UNIVERSIDADE FEDERAL DE VIÇOSA

A Framework for Semantic and Musical Hyperlapses

Raphael Carmo Silva Nepomuceno
Magister Scientiae

VIÇOSA - MINAS GERAIS
2025

RAPHAEL CARMO SILVA NEPOMUCENO

A Framework for Semantic and Musical Hyperlapses

Dissertation submitted to the Computer Science Graduate Program of the Universidade Federal de Viçosa in partial fulfillment of the requirements for the degree of *Magister Scientiae*.

Adviser: Michel Melo da Silva

VIÇOSA - MINAS GERAIS
2025

Ficha catalográfica elaborada pela Biblioteca Central da Universidade Federal de Viçosa - Campus Viçosa

T
N441f
2025

Nepomuceno, Raphael Carmo Silva, 1997-
A framework for semantic and musical hyperlapses / Raphael Carmo Silva Nepomuceno. – Viçosa, MG, 2025.
1 dissertação eletrônica (68 f.): il. (algumas color.).

Texto em inglês.
Orientador: Michel Melo da Silva.
Dissertação (mestrado) - Universidade Federal de Viçosa, Departamento de Informática, 2025.
Referências bibliográficas: f. 63-68.
DOI: https://doi.org/10.47328/ufvbbt.2025.688
Modo de acesso: World Wide Web.

1. Gravação em vídeo - Simulação por computador . I. Silva, Michel Melo da, 1990-. II. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDD 22. ed. 006.7

Bibliotecário(a) responsável: Euzébio Luiz Pinto CRB-6/3317

RAPHAEL CARMO SILVA NEPOMUCENO

A Framework for Semantic and Musical Hyperlapses

Dissertation submitted to the Computer Science Graduate Program of the Universidade Federal de Viçosa in partial fulfillment of the requirements for the degree of *Magister Scientiae*.

APPROVED: January 31, 2025.

Assent:

Raphael Carmo Silva Nepomuceno
Author

Michel Melo da Silva
Adviser

ACKNOWLEDGMENTS

This work has been sponsored by the following Brazilian research agencies: Coordination for the Improvement of Higher Education Personnel (CAPES; Financing code 001), Minas Gerais State Foundation for Research Aid (FAPEMIG) and National Council of Scientific and Technological Development (CNPq).

ABSTRACT

NEPOMUCENO, Raphael Carmo Silva, M.Sc., Universidade Federal de Viçosa, January, 2025. **A Framework for Semantic and Musical Hyperlapses**. Adviser: Michel Melo da Silva.

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 egocentric 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 dissertation, 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; egocentric videos; hyperlapse; loudness

RESUMO

NEPOMUCENO, Raphael Carmo Silva, M.Sc., Universidade Federal de Viçosa, janeiro de 2025. **Uma Abordagem para *Hyperlapses* Semânticos e Musicais**. Orientador: Michel Melo da Silva.

Com a crescente prevalência de câmeras portáteis—como smartphones, câmeras de ação e óculos inteligentes—gravar vídeos em primeira pessoa das atividades diárias tornou-se cada vez mais comum. No entanto, essas gravações frequentemente sofrem com filmagens tremidas causadas pelos movimentos contínuos do usuário, tornando-as fisicamente desconfortáveis de assistir, além de incluírem segmentos repetitivos ou irrelevantes que as tornam tediosas de acompanhar. Para lidar com esses desafios, os métodos de *hyperlapse* aceleram vídeos egocêntricos enquanto estabilizam o movimento da câmera, e os métodos de *hyperlapse* semântico preservam adicionalmente os segmentos mais importantes. Embora o áudio seja uma parte importante da experiência de assistir vídeos, ele é frequentemente negligenciado na criação de *hyperlapses*, deixando a escolha da trilha sonora para o usuário. Nesta dissertação, apresentamos um algoritmo multimodal de *hyperlapse* que otimiza conjuntamente a retenção de conteúdo semântico, a estabilidade visual e o alinhamento da reprodução com o volume de uma música escolhida pelo usuário. Especificamente, o *hyperlapse* desacelera durante as partes silenciosas da música para destacar quadros importantes e acelera durante os segmentos mais altos para minimizar a ênfase em conteúdos menos críticos. Além disso, propomos estratégias para selecionar músicas que melhor complementem o *hyperlapse*. Nossos experimentos mostram que essa abordagem supera os métodos existentes em retenção semântica e na correlação entre intensidade sonora e velocidade, enquanto mantém níveis comparáveis de estabilidade de câmera e continuidade temporal.

Palavras-chave: resumo de vídeo; aceleração semântica; vídeos egocêntricos; *hyperlapse*; intensidade sonora

List of Figures

List of Tables

List of Algorithms

Contents

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 action cameras, 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 (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; 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 handheld, body-mounted, or head-mounted cameras, as in Figure 1, 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.

Figure 1 – Three types of cameras which facilitate recording of first-person videos: smartphones, action cameras and smart glasses, respectively.[1]

Sound also 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.

1.1 Contributions

In this work, we introduce 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 a dynamic programming formulation that selects the optimal set of frames to retain in the hyperlapse by minimizing a cost function that accounts for semantic retention, playback alignment with song loudness, smoothness of speed transitions, and visual stability. The resulting *semantic and musical hyperlapses* are both useful and enjoyable to watch.

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

[1] Available at: https://www.pexels.com/photo/person-holding-silver-iphone-6-taking-photo-105254/; https://gopro.com/en/us/shop/mounts-accessories/sports-kit/AKTAC-001.html; https://www.meta.com/smart-glasses/skyler-shiny-chalky-gray-pink/. Accessed January 11, 2025.

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). Rank normalization (Tsodikov et al., 2002) is applied to both 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 usage, memory efficiency, and parallelism to improve the viability of its implementation on real hardware.

We compare the frame sampling performance of our method to two state-of-the-art competitors that share the same primary goal as our work: 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. Our method outperforms both in semantic retention and is the only one to consistently achieve a non-zero loudness–speed correlation, while maintaining comparable performance in video stability and speed smoothness.

We also 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 approach which combines the two. We compare our proposed methods to a baseline loosely inspired by the song selector in Matos et al. (2023), which otherwise is not directly comparable due to differences in optimization goals and the need to explicitly account for loudness–speed correlation.

The reference implementation of our method is available at: `https://github.com/MaVILab-UFV/SemanticMusicalHyperlapse_Dissertation`.[2]

Part of this work was published at the 37th Conference on Graphics, Patterns and Images (SIBGRAPI) in 2024. An extended version was published at the Journal of the Brazilian Computer Society (JBCS) in 2025.

1.2 Document Structure

The remainder of this document is organized as follows. Chapter 2 introduces the theoretical background, and chapter 3 reviews related work for video summarization and

[2] Accessed January 11, 2025.

hyperlapse. Chapter 4 describes our methodology, including feature extraction, frame selection, composition, and song selection. The experimental setup and results are presented in chapter 5 and chapter 6, respectively. Finally, chapter 7 summarizes the contributions, limitations and future work.

2 Theoretical Background

In this chapter, we present the core concepts that underpin our method. These include video semantic, which serves as a measure of content importance; homography, used to estimate shakiness during frame transitions; and audio loudness, which defines the musical feature to which playback speed is aligned. We also review supporting mathematical tools: rank normalization, which improves the robustness of frame selection, and correlation, which quantifies the alignment between playback speed and loudness.

2.1 Video Semantic

Video semantic refers to a measure of importance assigned to parts of a video based on user-defined criteria. Its definition is inherently open-ended and depends on the recorder's or viewer's preferences regarding the relevance of objects, regions, or events over time—for example, emphasizing visually salient content, navigational cues, or social interactions.

In addition to manual annotations, automatic annotations of video semantics include examples such as the path traversed by the recorder (Okamoto; Yanai, 2014), the presence of people or objects in the scene (Ramos et al., 2016; Silva et al., 2016; Ramos et al., 2020b), acoustic features extracted from the video's audio stream (Furlan et al., 2018), the recorder's attention inferred from gaze patterns (Neves et al., 2020), and the emotions induced by the video content (Matos et al., 2021).

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. Conversely, the absence of people or their distance from the viewer signifies lower relevance. This choice is motivated by the existence of the Annotated Semantic Dataset (Silva et al., 2016), which consists of eleven videos of daily activities such as walking, biking, and driving, along with their semantic annotations.

Formally, for each video frame i, an object detection algorithm identifies a set K_i of objects of interest. Then, the semantic score V_i of the frame is computed as:

$$V_i = \sum_{k \in K_i} C(k) \cdot G_\sigma(k) \cdot A(k), \tag{2.1}$$

where k denotes a detected object (e.g., a person), $C(k)$ represents the detection confidence for k, $G_\sigma(k)$ evaluates the object's centrality in the frame, and $A(k)$ corresponds to the area of its bounding box.

Intuitively, objects with high detection confidence, close to the center of the frame, and with large bounding boxes (*i.e.*, appearing closer) are more likely to attract the viewer's attention and are therefore assigned higher scores.

2.2 Homography

In computer vision, a homography is a transformation that relates the coordinates of points in one image to those in another through a simple geometric mapping. It is commonly used in tasks such as camera motion estimation, image stitching, and frame-to-frame alignment in video processing. Formally, given a point p in the first image and its corresponding point p' in the second, both in homogeneous coordinates, there exists a 3×3 homography matrix H such that p' and Hp are equivalent in projective space.

To estimate a homography from a pair of images, one must first detect keypoints and compute feature descriptors. ORB (Oriented FAST and Rotated BRIEF) (Rublee et al., 2011) is widely used for this purpose due to its speed, rotation invariance, and computational efficiency. Once descriptors are extracted, keypoints between the two images are matched by comparing their similarity—using, for example, the Hamming distance (Hamming, 1950) for binary descriptors like BRIEF. Heuristics such as cross-checking are then applied to retain only mutually consistent matches. The resulting correspondences are used to compute the homography matrix that best aligns the two images.

In practice, many correspondences are incorrect due to noise, occlusion, or visually similar but unrelated features. To estimate the homography robustly in the presence of such outliers, algorithms such as RANSAC (Random Sample Consensus) (Fischler; Bolles, 1981) are commonly used. RANSAC iteratively selects random subsets of correspondences, computes candidate homographies, and evaluates how many matches agree with the model. MAGSAC++ (Baráth et al., 2020), a recent improvement over RANSAC, incorporates probabilistic scoring and more efficient sampling strategies to achieve higher accuracy and robustness.

In video processing, homographies can be used to estimate inter-frame camera motion. When applied to consecutive frames, they quantify how the camera's viewpoint has shifted over time. This is particularly useful in applications such as video stabilization and hyperlapse generation, where the goal is to reduce visual shakiness. In our work, we adopt this approach to assess and minimize camera motion as part of our frame selection strategy.

More recently, learning-based approaches for keypoint detection and description, such as XFeat (Potje et al., 2024), have shown promising results in both quality and speed. While we were not aware of these methods during our experiments, we believe they could serve as drop-in replacements for ORB in our use case.

2.3 Loudness and Its Alternatives

Loudness is a psychoacoustic property that represents how intense a sound is perceived by a listener. Unlike the purely physical measure of energy of a sound wave, loudness accounts for human auditory sensitivity across different frequencies. For humans, the intensity of a sound correlates with arousal (Lu et al., 2006), where higher loudness can evoke excitement or tension, while lower levels are associated with calmness or relaxation.

The ITU-R BS.1770 (International Telecommunication Union, 2023) defines algorithms to measure subjective loudness, incorporating frequency weighting and gating methods to approximate generalized human auditory perception. Building upon this, the EBU Tech 3341 (European Broadcasting Union, 2023) introduces, among other metrics, the momentary loudness, measured over a 400 ms sliding window to smooth short-term fluctuations. These standards are widely used in broadcasting and audio production for consistent loudness measurement.

Other psychoacoustic features include sharpness, fluctuation strength, and roughness, which quantify high-frequency content, slow amplitude modulations, and rapid modulations, respectively. These are commonly used to characterize aversive or fatiguing sound qualities. Combined with loudness, they form the basis of *psychoacoustic annoyance* (Zwicker; Fastl, 2013), which Furlan et al. (2018) used to identify unpleasant segments in videos. In contrast, loudness captures perceived intensity without negative connotation, making it more appropriate for curated audio such as music.

Another alternative is the regression-based estimation of Thayer's valence–arousal model (Yang et al., 2008), used by Matos et al. (2021) to create musical hyperlapses. While expressive, this model relies on subjective emotion labels that are inherently ambiguous and often inconsistent across annotators. Loudness, by contrast, is a deterministic signal derived directly from the audio and less prone to interpretive variation.

Ultimately, the choice of musical feature is not fixed. While we use loudness in this work, other features may be more suitable in contexts with different technical constraints or creative goals.

2.4 Correlation and Rank

Correlation is a statistical measure that quantifies the degree to which two variables move in relation to each other. It captures the strength and direction of their association and is widely used in statistical analyses to understand relationships between datasets. A positive correlation implies that as one variable increases, the other tends to increase, while a negative correlation suggests that as one increases, the other tends to decrease. If the correlation is zero or near zero, the variables are considered uncorrelated.

One of the most commonly used measures of correlation is the *Pearson correlation coefficient* (Pearson, 1895), which measures linear relationships. It can be calculated for two vectors $\mathbf{x}$ and $\mathbf{y}$ as:

$$r_{\mathbf{xy}} = \frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s_{\mathbf{x}}} \right) \left(\frac{y_i - \bar{y}}{s_{\mathbf{y}}} \right), \tag{2.2}$$

where n is the number of elements, x_i and y_i are individual elements from the vectors, $\bar{x}$ and $\bar{y}$ are their means, and $s_{\mathbf{x}}$ and $s_{\mathbf{y}}$ are their standard deviations.

Ranking, in statistics, refers to the relative standing of a value within a collection when arranged in ascending or descending order. Ranking is especially useful for data without a known distribution or when relationships are non-linear. The rank of an element x_i in the vector $\mathbf{x}$ can be calculated as:

$$\mathrm{R}(x_i) = \sum_{j=1}^{n} \begin{cases} 1 & \text{if } x_i > x_j, \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

The *Spearman rank correlation coefficient* (Spearman, 1904) measures rank correlation by assessing how well the relationship between two variables can be described by a monotonic function. It is calculated by replacing the original data $\mathbf{x}$ and $\mathbf{y}$ with their ranks $\mathrm{R}(\mathbf{x})$ and $\mathrm{R}(\mathbf{y})$, and then applying the Pearson correlation formula to the ranked data:

$$\rho_{\mathbf{xy}} = r_{\mathrm{R}(\mathbf{x})\,\mathrm{R}(\mathbf{y})}. \tag{2.4}$$

The interpretation of Spearman's correlation follows the same scale as Pearson's correlation (-1 to $+1$). A value close to $+1$ indicates that as one variable increases, the rank of the other variable also increases consistently, even if the relationship is not linear. A value near -1 suggests an inverse monotonic relationship.

Since ranking imposes minimal assumptions and is robust to outliers, it forms the basis of a data normalization method called *rank normalization* (Tsodikov et al., 2002; Szabo et al., 2002; Qiu et al., 2013). Rank normalization is nearly identical to $\mathrm{R}(x_i)$, but it incorporates a normalization step to make the resulting values independent of the sample size. It is formally defined as:

$$x_i^{\star} = \frac{1}{n}\,\mathrm{R}(x_i), \tag{2.5}$$

such that the normalized values range from 0 for the smallest value in the vector to 1 for the largest.

3 Related Work

Video summarization techniques, designed to reduce video length while retaining important content, can be broadly classified into three categories based on the type of output they produce: *storyboards*, which select representative static frames; *video skimming*, which extracts a discontinuous subset of key segments; and *fast-forwarding*, which maintains temporal continuity by accelerating playback, either at a constant or variable rate (Molino et al., 2017). Our work focuses on fast-forwarding methods for first-person videos, which pose unique challenges due to continuous motion, unpredictable scene transitions, and the absence of explicit scene boundaries. Other summarization strategies, like storyboards or skimming, tend to introduce temporal discontinuities that may hinder a viewer's ability to follow the recorder's trajectory (Okamoto; Yanai, 2014) or understand the scene context (Silva et al., 2021).

This chapter is organized into four sections that reflect the research areas related to our approach. First, we examine *hyperlapse* techniques, which aim to reduce visual shakiness when fast-forwarding first-person videos. Next, we explore *semantic fast-forward* methods that adapt playback speed according to scene importance. We then discuss *semantic hyperlapse* approaches, which combine both motion stabilization and content-aware speed control. Finally, we consider *audio-driven semantic hyperlapse*, a relatively recent direction that integrates audio features—either from the audio of the video itself or an external song—to influence frame selection.

3.1 Hyperlapse

The concept of hyperlapse, which addresses the accentuated camera motion in fast-forward first-person videos, was introduced by Kopf et al. (2014). Their method reconstructs the scene in 3D using structure-from-motion and renders an accelerated video through a smooth virtual camera trajectory in 2D. Although the results are visually compelling, this approach is computationally expensive—on the order of minutes per frame—and depends on scenes with sufficient camera motion for successful reconstruction.

Karpenko (2014) proposed an alternative approach based on gyroscope metadata to estimate camera orientation and stabilize the video. The same metadata is used to compute the optimal zoom level needed to avoid introducing empty regions into the output. While this method relies on additional sensor data, it is notable for supporting real-time usage and for powering the Instagram Hyperlapse app.

Poleg et al. (2015) formulated hyperlapse generation as a graph traversal problem, modeling video frames as nodes and transitions between them as edges, weighted by

shakiness, velocity, and appearance costs. The final output consists of the frames along the shortest path. This framework was later extended by Halperin et al. (2018), who leveraged rotational camera motion to generate wide-view panoramas from frames not included in the final output.

Joshi et al. (2015) proposed an adaptive frame sampling method based on dynamic programming, inspired by dynamic time warping. Their algorithm selects frames that jointly minimize visual instability, deviation from a target speed-up, and abrupt speed transitions. The optimal frame set is extracted from a sparse dynamic programming matrix and used to compose the output hyperlapse.

Other works have explored hyperlapse generation using omnidirectional cameras (Ogawa et al., 2017; Rani et al., 2018) and multi-camera rigs (Wang et al., 2018). In contrast, our work focuses on monocular video, which is more commonly available in consumer devices such as smartphones, action cameras, and smart glasses.

3.2 Semantic Fast-Forward

Naively fast-forwarding a video ignores its content, potentially skipping over segments with important information. Semantic fast-forward methods address this issue by prioritizing the retention of meaningful content, selectively accelerating less relevant parts while preserving essential scenes. This helps viewers better understand and follow key events in the video.

Cheng et al. (2009) proposed an adaptive fast-forwarding algorithm based on domain-specific semantic rules, inspired by the metaphor of a car driver who slows down near areas of interest and speeds up through unexciting regions. In their system, viewer feedback is used to infer interest, and playback speed is adjusted through a greedy algorithm. Their work also included a user study, which showed that participants preferred gradual speed changes and disliked both silent playback and the high-pitched audio caused by fast-forwarding the original soundtrack.

Okamoto and Yanai (2014) proposed a semantic fast-forwarding method tailored for route guidance in first-person videos, addressing the temporal discontinuities that often confuse viewers about their current location. Their method assigns importance scores to frames based on the recorder's behavior—such as stopping, turning, or walking straight—as well as environmental features like crosswalks and intersections. Playback speed is then adjusted greedily according to this precomputed importance curve.

Ramos et al. (2020a) introduced a reinforcement learning-based semantic fast-forwarding method for instructional videos. The approach learns a cross-modal embedding between video frames and a textual summary using neural networks, and rewards the agent based on cosine similarity between the two. Ramos et al. (2022) extended this method by

incorporating temporal information into both the feature space and the reward function, enabling the system to honor user-defined target speed-up rates.

3.3 Semantic Hyperlapse

Ramos et al. (2016) proposed the first semantic hyperlapse algorithm, which slows down playback during high-importance segments while maintaining visual stability. They compute semantic scores based on the presence, size, and centrality of faces, and divide the video into segments that are either important or unimportant, each with its own target speed-up. These segments are then processed using the graph traversal algorithm from Poleg et al. (2015) and recombined into the final hyperlapse.

Building on this work, Silva et al. (2016) introduced a video stabilization algorithm specifically designed for semantic hyperlapses and published a dataset of videos with varying semantic content.

Subsequent work by Silva et al. (2018a) proposed a weighted sparse sampling method to select frames that minimize reconstruction error under a segment-wise speed-up constraint. This was later extended by Silva et al. (2021), who addressed the issue of abrupt jumps between segments.

Other definitions of semantic content have also been explored. Ramos et al. (2020b) proposed using textual features extracted from social media to infer user preferences and calculate semantic scores based on the importance of detected objects. Their method reuses the frame sampling algorithm from Ramos et al. (2016) and the stabilizer from Silva et al. (2016).

Neves et al. (2020) proposed a method that uses gaze tracking data in combination with object tracking to compute semantic scores. The model accounts for whether objects intersect with the gaze point, their proximity, duration on screen, and novelty. Their evaluation compares the methods proposed by Silva et al. (2018b) and Silva et al. (2018a) for the fast-forwarding step.

3.4 Audio-driven Semantic Hyperlapse

While most hyperlapse methods rely exclusively on visual features, only a few incorporate audio into the decision-making process.

Furlan et al. (2018) proposed a multimodal hyperlapse method that uses the audio stream to compute semantic scores. Their method assigns higher playback speeds to unpleasant segments—such as noisy streets or crowded scenes—based on the psychoacoustic annoyance metric (Zwicker; Fastl, 2013). Frame sampling is performed using the

multi-importance strategy from Silva et al. (2018b). However, the final hyperlapse does not preserve the original audio.

In the work of Matos et al. (2021), a video and user-selected song are combined to create a hyperlapse in which the emotion curves of the two are aligned. The video is processed using a two-dimensional convolutional neural network to extract a frame-level emotion curve, while the song is processed using a one-dimensional convolutional neural network to extract its own curve. These curves are then aligned using a dynamic programming formulation inspired by Joshi et al. (2015). Later, Matos et al. (2023) extended this approach by proposing a method to automatically select the best-matching song from a collection.

3.5 Differences to Previous Works

Table 1 provides a comparative overview of existing methods across five aspects: semantic awareness, camera motion smoothing, audio awareness, output length control, and the frame sampling strategy employed. This structured comparison situates our method within the broader research landscape and highlights the particular combination of features that distinguish our approach.

Table 1 – Comparison of related works based on semantic, motion, and audio awareness, along with whether the output length matches the target duration and the frame selection strategy used.

Method	**Semantic**	**Motion**	**Audio**	**Length**	**Sampling**[1]
Kopf et al. (2014)	✗	✓	✗	✗	SfM
Karpenko (2014)	✗	✓	✗	✗	Unspecified
Poleg et al. (2015)	✗	✓	✗	✗	Graph
Joshi et al. (2015)	✗	✓	✗	✗	DP
Cheng et al. (2009)	✓	✗	✗	✗	Greedy
Okamoto and Yanai (2014)	✓	✗	✗	✗	Greedy
Ramos et al. (2020a, 2022)	✓	✗	✗	✗	RL
Ramos et al. (2016)	✓	✓	✗	✗	Graph
Silva et al. (2016)	✓	✓	✗	✗	Graph
Silva et al. (2018a, 2021)	✓	✓	✗	✗	SS
Neves et al. (2020)	✓	✓	✗	✗	Graph, SS[2]
Furlan et al. (2018)	✓	✓	✓[3]	✗	SS[4]
Matos et al. (2021, 2023)	✓	✓	✓	✓	DP
Ours	✓	✓	✓	✓	DP

[1] Frame selection strategy: Structure from Motion (SfM), Dynamic Programming (DP), Reinforcement Learning (RL), Sparse Sampling (SS), graph-based, or greedy.
[2] Compares the sampling methods from Silva et al. (2018a) and Silva et al. (2018b).
[3] Uses audio from the input video to derive semantics but does not retain the audio in the output.
[4] Reuses the sampling method from Silva et al. (2018b).

Our method builds upon the multimodal fast-forwarding formulation introduced by Matos et al. (2021), which in turn extends the dynamic programming framework proposed by Joshi et al. (2015). Like these prior works, we model frame selection as an optimization problem over a dynamic programming matrix, where each path corresponds to a possible frame sequence in the final video.

Compared to Joshi et al. (2015), which only considers camera smoothness and playback speed consistency, our formulation introduces two additional objectives: semantic retention and audio alignment. To incorporate semantic awareness, we use task-specific scores such as the presence and proximity of people, following the strategy of Ramos et al. (2016). This allows the output hyperlapse to emphasize content likely to be more meaningful to the viewer.

Relative to the multimodal algorithm of Matos et al. (2021), our approach adopts a different cost formulation and includes several algorithmic optimizations. While their method aligns emotion curves estimated from both the video and the music, we treat each modality independently. In particular, our audio alignment is based on momentary loudness—a deterministic, perceptually grounded feature—which we align with playback speed rather than the video's emotional content. As in their work, we ensure that the hyperlapse duration matches the length of the selected song exactly, avoiding situations where the video or audio ends prematurely.

4 Methodology

In this section, we describe our method for creating semantic hyperlapses aligned with music, as shown 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.

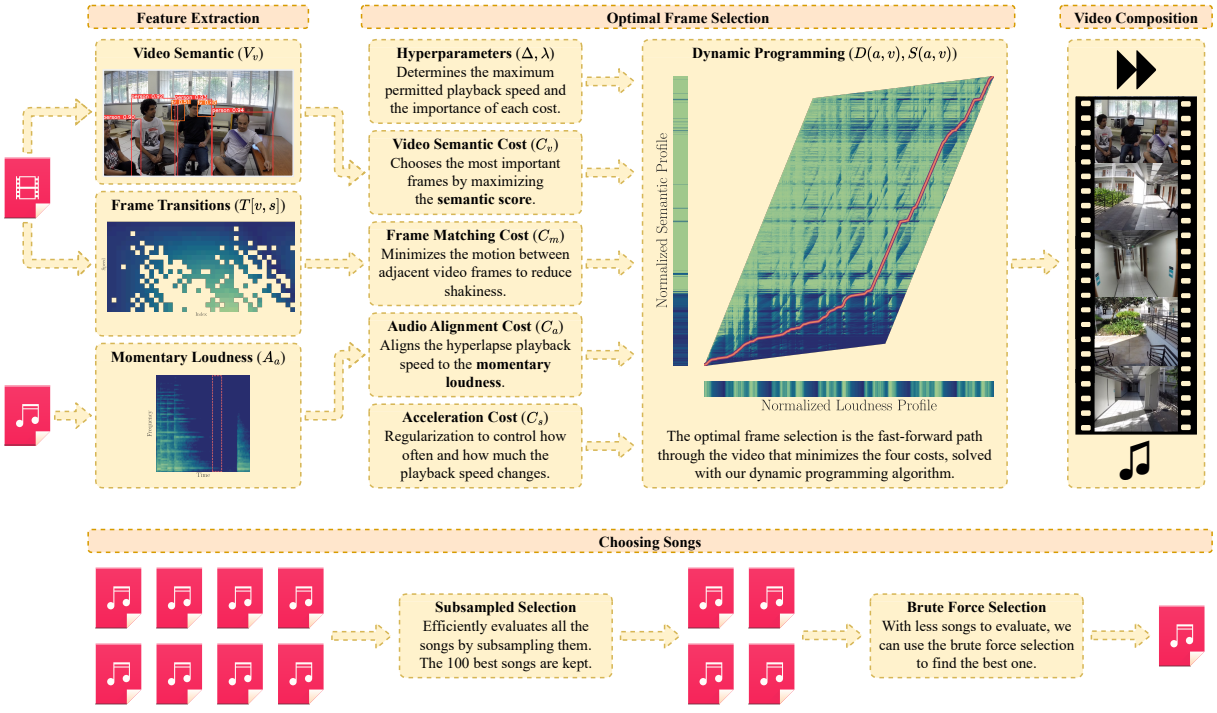


Figure 2 – High-level overview of our method. The **feature extraction** prepares the inputs for our **optimal frame selection**, which in turn produces a path through the video which the **video composition** traverses to create a hyperlapse. Lastly, **choosing songs** automates how to pick a song from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

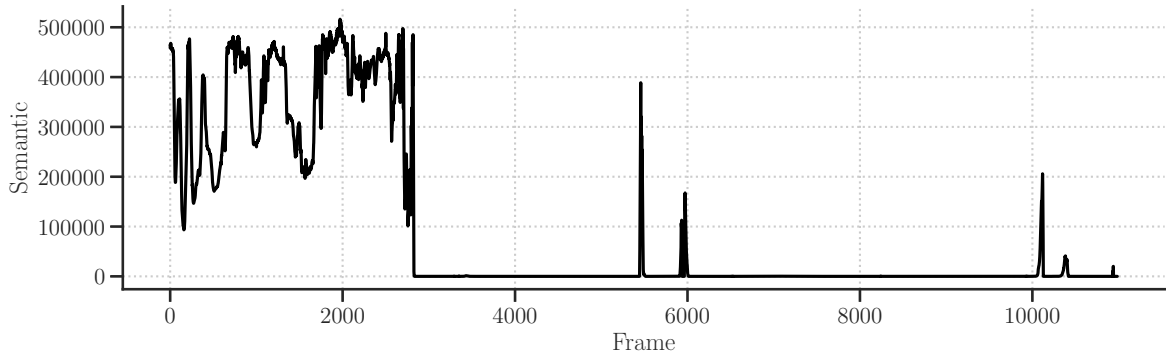
4.1 Feature Extraction

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

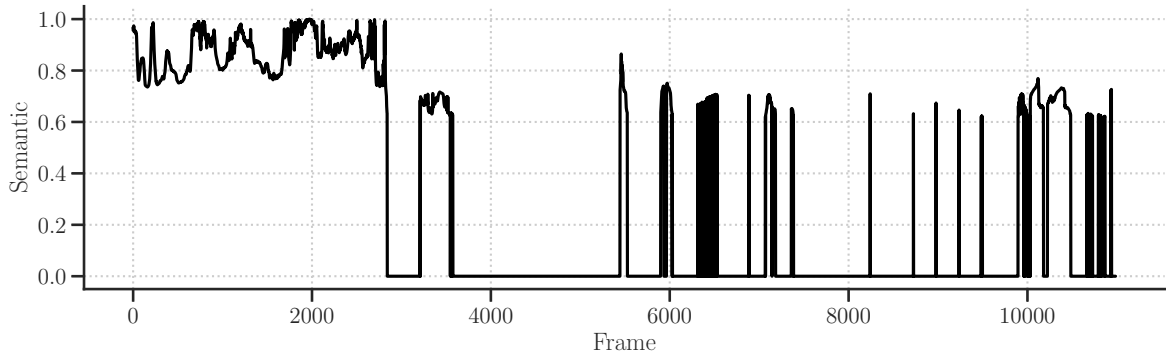
4.1.1 Semantic Scores

We compute semantic scores for each video frame using the definition proposed by Ramos et al. (2016), which considers the presence and proximity of people as indicators of higher importance. These scores, detailed in Section 2.1, are represented as a vector $V = \langle V_0, \ldots, V_{n-1} \rangle$. Hereafter, we denote its length as $|V| = n$.

Semantic scores are sensitive to outliers. For example, if a large crowd appears in a few frames, those scores may spike, making other important frames with fewer people appear comparatively less relevant. To mitigate this, we apply rank normalization (Tsodikov et al., 2002) to V, as described in Equation 2.5, yielding the normalized vector $V^\star$. Tied values are assigned the minimum rank among all tied entries.[1] An example of this normalization is shown in Figure 3.



(a) Pre-normalization semantic curve.



(b) Rank-normalized semantic curve.

Figure 3 – Example of rank normalization applied to the semantic curve of the video *Walking 25p* from the *Annotated Semantic Dataset* (Silva et al., 2016). Many frames with scores near zero in the original curve are assigned higher values after normalization, reflecting their relative importance compared to frames with zero scores.

A trade-off of this approach is increased sensitivity to noise: in videos with few

[1] This tie-breaking approach is referred to as "min" or "competition" ranking by https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.rankdata.html. Accessed January 11, 2025.

important frames, distant people (*e.g.*, Figure 4) or objects mistakenly classified as people may receive disproportionately high ranks, causing the video to slow down without clear justification. Despite this limitation, we prioritize recall over precision, accepting this trade-off to ensure that important frames are retained.

Figure 4 – A hard-to-spot person in a frame from the video *Walking 25p* (Silva et al., 2016). The frame initially received a very low score but was ranked above 60% after normalization, as most frames in the video had near-zero scores.

4.1.2 Pairwise Distance Matrix

The pairwise distance matrix $T[v, s]$ quantifies frame-to-frame camera motion and serves as a proxy for visual shakiness in the output hyperlapse. To compute it, we calculate how far the image center shifts between frames, normalized to the frame size:

$$T[v, s] = \frac{\|\hat{\mathbf{M}} - \mathbf{M}\|_2}{\|\mathbf{M}\|_2}, \tag{4.1}$$

where $\mathbf{M}$ denotes the 2D coordinates of the image center, and $\hat{\mathbf{M}}$ is the result of reprojecting $\mathbf{M}$ through the homography $H(v - s, v)$ that maps frame $v - s$ to frame v. If homography estimation fails or yields unreliable results, we set $T[v, s] = 1$. This formulation combines the *frame matching cost* from Joshi et al. (2015) with the *shaking ratio* metric proposed by Ramos et al. (2020b).

As described in Section 2.2, to estimate $H(v - s, v)$, we detect keypoints and compute descriptors using the ORB feature extractor (Rublee et al., 2011). Keypoints are matched between frames $v - s$ and v using a brute-force matcher with Hamming distance

(Hamming, 1950), retaining only bidirectional matches through cross-checking. We then estimate the homography using MAGSAC++ (Baráth et al., 2020), requiring at least 10 valid matches for a solution to be considered. To validate the result, we compute the average reprojection error between matched keypoints. If this error exceeds $0.2\|\mathbf{M}\|_2$, the homography is considered unreliable and discarded.

For efficiency, we compute $T[v, s]$ only for values of s in the range $1 \leq s \leq \Delta$, where Δ is the maximum playback speed allowed by the hyperlapse. Since our method guarantees that playback speed never exceeds Δ, limiting the distance matrix to this range reduces both computational and memory costs without affecting correctness.

4.1.3 Loudness Profile

The loudness profile represents the momentary loudness of the input song and serves as the primary audio feature for guiding playback speed. As discussed in Section 2.3, loudness is a psychoacoustic property that reflects the perceived intensity of a sound, incorporating human sensitivity across different frequencies. Psychologically, high loudness levels are often associated with excitement or tension, while lower levels convey calmness or relaxation (Lu et al., 2006).

We compute momentary loudness as specified by EBU Tech 3341 (European Broadcasting Union, 2023), which defines a 400 ms sliding window to smooth short-term fluctuations. To align each loudness value with a corresponding frame in the output video, we set the stride of this sliding window to match the duration of a single frame—for example, 33.3 ms for videos at 30 frames per second or 16.6 ms for 60 FPS. Each window is centered on the target frame to ensure temporal alignment between audio and video. Formally, the loudness profile is represented as a vector $A = \langle A_0, \ldots, A_{n-1} \rangle$, where each A_i corresponds to the momentary loudness aligned with video frame i. We denote the length of this vector as $|A| = n$.

As with the semantic scores described in Subsection 4.1.1, we apply rank normalization (Tsodikov et al., 2002) to the loudness profile to reduce the influence of outliers and ensure robustness to unknown distributions. The normalized vector $A^\star$ is computed using Equation 2.5, mapping the lowest and highest loudness values to 0 and 1, respectively.

4.2 Optimal Frame Selection

Our objective is to select a sequence of video frames that minimizes a cost function under a set of constraints, yielding a fast-forwarded video.

The cost function balances four objectives. First, it prioritizes frames deemed important based on a semantic score. Second, it promotes alignment between playback speed and song loudness—slowing down during quieter segments and speeding up during louder

ones. Third, it penalizes abrupt changes in playback speed to ensure temporal smoothness. Fourth, it reduces visual shakiness by discouraging sudden shifts in camera motion.

The constraints define the set of valid frame sequences that can compose the hyperlapse. The output video must match the length of the selected song exactly, preventing two undesirable outcomes: the song ending before the video, leaving a silent segment, or the video ending before the song, causing an abrupt cutoff. Additionally, playback speed must be strictly positive—to avoid rewinds or pauses—and must not exceed a maximum value Δ, which limits large temporal skips that break continuity and hinder stabilization. Finally, we fix the first and last frames to match the beginning and end of the input video. While not strictly required, this assumption simplifies both the formulation and certain optimizations.

Following previous works (Joshi et al., 2015; Matos et al., 2021), we model frame selection as a dynamic programming problem, differing in both algorithmic structure and cost definition. In particular, we extend Joshi et al. (2015) by incorporating an audio modality and enforcing that the hyperlapse duration equals the song length. Compared to Matos et al. (2021), we introduce optimizations for computational efficiency.

Let $D[a, v]$ denote the minimum cumulative cost of selecting frames and playback speeds up to audio timestep a and video frame v. The recurrence is defined as:

$$D[a, v] = \min_{s \in [1, \Delta]} \mathrm{C}(a, v, s, S[a-1, v-s]) + D[a-1, v-s], \tag{4.2}$$

with a corresponding traceback matrix $S[a, v]$ storing the playback speed s that minimizes the cost:

$$S[a, v] = \arg\min_{s \in [1, \Delta]} \mathrm{C}(a, v, s, S[a-1, v-s]) + D[a-1, v-s]. \tag{4.3}$$

Here, a is the current audio frame index, v is the video frame index, s is the playback speed—*i.e.*, the number of video frames skipped between steps—and Δ is the maximum speed. The construction of $D[a, v]$ and $S[a, v]$ is described 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 $\mathcal{O}(\Delta \cdot |A| \cdot |V|)$.

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.

The cost minimized by the dynamic programming algorithm depends on the video frame index (v), the song timestep (a), and the current and previous playback speeds

Algorithm 1 Fast-Forward Matrix Construction

Input: The song loudness curve A and the video semantic curve V.
Output: The cost matrix D and the traceback matrix S.

Initialize $D[a, v] = \infty$ for all a, v.
Initialize $S[a, v] = 0$ for all a, v.
$D[0, 0] \leftarrow 0$
$S[0, 0] \leftarrow \lfloor \frac{|V|}{|A|} \rfloor$
for $a \in [1, |A|)$ **do**
    for $v \in [1, |V|)$ **do**
        $s_{\min} \leftarrow 1$
        $s_{\max} \leftarrow \min(\Delta, v)$ ▷ Prevents out-of-bounds access.
        for $s \in [s_{\min}, s_{\max}]$ **do**
            if $D[a-1, v-s] \neq \infty$ **then**
                $s' \leftarrow S[a-1, v-s]$
                $c \leftarrow D[a-1, v-s] + C(a, v, s, s')$
                if $c < D[a, v]$ **then**
                $D[a, v] \leftarrow c$
                $S[a, v] \leftarrow s$
return S

Algorithm 2 Fast-Forward Matrix Traversal

Input: The traceback matrix S.
Output: The list of selected frames I.

$a \leftarrow |A|$
$v \leftarrow |V|$
Initialize I as an empty list.
while $a \geq 1$ and $v \geq 1$ **do**
    Prepend v to I
    $s \leftarrow S[a, v]$
    $a \leftarrow a - 1$
    $v \leftarrow v - s$
return I

(s and s', respectively). It is defined as a weighted sum of four components—the *video semantic cost*, *audio alignment cost*, *acceleration cost*, and *frame matching cost*:

$$\mathrm{C}(a, v, s, s') = \lambda_v \, \mathrm{C_v}(v, s) + \lambda_a \, \mathrm{C_a}(a, s) + \lambda_s \, \mathrm{C_s}(s, s') + \lambda_m \, \mathrm{C_m}(v, s). \tag{4.4}$$

Each term in this cost function is defined over the range $[0, 1]$ to facilitate the choice of the weighting coefficients (λ). The following subsections present the definition of each component.

4.2.1 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** ($\mathrm{C_v}$) as:

$$\mathrm{C_v}(v, s) = \begin{cases} 1 - V_v^\star, & \text{if } s < \frac{|V|}{|A|}, \\ 1 - \alpha V_v^\star, & \text{otherwise,} \end{cases} \tag{4.5}$$

where $V_v^\star$ is the rank-normalized semantic score from Subsection 4.1.1. Since our framework is formulated as a minimization problem, minimizing $\mathrm{C_v}(v, s)$ corresponds to maximizing $V_v^\star$. 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 α (*e.g.*, 0.05).

4.2.2 Audio Alignment Cost

The **audio alignment cost** ($\mathrm{C_a}$) 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:

$$\hat{s}_a = 1 + (\Delta - 1) \cdot A_a^\star, \tag{4.6}$$

where $A_a^\star$ is the rank-normalized loudness score from Subsection 4.1.3. 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:

$$\mathrm{C_a}(a, s) = \left(\frac{\hat{s}_a - s}{\Delta - 1} \right)^2, \tag{4.7}$$

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

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 $\mathrm{C_a}(a, s)$. 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.

4.2.3 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 ($\mathrm{C_s}$) is used to penalize large variations in speed. It is defined as:

$$\mathrm{C_s}(s, s') = \left(\frac{s' - s}{\Delta - 1}\right)^2, \tag{4.8}$$

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 Δ).

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

4.2.4 Frame Matching Cost

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

$$\mathrm{C_m}(v, s) = T[v, s], \tag{4.9}$$

where T is the pairwise distance matrix described in Equation 4.1. 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.

4.2.5 Reducing the 3D Cost Matrix to 2D

In this subsection, we show that the 3D dynamic programming formulation used by Matos et al. (2021) can be reformulated as an equivalent 2D version that is both conceptually simpler and computationally more efficient. This reformulation lays the foundation for our cost matrix in Equation 4.2.

The 3D recurrence from Matos' work can be defined as:

$$D[i, j, k] = \mathrm{C}(i, j, k) + \min_{h \in [1, w]} D[i - h, i, k - 1], \tag{4.10}$$

where i and j are video frame indices, and k is the index of a segment in the song. The cost function $\mathrm{C}(i, j, k)$ evaluates the cost of transitioning from frame i to frame j at segment k.

To simplify this recurrence, we first observe that i and j always appear as a pair, with $i < j$. We can therefore replace i with the offset $s = j - i$, and reparameterize the

recurrence as:

$$\hat{D}[s, j, k] = \mathrm{C}(j - s, j, k) + \min_{h \in [1,w]} \hat{D}[h, j - s, k - 1]. \tag{4.11}$$

This makes the dependence on the jump size s explicit. Next, we define an auxiliary function that computes the minimum cost over all jump sizes:

$$D^{\star}[j, k] = \min_{h \in [1,w]} \hat{D}[h, j, k]. \tag{4.12}$$

Note that the inner minimization in Equation 4.11 is equivalent to the definition of $D^{\star}$ in Equation 4.12:

$$\min_{h \in [1,w]} \hat{D}[h, j - s, k - 1] = D^{\star}[j - s, k - 1]. \tag{4.13}$$

Substituting this into Equation 4.11, we obtain:

$$\hat{D}[s, j, k] = \mathrm{C}(j - s, j, k) + D^{\star}[j - s, k - 1]. \tag{4.14}$$

Finally, substituting this into Equation 4.12 yields a 2D formulation equivalent to the original 3D recurrence in Equation 4.10:

$$D^{\star}[j, k] = \min_{h \in [1,w]} \mathrm{C}(j - h, j, k) + D^{\star}[j - h, k - 1]. \tag{4.15}$$

While both are mathematically equivalent, the 2D formulation is more efficient in practice. The 3D version requires either full storage or a sparse representation, introducing overhead even when only a subset of entries is used.

Our cost matrix in Equation 4.2 follows the same structure as $D^{\star}[j, k]$, differing only in the cost function's definition. This change not only supports our own method but can also be trivially applied to optimize the original algorithm by Matos et al. (2021).

4.2.6 Memory Usage Optimization

Algorithm 1 prioritizes clarity over efficiency, suffering from excessive memory consumption, which limits its applicability to long videos. In this section, we present an optimization to overcome this limitation while maintaining the algorithm's original behavior.

As shown in Algorithm 1, the entire $D[a, v]$ matrix is stored, even though only the current row (*i.e.*, the $D[a, v]$ values) and previous row (*i.e.*, the $D[a - 1, v - s]$ values) are accessed at any given time. Additionally, as shown in Algorithm 2, only the traceback matrix $S[a, v]$ is required for determining the optimal frame selection. Instead of main-

taining the entire quadratic matrix $D[a, v]$, we can use two vectors: $D[v]$ for the current cost and $D'[v]$ for the previous cost. We highlight this change in Algorithm 3.

Algorithm 3 Forward Matrix Construction with memory optimization. Changes compared to Algorithm 1 are highlighted.

Initialize $D[v] \leftarrow \infty$ for all v.
Initialize $D'[v] \leftarrow \infty$ for all v.
Initialize $S[a, v] \leftarrow 0$ for all a, v.
$D[0] \leftarrow 0$
$S[0, 0] \leftarrow \lfloor \frac{|V|}{|A|} \rfloor$
for $a \in [1, |A|)$ **do**
    Swap D and D'.
    Reset $D[v] \leftarrow \infty$ for all v.
    for $v \in [1, |V|)$ **do**
        $s_{\min} \leftarrow 1$
        $s_{\max} \leftarrow \min(\Delta, v)$
        for $s \in [s_{\min}, s_{\max}]$ **do**
            if $D'[v - s] \neq \infty$ **then**
                $s' \leftarrow S[a - 1, v - s]$
                $c \leftarrow D'[v - s] + C(a, v, s, s')$
                if $c < D[v]$ **then**
                    $D[v] \leftarrow c$
                    $S[a, v] \leftarrow s$
return S

In an implementation where $D[a, v]$ is a matrix of doubles (8 bytes per element) and $S[a, v]$ is a matrix of bytes (1 byte per element), this optimization uses approximately one-ninth the memory of the naive approach. For example, an hour-long video at 60 FPS has 216 000 frames. To fast-forward it by 10× using our method requires a six-minute song, or 21 600 audio frames. The total memory needed for $D[a, v]$ and $S[a, v]$ is:

$$216\,000 \times 21\,600 \times 9 \text{ B} \approx 41.99 \text{ GB}, \tag{4.16}$$

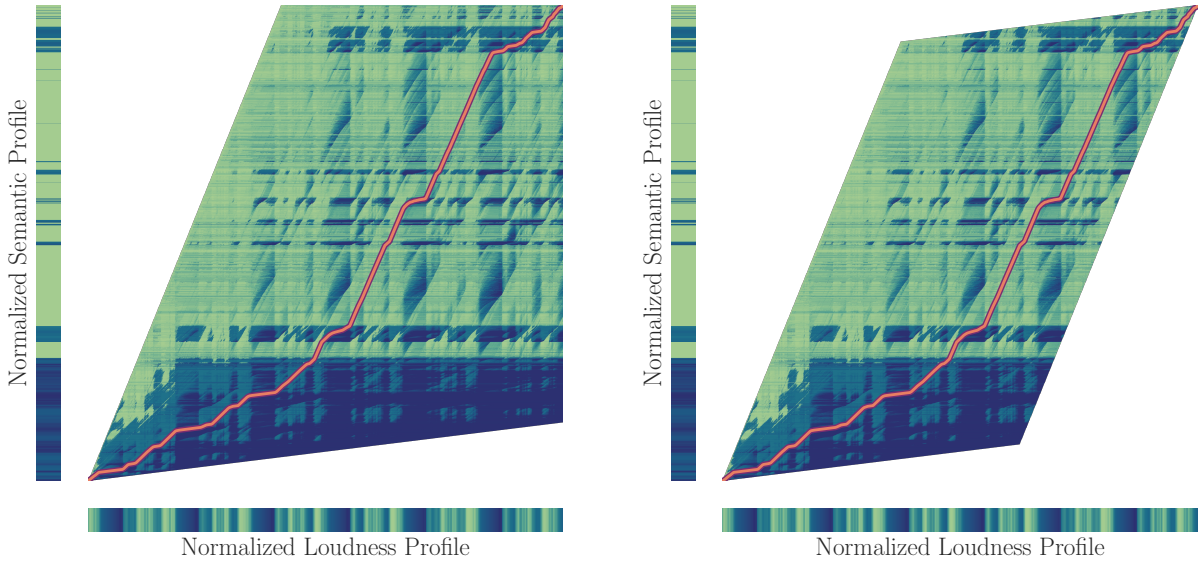
which exceeds the capacity of a high-end user computer with 32 GB of RAM. By optimizing the implementation to avoid storing $D[a, v]$ as a full matrix and retaining only two active rows, memory usage becomes dominated by $S[a, v]$. This reduces the requirement to:

$$216\,000 \times 21\,600 \times 1 \text{ B} \approx 4.67 \text{ GB}, \tag{4.17}$$

comfortably within the capabilities of a typical consumer computer or smartphone with 8 GB of memory.

4.2.7 Avoiding Infeasible Positions

The second optimization reduces computation by skipping infeasible entries in the dynamic programming matrix during the forward pass. These are positions that either cannot be reached from the start or cannot reach the end of the video and song while respecting playback constraints. Figure 5 illustrates this pruning: the unoptimized version evaluates all reachable positions, while the optimized version restricts computation to those that can also reach the final frame. Algorithm 4 highlights these changes relative to Algorithm 3.



(a) Unoptimized $S[a, v]$. Only the positions reachable from the origin are evaluated; the remaining 26.5% of the matrix is marked as infeasible.

(b) Optimized $S[a, v]$. Only positions that can also reach the end of the song and video are evaluated. Thus, 45.2% of the matrix is marked as infeasible.

Figure 5 – Illustration of the second optimization. White regions in the speed-up matrix represent infeasible positions during the forward pass (*i.e.*, $D[v] = \infty$). The red line shows the path selected in the backward pass, from the top-right.

In Algorithm 3, the matrix is filled bottom-up, starting from the origin, which corresponds to the first frame of the video and the first window of the song (*i.e.*, $\langle a, v\rangle = \langle 0, 0\rangle$). This exhaustive process includes positions that cannot reach $\langle a, v\rangle = \langle |A|, |V|\rangle$, as doing so would require speeds $s \notin [1, \Delta]$, making them infeasible in the backward pass (Algorithm 2).

Intuitively, the bounds of the interval of feasible positions at timestep a in the forward pass represent the smallest position reachable when always moving at the minimum speed (*i.e.*, $s = 1$) and the farthest position for the maximum speed (*i.e.*, $s = \Delta$). Thus, the forward-pass interval is:

$$\mathbb{V}_{\text{forward}} = \left[\sum_{i=1}^{a} 1, \sum_{i=1}^{a} \Delta\right] = [a \cdot 1, a \cdot \Delta]\,. \tag{4.18}$$

Algorithm 4 Forward Matrix Construction without infeasible positions. Changes compared to Algorithm 3 are highlighted.

Initialize $S[a, v] \leftarrow 0$ for all a, v.
Initialize $D[v] \leftarrow \infty$ for all v.
Initialize $D'[v] \leftarrow \infty$ for all v.
$D[0] \leftarrow 0$
$S[0, 0] \leftarrow\leftarrow \lfloor \frac{|V|}{|A|} \rfloor$
for $a \in [1, |A|)$ **do**
    Swap D and D'
    Reset $D[v] \leftarrow \infty$ for all v.
    $\bar{a} \leftarrow |A| - a$
    $v_{\min} \leftarrow \max(a, |V| - \bar{a} \cdot \Delta)$
    $v_{\max} \leftarrow \min(a \cdot \Delta, |V| - \bar{a})$
    for $v \in [v_{\min}, v_{\max}]$ **do**
        $s_{\min} \leftarrow 1$
        $s_{\max} \leftarrow \min(\Delta, v)$
        for $s \in [s_{\min}, s_{\max}]$ **do**
            if $D'[v - s] \neq \infty$ **then**
                $s' \leftarrow S[a - 1, v - s]$
                $c \leftarrow D'[v - s] + C(a, v, s, s')$
                if $c < D[v]$ **then**
                    $D[v] \leftarrow c$
                    $S[a, v] \leftarrow s$
return S

Similarly, during the backward pass, the bounds of the interval depend on the remaining frames (*i.e.*, $\bar{a} = |A| - a$). Choosing the maximum speed brings us closer to the origin, defining the lower bound, and the minimum speed defines the upper bound. Thus, the backward-pass interval is:

$$\mathbb{V}_{\text{backward}} = \left[|V| - \sum_{i=1}^{\bar{a}} \Delta, |V| - \sum_{i=1}^{\bar{a}} 1 \right] = \left[|V| - \bar{a} \cdot \Delta, |V| - \bar{a} \cdot 1 \right]. \tag{4.19}$$

To avoid infeasible positions, the dynamic programming algorithm restricts iterations to the intersection of the two intervals:

$$\mathbb{V}_{\text{forward}} \cap \mathbb{V}_{\text{backward}} = [v_{\min}, v_{\max}], \tag{4.20}$$

where:

$$v_{\min} = \max(a \cdot 1, |V| - \bar{a} \cdot \Delta), \text{ and} \tag{4.21}$$

$$v_{\max} = \min(a \cdot \Delta, |V| - \bar{a} \cdot 1). \tag{4.22}$$

In Algorithm 3, the $D'[v - s] \neq \infty$ condition serves the same role as $\mathbb{V}_{\text{forward}}$. This can be seen in Figure 5a, which is partially marked as infeasible. However, by explicitly defining $\mathbb{V}_{\text{forward}}$, we avoid repeatedly checking $D'[v - s] \neq \infty$ for each $s \in [s_{\min}, s_{\max}]$, improving efficiency.

4.2.8 Implementing Internal Parallelism

Our third and last optimization introduces parallelism into the algorithm, enabling multiple the usage of multiple CPU cores to process a single video–audio pair.

In Algorithm 4, the loop over $v \in [v_{\min}, v_{\max}]$ (line 12) is trivially parallelizable, as each value of v is computed solely from pre-existing values in $D'[v - s]$ and does not depend on updates to other $D[v]$ values during the same iteration. No changes to the pseudocode are made, as the parallelization is heavily language-dependent.

4.3 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.

Another approach 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 6.

4.4 Choosing Songs

When access to a large collection of songs is available, there is an opportunity to select one that complements the video well. However, manually evaluating all possible

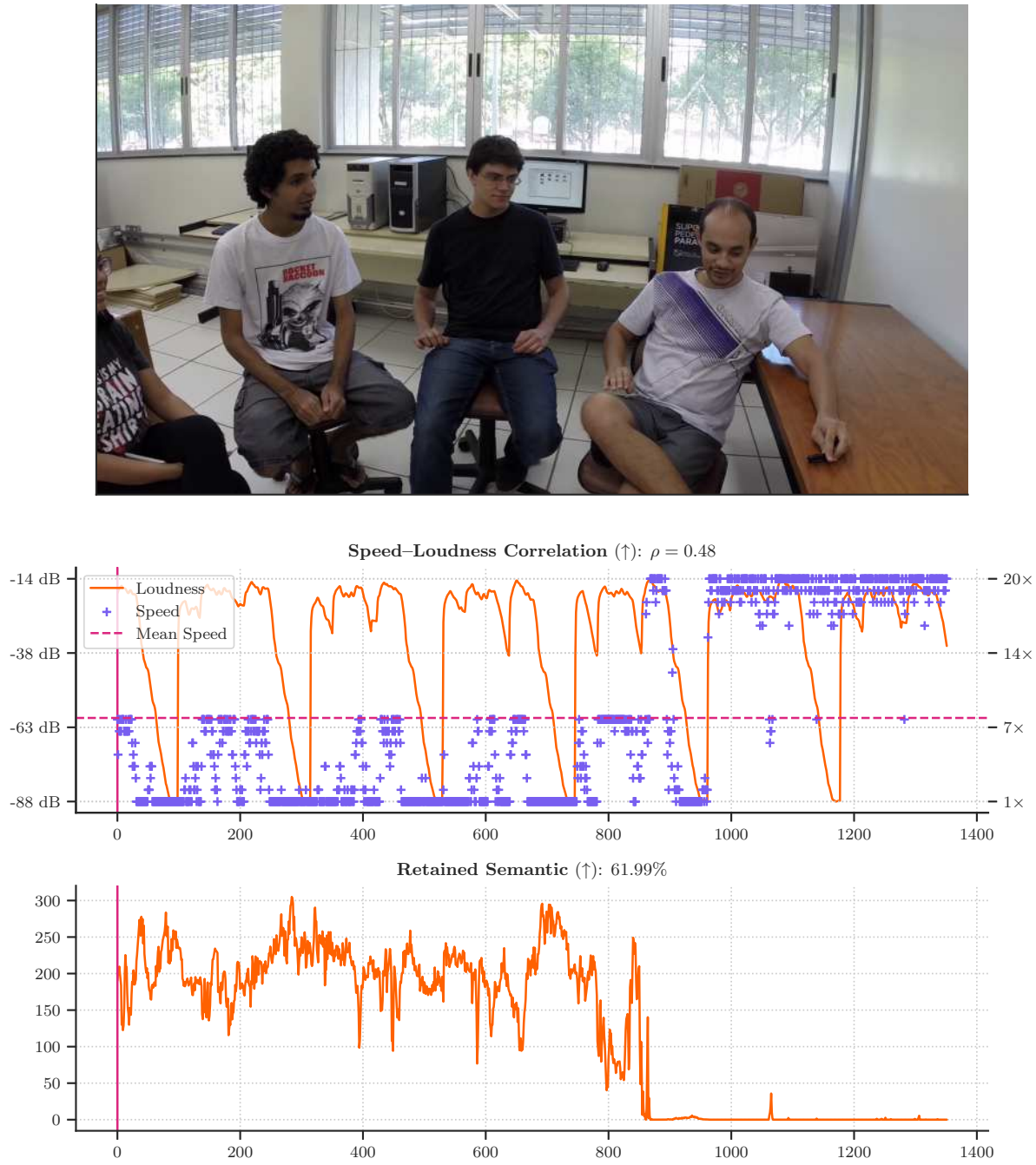


Figure 6 – Dashboard visualization for evaluation. The top image is a frame from the hyperlapse. The middle plot shows speed (blue crosses) and loudness (orange line), with a mean speed reference (dashed pink line). The bottom plot tracks retained semantic content. The vertical purple line serves as a cursor, indicating the current frame's position.

pairings becomes time-consuming and impractical at scale. To address this, we propose automating the selection process using a simplified fitness model that balances two objectives: semantic retention and correlation between playback speed and song loudness.

Pairing a song with a video can yield varying outcomes. At one extreme, a well-matched pair enables both high semantic retention and strong loudness–speed correlation.

At the other, a poor match forces a trade-off, where improving one metric degrades the other. In this section, we formalize this trade-off and use it as the basis for our song selection algorithm.

4.4.1 Balancing Semantic and Correlation

To balance semantic retention and loudness–speed correlation, we adopt Tchebycheff scalarization from multi-objective optimization (Steuer; Choo, 1983), which evaluates a solution by its worst-performing objective to prevent extreme compromises. The original formulation is defined for minimization and uses the max operator; since our objectives are to be maximized, we invert it and define:

$$\mathrm{F}_{\min}(\mathbf{x}, \mathbf{s}) = \min\{\mathrm{F}_{\mathrm{v}}(\mathbf{x}), \mathrm{F}_{\mathrm{a}}(\mathbf{s})\}, \tag{4.23}$$

where $\mathrm{F}_{\mathrm{v}}(\mathbf{x})$ is the semantic score of the selected video frames $\mathbf{x}$, and $\mathrm{F}_{\mathrm{a}}(\mathbf{s})$ is the correlation between playback speed $\mathbf{s}$ and the loudness curve.

The **semantic score** $\mathrm{F}_{\mathrm{v}}(\mathbf{x})$ quantifies how much semantic content is retained relative to the best possible selection of the same length (Silva et al., 2018b):

$$\mathrm{F}_{\mathrm{v}}(\mathbf{x}) = \frac{\sum_{i \in \mathbf{x}} V_i}{\sum_{j \in \hat{\mathbf{x}}} V_j}, \tag{4.24}$$

where $\hat{\mathbf{x}}$ is the set of frames with the highest semantic scores, constrained to the same length as $\mathbf{x}$. Since this comparison does not consider the maximum skip (Δ) between frames, perfect (100%) scores are not always attainable.

The **correlation score** $\mathrm{F}_{\mathrm{a}}(\mathbf{s})$ measures how closely the speed-up vector aligns with the song's momentary loudness. It is defined as:

$$\mathrm{F}_{\mathrm{a}}(\mathbf{s}) = \rho_{\mathbf{as}}, \tag{4.25}$$

where $\mathbf{a}$ is the loudness curve and $\rho_{\mathbf{as}}$ is the Spearman rank correlation coefficient (Spearman, 1904), as described in Equation 2.4. The use of Spearman correlation avoids assumptions of linearity between loudness and playback speed. Values near 1 indicate that speed increases with loudness, values near 0 suggest little or no alignment, and negative values imply an inverse relationship.

4.4.2 Algorithms for Choosing Songs

We propose three approaches for selecting the optimal song to pair with a video, each offering a different trade-off between computational cost and accuracy: (i) a brute-force method that evaluates all candidates exhaustively; (ii) an approximate method that

operates on subsampled video and audio curves; and (iii) a hybrid approach that combines both strategies to reduce cost while maintaining precision.

The **brute force selection** evaluates every song in the collection by running our hyperlapse algorithm (*i.e.*, Algorithm 4 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 **subsampled selection** yields an approximate solution by downsampling the video semantic and audio loudness curves using a maximum filter with a window size of 10, reducing their lengths to one-tenth while preserving salient peaks. Since Algorithm 1 has time complexity $\mathcal{O}(\Delta \cdot |A| \cdot |V|)$, this results in an expected 100× reduction in computation. After scoring all downsampled inputs, the original, non-downsampled version of the highest-ranked song is selected. While less precise than the brute-force selection, this method is computationally efficient and well-suited to large song libraries.

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.

In all three approaches, the video–song evaluations are independent and therefore highly parallelizable. As discussed in Subsection 4.2.8, distributing these evaluations across available CPU cores is more effective than relying on internal parallelism within the hyperlapse algorithm. Our implementation leverages this by parallelizing over song candidates during the selection process.

5 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.

5.1 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 4.1.1. 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 2 lists the lengths of the videos and the speed-ups used for them in the experiments, together with other properties. Since the dataset does not provide the pairwise distance matrices required by our method, we compute them as described in Subsection 4.1.2.

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[1] library, as described in Subsection 4.1.3.

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

[1] Available at: https://essentia.upf.edu/reference/std_LoudnessEBUR128.html. Accessed January 11, 2025.

Table 2 – 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**[1]	**Resolution**[2]	**Speed-up**[3]
Biking 0p	17 949	60	4:59	1280 × 720	6.6×
Biking 25p	17 071	30	9:29	1920 × 1080	12.6×
Biking 50p	26 954	60	7:29	1280 × 720	10.0×
Biking 50p 2	14 939	60	4:09	1280 × 720	5.5×
Driving 0p	9 463	30	5:15	1920 × 1080	7.0×
Driving 25p	7 989	30	4:26	1920 × 1080	5.9×
Driving 50p	10 379	30	5:46	1920 × 1080	7.7×
Walking 0p	8 219	30	4:34	1920 × 1080	6.1×
Walking 25p	10 982	30	6:06	1920 × 1080	8.1×
Walking 50p	11 570	30	6:26	1920 × 1080	8.6×
Walking 75p	15 481	30	8:36	1920 × 1080	11.5×

[1] Duration in minutes and seconds (mm:ss).
[2] Pixel resolution, given as width and height.
[3] Speed-up factor required to align the video with a 45-second-long song.

in optimization criteria and the absence of semantic labels in their dataset, we chose to omit it from the comparison.

5.2 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 4.24. 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 Equation 4.25. 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:

$$\mathrm{F_s}(\mathbf{s}) = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n-1}(s_{i+1}-s_i)^2}, \tag{5.1}$$

i.e., the root mean square of successive differences, which is a measure of variance for time series (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 4.1) in selecting frame-to-frame transitions with valid homography transformations. To determine failures in homography estimation, we apply the criteria outlined in Subsection 4.1.2. We then define the failure rate as:

$$\mathrm{F_m}(\mathbf{x}) = \frac{1}{n-1}\sum_{i=2}^{n}\begin{cases}0, & \text{if } H(x_{i-1}, x_i) \text{ exists,}\\ 1, & \text{otherwise.}\end{cases} \tag{5.2}$$

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.

5.3 Hyperparameters

Our method has five hyperparameters: the maximum speed-up (Δ) and the four weights for Equation 4.4 (*i.e.*, λ_v, λ_a, λ_m, λ_s). For our experiments, these parameters were set empirically.

The choice of weights ultimately depends on user preferences. For our experiments, we prioritized video semantics, using the song as a secondary element to enhance the hyperlapse. Acceleration regularization and camera stability were treated as tertiary concerns. Accordingly, we set:

$$\langle \lambda_v, \lambda_a, \lambda_s, \lambda_m \rangle = \langle 4, 2, 1, 1 \rangle. \tag{5.3}$$

We emphasize, however, that this arrangement is arbitrary and not a requirement for the weights.

For the maximum speed-up, our goal was to push Δ 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.

5.4 Competitors

We evaluate the frame sampling performance of our method against two state-of-the-art competitors that share similar 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 include the uniform fast-forward method, described in Algorithm 5, as a weak baseline.

Algorithm 5 Uniform Fast-Forward

Input: The song loudness curve A and the video semantic curve V.
Output: The list of selected frames I.

```
Initialize I as an empty list.
for a ∈ [0, |A|) do
    v ← ⌊a · |V|/|A|⌋
    Append v to I
return I
```

To evaluate SAS2, we use their Locality-constrained Linear Coding (LLC) sampler with the speed-ups listed in Table 2 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.

For MH, we used the authors' neural networks to extract emotion features from the songs and videos. We reimplemented their frame sampling algorithm based on the equivalence described in Subsection 4.2.5, preserving the original cost function and hyperparameters. This reimplementation was necessary to evaluate all video–song pairs efficiently, as the original Python code was too slow and would have made the evaluation unfeasible.

5.5 Ablation Study

We conduct an ablation study to provide a comprehensive comparison between our current method and its earlier version, referred to as SMH1, which was published in (Nepomuceno et al., 2024). SMH1 introduced the core idea of combining video semantics with music for hyperlapse generation but lacked several of the improvements presented

here. In this study, each modification introduced since SMH1 is evaluated both in isolation and in combination, with their effects discussed accordingly.

For the **semantic cost**, in Equation 4.5, 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 $\mathrm{C_v}(v, s)$ with:

$$\mathrm{C_v}(v) = 1 - V_v^\star \tag{5.4}$$

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

For the **acceleration cost**, with Equation 4.8, 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:

$$\mathrm{C_s}(s) = \left(\frac{s-1}{\Delta-1}\right)^2, \tag{5.5}$$

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 4.7) and the acceleration cost (Equation 4.8). In our prior approach, we followed Joshi et al. (2015) and Matos et al. (2021), where speed-related costs exceeding the threshold τ were truncated. In this ablation study, we apply the same truncation:

$$\mathrm{C_a}(a, s) = \frac{1}{\tau}\min\left\{(\hat{s}_a - s)^2, \tau\right\}, \tag{5.6}$$

and

$$\mathrm{C_s}(s, s') = \frac{1}{\tau}\min\left\{(s' - s)^2, \tau\right\}. \tag{5.7}$$

As Joshi et al. (2015) noted that the results were not particularly sensitive to the exact value of τ, we adopt the same $\tau = 200$ that was used previously by these works.

We also assess the impact of replacing our proposed dynamic programming algorithm with a computationally cheaper **greedy algorithm**, as described in Algorithm 6. While both use the same cost function as in Equation 4.4, 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.

Lastly, we evaluate two performance optimizations: the **pruning of infeasible positions** (Subsection 4.2.7) and the use of **internal parallelism** (Subsection 4.2.8).

Algorithm 6 Greedy Frame Sampling

$\mu \leftarrow \frac{|V|}{|A|}$
$I \leftarrow [0]$
$S \leftarrow [\mu]$
for $a \in [1, |A|)$ **do**
    $c_{\min} \leftarrow \infty$
    $s_{\min} \leftarrow 0$
    $v_{\min} \leftarrow 0$
    for $s \in [1, \Delta]$ **do**
        $v \leftarrow I[a-1] + s$
        if $v < |V|$ **then**
            $c \leftarrow C(a, v, s, S[a-1])$
            if $c < c_{\min}$ **then**
                $c_{\min} \leftarrow c$
                $s_{\min} \leftarrow s$
                $v_{\min} \leftarrow v$
    if $c_{\min} = \infty$ **then**
        break
    Append $s_{\min}$ to S
    Append $v_{\min}$ to I
return I

To isolate the effects of the algorithmic improvements and simulate a worst-case scenario for branch prediction, the tests used a randomized semantic curve equivalent to an hour-long video recorded at 60 FPS (216 000 frames) and a randomized audio loudness curve equivalent to a six-minute song (21 600 frames).

To quantify the benefits of pruning infeasible positions, we compared runtimes before and after the optimization introduced in Algorithm 4. For internal parallelism, we benchmarked the parallelized implementation using up to 15 threads and 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. The experiments were conducted on a system equipped with an AMD Ryzen 7 7800X3D (16 threads, 8 cores) and 32 GB of memory.

5.6 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 5.3 and Equation 4.4. 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.

5.7 Song Selection

We evaluate the song selection methods proposed in Subsection 4.4.2, along with two baseline methods, by comparing their effectiveness to the brute-force method—which serves as ground truth—using two new metrics: overestimation and underestimation.

The **naive method** is loosely inspired by the song selector proposed by Matos et al. (2023), which otherwise is not directly comparable due to different optimization goals and the absence of explicit modeling of loudness–speed correlation. It uniformly resamples the video semantic curve to match the length of each song's loudness curve, selecting the one with the most negative Spearman correlation (Spearman, 1904) between the two. This follows the naive assumption that if playback speed correlates positively with loudness and negatively with semantics, then the resampled semantic and loudness curves should be negatively correlated.

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

The **overestimation** metric measures how highly an alternate method ranks a candidate that brute-force selection deems suboptimal. Specifically, it reflects where the alternate method's top pick appears in the brute-force ranking. If this candidate ranks poorly under brute-force evaluation, it indicates that the alternate method has overestimated its quality. This metric evaluates the reliability of the alternate method as a direct substitute for brute-force selection.

The **underestimation** metric measures how poorly an alternate method ranks the optimal candidate found by brute-force selection. It reflects the position of the brute-force winner in the alternate method's ranking. A poor ranking suggests that the method has underestimated the candidate's quality. This metric is especially relevant for hybrid selection, where brute-force refinement is applied only to the top-ranked subset.

6 Results

In this chapter, 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.

6.1 Frame Sampling

This section presents the results of our frame sampling experiments, as described in Section 5.2. We report semantic score, correlation score, video instability, and acceleration score to evaluate the quality of our approach. The results of this analysis are summarized in Figure 7 and are discussed throughout the rest of this section.

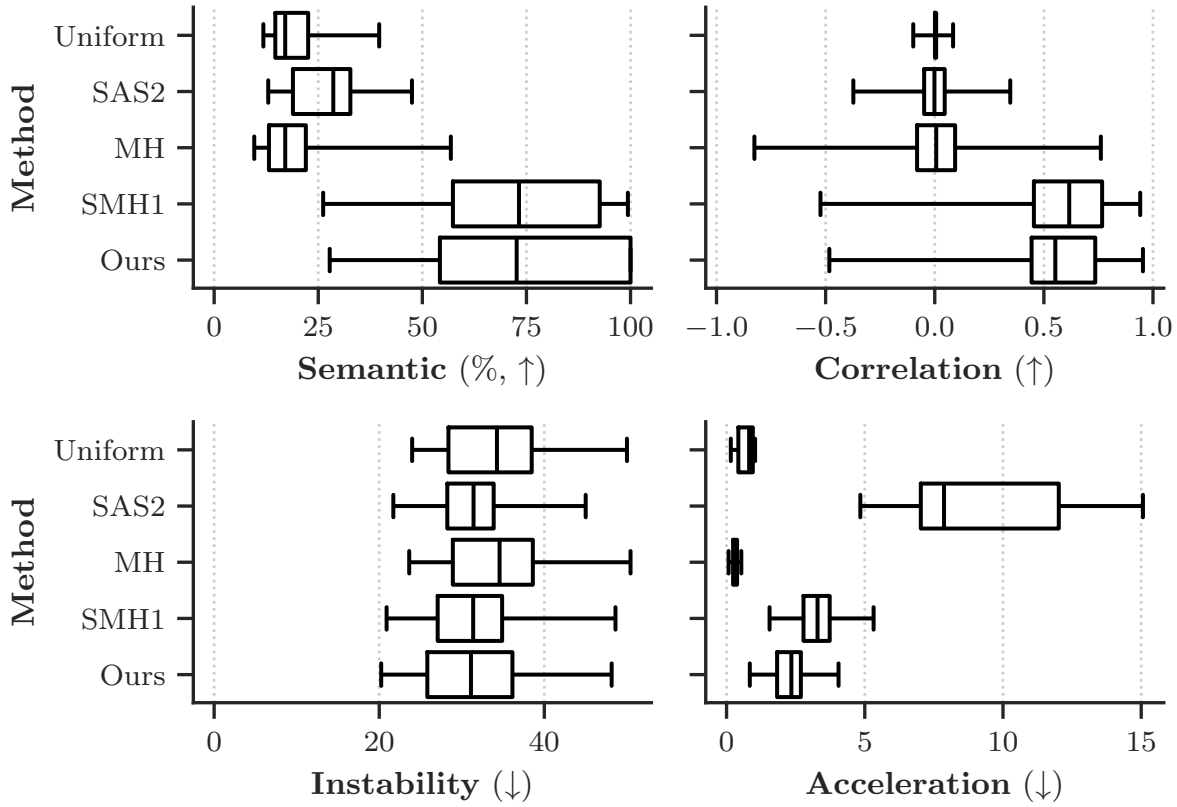


Figure 7 – Distribution of scores for each method across all videos, shown as box plots for the four main evaluation metrics: semantic score, correlation score, video instability, and acceleration score. Higher semantic and correlation scores are better; lower instability and acceleration scores are preferred.

6.1.1 Semantic Score

The results of the semantic score evaluation are presented in Table 3. 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.

Table 3 – Semantic score measuring preserved semantic information in the accelerated video, relative to the maximum possible for the same number of frames, without enforcing frame skip constraints. Since the evaluated methods must respect these constraints, perfect scores are not always attainable.

Video	**Semantic** (%, ↑)				**Low-speed Semantic** (%, ↑)			
	Uniform	SAS2	MH	Ours	Uniform	SAS2	MH	Ours
Biking 0p	14.71	15.28	14.86	**100.00**	5.28	9.12	4.86	**100.00**
Biking 25p	12.24	28.56	11.89	**35.26**	4.44	25.87	9.58	**35.21**
Biking 50p	19.78	27.10	19.33	**48.76**	0.40	19.04	10.47	**48.34**
Biking 50p 2	22.76	31.92	22.26	**79.58**	10.51	24.48	7.45	**79.08**
Driving 0p	13.64	34.36	12.54	**100.00**	5.84	29.66	6.69	**100.00**
Driving 25p	17.19	28.61	16.40	**99.97**	1.50	23.55	13.43	**99.97**
Driving 50p	13.47	18.92	12.40	**72.46**	4.13	12.43	11.22	**72.22**
Walking 0p	15.94	12.99	17.92	**100.00**	14.27	4.80	14.56	**100.00**
Walking 25p	20.10	32.71	19.62	**60.45**	17.64	27.66	8.09	**60.44**
Walking 50p	23.92	29.31	23.50	**58.90**	10.55	19.50	2.81	**58.80**
Walking 75p	39.24	47.51	39.08	**53.71**	21.44	30.60	9.17	**51.33**
Overall	19.36	27.93	19.07	**73.55**	8.73	20.61	8.94	**73.22**

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

A potential drawback of frame-level granularity is that important frames may be selected at playback speeds too high to be properly perceived. To address this, we added a penalty term in Equation 4.5 that discourages assigning high playback speeds to such frames. This is supported by the *low-speed semantic score*, which remains nearly unchanged for our method—indicating that important frames are consistently shown at low speeds, unlike in other methods where a significant drop is observed. The trade-off introduced by this constraint is discussed in the ablation study.

An example of how each method performs on a particular instance is shown in Figure 8, which illustrates how semantic methods are often forced to select important frames outside the ground truth. This highlights the unrealistic nature of the ground truth:

it does not respect the maximum allowed frame skip (Δ) and leaves unacceptable gaps. As noted in Equation 4.24, this means perfect (100%) scores are not always achievable under this evaluation.

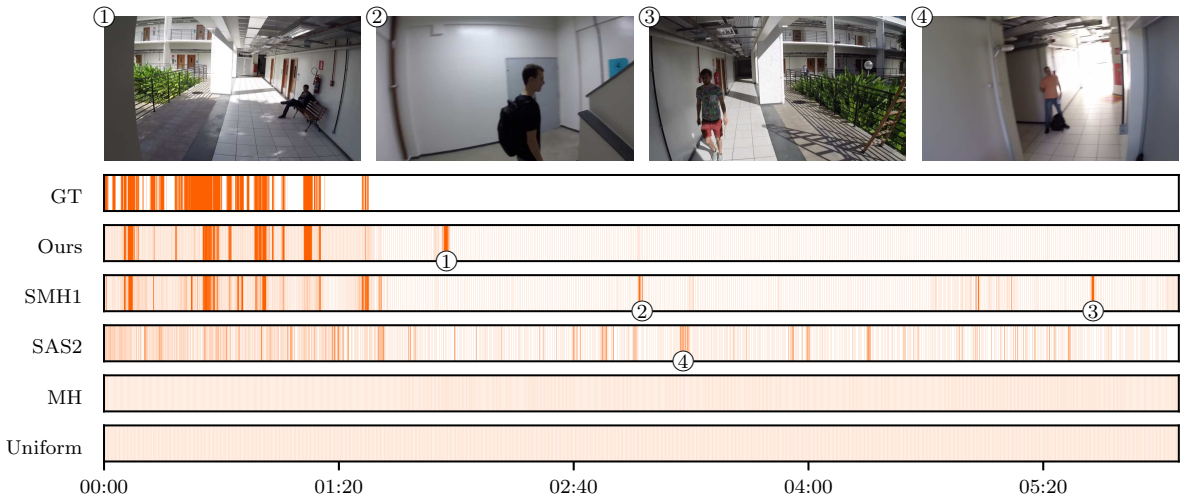


Figure 8 – Comparison of frame sampling methods with the semantic ground truth for *Walking 25p*, where dark regions indicate sampled frames and white regions indicate skipped ones. The semantic ground truth marks the most important frames. Numbered segments highlight examples of selected frames not included in the ground truth.

6.1.2 Correlation Score

As shown in Table 4, our methods are the only ones to achieve a high correlation between song loudness and video speed-up. This result aligns with expectations for the uniform fast-forward and SAS2, which are non-musical methods. The poor performance of MH on this metric, despite being a musical method, suggests that aligning induced emotions between video and music is insufficient to the goals of our method.

Figure 9 illustrates the impact of song–video pairings on the trade-off between semantic retention and loudness–speed correlation. In the first example, the songs increasing loudness over time aligns naturally with the videos speed-up curve, allowing both objectives to be satisfied. In the second, the songs decreasing loudness conflicts with the videos structure, forcing the algorithm to prioritize semantic retention over correlation. This explains why, as seen in Figure 7, some outputs exhibit negative correlation scores: in poorly matched pairs, such trade-offs are unavoidable. These cases reinforce the need for an informed song selection process, as discussed in Section 4.4.

6.1.3 Acceleration Score

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

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

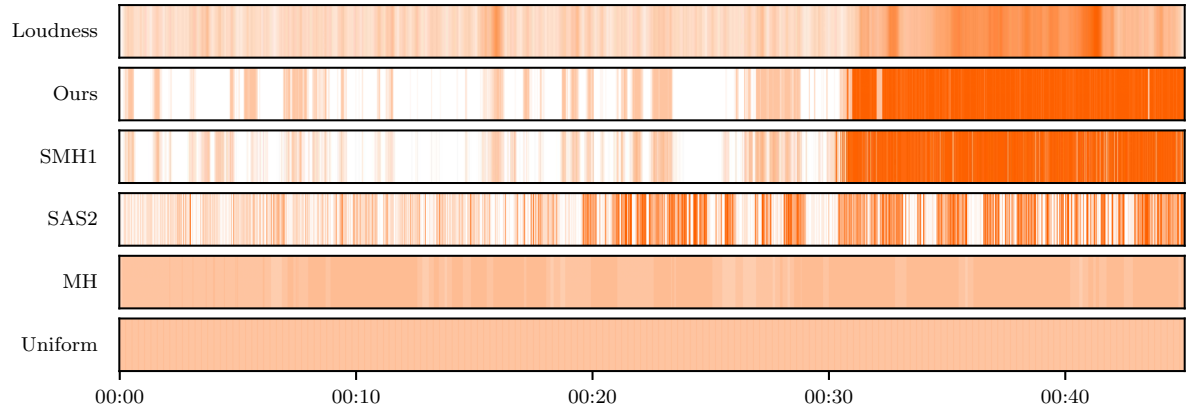
Video	**Correlation** (↑)			
	Uniform	SAS2	MH	Ours
Biking 0p	0.00	0.00	0.01	**0.82**
Biking 25p	0.00	-0.01	0.01	**0.47**
Biking 50p	0.01	0.00	0.01	**0.63**
Biking 50p 2	0.00	0.00	0.01	**0.49**
Driving 0p	0.02	-0.01	0.00	**0.86**
Driving 25p	0.00	0.00	0.01	**0.40**
Driving 50p	0.00	0.00	0.01	**0.48**
Walking 0p	0.01	0.01	0.00	**0.80**
Walking 25p	0.01	0.01	0.01	**0.40**
Walking 50p	0.00	-0.01	0.02	**0.49**
Walking 75p	0.00	0.01	0.01	**0.45**
Overall	0.00	0.00	0.01	**0.57**

possible, the uniform fast-forward achieves fractional speed-ups by alternating between integer speeds. For instance, a speed-up of 11.5× (*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.

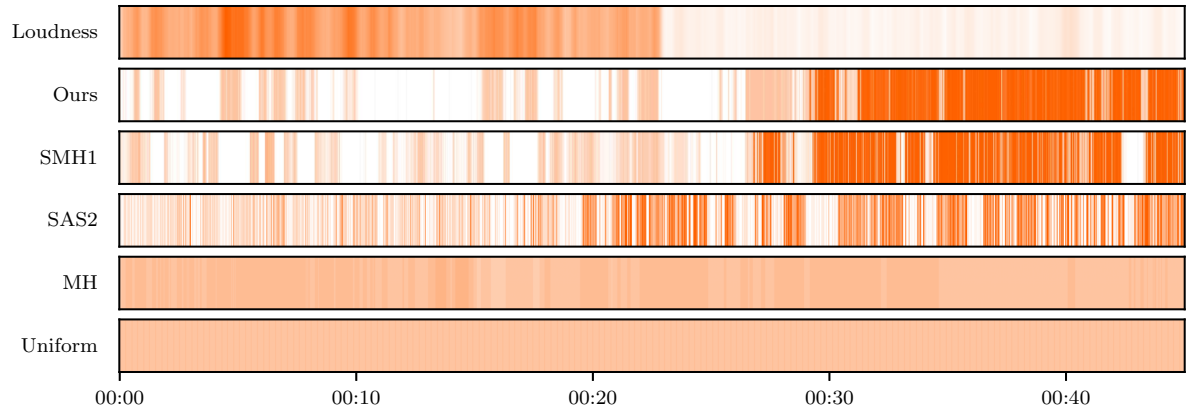
Table 5 – Evaluation of acceleration, measured as the root mean square of successive differences in playback speed.

Video	**Acceleration** (↓)			
	Uniform	SAS2	MH	Ours
Biking 0p	0.84	6.25	**0.22**	1.12
Biking 25p	0.91	15.06	**0.33**	2.48
Biking 50p	**0.23**	11.63	0.45	2.97
Biking 50p 2	0.97	7.64	**0.27**	2.37
Driving 0p	**0.19**	7.24	0.37	1.70
Driving 25p	0.43	7.03	**0.41**	2.21
Driving 50p	0.81	8.64	**0.27**	3.75
Walking 0p	0.43	4.84	**0.26**	1.69
Walking 25p	0.54	7.87	**0.28**	2.22
Walking 50p	0.95	12.66	**0.27**	2.78
Walking 75p	0.99	12.02	**0.23**	2.33
Overall	0.66	9.17	**0.30**	2.33

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



(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.



(b) The song starts loud and ends quietly, clashing with the videos structure. In such cases, our method prioritizes semantic content over correlation.

Figure 9 – Examples of well-matching and poorly-matching songs for the speed-up curve of the video in Figure 8, 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 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 leaves large gaps between scenes, which are interpreted as abrupt jumps in speed by the acceleration score.

Our method achieves acceleration scores between those of MH and SAS2. Scores as low as MH are undesirable, as we intentionally use speed changes to convey information. On the other hand, high acceleration scores, as seen in SAS2, indicate abrupt gaps between scenes that break the video's temporal continuity. In our approach, the constraint of never exceeding a maximum speed (*i.e.*, Δ) avoids such gaps.

6.1.4 Video Instability

The results of the **video instability** evaluation are presented in Table 6, which measures instability as the pixel-wise standard deviation of consecutive frames within a sliding window. Our method performs comparably to SAS2 and outperforms 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** (↓)			
	Uniform	SAS2	MH	Ours
Biking 0p	24.05	21.71	24.08	**21.70**
Biking 25p	49.98	**45.01**	50.21	47.32
Biking 50p	32.97	**29.22**	32.71	30.98
Biking 50p 2	24.55	23.99	24.77	**21.96**
Driving 0p	42.81	**37.63**	42.80	37.89
Driving 25p	36.67	31.42	36.54	**30.79**
Driving 50p	38.54	33.87	38.72	**32.38**
Walking 0p	28.37	28.25	28.63	**25.23**
Walking 25p	32.45	30.06	32.74	**28.48**
Walking 50p	34.28	33.30	34.54	**32.47**
Walking 75p	37.69	**33.51**	38.00	36.31
Overall	34.76	31.63	34.89	**31.41**

We suspect that this outcome is influenced by differences in playback speed variation. Semantic-based methods like ours and SAS2 tend to slow down during important segments, selecting many similar frames in sequence and thereby increasing local frame similarity. In contrast, uniform and MH maintain more consistent speed-ups, as shown in Table 5.

We further hypothesize that instability is minimized when playback alternates between short and long skips: local similarity is highest for small skips, quickly decreases with intermediate ones, and then plateaus as frame differences saturate. However, we were not able to ascertain whether this is truly the case.

6.1.5 Ablation Study

The results of the ablation study are presented in Figure 10 and Table 7 and discussed throughout this subsection. For reference, we also include the previous version of our method (SMH1) (Nepomuceno et al., 2024), which simultaneously incorporates the *semantic cost*, *acceleration cost* and *cost normalization* ablations.

The results of the **semantic cost** (C_v) ablation show that our updated formulation trades off some overall semantic score to better emphasize important frames. In the

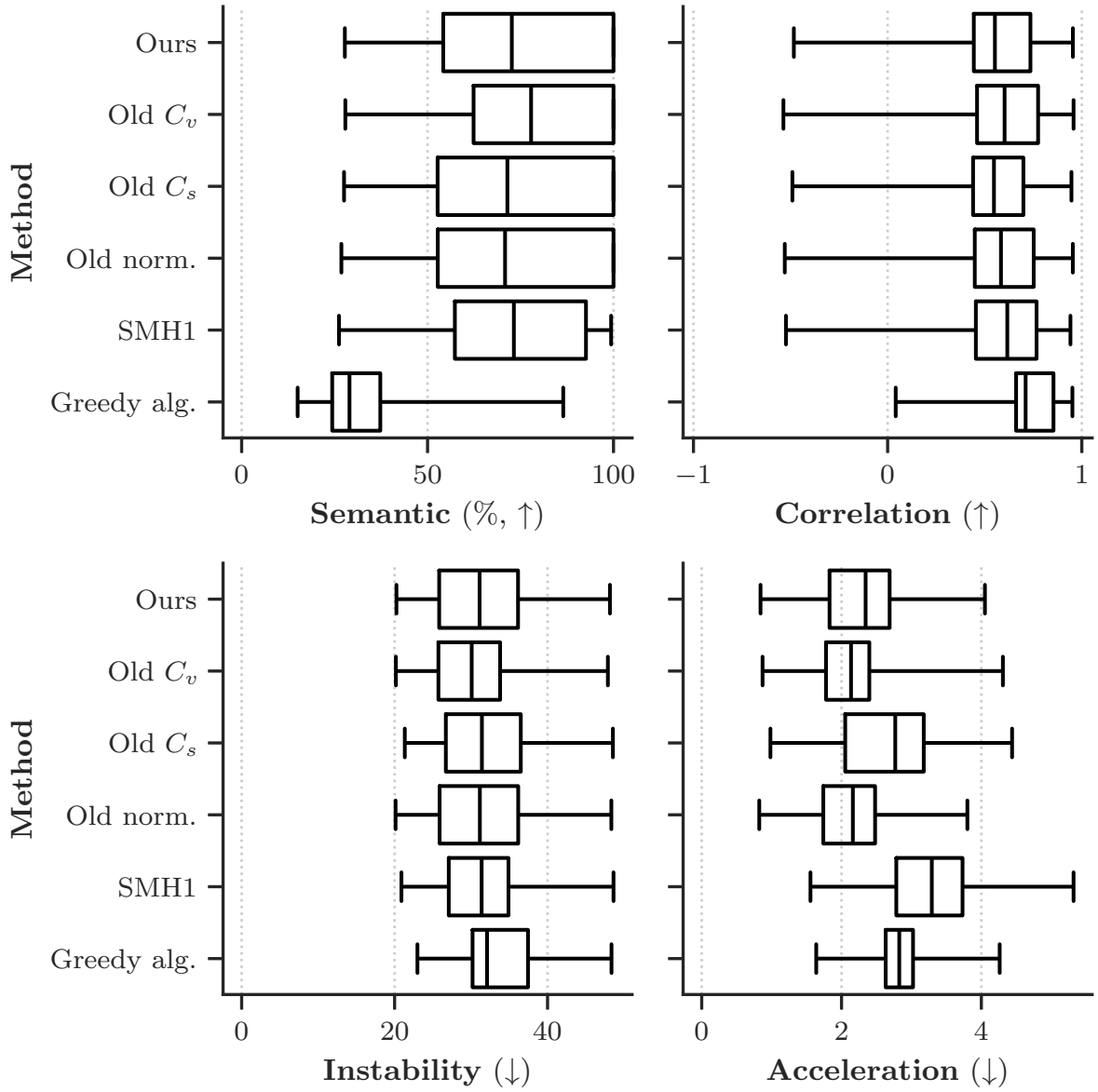


Figure 10 – Box plots showing results for: (i) our unablated method, (ii) semantic cost ablation, (iii) acceleration cost ablation, (iv) cost normalization ablation, (v) the previous version of our method, and (vi) the greedy algorithm.

previous version (SMH1), the lack of a playback speed penalty led to some important frames being shown too quickly. To address this, we introduced a term in Equation 4.5 that penalizes high playback speeds for high-semantic regions. While this adjustment slightly reduces performance on high-semantic videos (*e.g.*, 50p and 75p), it improves results on low-semantic ones. Furthermore, as shown in Table 7, our unablated method outperforms both SMH1 and the older C_v under the low-speed semantic score, indicating better preservation of important content during slowed playback.

The ablation results show that the unablated version of the **acceleration cost** (C_s) more effectively smooths changes in playback speed without degrading other metrics. Our current method replaces the earlier *speed-up cost*, which simply penalized high speeds,

Table 7 – Ablation study results. The low-speed semantic score quantifies how much semantic information is preserved in frames shown below the target speed-up. Truncation measures the percentage of video or audio left out due to the greedy alignment algorithm.

Video	**Low-speed Semantic** (%, ↑)			**Truncation** (%, ↓)	
	Ours	Old C_v	SMH1	Video	Audio
Biking 0p	**100.00**	98.07	89.31	**0.01**	31.42
Biking 25p	**35.21**	32.25	30.32	30.08	**0.00**
Biking 50p	**48.34**	45.28	42.42	22.59	**0.00**
Biking 50p 2	**79.08**	76.14	74.00	**0.01**	16.99
Driving 0p	**100.00**	96.18	90.56	**0.01**	24.91
Driving 25p	**99.97**	96.03	88.69	**0.01**	28.41
Driving 50p	**72.22**	71.02	67.02	**0.06**	2.90
Walking 0p	**100.00**	99.16	94.80	**0.01**	34.86
Walking 25p	**60.44**	58.25	54.71	1.74	**1.48**
Walking 50p	**58.80**	54.46	52.00	23.61	**0.00**
Walking 75p	**51.33**	40.65	44.14	35.27	**0.00**
Overall	**73.22**	69.77	66.18	**10.31**	12.82

with a more targeted *acceleration cost* (Equation 4.8) that discourages abrupt transitions. The old C_s performs worse on the acceleration metric while yielding similar results on the others, justifying its replacement.

We find that modifying the **cost normalization** in Equation 4.7 and Equation 4.8 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 Δ controls the maximum playback speed, whereas previously, τ also had to be adjusted simultaneously.

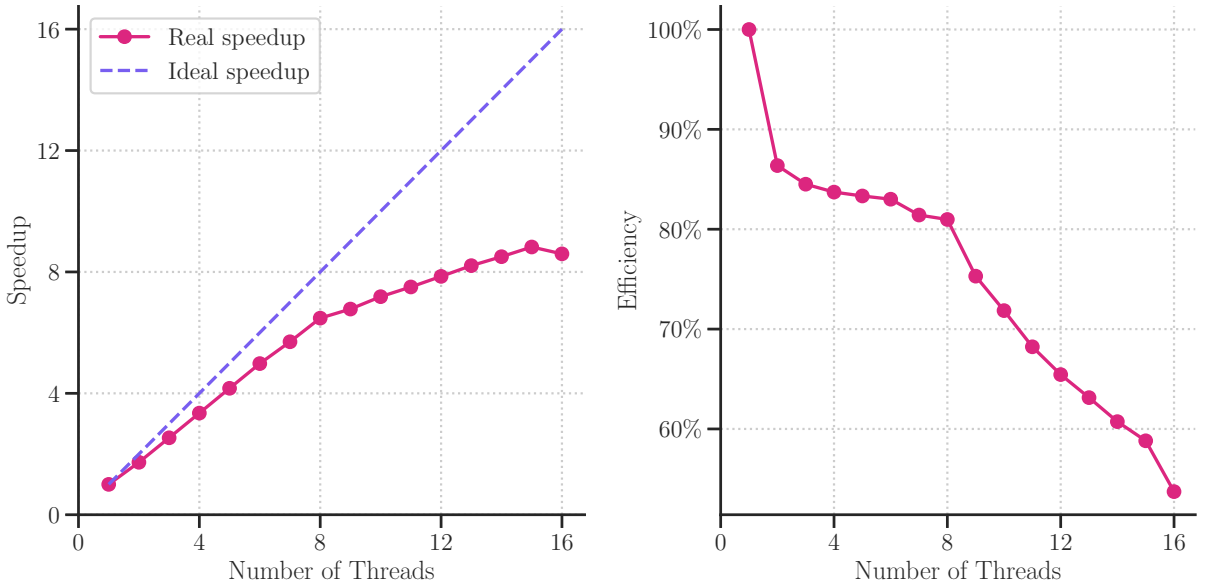
Combined, these changes refine our method relative to SMH1 by providing better emphasis on important frames, smoother playback transitions, and a simpler interface for parameter selection. While the semantic cost introduces a slight trade-off in high-semantic videos, it improves performance in low-semantic ones and yields higher low-speed semantic scores. The new acceleration cost reduces abrupt speed transitions without degrading other metrics, and the normalization change simplifies the formulation without impacting results.

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 7 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.

The **internal parallelism** ablation demonstrates significant improvements in execution time, as illustrated in Figure 11. 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.



(a) Speedup is the ratio between the wall clock time before and after parallelization, measuring the raw improvement compared to an idealized, linear speedup.

(b) Efficiency is the ratio between speedup and number of threads, and measures how effectively the CPU cores are being utilized when executing the parallel algorithm.

Figure 11 – Scaling study for our algorithm. The parallelization achieves an almost linear scaling for up to 8 threads; further increases in threads have lower efficiency, which we attribute to the distinction between physical and virtual cores in the experimental setup.

The final item in our ablation study is the **pruning of infeasible positions** shown in Figure 5. Before this change, we measured a total runtime of 395 seconds. After the change, the measured time was 279 seconds—a 29% reduction. No changes in memory

usage were observed, as $S[a, v]$ is allocated as a contiguous block regardless of its sparsity. Since the pruning is exact and introduces no approximations, this change offers a clear benefit with no downside once implemented.

6.2 Sensitivity Analysis

Figure 12 summarizes the sensitivity analysis results, showing how the removal of each cost component influences our method's performance across the evaluated metrics.

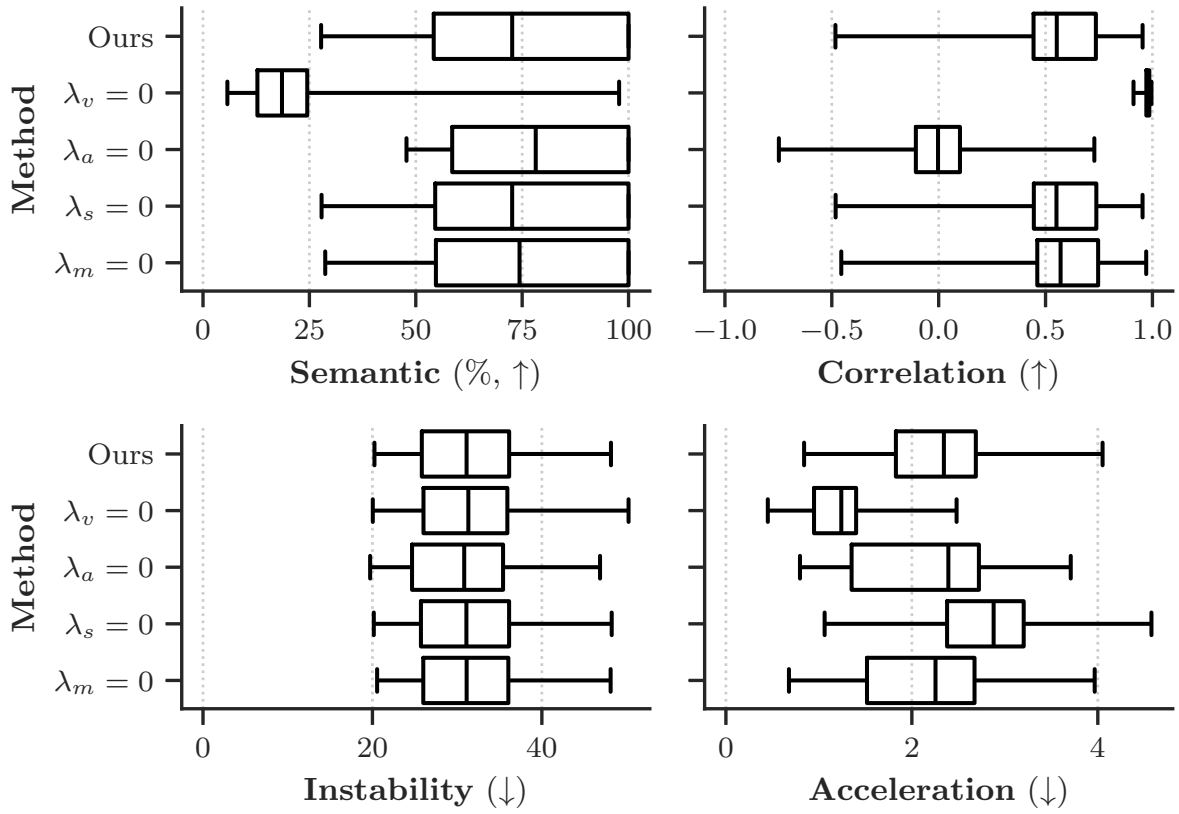


Figure 12 – Sensitivity analysis results, aggregated over the entire dataset.

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 noticeably affect the evaluated metrics—including, unexpectedly, the instability metric. This may hint at the relationship between instability and playback speed hypothesized in Subsection 6.1.4. Further analysis shows that this removal leads to a tenfold increase in the *homography failure rate*, as shown in Table 8. We therefore 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 8 – Proportion of frame-to-frame transitions in the output video where homography estimation failed. The worst value in each row is *italicized*.

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

6.3 Song Selection

Here, we provide the results of our song selection experiments described in Section 5.7. In Table 9, we assess each method's agreement with the brute-force rankings using overestimation and underestimation metrics. Additionally, Table 10 compares the runtime of the methods to highlight their computational efficiency. Overestimation and underestimation are both zero if and only if the alternate method ranks the same candidate as the brute-force method. In this case, the approximate method successfully identifies the best song.

Table 9 – 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.

Video	**Overestimation** (↓)			**Underestimation** (↓)		
	Naive	Random[1]	Subsampled	Naive	Random[1]	Subsampled
Biking 0p	1742	**13**	**38**	1470	–	**31**
Biking 25p	1626	**5**	**1**	1358	–	**68**
Biking 50p 2	1587	**5**	**15**	640	–	**28**
Biking 50p	1708	**3**	**6**	1483	–	585
Driving 0p	**75**	**54**	**14**	1661	–	**35**
Driving 25p	1320	**24**	**0**	1397	–	**0**
Driving 50p	298	**2**	**21**	1177	–	**8**
Walking 0p	1738	**1**	180	942	–	**57**
Walking 25p	1736	**14**	**13**	1432	–	**2**
Walking 50p	1582	**0**	**11**	861	**0**	**17**
Walking 75p	1723	**6**	**6**	1424	–	206
Average	1376	**13**	**28**	1259	–	**94**

[1] The random method is unordered by default; in this table, we include its results after sorting the selected candidates using the brute-force method. Entries marked as – mean that the best song is not in the top-100.

While the **naive method** runs in just 2–4 seconds, its simplicity leads to significant overestimation and underestimation. This limitation arises because uniform resampling fails to account for the behavior in hyperlapses where important parts of the video are stretched out and unimportant parts are squished.

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 three methods from Subsection 4.4.2 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 9, it often is able to identify the optimal song due to the underestimation of the subsampled method being below 100, effectively combining its strengths with those of the brute-force method. In 9

Table 10 – Runtime comparison of song selection methods for each video. Times are calculated assuming no parallelization.

Video	**Runtime** (mm:ss, ↓)				
	Brute Force	Subsampled	Hybrid	Random[1]	Naive
Biking 0p	116:36	1:01	7:42	6:37	0:03
Biking 25p	45:23	0:21	2:56	2:35	0:03
Biking 50p 2	109:33	0:57	7:11	6:13	0:02
Biking 50p	179:35	1:27	11:35	10:12	0:04
Driving 0p	34:35	0:17	2:16	1:58	0:02
Driving 25p	31:52	0:16	2:04	1:49	0:02
Driving 50p	42:03	0:21	2:44	2:23	0:02
Walking 0p	27:35	0:16	1:51	1:34	0:02
Walking 25p	41:21	0:20	2:42	2:21	0:02
Walking 50p	39:41	0:21	2:37	2:16	0:02
Walking 75p	39:34	0:23	2:38	2:14	0:02

[1] The random selection step takes negligible time; the reported runtime corresponds to the brute-force evaluation used to sort the 100 randomly sampled songs.

of the 11 cases, a hit was found within the top 100 candidates, and in 7 cases, a top 50 would also have worked. One case resulted in a clear failure, suggesting that the method is generally reliable but susceptible to outliers.

7 Conclusions

In this dissertation, we introduced a method that combines concepts from *semantic hyperlapses* and *musical hyperlapses*. Building on the common goals of semantic hyperlapse methods—fast-forwarding egocentric videos of daily activities while minimizing discomfort from camera motion and allocating more time to the interesting parts of the video—we also adopt the goal of musical hyperlapses: incorporating a song into the hyperlapse in a way that complements the video. These ideas tie into the goal of creating *semantic and musical hyperlapses* that are both useful and enjoyable to watch.

To that end, we proposed a method to create hyperlapses with the following goals: (i) maximize the total semantic content; (ii) maximize the correlation between playback speed and the loudness of a song chosen by the user; (iii) minimize shaking in the video; and (iv) minimize abrupt changes in playback speed.

Segments where the playback speed is reduced to show important frames must coincide with the quiet parts of the song, drawing attention to the video. Conversely, segments with higher speed-ups should align with the louder parts of the song. Frame transitions with a valid homography and minimal motion are preferred to aid homography-based video stabilizers. Additionally, minimizing abrupt or frequent changes in playback speed ensures the video remains easy to follow.

Beyond the simpler, didactic version of the algorithm, we also described three optimizations, including parallelization, that make it more suitable for practical use on real computers.

We compared our method to state-of-the-art competitors for semantic hyperlapses and musical hyperlapses. Our experiments show that our approach performs significantly better than the competitors in retaining the semantic content of the video. Our method is the only one to achieve a non-zero score in loudness–speed correlation, which is expected, as it is the only method explicitly designed to optimize this metric. These results were achieved without negatively impacting other aspects of the hyperlapse—our method demonstrates comparable performance to the competitors in both video stability and avoiding abrupt changes to speed.

When the video and song are well-matched, both the semantic score and the correlation score can be maximized. However, a poor match forces a trade-off, where improving one score reduces the other. With a large collection of songs available, this trade-off can be mitigated by selecting the song that best matches the video. Manually evaluating each song, however, is a tedious and time-consuming process. To address this, we proposed three approaches to automate the selection: (i) the exact but computationally

intensive brute-force method; (ii) an approximate method that subsamples the video and the song; and (iii) a hybrid approach that first uses the approximate method to select the top 100 candidates, followed by the brute-force method applied to this subset.

When comparing the song selection methods, we found that the hybrid approach offers the best balance between runtime and quality. In most cases, it is able to identify the best matching song while maintaining a reasonable execution time.

7.1 Limitations

We identified three key limitations in our method. First, the pre-processing phase, and the video composition phase if stabilization is applied, take significantly longer than the main algorithm. While this cost is amortized across multiple runs, it can be prohibitive if the user only wishes to create a single hyperlapse for one video and song.

Second, our method treats playback speeds as linear, even though human perception of speed changes is not. For example, the difference between 1× and 2× speed feels much larger than the difference between 2× and 3× speed. This limitation makes subtle speed changes harder to perceive at higher playback rates. Although our attempts with non-linear playback speeds did not yield satisfactory results, we believe this approach has potential and warrants further exploration.

Finally, the global optimization nature of our frame sampling algorithm means that changes in one part of the video can influence decisions in entirely unrelated regions. We believe this could result in cases where the hyperlapse slows down arbitrarily in one section to improve the overall score in a different, unrelated part of the video. While we have not observed specific examples of this behavior, it remains a potential limitation that warrants consideration.

7.2 Future Works

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

Another potential direction is to explore the use of 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.*, 2 hours or more), finding songs of the required length, let alone ones that match well, can be challenging.

Generating the song would allow users to instantly obtain a soundtrack tailored to the video's length and desired speed-up rate.

Bibliography

ALJANAKI, A.; YANG, Y.-H.; SOLEYMANI, M. Developing a benchmark for emotional analysis of music. **PLOS ONE**, Public Library of Science, v. 12, n. 3, p. 1–22, Mar. 2017. DOI: 10.1371/journal.pone.0173392. Available from: <https://doi.org/10.1371/journal.pone.0173392>. Cit. on p. 40.

BARÁTH, D.; NOSKOVA, J.; IVASHECHKIN, M.; MATAS, J. MAGSAC++, a Fast, Reliable and Accurate Robust Estimator. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], June 2020. P. 1301–1309. DOI: 10.1109/CVPR42600.2020.00138. Cit. on pp. 16, 27.

CHENG, K.-Y.; LUO, S.-J.; CHEN, B.-Y.; CHU, H.-H. SmartPlayer: user-centric video fast-forwarding. In: CHI '09: CHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS. **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**. Boston, MA, USA: ACM, Apr. 2009. P. 789–798. ISBN 9781605582467. DOI: 10.1145/1518701.1518823. Cit. on pp. 12, 20, 22, 31.

EUROPEAN BROADCASTING UNION. **Loudness Metering: 'EBU Mode' Metering to Supplement Loudness Normalisation**. [S.l.], 2023. Available from: <https://tech.ebu.ch/publications/tech3341>. Cit. on pp. 13, 17, 27.

FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 24, n. 6, p. 381–395, June 1981. ISSN 0001-0782. DOI: 10.1145/358669.358692. Available from: <https://doi.org/10.1145/358669.358692>. Cit. on p. 16.

FURLAN, V. S.; BAJCSY, R.; NASCIMENTO, E. R. Fast Forwarding Egocentric Videos by Listening and Watching. In: IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR) WORKSHOPS. **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops**. [S.l.: s.n.], June 2018. P. 2504–2507. DOI: 10.48550/arXiv.1806.04620. Cit. on pp. 15, 17, 21, 22.

HALPERIN, T.; POLEG, Y.; ARORA, C.; PELEG, S. EgoSampling: Wide View Hyperlapse From Egocentric Videos. **IEEE Transactions on Circuits and Systems**

for Video Technology, v. 28, n. 5, p. 1248–1259, May 2018. ISSN 1558-2205. DOI: 10.1109/TCSVT.2017.2651051. Cit. on p. 20.

HAMMING, R. W. Error detecting and error correcting codes. **The Bell System Technical Journal**, v. 29, n. 2, p. 147–160, Apr. 1950. ISSN 0005-8580. DOI: 10.1002/j.1538-7305.1950.tb00463.x. Cit. on pp. 16, 27.

INTERNATIONAL TELECOMMUNICATION UNION. **Algorithms to Measure Audio Programme Loudness and True-Peak Audio Level**. [S.l.], 2023. Available from: <https://www.itu.int/rec/R-REC-BS.1770>. Cit. on p. 17.

JOSHI, N.; KIENZLE, W.; TOELLE, M.; UYTTENDAELE, M.; COHEN, M. F. Real-Time Hyperlapse Creation via Optimal Frame Selection. **ACM Trans. Graph.**, Association for Computing Machinery, v. 34, n. 4, July 2015. ISSN 0730-0301. DOI: 10.1145/2766954. Cit. on pp. 11, 20, 22, 23, 26, 28, 31, 44.

KARPENKO, A. **The Technology behind Hyperlapse from Instagram**. [S.l.: s.n.], Aug. 2014. Available from: <http://instagram-engineering.tumblr.com/post/95922900787/hyperlapse>. Cit. on pp. 19, 22.

KOPF, J.; COHEN, M.; SZELISKI, R. First-person Hyperlapse Videos. In: ACM Transactions on Graphics (Proc. SIGGRAPH 2014). [S.l.]: ACM - Association for Computing Machinery, Aug. 2014. v. 33. DOI: 10.1145/2601097.2601195. Cit. on pp. 19, 22.

LU, L.; LIU, D.; ZHANG, H.-J. Automatic mood detection and tracking of music audio signals. **IEEE Transactions on Audio, Speech, and Language Processing**, v. 14, n. 1, p. 5–18, 2006. DOI: 10.1109/TSA.2005.860344. Cit. on pp. 17, 27.

MATOS, D. d.; RAMOS, W.; ROMANHOL, L.; NASCIMENTO, E. R. Musical Hyperlapse: A Multimodal Approach to Accelerate First-Person Videos. In: 34TH SIBGRAPI Conference on Graphics, Patterns and Images. [S.l.: s.n.], Oct. 2021. P. 184–191. DOI: 10.1109/SIBGRAPI54419.2021.00033. Cit. on pp. 12, 13, 15, 17, 22, 23, 28, 31, 32, 40, 42–44.

MATOS, D. d.; RAMOS, W.; SILVA, M.; ROMANHOL, L.; NASCIMENTO, E. R. A multimodal hyperlapse method based on video and songs emotion alignment. **Pattern Recognition Letters**, v. 166, p. 174–181, 2023. ISSN 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2022.08.014. Cit. on pp. 13, 22, 46.

MOLINO, A. G. del; TAN, C.; LIM, J.-H.; TAN, A.-H. Summarization of Egocentric Videos: A Comprehensive Survey. **IEEE Transactions on Human-Machine Systems**, v. 47, n. 1, p. 65–76, Feb. 2017. ISSN 2168-2305. DOI: 10.1109/THMS.2016.2623480. Cit. on pp. 11, 19.

NEPOMUCENO, R.; FERREIRA, L.; SILVA, M. A Multimodal Frame Sampling Algorithm for Semantic Hyperlapses with Musical Alignment. In: PROCEEDINGS of the 37th Conference on Graphics, Patterns and Images (SIBGRAPI). [S.l.: s.n.], 2024. DOI: 10.1109/SIBGRAPI62404.2024.10716336. Available from: <http://urlib.net/ibi/8JMKD3MGPEW34M/4BU46CP>. Cit. on pp. 43, 52.

NEUMANN, J. von; KENT, R. H.; BELLINSON, H. R.; HART, B. I. The Mean Square Successive Difference. **The Annals of Mathematical Statistics**, Institute of Mathematical Statistics, v. 12, n. 2, p. 153–162, 1941. DOI: 10.1214/aoms/1177731746. Available from: <https://doi.org/10.1214/aoms/1177731746>. Cit. on p. 42.

NEVES, A. C.; SILVA, M. M.; CAMPOS, M. F. M.; NASCIMENTO, E. R. 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). [S.l.: s.n.], June 2020. P. 1–4. DOI: 10.48550/arXiv.2006.05569. Available from: <https://arxiv.org/abs/2006.05569>. Cit. on pp. 15, 21, 22.

OGAWA, M.; YAMASAKI, T.; AIZAWA, K. Hyperlapse generation of omnidirectional videos by adaptive sampling based on 3D camera positions. In: IEEE International Conference on Image Processing (ICIP). [S.l.: s.n.], Sept. 2017. P. 2124–2128. DOI: 10.1109/ICIP.2017.8296657. Cit. on p. 20.

OKAMOTO, M.; YANAI, K. Summarization of Egocentric Moving Videos for Generating Walking Route Guidance. In: KLETTE, R.; RIVERA, M.; SATOH, S. (Eds.). **Image and Video Technology**. [S.l.: s.n.], 2014. P. 431–442. ISBN 978-3-642-53842-1. DOI: 10.1007/978-3-642-53842-1-37. Cit. on pp. 11, 15, 19, 20, 22.

PEARSON, K. Note on Regression and Inheritance in the Case of Two Parents. **Proceedings of the Royal Society of London Series I**, v. 58, p. 240–242, Jan. 1895. DOI: 10.1098/rspl.1895.0041. Cit. on p. 18.

POLEG, Y.; HALPERIN, T.; ARORA, C.; PELEG, S. EgoSampling: Fast-forward and stereo for egocentric videos. In: IEEE Conf. on Computer Vision and Pattern

Recognition (CVPR). [S.l.: s.n.], June 2015. P. 4768–4776. DOI: 10.1109/CVPR.2015.7299109. Cit. on pp. 19, 21, 22.

POTJE, G.; CADAR, F.; ARAUJO, A.; MARTINS, R.; NASCIMENTO, E. R. XFeat: Accelerated Features for Lightweight Image Matching. In: PROCEEDINGS of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], June 2024. P. 2682–2691. Cit. on p. 16.

QIU, X.; WU, H.; HU, R. The impact of quantile and rank normalization procedures on the testing power of gene differential expression analysis. **BMC Bioinformatics**, v. 14, n. 1, p. 124, Apr. 2013. ISSN 1471-2105. DOI: 10.1186/1471-2105-14-124. Available from: <https://doi.org/10.1186/1471-2105-14-124>. Cit. on p. 18.

RAMOS, W.; SILVA, M.; ARAUJO, E.; MARCOLINO, L. S.; NASCIMENTO, E. Straight to the Point: Fast-Forwarding Videos via Reinforcement Learning Using Textual Data. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], June 2020. P. 10928–10937. DOI: 10.1109/CVPR42600.2020.01094. Cit. on pp. 20, 22.

RAMOS, W.; SILVA, M.; ARAUJO, E.; MOURA, V.; OLIVEIRA, K.; SORIANO MARCOLINO, L.; NASCIMENTO, E. Text-driven video acceleration: A weakly-supervised reinforcement learning method. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, p. 1–1, 2022. ISSN 1939-3539. DOI: https://doi.org/10.1109/TPAMI.2022.3157198. Cit. on pp. 20, 22.

RAMOS, W. L. S.; SILVA, M. M.; ARAUJO, E. R.; NEVES, A. C.; NASCIMENTO, E. R. Personalizing Fast-Forward Videos Based on Visual and Textual Features from Social Network. In: IEEE Winter Conference on Applications of Computer Vision (WACV). [S.l.: s.n.], Mar. 2020. P. 3260–3269. DOI: 10.1109/WACV45572.2020.9093330. Cit. on pp. 15, 21, 26.

RAMOS, W. L. S.; SILVA, M. M.; CAMPOS, M. F. M.; NASCIMENTO, E. R. Fast-forward video based on semantic extraction. In: IEEE International Conference on Image Processing (ICIP). [S.l.: s.n.], Sept. 2016. P. 3334–3338. DOI: 10.1109/ICIP.2016.7532977. Cit. on pp. 11, 13, 15, 21–23, 25.

RANI, P.; JANGID, A.; NAMBOODIRI, V. P.; VENKATESH, K. S. Visual Odometry Based Omni-directional Hyperlapse. In: RAMESHAN, R.; ARORA, C.; DUTTA ROY, S. (Eds.). **Computer Vision, Pattern Recognition, Image**

Processing, and Graphics. [S.l.: s.n.], 2018. P. 3–13. ISBN 978-981-13-0020-2. DOI: 10.1007/978-981-13-0020-2_1. Cit. on p. 20.

RUBLEE, E.; RABAUD, V.; KONOLIGE, K.; BRADSKI, G. ORB: An efficient alternative to SIFT or SURF. In: 2011 International Conference on Computer Vision. [S.l.: s.n.], 2011. P. 2564–2571. DOI: 10.1109/ICCV.2011.6126544. Cit. on pp. 16, 26.

SILVA, M.; RAMOS, W.; CAMPOS, M.; NASCIMENTO, E. R. A Sparse Sampling-Based Framework for Semantic Fast-Forward of First-Person Videos. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 43, n. 4, p. 1438–1444, Apr. 2021. ISSN 1939-3539. DOI: 10.1109/TPAMI.2020.2983929. Cit. on pp. 13, 19, 21, 22, 43, 51.

SILVA, M.; RAMOS, W.; FERREIRA, J.; CHAMONE, F.; CAMPOS, M.; NASCIMENTO, E. R. 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. [S.l.: s.n.], 2018. P. 2383–2392. DOI: 10.1109/CVPR.2018.00253. Cit. on pp. 21, 22.

SILVA, M. M.; RAMOS, W. L. S.; CHAMONE, F. C.; FERREIRA, J. P. K.; CAMPOS, M. F. M.; NASCIMENTO, E. R. 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**, v. 53, p. 55–64, 2018. ISSN 1047-3203. DOI: 10.1016/j.jvcir.2018.02.013. Cit. on pp. 21, 22, 38, 42.

SILVA, M. M.; RAMOS, W. L. S.; FERREIRA, J. P. K.; CAMPOS, M. F. M.; NASCIMENTO, E. R. Towards Semantic Fast-Forward and Stabilized Egocentric Videos. In: EUROPEAN Conference on Computer Vision Workshops. [S.l.: s.n.], Oct. 2016. P. 557–571. ISBN 978-3-319-46604-0. DOI: 10.1007/978-3-319-46604-0_40. Cit. on pp. 11, 15, 21, 22, 25, 26, 36, 40, 57.

SPEARMAN, C. The Proof and Measurement of Association between Two Things. **The American Jour. of Psych.**, University of Illinois Press, v. 15, n. 1, p. 72–101, 1904. ISSN 00029556. DOI: 10.2307/1412159. Cit. on pp. 18, 38, 41, 46.

STEUER, R. E.; CHOO, E.-U. An interactive weighted Tchebycheff procedure for multiple objective programming. **Mathematical programming**, Springer, v. 26, p. 326–344, 1983. DOI: 10.1007/BF02591870. Cit. on p. 38.

SZABO, A.; BOUCHER, K.; CARROLL, W.; KLEBANOV, L.; TSODIKOV, A.; YAKOVLEV, A. Variable selection and pattern recognition with gene expression data generated by the microarray technology. **Mathematical Biosciences**, Elsevier BV, v. 176, n. 1, p. 71–98, Mar. 2002. ISSN 0025-5564. DOI: 10.1016/S0025-5564(01)00103-1. Cit. on p. 18.

TSODIKOV, A.; SZABO, A.; JONES, D. Adjustments and measures of differential expression for microarray data. **Bioinformatics**, Oxford University Press (OUP), v. 18, n. 2, p. 251–260, Feb. 2002. ISSN 1367-4803. DOI: 10.1093/bioinformatics/18.2.251. Cit. on pp. 13, 18, 25, 27.

WANG, M.; LIANG, J.-B.; ZHANG, S.-H.; LU, S.-P.; SHAMIR, A.; HU, S.-M. Hyper-Lapse From Multiple Spatially-Overlapping Videos. **IEEE Transactions on Image Processing**, v. 27, n. 4, p. 1735–1747, Apr. 2018. ISSN 1941-0042. DOI: 10.1109/TIP.2017.2749143. Cit. on p. 20.

YANG, Y.-H.; LIN, Y.-C.; SU, Y.-F.; CHEN, H. H. A Regression Approach to Music Emotion Recognition. **IEEE Transactions on Audio, Speech, and Language Processing**, v. 16, n. 2, p. 448–457, Feb. 2008. ISSN 1558-7924. DOI: 10.1109/TASL.2007.911513. Cit. on p. 17.

ZWICKER, E.; FASTL, H. **Psychoacoustics: Facts and models**. [S.l.]: Springer Science & Business Media, 2013. v. 22. DOI: 10.1007/978-3-540-68888-4. Cit. on pp. 17, 21.