampling is solved as a shortest path problem, and the frames along the path are included in the output hyperlapse. Later, an extension [Halperin *et al.*, 2018] was proposed to exploit camera motion to construct wide-view panoramas from unused frames.

Joshi *et al.* [2015] developed an adaptive frame sampling method inspired by dynamic time warping to jointly optimize camera smoothness while achieving a desired speed-up rate. Their algorithm scores frame-to-frame transitions in order to achieve smooth visual transitions with minimal cropping, while avoiding large deviations from the desired speed-up and abrupt changes in momentary speed-ups. This is achieved by building a sparse dynamic programming matrix, from which an optimal frame selection that minimizes the overall cost is retrieved and used to render the hyperlapse.

Approaches based on omnidirectional cameras [Ogawa *et al.*, 2017; Rani *et al.*, 2018] and multiple cameras [Wang *et al.*, 2018] also exist, but we focus solely on regular, single-video sources.

## 2.2 Semantic Fast-Forward

Naively fast-forwarding a video overlooks the content, accelerating through segments that contain critical information. Semantic fast-forward addresses this issue by prioritizing the retention of significant content, attempting to avoid missing key details in the accelerated playback. This approach selectively speeds up less important sections while preserving essential scenes, aiding the viewer in perceiving and understanding important events in the video.

Cheng *et al.* [2009] proposed an adaptive fast-forwarding algorithm based on domain-specific semantic rules, designed around the metaphor of a car driver who slows down near areas of interest and speeds through unexciting areas. In this

framework, the algorithm receives feedback from the viewer to determine their interests, which then informs decisions about when to increase or reduce the playback speed through a greedy approach. Their work also contributed a user research, which revealed that users prefer gradual rather than sudden or dramatic increases in playback speed, and that although most participants did not like watching videos without any audio, they also did not want high-pitched audio resulting from high-speed fast-forwarding.

Okamoto and Yanai [2014] proposed a semantic fast-forwarding method designed for route guidance in first-person videos, addressing the temporal discontinuities that can confuse viewers about their current location. Their approach assigns frame scores based on the recorder's actions—such as turning, stopping, or moving forward—and the presence of crosswalks and turns. During playback, the speed-up is greedily adjusted based on this precomputed list of importance scores for each video section.

Ramos *et al.* [2020a] introduced a semantic fast-forwarding approach that leverages reinforcement learning to adaptively accelerate instructional videos. By employing neural networks, their method learns a cross-modal embedding space between text and video. This embedding guides the agent, which is rewarded or penalized based on the cosine similarity between text and frame features. This design encourages the selection of relevant frames while skipping irrelevant ones, with the available action space ensuring temporal continuity. Later, Ramos *et al.* [2022] enhanced this framework by adding temporal information into both the feature space and the reward function, enabling the method to meet user-defined target speed-up rates.

## 2.3 Semantic Hyperlapse

Ramos *et al.* [2016] introduced the first semantic hyperlapse algorithm, which assigns lower playback speeds to high-importance segments while maintaining camera stability. They calculate a semantic score for each frame based on the presence, size, and centrality of faces, then divide the video into important and unimportant segments, each with different target playback speeds. The segments are accelerated using the graph traversal algorithm proposed by Poleg *et al.* [2015] and then combined to produce the final hyperlapse video. Building on this work, Silva *et al.* [2016] introduced a stabilization algorithm specifically tailored for semantic hyperlapses, along with a dataset of videos featuring varying total semantic scores.

Further contributions by Silva *et al.* [2018a] proposed a weighted sparse sampling approach to select a set of frames that minimize the reconstruction error of the original video given the required speed-up in each video segment, and subsequently aimed to solve the problem of abrupt jumps between segments [Silva *et al.*, 2021].

Regarding alternative semantic definitions, Ramos *et al.* [2020b] proposed using textual features extracted from social networks to represent user preferences and determine the importance of objects when calculating semantic scores. Their method employs the same sampling algorithm as Ramos *et al.* [2016] and uses the video stabilizer from Silva *et al.* [2016].

Neves *et al.* [2020] proposed a method that combines object tracking with gaze information from an eye tracker to calculate semantic scores for frames in first-person videos. Their model evaluates the intersection of objects with the user’s gaze, their proximity to the gaze point, the duration they remain in view, and their novelty. For fast-forwarding, they compared the performance of the methods proposed by Silva *et al.* [2018b] and Silva *et al.* [2018a].

## 2.4 Audio-driven Semantic Hyperlapse

Most hyperlapse methods focus primarily on visual features and do not consider the input or output audio streams. To the best of our knowledge, only two works have proposed an audio-driven approach to semantic fast-forwarding.

Furlan *et al.* [2018] proposed a multimodal hyperlapse approach that utilizes the audio stream to compute semantic scores for video frames. The method assigns higher playback speeds to unpleasant segments, such as those containing noisy crowds or streets, based on the psychoacoustic annoyance metric [Zwicker and Fastl, 2013], and employs the multi-importance fast-forwarding technique introduced by Silva *et al.* [2018b] for frame sampling. However, their resulting hyperlapse does not use the original audio.

In the work of Matos *et al.* [2021], a video and a song chosen by the user are used to create a hyperlapse in which the induced emotions of the video and the song are aligned in time. The video is processed frame-by-frame by a 2D convolutional neural network to estimate an emotion curve based on valence and arousal. Similarly, the song is processed by a 1D convolutional neural network to estimate its emotion curve. The method employs a dynamic programming algorithm, inspired by Joshi *et al.* [2015], to fast-forward the video while aligning the emotion curves. Matos *et al.* [2023] later extended this work by developing a method to automatically select the best matching song from a library.

In our previous work [Nepomuceno *et al.*, 2024], we introduced a method that combines the concepts of semantic hyperlapses and musical hyperlapses. The goal was to fast-forward first-person videos while maximizing semantic content and aligning the hyperlapse playback speed with the loudness of a song. To achieve this, we proposed an efficient dynamic programming formulation and introduced the use of rank normalization [Tsodikov *et al.*, 2002] in this context. Our method outperformed existing approaches in both semantic content retention and the novel loudness–speed correlation, while achieving comparable results in camera stability and temporal continuity.

## 2.5 Differences to Previous Works

Our method is directly derived from the works of Joshi *et al.* [2015] and Matos *et al.* [2021], which approach the video fast-forward problem through the construction and traversal of a dynamic programming matrix. Unlike both methods, in addition to optimizing the visuals or aligning the hyperlapse with the song, we also attempt to maximize an arbitrary semantic score—such as the presence of faces from Ramos *et al.* [2016]. This allows our approach to balance semantic

relevance, camera stability, and audio alignment in a unified framework.

Our approach also differs from Joshi *et al.* [2015] by precisely achieving the requested video length, as is done by Matos *et al.* [2021]. This is important when including a song in the output hyperlapse: if the hyperlapse is longer, it will extend beyond the song, leaving part of the video without accompanying audio. Conversely, if the song is longer, it will be truncated to fit the hyperlapse.

In contrast to Matos *et al.* [2021], we avoid emotion-based features, which are subjective and require learning-based methods to estimate. Instead, we use loudness, a handcrafted feature derived from objective characteristics of human hearing.

## 3 Methodology

In this section, we describe our method for creating semantic hyperlapses aligned with music. The end-to-end pipeline and data flow are summarized in Figure 2. First, we compute per-frame semantic scores and homography-based transition costs from the input video, along with a loudness profile from a song. Next, our optimizer selects frames by minimizing a cost function based on these extracted features. Then, we combine the selected frames into a video using the chosen song as background music. Finally, we propose an automated method to select a suitable song from a large collection. This division of steps arises naturally from their outputs: the extracted features can be reused across many video–audio pairings, and the selected frames can be evaluated independently of the compositing step.

### 3.1 Feature Extraction

The first stage of our method analyzes the video and the song to extract features to guide the frame selection. This step generates three outputs: (i) the semantic scores for each video frame, (ii) the pairwise distance matrix between consecutive video frames, and (iii) the momentary loudness profile of the song.

#### 3.1.1 Semantic Scores

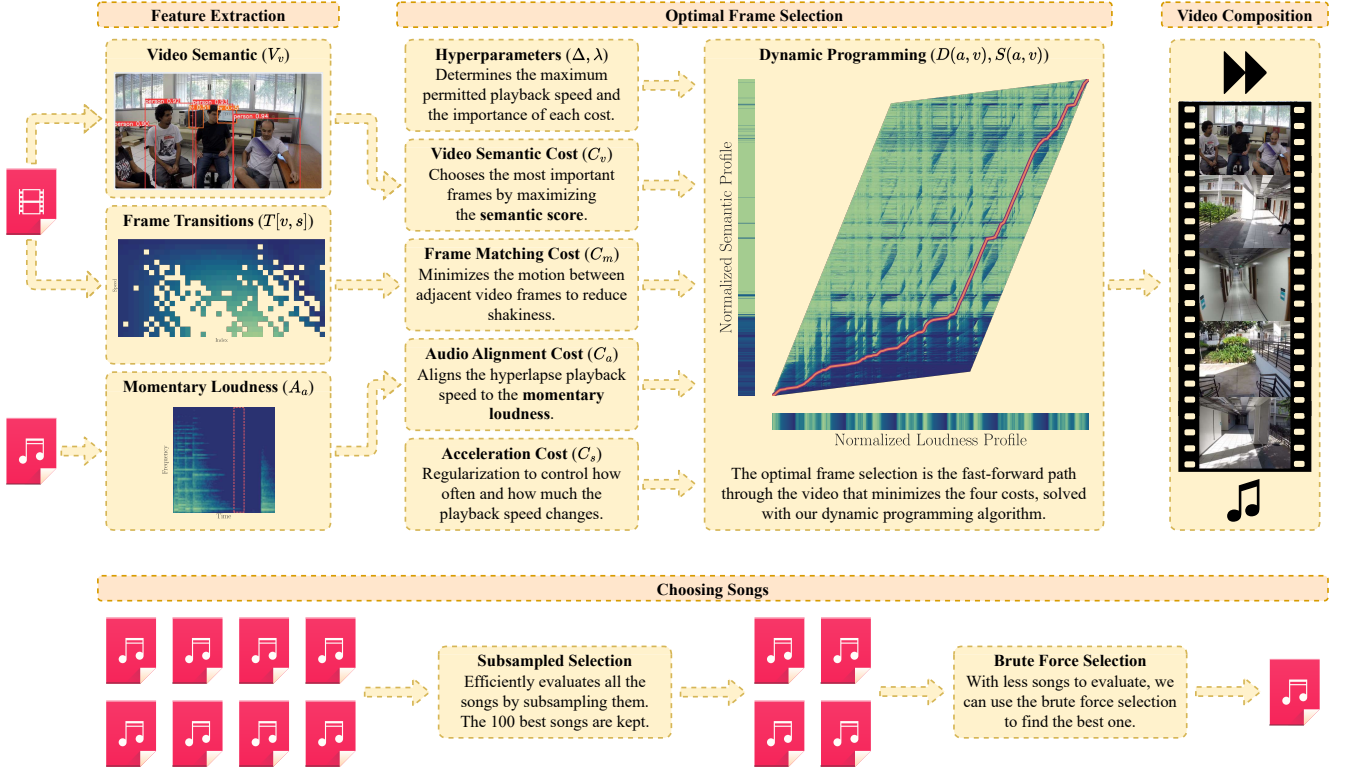
We compute semantic scores for each frame using the definition proposed by Ramos *et al.* [2016], which considers the presence and proximity of people as indicators of higher importance. Formally, for each video frame  $i$ , an object detection algorithm identifies a set  $K_i$  of objects of interest. The semantic score  $V_i$  is then calculated as:

$$V_i = \sum_{k \in K_i} C(k) \cdot G_\sigma(k) \cdot A(k), \quad (1)$$

where  $k$  denotes a detected object (*e.g.*, a person),  $C(k)$  is the detection confidence,  $G_\sigma(k)$  measures the object’s centrality in the frame, and  $A(k)$  is the area of its bounding box.

Semantic scores are sensitive to outliers. For example, if a large crowd appears in a few frames, their scores may spike, making other important frames with fewer people appear less significant. To mitigate this, we apply rank normalization





**Figure 2.** High-level overview of our method. The **feature extraction** prepares the inputs for our **optimal frame selection**, which in turn produces a path through the video which the **video composition** traverses to create a hyperlapse. Lastly, **choosing songs** automates how to pick a song from a large collection.

[Tsodikov *et al.*, 2002] to  $V$ , producing the normalized vector  $V^*$ :

$$V_i^* = \frac{1}{|V|} \sum_{j=1}^{|V|} \begin{cases} 1, & \text{if } V_i > V_j, \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

where  $|V|$  is the total number of frames.

A trade-off of this approach is increased sensitivity to noise: in videos with few important frames, distant people or objects mistakenly classified as people may receive disproportionately high ranks, which may cause the video to slow down seemingly without justification. Despite this limitation, we prioritize recall over precision, accepting this trade-off to ensure that important frames are retained.

### 3.1.2 Pairwise Distance Matrix

The pairwise distance matrix  $T[v, s]$  quantifies camera motion and is used to minimize shakiness in the fast-forward video:

$$T[v, s] = \frac{\|H(v - s, v) \cdot \mathbf{M} - \mathbf{M}\|_2}{\|\mathbf{M}\|_2}, \quad (3)$$

where  $H(v - s, v)$  is the homography mapping frame  $v - s$  to frame  $v$ , and  $\mathbf{M}$  is the 2D coordinate vector of the image center. If homography estimation fails or produces unreliable results, we set  $T[v, s] = 1$ . Thus,  $T[v, s]$  combines the *frame matching cost* from Joshi *et al.* [2015] and the *shaking ratio* metric from Ramos *et al.* [2020b].

To estimate homography matrices, we first detect keypoints and compute descriptors using the ORB feature extractor [Rublee *et al.*, 2011]. Then, for each frame pair  $\langle v - s, v \rangle$ , we match keypoints using a brute-force matcher with Hamming distance [Hamming, 1950], applying cross-checking to

retain only bidirectional matches. Finally, we estimate the homography matrix using MAGSAC++ [Baráth *et al.*, 2020], requiring at least 10 valid matches for reliable estimation.

Each homography matrix is validated by checking its reprojection error, which measures how well the detected keypoints in one frame align with their corresponding locations in the other frame under the computed homography. If the average reprojection error exceeds  $0.2 \|\mathbf{M}\|_2$ , the homography is deemed unreliable due to false positives.

### 3.1.3 Loudness Profile

Loudness is a psychoacoustic property that represents the perceived intensity of a sound. Unlike the purely physical measure of a sound wave's energy, loudness accounts for human auditory sensitivity across different frequencies. Furthermore, higher loudness levels are associated with excitement or tension, while lower levels correspond to calmness or relaxation Lu *et al.* [2006].

As specified by European Broadcasting Union [2023], momentary loudness is computed using a sliding time window of 400 ms. To align loudness with playback speed, each audio window in our work is centered on the corresponding video frame in the output. The stride of the sliding window is set to match the duration of a single video frame (e.g., 33.3 ms for videos at 30 frames per second).

We represent the loudness profile as the vector  $A = \langle A_0, \dots, A_{n-1} \rangle$ , with length  $|A| = n$ . Similarly to the semantic scores in Equation 2, we apply rank normalization [Szabo *et al.*, 2002] to ensure robustness against outliers and variations in distribution.

### 3.2 Optimal Frame Selection

Broadly speaking, our objective is to select a sequence of frames from a video that minimizes a cost function while satisfying a set of constraints, resulting in a fast-forwarded video.

Similarly to previous works [Joshi *et al.*, 2015; Matos *et al.*, 2021], we formulate this fast-forwarding problem as a dynamic programming algorithm, differing in both the algorithm formulation and the cost function. Specifically, our work differs from Joshi *et al.* [2015] by incorporating a song modality and ensuring that the hyperlapse length matches the song length. Compared to Matos *et al.* [2021], our approach introduces optimizations for computational efficiency.

The recurrence relation  $D[a, v]$  represents the minimum cumulative cost of selecting frames and playback speeds from the start of the sequence up to audio timestep  $a$  and video frame  $v$ :

$$D[a, v] = \min_{s \in [1, \Delta]} C(a, v, s, S[a-1, v-s]) + D[a-1, v-s], \quad (4)$$

with the traceback matrix  $S[a, v]$  recording the playback speeds that yield the lowest cumulative cost:

$$S[a, v] = \arg \min_{s \in [1, \Delta]} C(a, v, s, S[a-1, v-s]) + D[a-1, v-s]. \quad (5)$$

Here,  $a$  denotes the current audio timestep,  $v$  is the video frame index,  $s$  is the playback speed—*i.e.*, the number of frames to skip—and  $\Delta$  is the maximum playback speed. The construction of  $D[a, v]$  and  $S[a, v]$  is detailed in Algorithm 1. With  $C(a, v, s, s')$  as the basic operation, its time complexity depends on the nested loops over  $a$ ,  $v$  and  $s$ , resulting in  $\mathcal{O}(\Delta \cdot |A| \cdot |V|)$ .

---

**Algorithm 1** Fast-Forward Matrix Construction
 

---

```

1: Initialize  $S[a, v] \leftarrow 0$  for all  $a, v$ .
2: Initialize  $D[v] \leftarrow \infty$  for all  $v$ .
3: Initialize  $D'[v] \leftarrow \infty$  for all  $v$ .
4:  $D[0] \leftarrow 0$ 
5:  $S[0, 0] \leftarrow \frac{|V|}{|A|}$ 
6: for  $a \in [1, |A|)$  do
7:   Swap  $D$  and  $D'$ 
8:   Reset  $D[v] \leftarrow \infty$  for all  $v$ .
9:    $\bar{a} \leftarrow |A| - a$ 
10:   $v_{\min} \leftarrow \max(a, |V| - \bar{a} \cdot \Delta)$ 
11:   $v_{\max} \leftarrow \min(a \cdot \Delta, |V| - \bar{a})$ 
12:  for  $v \in [v_{\min}, v_{\max}]$  do
13:     $s_{\min} \leftarrow 1$ 
14:     $s_{\max} \leftarrow \min(\Delta, v)$ 
15:    for  $s \in [s_{\min}, s_{\max}]$  do
16:      if  $D'[v-s] \neq \infty$  then
17:         $s' \leftarrow S[a-1, v-s]$ 
18:         $c \leftarrow D'[v-s] + C(a, v, s, s')$ 
19:        if  $c < D[v]$  then
20:           $D[v] \leftarrow c$ 
21:           $S[a, v] \leftarrow s$ 
22: return  $S$ 

```

---

In Equation 4, only  $D[a, v]$  and  $D[a-1, v-s]$  are accessed, which Algorithm 1 exploits by storing  $D[a, \dots]$  as  $D[\dots]$  and

$D[a-1, \dots]$  as  $D'[\dots]$ . This optimization reduces memory usage to approximately one-ninth of the unoptimized version. Additionally, the loop over  $v \in [v_{\min}, v_{\max}]$  is inherently parallelizable since each  $D[v]$  is computed independently from values in  $D'[v-s]$ , without any dependency on other  $D[v]$  values within the same iteration.

The optimal path is determined through a backward traversal of the traceback matrix  $S[a, v]$ . Starting from  $\langle a, v \rangle$  at the end of the audio and video sequences, we recursively move to  $\langle a-1, v-S[a, v] \rangle$ , recording the values of  $v$  at each step until reaching  $\langle a, v \rangle = \langle 0, 0 \rangle$ . By starting at the end of both sequences, this traversal ensures that the fast-forward video precisely matches the length of the song, preventing either from being cut short. Formally, this process is described in Algorithm 2.

---

**Algorithm 2** Fast-Forward Matrix Traversal
 

---

```

Input: The traceback matrix  $S$ .
Output: The list of selected frames  $I$ .
1:  $a \leftarrow |A| - 1$ 
2:  $v \leftarrow |V| - 1$ 
3:  $I[a] \leftarrow v$ 
4: while  $a > 0$  do
5:    $v \leftarrow v - S[a, v]$ 
6:    $a \leftarrow a - 1$ 
7:    $I[a] \leftarrow v$ 
8: return  $I$ 

```

---

The cost minimized by the dynamic programming algorithm depends on the video frame ( $v$ ), the song timestep ( $a$ ), and the current and previous playback speeds ( $s$  and  $s'$ , respectively). It is defined as a weighted combination of the *video semantic cost*, *audio alignment cost*, *acceleration cost*, and *frame matching cost*:

$$C(a, v, s, s') = \lambda_v C_v(v, s) + \lambda_a C_a(a, s) + \lambda_s C_s(s, s') + \lambda_m C_m(v, s). \quad (6)$$

The specifics of each term in this cost function are detailed in the following subsections. Each term is normalized within the range  $[0, 1]$  to simplify weight selection for  $\lambda$ .

#### 3.2.1 Video Semantic Cost

Our objective is to retain as many important frames from the original video as possible by adjusting the playback speed—slowing down for important frames and speeding up for unimportant ones.

To achieve this, we define the **video semantic cost** ( $C_v$ ) as:

$$C_v(v, s) = \begin{cases} 1 - V_v^*, & \text{if } s < \frac{|V|}{|A|}, \\ 1 - \alpha V_v^*, & \text{otherwise,} \end{cases} \quad (7)$$

where  $V_v^*$  is the rank-normalized semantic score from subsection 3.1.1. Since our framework is formulated as a minimization problem, minimizing  $C_v(v, s)$  corresponds to maximizing  $V_v^*$ . To penalize choosing playback speeds higher than the target speed-up rate  $\frac{|V|}{|A|}$  for important frames, we downweight their contribution using a small positive constant  $\alpha$  (e.g., 0.05).

### 3.2.2 Audio Alignment Cost

The **audio alignment cost** ( $C_a$ ) adjusts the hyperlapse playback speed to align with the music’s loudness, slowing down during quieter sections and speeding up during louder ones.

To achieve this, we define a target playback speed  $\hat{s}_a$  at each song timestep  $a$ , computed using linear interpolation:

$$\hat{s}_a = 1 + (\Delta - 1) \cdot A_a^*, \quad (8)$$

where  $A_a^*$  is the rank-normalized loudness score from subsection 3.1.3. The playback speed reaches its maximum value ( $\hat{s}_a = \Delta$ ) at peak loudness and its minimum ( $\hat{s}_a = 1$ ) at the lowest loudness.

Next, we define a penalty for deviations from the target playback speed:

$$C_a(a, s) = \left( \frac{\hat{s}_a - s}{\Delta - 1} \right)^2, \quad (9)$$

which is normalized by the maximum possible speed variation (*i.e.*, from 1 to  $\Delta$ ).

The alignment between loudness and semantic importance results from the joint minimization of the video semantic cost and the audio alignment cost. The video must slow down to highlight important frames, but doing so during loud audio segments increases  $C_a(a, s)$ . Therefore, the optimal strategy is to synchronize important video segments with quieter audio periods and less important segments with louder audio.

In practice, the quiet–slow and loud–fast relationship is based on empirical findings, as it felt more intuitive in our experiments. However, this relationship could be reversed to accommodate different preferences or use cases.

### 3.2.3 Acceleration Cost

Users tend to prefer smooth, gradual changes in playback speed rather than abrupt shifts [Cheng *et al.*, 2009]. To achieve this, an acceleration cost ( $C_s$ ) is used to penalize large variations in speed. It is defined as:

$$C_s(s, s') = \left( \frac{s' - s}{\Delta - 1} \right)^2, \quad (10)$$

where  $s$  and  $s'$  represent the current candidate speed and the previous speed, respectively. The difference between them is normalized by the maximum possible change in speed (*i.e.*, from 1 to  $\Delta$ ).

This acceleration cost was originally proposed by Joshi *et al.* [2015] to reduce visual jumps caused by sudden accelerations. Our formulation of  $C_s(s, s')$  differs in its normalization to be consistent with the other cost terms in our algorithm.

### 3.2.4 Frame Matching Cost

The frame matching cost ( $C_m$ ) reduces camera shakiness by penalizing motion with:

$$C_m(v, s) = T[v, s], \quad (11)$$

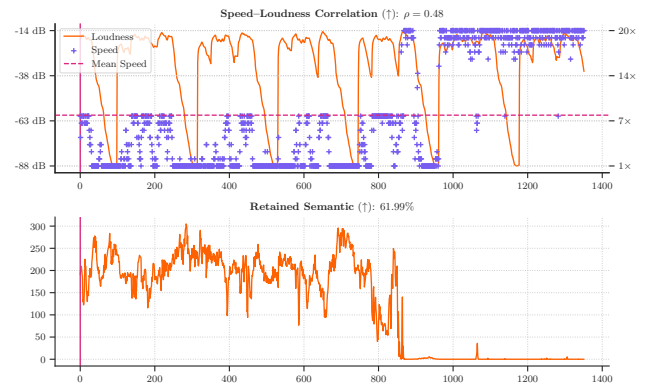
where  $T$  is the pairwise distance matrix described in Equation 3.

The value of  $T[v, s]$  represents the amount of camera movement in the transition from frame  $v - s$  to  $v$ . It ranges from 0, indicating no movement, to 1, which corresponds to movements exceeding half the diagonal of the video frame or cases of homography failure.

## 3.3 Video Composition

The output from the previous step is a sequence of frame indices that should be included in the hyperlapse. However, to be useful for humans, this sequence must be made into a video. The chosen frames can be either concatenated directly to produce a video, or processed with the stabilization algorithm proposed by Silva *et al.* [2016] for first-person fast-forward videos. The original audio track of the video is discarded, and the chosen song is added as the audio stream of the hyperlapse to complete the output of our method.

If the stabilization algorithm proposed by Silva *et al.* [2016] is used, the camera motions between the frames sampled by our algorithm are smoothed using weighted homographies, while the discarded frames are used to cover blank spaces produced by the homographies. Images corrupted by the smoothing step are replaced with discarded frames that have a high semantic score.



**Figure 3.** Dashboard visualization for evaluation. The top image is a frame from the hyperlapse. The middle plot shows speed (blue crosses) and loudness (orange line), with a mean speed reference (dashed pink line). The bottom plot tracks retained semantic content. The vertical purple line serves as a cursor, indicating the current frame’s position.

Another approach we used during testing was generating a “dashboard” view of the hyperlapse. This visualization presents the accelerated video along with plots of the selected speed-ups and song loudness, helping to assess their alignment. It also includes a plot of each frame’s semantic content, which helps identify why the playback speed changes, particularly in cases of false positives. An example of this

visualization is shown in Figure 3.

### 3.4 Choosing Songs

When access to a large collection of songs is available, there is an opportunity to select one that achieves a good match with the video. However, due to the size of the collection, evaluating matches manually becomes time-consuming and impractical. To address this, we propose automating the selection process using a simplified model of fitness based on the trade-off between semantic and correlation scores.

Pairing a song with a video can lead to a range of outcomes. At one extreme, the video and song are well-matched, allowing both the semantic score and correlation score to be maximized. At the other extreme, a poor match forces a trade-off, where improving one score reduces the other.

In this section, we formalize this trade-off and present it as the guiding criterion for our song selection algorithm.

#### 3.4.1 Balancing Semantic and Correlation

To balance semantic and correlation, we combined them using Tchebycheff scalarization from the area of multi-objective optimization [Steuer and Choo, 1983], which evaluates the solution by the worst of the two scores. This ensures no extreme compromises in either objective.

The original Tchebycheff scalarization assumes minimization problems and therefore takes the max of the objectives. Since ours is a maximization problem, we instead define:

$$F_{\min}(\mathbf{x}, \mathbf{s}) = \min\{F_v(\mathbf{x}), F_a(\mathbf{s})\}, \quad (12)$$

where  $F_v(\mathbf{x})$  represents the total semantic score computed from the chosen video frames  $\mathbf{x}$ , and  $F_a(\mathbf{s})$  represents the correlation score derived from the chosen speed-up vector  $\mathbf{s}$ .

The **semantic score** ( $F_v$ ) quantifies the retention of semantic information relative to the maximum possible, given an equal number of frames [Silva *et al.*, 2018b]:

$$F_v(\mathbf{x}) = \frac{\sum_{i \in \mathbf{x}} V_i}{\sum_{j \in \hat{\mathbf{x}}} V_j}, \quad (13)$$

where  $\hat{\mathbf{x}}$  represents the vector of frames with the highest semantic scores from the original video, constrained to have the same length as  $\mathbf{x}$ . As this calculation doesn't account for the maximum skip ( $\Delta$ ) allowed between frames, a score of 100% isn't necessarily achievable in practice.

The **correlation score** ( $F_a$ ) evaluates how closely the output speed-up curve aligns with the song's momentary loudness, calculated using the Spearman correlation coefficient [Spearman, 1904]. When  $F_a(\mathbf{s}) \approx 1$ , playback speed increases with loudness; scores near zero suggest no significant alignment; and when  $F_a(\mathbf{s}) \approx -1$ , higher loudness corresponds to slower playback. The choice of this correlation minimizes assumptions about the relationship between playback speed and loudness, accommodating their non-linear nature.

#### 3.4.2 Algorithms for Choosing Songs

We propose three approaches for selecting the optimal song to pair with a video, each balancing computational cost and precision differently.

The **brute force selection** evaluates every song in the collection by running our hyperlapse algorithm (*i.e.*, Algorithm 1 and Algorithm 2) on each song and computing its  $F_{\min}$  score. The song with the highest  $F_{\min}$  score is selected as the best match. This approach is computationally intensive, as it requires evaluating the complete video-song pair for every candidate.

The **subsampling selection** method approximates the brute-force approach by first downsampling the video and audio data using a maximum filter with a window size of 10. Algorithm 1 has a nested iteration over each audio timestep  $a$ , over a range of video frames  $v$ , and over the playback speeds  $s$  from 1 to  $\Delta$ , resulting in a time complexity of  $\mathcal{O}(\Delta \cdot |A| \cdot |V|)$ . Consequently, the subsampling provides an expected speed-up of  $100\times$ . After evaluating all songs on the subsampled data, the song with the highest  $F_{\min}$  score is selected. Although less precise than the brute-force approach, this method is computationally efficient and well-suited for large datasets.

The third method, **hybrid selection**, combines the subsampling and the brute force methods. First, the subsampled method is used to rank all songs, and the top 100 candidates with the highest  $F_{\min}$  scores are selected. Then, the brute force method is applied only to these top 100 candidates, fully evaluating their scores to determine the best song. Our experiments show that this two-step process is often enough to find the optimal song at a fraction of the computational cost of the brute force algorithm.

## 4 Experiments

In this section, we describe the experimental setup used to evaluate our proposed methods. We begin by presenting the *datasets* used in our experiments. Next, we define the *evaluation criteria* used for comparison, followed by a discussion of the *hyperparameters* chosen for our approach. We then assess our *frame sampling* method through an analysis against *competing methods*. Following this, we perform an *ablation study* to isolate the contributions of our algorithmic choices and conduct a *sensitivity analysis* to examine the impact of individual cost components. Finally, we evaluate the effectiveness of our *song selection* algorithms.

### 4.1 Datasets

We evaluate our method using videos from the *Annotated Semantic Dataset* (ASD) [Silva *et al.*, 2016] and songs from the *Database for Emotional Analysis of Music* (DEAM) [Aljanaki *et al.*, 2017].

The ASD consists of eleven videos of daily activities such as walking, biking, and driving, along with their semantic annotations as described in subsection 3.1.1. These videos vary in semantic content, ranging from those where the semantic information is present in approximately 0% of the frames (*i.e.*, “0p”) to those where it is present in up to 75% of the frames (*i.e.*, “75p”). They also differ in camera motion and duration, with lengths between 4 and 10 minutes. Table 1 lists the lengths of the videos and the speed-ups used for them in the experiments. Since the dataset does not provide



the pairwise distance matrices required by our method, we compute them as described in subsection 3.1.2.

**Table 1.** Details of the Annotated Semantic Dataset, with the target speed-up factors needed to pair with the songs from the Database for Emotional Analysis of Music.

Name	Frames	FPS	Length (mm:ss)	Speed-up (to 45s)
Biking 0p	17,949	60	4:59	6.6×
Biking 25p	17,071	30	9:29	12.6×
Biking 50p	26,954	60	7:29	10.0×
Biking 50p 2	14,939	60	4:09	5.5×
Driving 0p	9,463	30	5:15	7.0×
Driving 25p	7,989	30	4:26	5.9×
Driving 50p	10,379	30	5:46	7.7×
Walking 0p	8,219	30	4:34	6.1×
Walking 25p	10,982	30	6:06	8.1×
Walking 50p	11,570	30	6:26	8.6×
Walking 75p	15,481	30	8:36	11.5×

For the songs, we use the 1,744 excerpts from the DEAM dataset, each approximately 45 seconds long. The dataset also includes 58 full-length songs, which we exclude to ensure consistency in length for the experiments. Instead of the emotion annotations provided in the dataset, we compute the loudness curves for each song using the Essentia<sup>1</sup> library, as described in subsection 3.1.3.

To evaluate our method comprehensively, we tested all possible combinations of ASD videos and DEAM songs, resulting in a total of 19,184 pairs.

Although Matos *et al.* [2021] also introduces a dataset in their work, it is designed to measure the emotional similarity between the song and the video. Given the difference in optimization criteria and the absence of semantic labels in their dataset, we chose to omit it from the comparison.

## 4.2 Evaluation Criteria

Our quantitative analysis of the output fast-forward video evaluates four aspects: (i) the semantic of the output video, (ii) the correlation between playback speed and song loudness, (iii) the variability in playback speed, and (iv) the visual smoothness of the accelerated video.

The **semantic score** quantifies how much semantic information is preserved in the accelerated video relative to the maximum possible for the same number of frames, as defined in Equation 13. To evaluate whether a method explicitly slows down for important frames, we also compute a modified version that considers only frames played below the target speed-up, referred to as the **low-speed semantic score**.

The **correlation score** measures how well the playback speed aligns with song loudness using the Spearman correlation coefficient [Spearman, 1904], as described in subsection 3.4.1. A score near 1 indicates strong alignment, where the loudest parts of the song correspond to the fastest video segments and the quietest parts to the slowest. A score close

to 0 suggests weak or ineffective alignment between speed and loudness.

The **acceleration score** measures how gradual the changes in playback speed are, as large or frequent changes can make it difficult to anticipate and follow important scenes. It is defined as:

$$F_s(s) = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n-1} (s_{i+1} - s_i)^2}, \quad (14)$$

*i.e.*, the root mean square of successive differences, which is a measure of variance for time series [von Neumann *et al.*, 1941], over the chosen speeds.

The **instability score** quantifies shakiness as the standard deviation of pixel values within a sliding window of seven neighboring frames, averaged over the entire video [Silva *et al.*, 2018b]. Since our goal is to compare frame sampling strategies, we evaluate instability on the raw frames of the original video, without any stabilization post-processing.

In the sensitivity analysis, we use the **homography failure rate** to assess the effectiveness of our *frame matching cost* (Equation 3) in selecting frame-to-frame transitions with valid homography transformations. To determine failures in homography estimation, we apply the criteria outlined in subsection 3.1.2. We then define the failure rate as:

$$F_m(\mathbf{x}) = \frac{1}{n-1} \sum_{i=2}^n \begin{cases} 0, & \text{if } H(x_{i-1}, x_i) \text{ exists,} \\ 1, & \text{otherwise.} \end{cases} \quad (15)$$

Here,  $\mathbf{x}$  represents the vector of selected frame indices,  $n$  is the total number of selected frames, and  $H(x_{i-1}, x_i)$  denotes a valid homography matrix from  $x_{i-1}$  to  $x_i$ .

## 4.3 Hyperparameters

Our method has five parameters: the maximum speed-up ( $\Delta$ ) and the four weights for Equation 6 (*i.e.*,  $\lambda_v$ ,  $\lambda_a$ ,  $\lambda_m$ ,  $\lambda_s$ ). For our experiments, these parameters were set empirically.

The choice of weights depends on user preferences. For the experiments, we chose the video semantics as the primary focus, using the song as a secondary medium to enhance the hyperlapse. Acceleration regularization and camera stability were treated as tertiary factors. Thus, we set:

$$\langle \lambda_v, \lambda_a, \lambda_s, \lambda_m \rangle = \langle 4, 2, 1, 1 \rangle. \quad (16)$$

For the maximum speed-up, our goal was to push  $\Delta$  as high as we felt comfortable, particularly for fast-moving videos such as *Driving* and *Biking*. We settled on  $\Delta = 20$ , slightly higher than the maximum speed-up of 16 used by Matos *et al.* [2021]. A higher maximum speed allows for greater time savings on unimportant frames but can create gaps between scenes, breaking the temporal continuity of the video. Additionally, larger distances between frames increase the likelihood of homography estimation failures, making it harder to stabilize the video.

## 4.4 Competitors

We evaluate the frame sampling performance of our method against two state-of-the-art competitors that share similar

<sup>1</sup>Available at: [https://essentia.upf.edu/reference/std\\_LoudnessEBUR128.html](https://essentia.upf.edu/reference/std_LoudnessEBUR128.html). Accessed January 11, 2025.

goals: the expanded Sparse Adaptive Sampling (SAS2) [Silva *et al.*, 2021] for semantic hyperlapses and the Musical Hyperlapse (MH) [Matos *et al.*, 2021] for musical hyperlapses. Additionally, we compare our method to its previously published version (SMH1) [Nepomuceno *et al.*, 2024] to assess the improvements introduced in this work. Finally, we include the uniform speed-up method as a baseline.

To evaluate SAS2, we use their Locality-constrained Linear Coding (LLC) sampler with the speed-ups listed in Table 1 as the targets. This method treats the target speed-up on a best-effort basis, meaning it may exceed or fall below the intended value. This is undesirable, as perfectly matching the target speed-up is particularly important when including a song in the hyperlapse. If the hyperlapse is longer, part of the video will be left without an accompanying song. Conversely, if the song is longer, it must be truncated to fit the hyperlapse.

#### 4.5 Ablation Study

We conduct an ablation study to provide a comprehensive comparison between our method and its previously published version (SMH1). Each modification introduced since then is evaluated in isolation, with its results discussed separately.

For the **semantic cost**, in Equation 7, we place a soft constraint that the hyperlapse must slow down for important frames. This is an addition in relation to our previous work and is the first subject of our ablation study. Here, we replace  $C_v(v, s)$  with:

$$C_v(v) = 1 - V_v^* \quad (17)$$

to maximize the semantic cost without explicitly requiring the video to slow down.

For the **acceleration cost**, with Equation 10, our goal is to minimize the magnitude and frequency of changes in playback speed. This replaces the speed-up cost in our previous work and is the second subject of our ablation study. The speed-up cost, which is evaluated here, is defined as:

$$C_s(s) = \left( \frac{s-1}{\Delta-1} \right)^2, \quad (18)$$

to penalize large playback speeds.

The third modification to the cost function from our previous work is in the **cost normalization** of the audio alignment cost (Equation 9) and the acceleration cost (Equation 10). In our prior approach, we followed Joshi *et al.* [2015] and Matos *et al.* [2021], where speed-related costs exceeding the threshold  $\tau$  were truncated. In this ablation study, we apply the same truncation:

$$C_a(a, s) = \frac{1}{\tau} \min \{ (\hat{s}_a - s)^2, \tau \}, \quad (19)$$

and

$$C_s(s, s') = \frac{1}{\tau} \min \{ (s' - s)^2, \tau \}. \quad (20)$$

Joshi *et al.* [2015] noted that the results were not particularly sensitive to the exact value of  $\tau$ . Thus, we adopt the same  $\tau = 200$  that was used previously by the three works.

Another difference from our previous work is the addition of **parallelism** to Algorithm 1. To estimate its impact, we conducted experiments simulating an hour-long video recorded

at 60 FPS, resulting in 216,000 video frames, paired with a six-minute-long song consisting of 21,600 audio frames. Randomized data was used for both the video and song to simulate a worst-case scenario for the branch predictor, ensuring the results reflect the algorithmic improvements alone. We measured the speedup, defined as the ratio of wall clock time before and after parallelization, to capture the raw improvement compared to an idealized linear speedup. We tested the parallelized algorithm with up to 16 threads and compared its runtime to that of the sequential version. The tests were performed on a system with an AMD Ryzen 7 7800X3D processor (16 threads, 8 physical cores) and 32 GB of memory.

Lastly, we assess the impact of replacing our proposed dynamic programming algorithm with a computationally cheaper **greedy algorithm**, as described in Algorithm 3. While both use the same cost function as in Equation 6, the greedy algorithm no longer evaluates every possible path through the video, eliminating the need to compute and store the large dynamic programming matrix. Thus, our goal is to determine whether the dynamic programming algorithm achieves sufficiently better results to justify its higher computational requirements.

---

#### Algorithm 3 Greedy Frame Sampling

---

```

1:  $I \leftarrow [0]$ 
2:  $S \leftarrow [\mu]$   $\triangleright$  where  $\mu = \frac{|V|}{|A|}$ 
3: for  $a \in [1, |A|]$  do
4:    $c_{\min} \leftarrow \infty$ 
5:    $s_{\min} \leftarrow 0$ 
6:    $v_{\min} \leftarrow 0$ 
7:   for  $s \in [1, \Delta]$  do
8:      $v \leftarrow I[a-1] + s$ 
9:     if  $v < |V|$  then
10:       $c \leftarrow C(a, v, s, S[a-1])$ 
11:      if  $c < c_{\min}$  then
12:         $c_{\min} \leftarrow c$ 
13:         $s_{\min} \leftarrow s$ 
14:         $v_{\min} \leftarrow v$ 
15:   if  $c_{\min} = \infty$  then
16:     break
17:   Append  $s_{\min}$  to  $S$ 
18:   Append  $v_{\min}$  to  $I$ 
19: return  $I$ 
```

---

#### 4.6 Sensitivity Analysis

In this sensitivity analysis, we evaluate the impact of each component in the cost function by setting its corresponding weight to zero, as described in Equation 16 and Equation 6. This approach allows us to assess: (i) the effect of loudness-speed alignment on semantic retention, and vice versa; (ii) whether smoothing the playback speed curve negatively affects other aspects of the hyperlapse; and (iii) the contribution of the homography-based frame transition cost in reducing video shakiness.

## 4.7 Song Selection

We evaluate the performance of the song selection methods proposed in subsection 3.4.2, along with a baseline method, by comparing their effectiveness to the brute-force method—which serves as the ground truth—using two new metrics: overestimation and underestimation.

The **random method** serves as a baseline to assess how effectively the hybrid approach narrows the search space before applying the brute-force algorithm. Instead of selecting the top 100 candidates using the subsampled algorithm, this method randomly selects 100 songs. Since the random sample is unordered, we then sort it using the greedy algorithm.

The **overestimation** metric measures how highly an alternate method ranks a candidate that the brute-force method considers suboptimal. It reflects the position of the best candidate chosen by the alternate method in the brute-force ranking. If this candidate appears much lower in the brute-force ranking, it indicates that the alternate method has overestimated its quality. This metric helps evaluate how reliable the song suggested by the alternate method is in lieu of a brute-force search.

The **underestimation** metric assesses how poorly an alternate method ranks a candidate that the brute-force method considers optimal. It identifies the position of the brute-force method’s top candidate in the alternate method’s ranking. If the optimal candidate appears far down in the alternate ranking, it suggests that the alternate method has underestimated its quality. This metric is useful for determining whether running the brute-force method on the top 100 candidates identified by the alternate method is likely to recover the best result.

## 5 Results

We present the outcomes of our experiments evaluating the proposed methods. First, we assess the performance of our *frame sampling* approach against competitors. Next, we report the results of our *ablation study* and *sensitivity analysis*. Finally, we evaluate the effectiveness of our *song selection* algorithms.

### 5.1 Frame Sampling

The results of the **semantic score** evaluation are presented in Table 2. An example of how each method performs on a particular instance is shown in Figure 4, which also illustrates the unrealistic nature of the ground truth for semantic content, as it leaves unacceptable gaps between frames. As expected, both the uniform fast-forward and MH, which do not attempt to select important frames, exhibit the worst performance in this metric, with no significant differences between their scores. Being a semantic-based method, SAS2 has the next best scores, but is outperformed by SMH1 and our proposed method.

We attribute the large disparity from SAS2 to SMH1 and our approach to a difference in granularity. While SAS2 assigns speeds to entire segments based on their importance, our method works at the frame level. This finer granularity enables our method to more effectively choose semantic

**Table 2.** Evaluation of the semantic score, which quantifies how much semantic information is preserved in the accelerated video relative to the maximum possible for the same number of frames.

Video	Semantic (% , $\uparrow$ )				
	Uniform	SAS2	MH	SMH1	Ours
Biking 0p	14.71	15.28	14.86	91.13	<b>100.00</b>
Biking 25p	12.24	28.56	11.89	33.42	<b>35.26</b>
Biking 50p	19.78	27.10	19.33	<b>49.99</b>	48.76
Biking 50p 2	22.76	31.92	22.26	<b>79.94</b>	79.58
Driving 0p	13.64	34.36	12.54	93.91	<b>100.00</b>
Driving 25p	17.19	28.61	16.40	92.66	<b>99.97</b>
Driving 50p	13.47	18.92	12.40	<b>72.97</b>	72.46
Walking 0p	15.94	12.99	17.92	95.51	<b>100.00</b>
Walking 25p	20.10	32.71	19.62	57.20	<b>60.45</b>
Walking 50p	23.92	29.31	23.50	<b>60.46</b>	58.90
Walking 75p	39.24	47.51	39.08	<b>58.48</b>	53.71
Overall	19.36	27.93	19.07	71.42	<b>73.55</b>

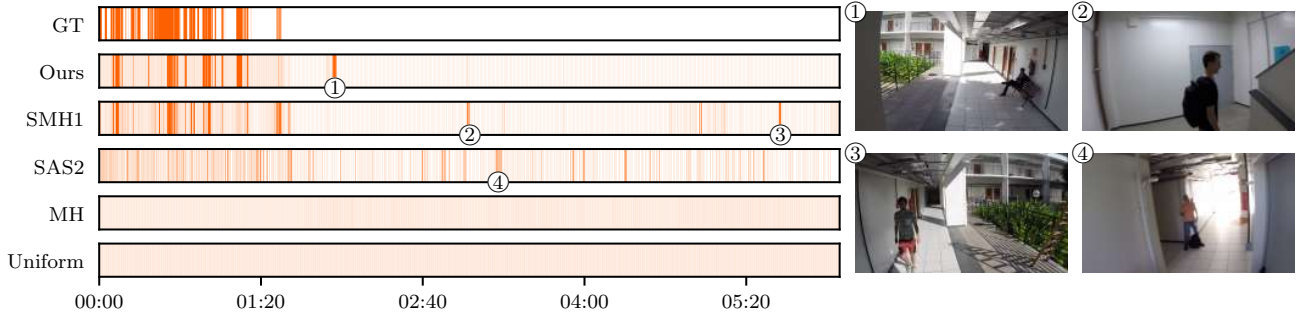
frames, especially in low-semantic videos (e.g., 0p and 25p) or within non-relevant segments that contain occasional important frames.

However, the frame-level granularity in SMH1 has a drawback: some important frames are played at high speeds, making them difficult to follow. To mitigate this, we introduced a penalty in Equation 7 for frames shown at high speeds, encouraging important frames to be played below the target speed-up rate. While this adjustment improves results for low-semantic videos, it leads to a decline in high-semantic ones (e.g., 50p and 75p). We justify this modification using the **low-speed semantic score** in Table 3. Under this evaluation, our method outperforms others in all cases, demonstrating that it explicitly slows down for important frames.

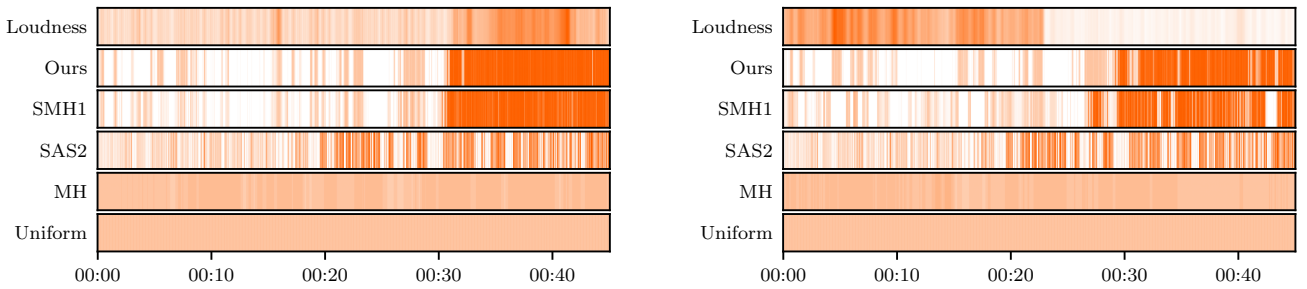
**Table 3.** Evaluation of the low-speed semantic score, which quantifies how much semantic information is preserved in frames shown below the target speed-up.

Video	Low-speed Semantic (% , $\uparrow$ )				
	Uniform	SAS2	MH	SMH1	Ours
Biking 0p	5.28	9.12	4.86	89.31	<b>100.00</b>
Biking 25p	4.44	25.87	9.58	30.32	<b>35.21</b>
Biking 50p	0.40	19.04	10.47	42.42	<b>48.34</b>
Biking 50p 2	10.51	24.48	7.45	74.00	<b>79.08</b>
Driving 0p	5.84	29.66	6.69	90.56	<b>100.00</b>
Driving 25p	1.50	23.55	13.43	88.69	<b>99.97</b>
Driving 50p	4.13	12.43	11.22	67.02	<b>72.22</b>
Walking 0p	14.27	4.80	14.56	94.80	<b>100.00</b>
Walking 25p	17.64	27.66	8.09	54.71	<b>60.44</b>
Walking 50p	10.55	19.50	2.81	52.00	<b>58.80</b>
Walking 75p	21.44	30.60	9.17	44.14	<b>51.33</b>
Overall	8.73	20.61	8.94	66.18	<b>73.22</b>

The results of the **correlation score** evaluation are presented in Table 4, showing that only our methods achieve a high correlation between song loudness and video speed-up. This is further illustrated in Figure 5, which presents examples of both high and low correlation. This result aligns with expectations for the uniform fast-forward and SAS2, which are non-musical methods. The poor performance of MH on



**Figure 4.** Comparison of frame sampling methods with the semantic ground truth, which represents the most important frames of the video, for *Walking 25p*. The dark regions indicate where frames were sampled, while the white regions represent skipped frames. The numbers highlight examples of segments selected despite not being part of the ground truth. This also demonstrates the unrealistic nature of the semantic ground truth, as it leaves unacceptable gaps between frames.



**(a)** The song starts quietly and becomes louder toward the end, making it a good fit for a hyperlapse that slows down at the start and accelerates towards the end. In this scenario, both the semantic content and the correlation can be maximized simultaneously.

**(b)** The song starts loud and ends quietly, which contrasts with the video's needs for a good match. In such cases, our method prioritizes maximizing semantic content over correlation.

**Figure 5.** Examples of well-matching and poorly-matching songs for the speed-up curve of the video in Figure 4, which slows down for important frames at the start and accelerates for less important frames at the end. This exemplifies the need for a song selection algorithm, as it is not always possible to simultaneously maximize both semantic content and the loudness-speed correlation.

this metric, despite being a musical method, suggests that their goal of aligning induced emotions between video and music is orthogonal to our goals of aligning loudness and playback speed.

**Table 4.** Correlation between video playback speed and song loudness. Each row presents the average correlation across all songs for the corresponding video.

Video	Correlation ( $\uparrow$ )				
	Uniform	SAS2	MH	SMH1	Ours
Biking 0p	0.00	0.00	0.01	0.78	<b>0.82</b>
Biking 25p	0.00	-0.01	0.01	<b>0.51</b>	0.47
Biking 50p	0.01	0.00	0.01	<b>0.76</b>	0.63
Biking 50p 2	0.00	0.00	0.01	0.45	<b>0.49</b>
Driving 0p	0.02	-0.01	0.00	0.80	<b>0.86</b>
Driving 25p	0.00	0.00	0.01	0.39	<b>0.40</b>
Driving 50p	0.00	0.00	0.01	0.45	<b>0.48</b>
Walking 0p	0.01	0.01	0.00	0.75	<b>0.80</b>
Walking 25p	0.01	0.01	0.01	0.39	<b>0.40</b>
Walking 50p	0.00	-0.01	0.02	<b>0.57</b>	0.49
Walking 75p	0.00	0.01	0.01	<b>0.71</b>	0.45
Overall	0.00	0.00	0.01	<b>0.60</b>	0.57

Compared to SMH1, our current method achieves a slightly lower correlation on average. This effect is particularly pronounced in videos such as *Walking 75p*, *Biking 50p*, and *Walking 50p*, where the decrease in correlation is signifi-

cantly higher. We attribute this decrease to the modification in Equation 7, which penalizes high playback speeds during important frames. Notably, when comparing their semantic scores in Table 2 to the semantic scores for frames below the mean speed in Table 3, these videos showed a significant drop in SMH1, while our current method remained nearly unaffected. This suggests that our current method imposes stronger constraints on speed, reducing its flexibility to align with loudness.

The results of the **acceleration score** evaluation are presented in Table 5, showing the acceleration behavior of each method. Because fractional skips between frames are not possible, the uniform fast-forward achieves fractional speed-ups by alternating between integer speeds. For instance, a speed-up of  $11.5\times$  (*Walking 75p*) is achieved by alternating between 11 and 12 for each frame. A fractional speed of 0.5 represents the worst-case scenario for this score. However, these high-frequency swaps are imperceptible, and we therefore consider the acceleration scores of the uniform method to be good baselines.

The MH achieves even lower acceleration scores than the uniform fast-forward, suggesting very few changes in speed. By observing its speed-up curves, we found that this method aligns the emotion curves between videos and songs while barely deviating from the mean speed-up.

On the other extreme, SAS2 has significantly higher acceleration scores than the other methods. This follows from the experiments made by Silva *et al.* [2021]: their LLC sampler

**Table 5.** Evaluation of acceleration, measured as the root mean square of successive differences in playback speed.

Video	Acceleration ( $\downarrow$ )				
	Uniform	SAS2	MH	SMH1	Ours
Biking 0p	0.84	6.25	<b>0.22</b>	1.92	1.12
Biking 25p	0.91	15.06	<b>0.33</b>	3.98	2.48
Biking 50p	<b>0.23</b>	11.63	0.45	3.50	2.97
Biking 50p 2	0.97	7.64	<b>0.27</b>	2.76	2.37
Driving 0p	<b>0.19</b>	7.24	0.37	2.93	1.70
Driving 25p	0.43	7.03	<b>0.41</b>	3.20	2.21
Driving 50p	0.81	8.64	<b>0.27</b>	4.77	3.75
Walking 0p	0.43	4.84	<b>0.26</b>	2.61	1.69
Walking 25p	0.54	7.87	<b>0.28</b>	3.19	2.22
Walking 50p	0.95	12.66	<b>0.27</b>	3.75	2.78
Walking 75p	0.99	12.02	<b>0.23</b>	3.64	2.33
Overall	0.66	9.17	<b>0.30</b>	3.29	2.33

leaves large gaps between scenes, which are interpreted as abrupt jumps in speed by the acceleration score.

Our method achieve acceleration scores between those of MH and SAS2. Having an acceleration score as low as MH is undesirable for our method, as we intentionally use speed changes to convey information. On the other hand, large acceleration scores are symptoms of gaps between scenes, as seen in SAS2, which break the temporal continuity of the video. In our work, the constraint to never exceed a maximum speed (*i.e.*,  $\Delta$ ) prevents gaps between scenes. Compared to SMH1, our current method replaces the *speed-up cost*, which simply penalized high speeds, with a more effective *acceleration cost* (Equation 10) that smooths speed changes.

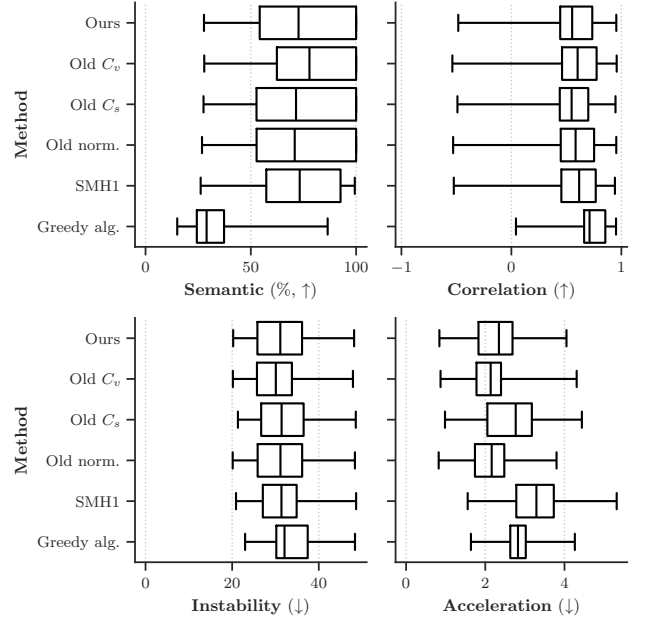
The results of the **video instability** evaluation are presented in Table 6. Despite incorporating four optimization objectives and our choice of hyperparameters assigning a lower weight to the *frame matching cost* compared to the other objectives, our method achieves, on average, comparable instability performance to SAS2 and SMH1 while outperforming uniform fast-forward and MH.

**Table 6.** Evaluation of video instability, measured as the pixel-wise standard deviation of consecutive frames within a sliding window.

Video	Instability ( $\downarrow$ )				
	Uniform	SAS2	MH	SMH1	Ours
Biking 0p	24.05	21.71	24.08	22.49	<b>21.70</b>
Biking 25p	49.98	<b>45.01</b>	50.21	47.52	47.32
Biking 50p	32.97	<b>29.22</b>	32.71	30.43	30.98
Biking 50p 2	24.55	23.99	24.77	<b>21.86</b>	21.96
Driving 0p	42.81	<b>37.63</b>	42.80	39.26	37.89
Driving 25p	36.67	31.42	36.54	31.59	<b>30.79</b>
Driving 50p	38.54	33.87	38.72	32.63	<b>32.38</b>
Walking 0p	28.37	28.25	28.63	26.58	<b>25.23</b>
Walking 25p	32.45	30.06	32.74	28.92	<b>28.48</b>
Walking 50p	34.28	33.30	34.54	<b>31.57</b>	32.47
Walking 75p	37.69	<b>33.51</b>	38.00	35.20	36.31
Overall	34.76	31.63	34.89	31.64	<b>31.41</b>

## 5.2 Ablation Study

The results of the ablation study are presented in Figure 6 and discussed throughout this subsection. For reference, we also include the previous version of our method (SMH1), which was analyzed in subsection 5.1.

**Figure 6.** Box plots showing results for: (i) our unablated method, (ii) semantic cost ablation, (iii) acceleration cost ablation, (iv) cost normalization ablation, (v) the previous version of our method, and (vi) the greedy algorithm.

The results of the **semantic cost** ( $C_v$ ) ablation align with the analysis of the *video semantic score* against competitors. Although the unablated method has a lower semantic score, this trade-off enables it to slow down for important frames, as shown in Table 7. Compared to SMH1, the version using the previous  $C_v$  achieves a higher semantic score, suggesting that the other modifications collectively make it more difficult for the hyperlapse to slow down during important segments.

**Table 7.** Evaluation of the low-speed semantic score, which quantifies how much semantic information is preserved in frames shown below the target speed-up.

Video	Low-speed Semantic (% $\uparrow$ )		
	Ours	Old $C_v$	SMH1
Biking 0p	<b>100.00</b>	98.07	89.31
Biking 25p	<b>35.21</b>	32.25	30.32
Biking 50p	<b>48.34</b>	45.28	42.42
Biking 50p 2	<b>79.08</b>	76.14	74.00
Driving 0p	<b>100.00</b>	96.18	90.56
Driving 25p	<b>99.97</b>	96.03	88.69
Driving 50p	<b>72.22</b>	71.02	67.02
Walking 0p	<b>100.00</b>	99.16	94.80
Walking 25p	<b>60.44</b>	58.25	54.71
Walking 50p	<b>58.80</b>	54.46	52.00
Walking 75p	<b>51.33</b>	40.65	44.14
Overall	<b>73.22</b>	69.77	66.18

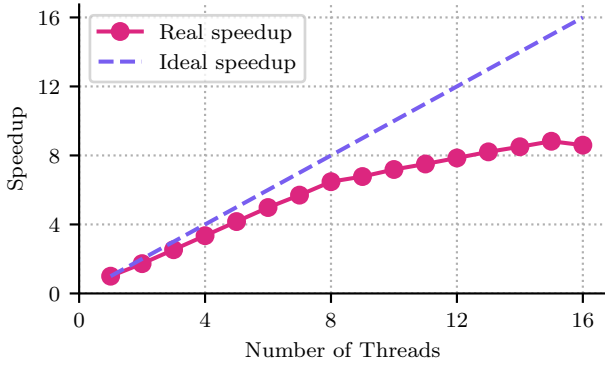
As observed in the sensitivity analysis (subsection 5.3), the unablated version of the **acceleration cost** ( $C_s$ ) effectively



smooths changes in playback speed without negatively impacting the other metrics. Specifically, we show here that the old  $C_s$  performs worse on the acceleration metric while yielding equal results on the other metrics, thus justifying its replacement.

We find that modifying the **cost normalization** in Equation 9 and Equation 10 has no significant impact on the results. Thus, while not strictly necessary, we consider this change a simplification for convenience: in our current formulation, only  $\Delta$  controls the maximum playback speed, whereas previously,  $\tau$  also had to be adjusted simultaneously.

The results of the **parallelism** ablation demonstrate significant improvements in execution time, as illustrated in Figure 7. The parallelized algorithm, executed with 15 threads, completed in 32 seconds compared to 279 seconds for the sequential version—an 88% reduction in wall clock time. In practice, since speedup is not perfectly linear, task-level parallelism can provide better performance for multiple independent runs (e.g., during hyperparameter tuning or song comparison) by reducing communication overhead. Conversely, larger problem instances benefit more from internal parallelism, as memory constraints limit the number of simultaneous runs, and greater efficiency is achieved by processing more work per iteration.



**Figure 7.** Parallelization achieves near-linear speedup up to 8 threads, with diminishing efficiency beyond that, likely due to the distinction between physical and virtual cores in the experimental setup.

We observed that the **greedy algorithm** prioritizes loudness–speed alignment significantly more than semantic retention, despite the higher weight placed on semantics. While the exact cause is not fully clear, we believe that the restriction on selecting an exact number of frames in our unablated algorithm acts as regularization for the *audio alignment cost*. By removing this restriction, the cost becomes unbounded across the entire sequence, amplifying its influence.

Furthermore, Table 8 shows that the greedy method fails to achieve the target hyperlapse length without large gaps. It resulted in large video truncations in 4 cases and audio truncations in 5 cases, with only 2 cases showing minor deviations. As previously discussed, these truncations are highly undesirable. Video truncation leaves a large gap at the end, potentially making the viewer unaware of any missing content. A possible workaround is selecting more frames, but this creates a segment of video without background music. Audio truncation, on the other hand, results in an abrupt and

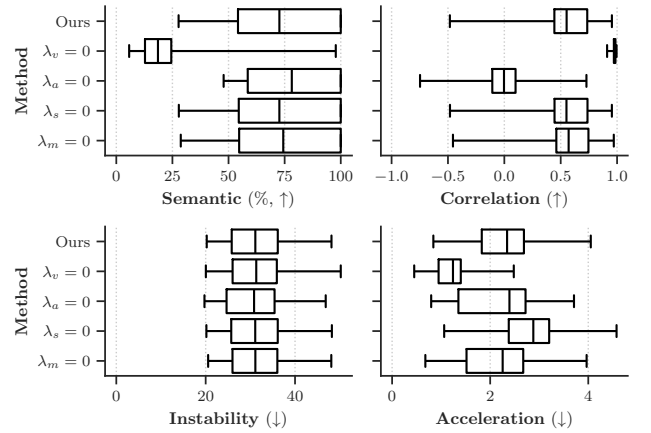
awkward stop as the song is cut short.

**Table 8.** Truncations caused by the greedy algorithm. Video truncation occurs when the song ends before the video is finished, while audio truncation happens when the video ends before the song is completed.

Video	Truncation (% , ↓)	
	Video	Audio
Biking 0p	<b>0.01</b>	31.42
Biking 25p	30.08	<b>0.00</b>
Biking 50p	22.59	<b>0.00</b>
Biking 50p 2	<b>0.01</b>	16.99
Driving 0p	<b>0.01</b>	24.91
Driving 25p	<b>0.01</b>	28.41
Driving 50p	<b>0.06</b>	2.90
Walking 0p	<b>0.01</b>	34.86
Walking 25p	1.74	<b>1.48</b>
Walking 50p	23.61	<b>0.00</b>
Walking 75p	35.27	<b>0.00</b>
Overall	<b>10.31</b>	12.82

### 5.3 Sensitivity Analysis

Figure 8 illustrates how removing each component of the cost function affects the results of our method.



**Figure 8.** Sensitivity analysis results, aggregated over the entire dataset.

Removing the **video semantic cost** ( $\lambda_v = 4 \rightarrow \lambda_v = 0$ ) results in a substantial decline in the semantic metric, as frame selection becomes dominated by the song. Consequently, the loudness–speed correlation consistently reaches its maximum value. Moreover, since the song loudness curve is inherently smooth, the playback speed curve becomes further smoothed through this correlation, leading to the lowest acceleration score among all ablations.

Removing the **audio alignment cost** ( $\lambda_a = 2 \rightarrow \lambda_a = 0$ ) reduces the median correlation between speed-up and loudness to zero. As shown in Table 4, this suggests that a method must explicitly optimize for alignment to consistently achieve high correlation. Compared to the unablated method, the increase in semantic score is relatively small, which we believe is due to (i) the originally lower weight of this term and (ii) how small changes in early frames can propagate and influence the alignment of the entire video.

Removing the **acceleration cost** ( $\lambda_s = 1 \rightarrow \lambda_s = 0$ ) worsens the acceleration score without affecting the other metrics. Thus, we consider this cost effective at smoothing out speed transitions, even with its smaller weight relative to the previous two costs.

Removing the **frame matching cost** ( $\lambda_m = 1 \rightarrow \lambda_m = 0$ ) does not impact the evaluated metrics, including, unexpectedly, the instability metric. However, further analysis revealed that this removal leads to a tenfold increase in the *homography failure rate*, as shown in Table 9. Therefore, we consider this term important in the optimization cost to support homography-based video stabilizers, such as the one proposed by Silva *et al.* [2016].

**Table 9.** Evaluation of the proportion of frame-to-frame transitions in the output video where homography estimation failed. The worst value in each row is underlined.

Video	Homography Failure Rate (%), $\downarrow$				
	Ours	$\lambda_v = 0$	$\lambda_a = 0$	$\lambda_s = 0$	$\lambda_m = 0$
Biking 0p	1.09	0.04	1.07	1.09	<u>19.31</u>
Biking 25p	3.09	0.10	4.00	3.03	<u>38.14</u>
Biking 50p	1.33	0.04	1.52	1.28	<u>29.39</u>
Biking 50p 2	0.36	0.04	0.59	0.48	<u>8.55</u>
Driving 0p	0.62	0.07	0.59	0.61	<u>38.85</u>
Driving 25p	3.34	0.07	3.11	3.27	<u>31.73</u>
Driving 50p	2.70	0.07	2.96	2.79	<u>27.99</u>
Walking 0p	1.76	1.62	1.70	1.71	<u>22.11</u>
Walking 25p	1.86	0.76	2.37	1.86	<u>19.19</u>
Walking 50p	1.52	0.52	2.22	1.46	<u>14.98</u>
Walking 75p	0.36	0.07	0.74	0.34	<u>11.19</u>
Overall	1.64	0.31	1.90	1.63	<u>23.77</u>

## 5.4 Song Selection

Here, we provide the results of our song selection experiments described in subsection 4.7. In Table 10, we assess each method’s agreement with the brute-force rankings using overestimation and underestimation metrics. Additionally, Table 11 compares the runtime of the methods to highlight their computational efficiency.

The **random method** avoids extreme overestimation or underestimation but struggles to consistently find the optimal match. Moreover, the marginal time savings over the hybrid method do not justify its limited accuracy, as the subsampled method takes only a small amount of time in the first place.

While all methods benefit from parallelization, we consider it essential for the **brute-force method** to reduce its runtime to a more manageable 1–4 minutes, making it viable in scenarios where computational resources are not a limiting factor.

If time is a critical concern and parallelization is unavailable, running only the **subsampled method** provides a reasonable alternative, delivering results with acceptable accuracy in a fraction of the time.

Ultimately, we consider the **hybrid method** to offer the best trade-off between runtime and quality. As indicated by the bolded values in Table 10, it often is able to identify the optimal song due to the underestimation of the subsampled

**Table 10.** Performance of song selection methods across different videos. Lower values indicate better agreement with the brute-force rankings. Values within the top-100 are emboldened. Entries marked as – means that the best song is not in the top-100.

Video	Overestimation ( $\downarrow$ )		Underestimation ( $\downarrow$ )	
	Random	Subsample	Random	Subsample
Biking 0p	<b>13</b>	<b>38</b>	–	<b>31</b>
Biking 25p	<b>5</b>	<b>1</b>	–	<b>68</b>
Biking 50p 2	<b>5</b>	<b>15</b>	–	<b>28</b>
Biking 50p	<b>3</b>	<b>6</b>	–	585
Driving 0p	<b>54</b>	<b>14</b>	–	<b>35</b>
Driving 25p	<b>24</b>	<b>0</b>	–	<b>0</b>
Driving 50p	<b>2</b>	<b>21</b>	–	<b>8</b>
Walking 0p	<b>1</b>	180	–	<b>57</b>
Walking 25p	<b>14</b>	<b>13</b>	–	<b>2</b>
Walking 50p	<b>0</b>	<b>11</b>	<b>0</b>	<b>17</b>
Walking 75p	<b>6</b>	<b>6</b>	–	206

**Table 11.** Runtime comparison of song selection methods for each video. Times are calculated assuming no parallelization.

Video	Runtime (mm:ss, $\downarrow$ )			
	Brute Force	Subsampled	Hybrid	Random
Biking 0p	116:36	1:01	7:42	6:37
Biking 25p	45:23	0:21	2:56	2:35
Biking 50p 2	109:33	0:57	7:11	6:13
Biking 50p	179:35	1:27	11:35	10:12
Driving 0p	34:35	0:17	2:16	1:58
Driving 25p	31:52	0:16	2:04	1:49
Driving 50p	42:03	0:21	2:44	2:23
Walking 0p	27:35	0:16	1:51	1:34
Walking 25p	41:21	0:20	2:42	2:21
Walking 50p	39:41	0:21	2:37	2:16
Walking 75p	39:34	0:23	2:38	2:14
Average	64:21	0:33	4:12	3:39

method being below 100, effectively combining its strengths with those of the brute-force method.

## 6 Conclusions

In this work, we introduced a method that combines concepts from *semantic hyperlapses* and *musical hyperlapses*. Building on the goal of fast-forwarding first-person videos while preserving important content and minimizing discomfort from camera motion, we also incorporate music in a way that complements the hyperlapse. To achieve this, we propose an approach that (i) maximizes semantic content retention, (ii) aligns playback speed with the loudness of a chosen song, (iii) reduces shakiness, and (iv) minimizes abrupt speed transitions. Our method produces *semantic and musical hyperlapses* that are both useful and enjoyable to watch.

Our experiments demonstrate that our method outperforms the previous state-of-the-art [Silva *et al.*, 2021] in semantic retention, achieving an average semantic score of 73.6% compared to 27.9%. Moreover, our method is the only one among the competitors to achieve a non-zero loudness–speed correlation, while maintaining comparable performance in

video stability and smooth speed transitions.

When the video and song are well-matched, both the semantic and correlation scores can be maximized, but a poor match forces a trade-off between them. With a large collection of songs available, this trade-off can be mitigated by selecting the best match, though manually evaluating each option is impractical. To automate this process, we proposed three approaches: (i) a brute-force method that is exact but computationally expensive, (ii) an approximate method that subsamples the video and song, and (iii) a hybrid approach that first narrows the selection to the top 100 candidates using the approximate method before applying brute-force refinement. Our results show that the hybrid approach offers the best balance between runtime and quality, consistently identifying the best match while maintaining reasonable execution time.

## 6.1 Limitations

We identified three main limitations in our method. First, the pre-processing phase—and the video composition phase when stabilization is applied—take significantly longer than the main algorithm. Second, our method treats playback speeds as linear, even though human perception of speed changes is nonlinear. For example, the difference between  $1\times$  and  $2\times$  speed feels greater than the difference between  $2\times$  and  $3\times$  speed. Finally, because our frame sampling algorithm relies on global optimization, changes in one part of the video can influence decisions in entirely unrelated sections, potentially causing the hyperlapse to slow down arbitrarily in one region to improve the overall score elsewhere.

## 6.2 Future Works

One potential direction for future research is developing an end-to-end neural network that processes the video in a single pass, replacing our feature extraction, dynamic programming algorithm, and composition steps. Neural networks have already been explored for semantic fast-forwarding, with reinforcement learning used to train fast-forward agents. In our case, a similar reinforcement learning approach could be adopted, or our method could be leveraged to generate ground-truth labels for a supervised learning setup.

Another promising avenue is exploring generative models to create songs that align with the speed-up curve of the video. In our current method, the speed-up rate is constrained by the length of the chosen song. For long videos (e.g., two hours or more), finding a suitable song of the required length is challenging. Generating the song would allow users to instantly obtain a soundtrack tailored to the video's duration and desired speed-up profile.

## Declarations

### Authors' Contributions

RN was responsible for the methodology, experiments, software, and original draft of the manuscript. LF contributed to the experiments

and manuscript review. MS was responsible for the conceptualization, funding acquisition, supervision, and manuscript review. All authors read and approved the final manuscript.

## Competing interests

The authors declare that they have no competing interests.

## Funding

The authors would like to thank CAPES, CNPq, FAPEMIG and Finep for supporting this project.

## Availability of data and materials

The implementation of the method proposed in this study is available at [https://github.com/MaVILab-UFV/SemanticMusicalHyperlapse\\_JBCS\\_2025](https://github.com/MaVILab-UFV/SemanticMusicalHyperlapse_JBCS_2025).

## References

- Aljanaki, A., Yang, Y.-H., and Soleymani, M. (2017). Developing a benchmark for emotional analysis of music. *PLOS ONE*, 12(3):1–22. DOI: 10.1371/journal.pone.0173392.
- Baráth, D., Noskova, J., Ivashechkin, M., and Matas, J. (2020). Magsac++, a fast, reliable and accurate robust estimator. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1301–1309. DOI: 10.1109/CVPR42600.2020.00138.
- Cheng, K.-Y., Luo, S.-J., Chen, B.-Y., and Chu, H.-H. (2009). Smartplayer: user-centric video fast-forwarding. In *Conference on Human Factors in Computing Systems*, pages 789–798. DOI: 10.1145/1518701.1518823.
- del Molino, A. G., Tan, C., Lim, J.-H., and Tan, A.-H. (2017). Summarization of egocentric videos: A comprehensive survey. *IEEE Transactions on Human-Machine Systems*, 47(1):65–76. DOI: 10.1109/THMS.2016.2623480.
- European Broadcasting Union (2023). *Loudness Metering: 'EBU Mode' Metering to Supplement Loudness Normalisation*. Available at: <https://tech.ebu.ch/publications/tech3341>.
- Furlan, V. S., Bajcsy, R., and Nascimento, E. R. (2018). Fast forwarding egocentric videos by listening and watching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshop on Sight and Sound*, pages 2504–2507. DOI: 10.48550/arXiv.1806.04620.
- Halperin, T., Poleg, Y., Arora, C., and Peleg, S. (2018). Egosampling: Wide view hyperlapse from egocentric videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(5):1248–1259. DOI: 10.1109/TCSVT.2017.2651051.
- Hamming, R. W. (1950). Error detecting and error correcting codes. *The Bell System Technical Journal*, 29(2):147–160. DOI: 10.1002/j.1538-7305.1950.tb00463.x.
- Joshi, N., Kienzle, W., Toelle, M., Uyttendaele, M., and Cohen, M. F. (2015). Real-time hyperlapse creation via optimal frame selection. *ACM Trans. Graph.*, 34(4). DOI: 10.1145/2766954.

- Karpenko, A. (2014). The technology behind hyperlapse from instagram. Available at: <http://instagram-engineering.tumblr.com/post/95922900787/hyperlapse>.
- Kopf, J., Cohen, M., and Szeliski, R. (2014). First-person hyperlapse videos. In *ACM Transactions on Graphics (Proc. SIGGRAPH 2014)*, volume 33. ACM - Association for Computing Machinery. DOI: 10.1145/2601097.2601195.
- Lu, L., Liu, D., and Zhang, H.-J. (2006). Automatic mood detection and tracking of music audio signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):5–18. DOI: 10.1109/TSA.2005.860344.
- Matos, D. d., Ramos, W., Romanhol, L., and Nascimento, E. R. (2021). Musical hyperlapse: A multimodal approach to accelerate first-person videos. In *34th SIBGRAPI Conference on Graphics, Patterns and Images*, pages 184–191. DOI: 10.1109/SIBGRAPI54419.2021.00033.
- Matos, D. d., Ramos, W., Silva, M., Romanhol, L., and Nascimento, E. R. (2023). A multimodal hyperlapse method based on video and songs' emotion alignment. *Pattern Recognition Letters*, 166:174–181. DOI: 10.1016/j.patrec.2022.08.014.
- Nepomuceno, R., Ferreira, L., and Silva, M. (2024). A multimodal frame sampling algorithm for semantic hyperlapses with musical alignment. In *Proceedings of the 37th Conference on Graphics, Patterns and Images (SIBGRAPI)*. DOI: 10.1109/SIBGRAPI62404.2024.10716336.
- Neves, A. C., Silva, M. M., Campos, M. F. M., and Nascimento, E. R. (2020). A gaze driven fast-forward method for first-person videos. In *Proceedings of the Sixth International Workshop on Egocentric Perception, Interaction and Computing (EPIC) at the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–4. DOI: 10.48550/arXiv.2006.05569.
- Ogawa, M., Yamasaki, T., and Aizawa, K. (2017). Hyperlapse generation of omnidirectional videos by adaptive sampling based on 3d camera positions. In *IEEE International Conference on Image Processing (ICIP)*, pages 2124–2128. DOI: 10.1109/ICIP.2017.8296657.
- Okamoto, M. and Yanai, K. (2014). Summarization of egocentric moving videos for generating walking route guidance. In Klette, R., Rivera, M., and Satoh, S., editors, *Image and Video Technology*, pages 431–442. DOI: 10.1007/978-3-642-53842-1\_37.
- Poleg, Y., Halperin, T., Arora, C., and Peleg, S. (2015). Egosampling: Fast-forward and stereo for egocentric videos. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4768–4776. DOI: 10.1109/CVPR.2015.7299109.
- Ramos, W., Silva, M., Araujo, E., Marcolino, L. S., and Nascimento, E. (2020a). Straight to the point: Fast-forwarding videos via reinforcement learning using textual data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10928–10937. DOI: 10.1109/CVPR42600.2020.01094.
- Ramos, W., Silva, M., Araujo, E., Moura, V., Oliveira, K., Soriano Marcolino, L., and Nascimento, E. (2022). Text-driven video acceleration: A weakly-supervised reinforcement learning method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1. DOI: 10.1109/TPAMI.2022.3157198.
- Ramos, W. L. S., Silva, M. M., Araujo, E. R., Neves, A. C., and Nascimento, E. R. (2020b). Personalizing fast-forward videos based on visual and textual features from social network. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 3260–3269. DOI: 10.1109/WACV45572.2020.9093330.
- Ramos, W. L. S., Silva, M. M., Campos, M. F. M., and Nascimento, E. R. (2016). Fast-forward video based on semantic extraction. In *IEEE International Conference on Image Processing (ICIP)*, pages 3334–3338. DOI: 10.1109/ICIP.2016.7532977.
- Rani, P., Jangid, A., Namboodiri, V. P., and Venkatesh, K. S. (2018). Visual odometry based omni-directional hyperlapse. In Rameshan, R., Arora, C., and Dutta Roy, S., editors, *Computer Vision, Pattern Recognition, Image Processing, and Graphics*, pages 3–13. DOI: 10.1007/978-981-13-0020-2\_1.
- Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571. DOI: 10.1109/ICCV.2011.6126544.
- Silva, M., Ramos, W., Campos, M., and Nascimento, E. R. (2021). A sparse sampling-based framework for semantic fast-forward of first-person videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(4):1438–1444. DOI: 10.1109/TPAMI.2020.2983929.
- Silva, M., Ramos, W., Ferreira, J., Chamone, F., Campos, M., and Nascimento, E. R. (2018a). A weighted sparse sampling and smoothing frame transition approach for semantic fast-forward first-person videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2383–2392. DOI: 10.1109/CVPR.2018.00253.
- Silva, M. M., Ramos, W. L. S., Chamone, F. C., Ferreira, J. P. K., Campos, M. F. M., and Nascimento, E. R. (2018b). Making a long story short: A multi-importance fast-forwarding egocentric videos with the emphasis on relevant objects. *Journal of Visual Communication and Image Representation*, 53:55–64. DOI: 10.1016/j.jvcir.2018.02.013.
- Silva, M. M., Ramos, W. L. S., Ferreira, J. P. K., Campos, M. F. M., and Nascimento, E. R. (2016). Towards semantic fast-forward and stabilized egocentric videos. In *European Conference on Computer Vision Workshops*, pages 557–571. DOI: 10.1007/978-3-319-46604-0\_40.
- Spearman, C. (1904). The proof and measurement of association between two things. *The American Jour. of Psych.*, 15(1):72–101. DOI: 10.2307/1412159.
- Steuer, R. E. and Choo, E.-U. (1983). An interactive weighted tchebycheff procedure for multiple objective programming. *Mathematical programming*, 26:326–344. DOI: 10.1007/BF02591870.
- Szabo, A., Boucher, K., Carroll, W., Klebanov, L., Tsodikov, A., and Yakovlev, A. (2002). Variable selection and pattern recognition with gene expression data generated by the microarray technology. *Mathematical Biosciences*, 176(1):71–98. DOI: 10.1016/S0025-5564(01)00103-1.
- Tsodikov, A., Szabo, A., and Jones, D. (2002). Adjustments and measures of differential expression for microarray data.

*Bioinformatics*, 18(2):251–260. DOI: 10.1093/bioinformatics/18.2.251.

von Neumann, J., Kent, R. H., Bellinson, H. R., and Hart, B. I. (1941). The Mean Square Successive Difference. *The Annals of Mathematical Statistics*, 12(2):153 – 162. DOI: 10.1214/aoms/1177731746.

Wang, M., Liang, J.-B., Zhang, S.-H., Lu, S.-P., Shamir, A., and Hu, S.-M. (2018). Hyper-lapse from multiple spatially-overlapping videos. *IEEE Transactions on Image Processing*, 27(4):1735–1747. DOI: 10.1109/TIP.2017.2749143.

Zwicker, E. and Fastl, H. (2013). *Psychoacoustics: Facts and models*, volume 22. Springer Science & Business Media. DOI: 10.1007/978-3-540-68888-4.