

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology



أكاديمية كاوست
KAUST ACADEMY

Contextual Understanding of Video Data for Surveillance Analysis

Shooq Tareq Alhatem
shooq.hatem@kaust.edu.sa

Yara Yaseer Alzahrani
yara.zahrani@kaust.edu.sa

Fatimah Nasser Albesher
fatema.besher@kaust.edu.sa

Manar Eyad Ali
manar.ali@kaust.edu.sa

Mentor: Tanveer Hussain
hussaint@edgehill.ac.uk

August 27, 2025

Abstract

The exponential growth of surveillance systems has resulted in vast amounts of continuous video data, creating significant challenges for manual monitoring and analysis. Traditional approaches are labor-intensive, prone to human error, and often unable to deliver timely responses. This project proposes an intelligent framework for long video understanding that leverages motion-based temporal segmentation as the foundation for improved retrieval and analysis. Using dense optical flow (RAFT) with enhancements such as diagonal normalization and hysteresis thresholding, surveillance footage is segmented into semantically meaningful activity-centered clips. These segments are then paired with concise, action-focused captions generated through transformer-based models refined with prompt engineering, ensuring that retrieval queries align closely with observable activities.

Our evaluation on the UCA dataset shows that the proposed segmentation method significantly outperforms PySceneDetect, achieving gains of +0.18 F1 at t=0.1, +0.18 at t=0.3, and +0.19 at t=0.5. This demonstrates that more accurate segmentation leads to clearer captions, richer video understanding, and improved retrieval performance. By combining segmentation, captioning, and retrieval in a unified workflow, this work contributes toward advancing automated video analysis in security and monitoring applications.
The source code for this project is publicly available at `https://github.com/ManarEyad7/VidConSurv.git`.

Contents

Abstract **1**

1 Introduction **5**
1.1 Objectives and Goals 5
1.2 Members Contribution 6

2 Related Work **6**

3 Methodology **7**
3.1 System Overview 8
3.1.1 Stage 1: Motion-Based Temporal Segmentation 8
3.1.2 Stage 2: Clip-Level Caption Generation 8
3.1.3 Stage 3: Text-to-Video Retrieval (Future Work) 10
3.2 Data Acquisition and Description 10
3.3 Code Experiments 10
3.3.1 Scene Detection Using PySceneDetect 10
3.3.2 Motion-Based Segmentation Using Optical Flow 11
3.3.3 Motion-Based Segmentation Using FlowNet 2.0 12
3.3.4 Scene Detection Using RAFT 13

4 Results Discussion **14**
4.1 Stage 1: Motion-Based Temporal Segmentation 14
4.2 Stage 2: Clip-Level Caption Generation 16

5 Reflections on Challenges **17**

6 Conclusion and Future Work **18**

List of Figures

Figure 1 Workflow of the proposed video retrieval system. The pipeline is organized into three main stages. In Stage 1, long surveillance videos are segmented into activity-centered clips using motion-based temporal segmentation using a pretrained RAFT optical flow model, replacing traditional low-level feature–based methods such as PySceneDetect. Stage 2 applies a vision–language model (VLM) to generate captions for each segmented clip, followed by clip–caption indexing. In Stage 3, the indexed clips are retrieved using natural-language queries. 7

Figure 2 Motion energy curve computed from RAFT optical flow. The raw motion signal (blue) is smoothed (orange), and activity boundaries are detected using hysteresis thresholding (dashed lines). This process highlights peaks of significant motion while discarding prolonged static intervals. 8

Figure 3 Example of verbose, narrative caption including environment and emotions 9

Figure 4 Example of concise, action-focused caption highlighting only observable interactions. 9

Figure 5 Temporal Segmentation by Motion Energy: Smoothed Signal and Exported Action Intervals. 12

Figure 6 Before segmentation: Uniformly sampled frames from the unprocessed CCTV video (no motion-based or temporal filtering). 12

Figure 7 After segmentation: The video split into clips after applying energy-based motion segmentation. 12

Figure 8 Distribution of segment lengths. 13

Figure 9 Smoothed motion magnitude (top) and motion change (bottom) with detected segment boundaries. 13

Figure 10 Ours outperforms PySceneDetect with +0.18 F1 at t=0.1, +0.18 at t=0.3, and +0.19 at t=0.5. 15

Figure 11 Example of our temporal segmentation. A raw video frame, ground-truth segments, and predicted segments labeled as TP, FP, and FN are shown. The bottom plot displays motion magnitude, illustrating how F1@τ is evaluated. 16

List of Tables

1 **Team Member Contributions** 6
2 **Comparison of prior work and the proposed system.** 7
3 Motion-based segmentation resultsF1@τ (higher is better). 14
4 Caption generation results.Captions are ranked by EVQA score, and both mean and median rank are reported (lower is better). 17

1 Introduction

The widespread adoption of surveillance systems has resulted in the continuous generation of massive volumes of video footage, making manual review increasingly impractical. Traditional monitoring methods are time-consuming, costly, and prone to human error, often leading to delayed responses and overlooked incidents. This problem is especially critical in high-security environments, where timely detection of abnormal or suspicious activity is essential.

Recent advances in computer vision and machine learning have motivated the development of automated video analysis tools, yet many existing approaches still struggle to handle long-duration surveillance footage effectively. Conventional segmentation methods, such as PySceneDetect [6], primarily rely on low-level visual cues like color histograms and frame differences. While efficient, these methods are not well suited for capturing subtle or extended activities, limiting their effectiveness in real-world surveillance contexts.

To address these challenges, this project proposes an intelligent solution based on motion-based video segmentation. By leveraging optical flow information to detect activity boundaries, the system automatically analyzes and divides lengthy surveillance footage into semantically meaningful segments. These segments can then be paired with concise, action-focused captions to enable faster retrieval of relevant events, improve situational awareness, and reduce the cognitive load on human operators.

1.1 Objectives and Goals

The primary objective of this project is to design an intelligent system capable of automatically retrieving and summarizing relevant segments from lengthy surveillance video footage. A central premise of this work is that segmentation represents the most critical stage in long video understanding, as effective retrieval and analysis rely on the accurate division of continuous streams into semantically coherent activities.
The specific goals of the project are as follows:

- To use a motion-based temporal segmentation technique that integrates temporal features in order to identify and separate meaningful events within long surveillance videos.
- To enhance the efficiency of video analysis by enabling rapid access to critical video segments, thereby minimizing the need for exhaustive manual review.
- To improve the accuracy and reliability of downstream tasks such as captioning, summarization, and retrieval, which are directly dependent on the quality of segmentation.
- To reduce human workload and error in surveillance monitoring by automating the structuring of raw video streams into organized, activity-centered segments.

By emphasizing segmentation as the cornerstone of long video analysis, this project seeks to advance video retrieval methodologies and contribute to the development of more effi-

cient and intelligent surveillance systems.

1.2 Members Contribution

Table 1: **Team Member Contributions**

Task	**Contributors**
Literature Review	Shooq Alhatem, Yara Alzahrani
Data Preparation & Exploration	Shooq Alhatem, Manar Eyad
Code Experiments	All members
Methodology	Yara Alzahrani, Shooq Alhatem
Segmentation Stage & Evaluation	Manar Eyad, Shooq Alhatem
Captioning Stage & Evaluation	Fatimah Albesher
Report Writing	All members
Integration & Finalization	All members

Note: While specific responsibilities were distributed among members, the project was a collaborative effort. All members worked together, shared feedback, and supported one another throughout the development and reporting process.

2 Related Work

LoVR[1] proposes a benchmarked pipeline for long-form video retrieval that combines scene segmentation, clip-level captioning, and multimodal indexing for text-to-video search. In common instantiations, segmentation is performed with conventional scene-change detectors (e.g., PySceneDetect), captions are generated with transformer-based models, and retrieval is conducted via vision–language embeddings. While effective for generic video understanding, this design inherits the limitations of appearance-only segmentation and may dilute action-centric signals needed in surveillance scenarios.

For scene segmentation, PySceneDetect [6] detects scene boundaries using content-based heuristics such as frame-to-frame histogram deltas and thresholded content metrics. These methods are computationally lightweight and robust to hard cuts, but they operate primarily on low-level appearance cues, making them sensitive to illumination shifts and less responsive to subtle or prolonged motions. Critically, they do not explicitly model motion fields, which are central to activity delineation in surveillance footage.

In contrast, our system replaces appearance-only segmentation with *motion-based temporal segmentation* computed from dense optical flow (RAFT) [8]. We further stabilize the motion-energy signal via diagonal normalization and adopt hysteresis thresholding to promote temporally consistent boundary decisions. Downstream, we constrain captioning with prompt engineering (and optional few-shot exemplars) to yield concise, activity-focused descriptions aligned with retrieval. This results in segments that more directly correspond to *what happened*, improving the semantic match between queries and indexed clips.

Table 2: **Comparison of prior work and the proposed system.**

Aspect	LoVR + PySceneDetect	Proposed System (RAFT)
Segmentation signal	Appearance-based (histograms, frame deltas)	Motion-based (optical flow energy)
Temporal sensitivity	Good for hard cuts, weak for subtle motion	Captures gradual and sustained motion
Captioning focus	Broad, descriptive summaries	Concise, activity-focused captions
Retrieval alignment	May misalign if not action-based	Strong alignment with activity semantics
Compute cost	Lightweight (CPU)	Higher (GPU), but more accurate

3 Methodology

The objective of the proposed pipeline is to advance the segmentation stage in long video understanding, with the aim of transforming extended-duration surveillance footage into semantically coherent, activity-centered clips. Segmentation is positioned as the foundation of the workflow, where a motion-based temporal approach is employed to capture meaningful activities that form the basis for subsequent retrieval through natural-language queries. To ensure both computational efficiency and semantic fidelity, the pipeline is organized as a three-stage cascade, as illustrated in Figure 1.

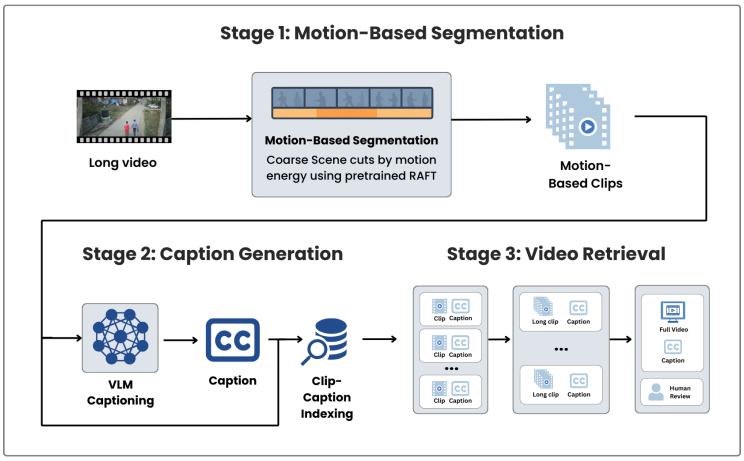


Figure 1: Workflow of the proposed video retrieval system. The pipeline is organized into three main stages. In Stage 1, long surveillance videos are segmented into activity-centered clips using motion-based temporal segmentation using a pretrained RAFT optical flow model, replacing traditional low-level feature–based methods such as PySceneDetect. Stage 2 applies a vision–language model (VLM) to generate captions for each segmented clip, followed by clip–caption indexing. In Stage 3, the indexed clips are retrieved using natural-language queries.

3.1 System Overview

3.1.1 Stage 1: Motion-Based Temporal Segmentation

Let $V = \{f_1, \ldots, f_T\}$ denote the untrimmed video stream consisting of T frames. To model motion information, we adopt the RAFT (Recurrent All-Pairs Field Transforms) optical flow framework[8], which provides dense motion estimates u_t between consecutive frames. The flow magnitudes $||u_t||^2$ are aggregated into a motion-energy curve that encodes temporal variations in activity throughout the video as illustrated in Figure 2.

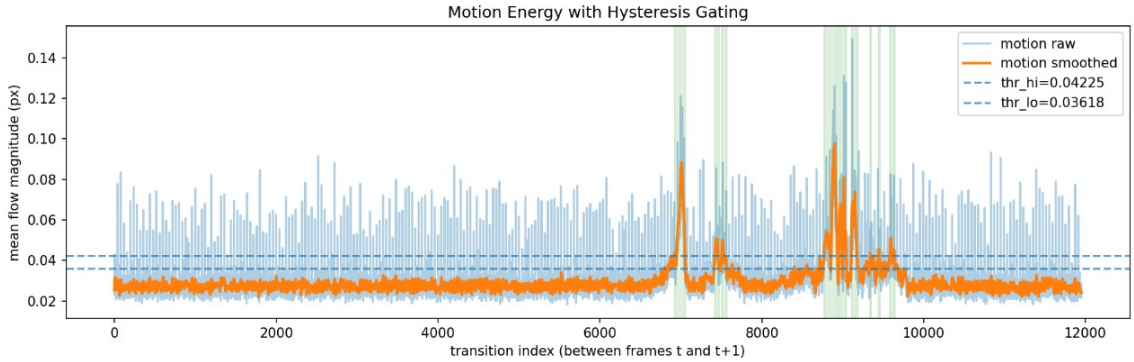


Figure 2: Motion energy curve computed from RAFT optical flow. The raw motion signal (blue) is smoothed (orange), and activity boundaries are detected using hysteresis thresholding (dashed lines). This process highlights peaks of significant motion while discarding prolonged static intervals.

To enhance robustness, the raw motion signal is processed in two ways. First, *diagonal normalization* is applied to reduce directional bias and stabilize the energy profile across varying motion patterns. Second, the signal is smoothed using a Savitzky–Golay filter to suppress local noise while preserving significant peaks. Finally, scene boundaries are identified using *hysteresis thresholding*, which employs a pair of adaptive thresholds to ensure that only persistent, high-confidence peaks exceeding the mean energy by $k\sigma$ are retained.
Each detected boundary yields a coarse scene clip $S(i)$. This segmentation procedure acts as a computationally efficient pre-filter, discarding prolonged static intervals while preserving activity-centered continuity, thereby providing a stronger foundation for subsequent caption generation and retrieval.

3.1.2 Stage 2: Clip-Level Caption Generation

Each action clip $A(i, j)$ is summarized by a concise textual description. Following standard practice, N uniformly spaced keyframes and a short temporal snippet are sampled and fed to a transformer-based video captioning model such as VideoBLIP-2 or ClipCap-ViT-G/14. The model generates a sentence of approximately 12–20 tokens that captures the actor, action, and scene context.

In the original LoVR implementation, the captioning pipeline was designed to generate broad, descriptive summaries of video content, often incorporating environmental aspects,

emotions, and stylistic elements as illustrated in Figure 3. While this approach proved effective for general video understanding, it was not well-suited for surveillance contexts, where the emphasis lies on observable actions and interactions rather than aesthetic or narrative features.

To address this limitation, we refined the pipeline through targeted *prompt engineering*. Instead of directing the model to produce lengthy narrative descriptions, the system prompt was revised to enforce succinct, action-focused captions that highlight *who performed what action, on whom, and where*. Environmental details were minimized unless directly relevant to the activity, and ambiguous identities were standardized into general terms such as *"person," "group,"* or *"vehicle"*.

Furthermore, the pipeline was extended to optionally incorporate ground truth annotations as few-shot examples. By randomly sampling annotated action segments and embedding them into the prompt, the model was guided to align more closely with the intended annotation style without requiring additional retraining. This hybrid strategy merging prompt engineering with example-guided in-context learning improves the relevance, consistency, and surveillance focus of the generated captions, while preserving the modular design of the original LoVR framework.
For downstream retrieval, both the generated caption and its CLIP embedding $e_{i,j} \in \mathbb{R}^d$ are stored together with a pointer to the corresponding video segment.


```
{"vid": "newV", "slice_num": 27, "path": "seg_0026_012191_012218.mp4",
"cap": "The video depicts a chaotic scene inside what appears to be a store or a similar commercial
establishment. The setting is filled with various objects and equipment, suggesting a busy environment.
In the foreground, there is a group of people engaged in a physical altercation. One individual,
wearing a black hoodie, is seen pushing another person who is being held by two others.
The person being pushed appears to be struggling against the force applied by the two individuals
holding them.
In the background, there are several other people observing the altercation.
Some of these individuals are standing close to the main group, while others are further back,
possibly trying to intervene or simply watching the situation unfold.
 The expressions on their faces suggest concern or shock at the ongoing conflict.
 The overall atmosphere of the video is tense and violent, with the physical altercation being
 the central focus. The presence of multiple witnesses indicates that this incident has caught the
 attention of others in the vicinity. The setting, combined with the intensity of the scene,
 creates a sense of urgency and immediacy.\n\nDespite the chaos, the video does not provide any
  additional context about the cause of the altercation or the identities of the individuals involved.
```


Figure 3: Example of verbose, narrative caption including environment and emotions


```
{"vid": "newV", "slice_num": 27, "path": "seg_0026_012191_012218.mp4",
"cap": "Action-only caption: People engage in a physical altercation inside a store.\n
Events:\n
- Person 1 pushes person 2\n
- Person 2 retaliates with a punch\n
- Person 1 grabs person 2's arm\n
- Person 2 tries to escape\n
- Person 1 follows and continues to push"}
```


Figure 4: Example of concise, action-focused caption highlighting only observable interactions.

3.1.3 Stage 3: Text-to-Video Retrieval (Future Work)

In the retrieval phase—reserved for future investigation—user queries q are embedded into the same multimodal space via a text encoder. Approximate nearest-neighbour search (e.g., FAISS or ScaNN) is performed over the embedding set $\{e_{i,j}\}$. Retrieved candidates may be re-ranked with a cross-modal similarity network to improve precision. The system ultimately returns an ordered list of clips ranked by semantic relevance to the query.

3.2 Data Acquisition and Description

This project employed two publicly available surveillance datasets to evaluate the performance of the proposed motion-based segmentation method (improved RAFT) and the quality of the generated captions.

- **UCA Dataset (UCF-Crime Annotation):** The UCA [9] dataset consists of 1,854 real-world surveillance videos with a total duration of 121.9 hours, of which 110.7 hours are annotated. It contains 23,542 sentence-level descriptions aligned with start and end times at a precision of 0.1 seconds. The videos were curated to ensure quality, and expert annotators meticulously labeled observable events—both normal and abnormal—using precise language. These annotations provide rich spatial and temporal context and have been validated by domain experts. The dataset is widely applicable to video captioning, temporal localization, and anomaly detection tasks, making it suitable for assessing the segmentation and captioning components of this project.

- **SurveillanceVQA-589K:** SurveillanceVQA-589K [4] is a large-scale benchmark dataset developed to advance open-ended video question answering (VQA) within surveillance domains. It comprises 589,380 question–answer pairs derived from 3,030 surveillance videos, totaling approximately 159 hours of footage collected from four established datasets: UCA, MSAD, MEVA, and NWPU. The dataset encompasses a diverse set of clips, including 27,962 normal and 3,586 abnormal segments, recorded across a variety of real-world environments such as public spaces, transportation hubs, and street intersections. Each clip is accompanied by multiple QA pairs targeting higher-level cognitive tasks, including temporal reasoning, spatial understanding, causal inference, and anomaly detection. For this project, SurveillanceVQA-589K was utilized to further evaluate the captioning pipeline, as its broad coverage and task diversity provide a challenging benchmark for semantic comprehension and contextual reasoning in complex surveillance settings.

3.3 Code Experiments

3.3.1 Scene Detection Using PySceneDetect

As part of the first stage in the proposed methodology, an experiment was conducted on video segmentation adapted from the LoVR paper [1]. This stage focuses on breaking down long surveillance videos into semantically meaningful clips.
To implement this, the open-source Python library `PySceneDetect` [6] was utilized. This tool detects scene transitions based on visual changes by computing histogram differences between adjacent frames in the HSV color space. If the computed content score exceeds a

specified threshold, the corresponding frame is marked as a scene boundary. For instance, in our code we used:

```
scene_manager.add_detector(ContentDetector(threshold=8.0))
```

A relatively low threshold (8.0) was selected to increase the detector's sensitivity to subtle changes, which are common in surveillance footage. A minimum scene length was also defined to prevent the detection of excessively short segments due to noise. Once scenes were detected, the video was split using:

```
split_video_ffmpeg([video_path], scene_list, output_dir)
```

Despite its ease of use, this approach proved unsuitable for our use case. Surveillance videos often include long durations of minimal motion or slow transitions, which do not generate enough visual difference to trigger content-based scene detection. As a result, this method underperformed in low-activity environments, making it unreliable for segmenting surveillance videos into meaningful clips.

3.3.2 Motion-Based Segmentation Using Optical Flow

As part of our exploratory phase, we experimented with an open-source Python implementation for dense optical flow tracking[5] . This code uses the Farnebäck method to compute optical flow between consecutive video frames, capturing both the direction and magnitude of motion at the pixel level.

The algorithm visualizes the flow using the HSV color space, where motion direction is encoded in hue and motion intensity is represented by brightness. The resulting output is a color-coded video that reveals movement patterns over time, aiding interpretation of activity in surveillance footage, even in low-motion or low-contrast conditions.

Although the original purpose of this code is primarily visual analysis, we recognized its potential to support the design of **Stage 1: Motion-Based Temporal Segmentation** in our pipeline. By extending the implementation to record average motion per frame and construct a motion-energy curve, we can apply smoothing and statistical thresholding to identify meaningful scene boundaries.

It is important to note that this remains an initial experiment and the code has not yet been formally integrated into our system. However, it provided valuable insight into how motion evolves over time in long surveillance videos and serves as a foundation for developing lightweight, motion-based segmentation methods in future stages of the project.

Building on this exploratory implementation, we added a lightweight temporal segmentation module that converts motion cues into action-centered clips. For each frame pair, per-pixel flow is reduced to a scalar motion-energy signal (mean squared magnitude), which is then smoothed with a Savitzky–Golay filter to suppress jitter while preserving peaks. An adaptive percentile threshold flags genuinely active periods, binary morphological closing bridges brief pauses so single actions are not over-fragmented, and a minimum-duration constraint removes spurious micro-bursts. Each contiguous high-motion interval is exported as a separate video clip, and a diagnostic plot is generated that shows raw and

smoothed motion energy, the threshold line, and shaded spans for detected actions. In practice, this suppressed long static intervals, adapted to both quiet and busy scenes, and yielded compact, review-ready segments suitable for downstream object segmentation and captioning.

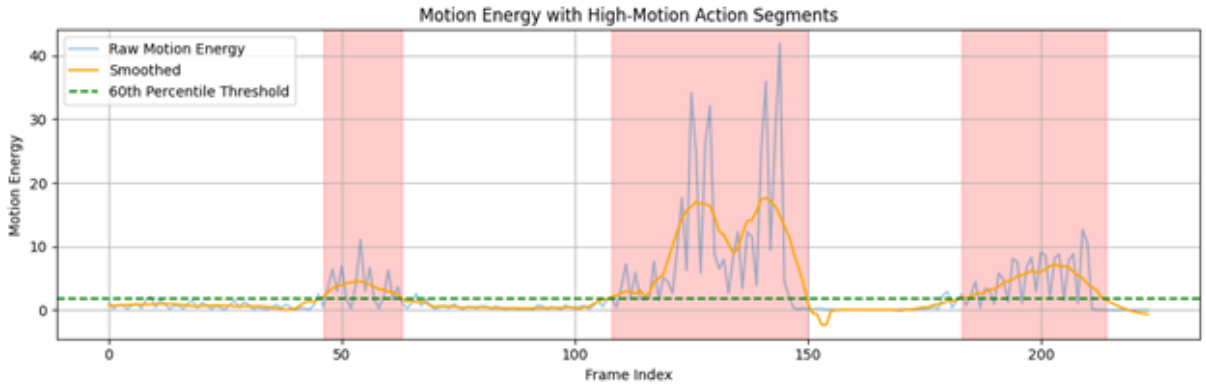


Figure 5: Temporal Segmentation by Motion Energy: Smoothed Signal and Exported Action Intervals.



Figure 6: Before segmentation: Uniformly sampled frames from the unprocessed CCTV video (no motion-based or temporal filtering).



Figure 7: After segmentation: The video split into clips after applying energy-based motion segmentation.

3.3.3 Motion-Based Segmentation Using FlowNet 2.0

Another experiment was conducted as part of the first stage in the proposed pipeline. FlowNet 2.0 [7] was employed to estimate dense motion fields between consecutive video frames. For implementation, the open-source PyTorch version of FlowNet 2.0 was used, enabling direct integration into the segmentation framework. The network takes two consecutive frames as input and outputs a dense optical flow map, where each pixel's vector encodes both motion direction and magnitude. These flow maps were subsequently processed to compute motion intensity scores, which served as cues for detecting temporal boundaries. Initially, a fixed threshold was applied for clipping, with three different threshold values tested. As shown in Figure 8, the fixed threshold produced clip length distributions concentrated around 30 frames. Then a dynamic method was adopted to improve the results. The dynamic approach identified boundaries using the nearest

local minimum of the smoothed motion curve , as shown in Figure 9, resulting in finer-grained clips that capture multiple motion events within the same scene.It implies that the video contains mostly static content except for one significant action event. FlowNet 2.0 demonstrated strong robustness in handling small motions and complex real-world scenes. This method will be compared against the LoVR dataset for evaluation.

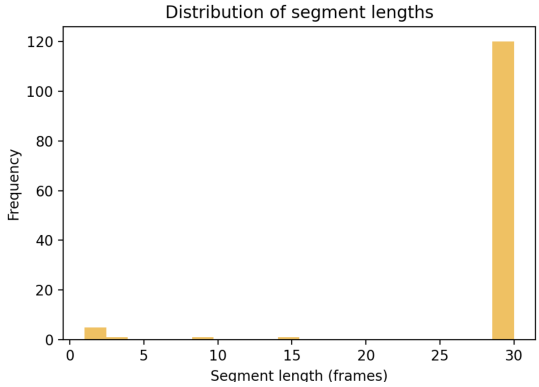


Figure 8: Distribution of segment lengths.

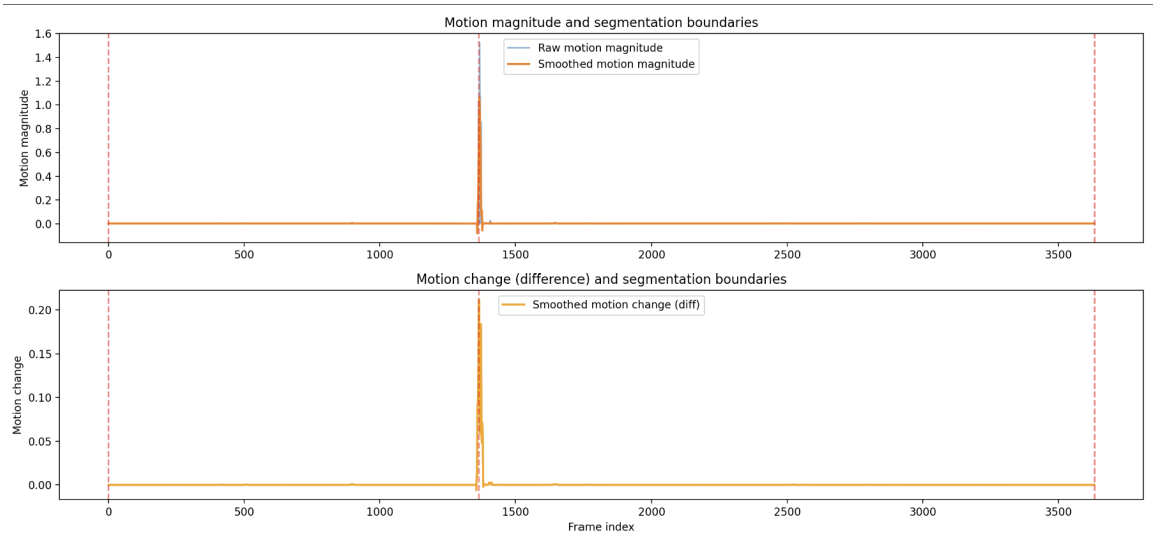


Figure 9: Smoothed motion magnitude (top) and motion change (bottom) with detected segment boundaries.

3.3.4 Scene Detection Using RAFT

As part of our exploration phase, we experimented with the Recurrent All-Pairs Field Transforms (RAFT) [8] model in our video segmentation pipeline. This model focuses on estimating optical flow, which is essential for detecting scene transitions in surveillance footage. RAFT generates 4D correlation volumes for all pairs of pixels between consecutive frames and uses a recurrent update process to iteratively refine the flow field. The model excels in handling large displacements and occlusions, providing accurate flow predictions, even for small and fast-moving objects. It utilizes feature extraction from input frames, followed by correlation computation and iterative updates to predict the optical

flow field efficiently.

We tested RAFT in our motion-based segmentation approach. The flow output from RAFT was used to generate a motion energy curve, which is crucial for detecting scene boundaries. We applied a Savitzky-Golay filter to smooth the signal and used statistical thresholding techniques to identify significant motion changes, marking potential scene boundaries. While RAFT performed exceptionally well, its application in real-world surveillance footage showed promising results. The model demonstrated its potential to improve video segmentation based on motion, providing a more accurate and effective approach for detecting scene transitions, especially in videos with intermittent or slow movement in the future.

4 Results Discussion

4.1 Stage 1: Motion-Based Temporal Segmentation

This stage was evaluated using the F1-score introduced in [2], denoted F1@τ. Given a predicted segment $[p_s, p_e]$ and a ground-truth segment $[g_s, g_e]$, we first compute the temporal Intersection-over-Union (tIoU):

$$\text{tIoU} = \frac{\max\big(0,\ \min(p_e, g_e) - \max(p_s, g_s)\big)}{\max(p_e, g_e) - \min(p_s, g_s)}. \tag{1}$$

A prediction is considered a true positive if its tIoU with a ground-truth segment exceeds a given threshold $\tau/100$, where we use $\tau \in \{0.1, 0.3, 0.5\}$. When multiple predictions overlap the same ground-truth action above the threshold, only one is counted as a true positive and the remaining matches are counted as false positives. The final score is the F1 harmonic mean of precision and recall:

$$\text{F1} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{2}$$

Table 3: Motion-based segmentation resultsF1@τ (higher is better).

Model	**@0.5**	**@0.3**	**@0.1**
Ours	**0.2053**	0.3072	0.4080
LoVR (PySceneDetect) [1]	0.0180	0.1238	0.2260

In Table 3, our RAFT-based segmentation clearly outperforms the benchmark based on the PySceneDetect segmentation method used in LoVR [1] across all thresholds. These results indicate that motion-energy segmentation derived from RAFT flow magnitudes is more effective at capturing temporal variations in surveillance videos. Nevertheless, both methods exhibit a notable drop in performance at the strictest threshold ($\tau = 0.5$), highlighting that precise boundary localization remains a challenge in this domain, as also

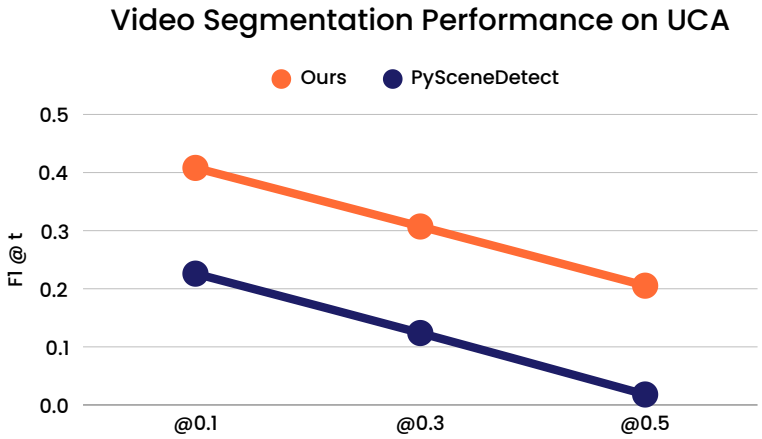


Figure 10: Ours outperforms PySceneDetect with +0.18 F1 at t=0.1, +0.18 at t=0.3, and +0.19 at t=0.5.

discussed in [2], where achieving fine-grained boundary accuracy is recognized as a central difficulty for modern architectures.

As shown in Figure. 11. It clarifies how predictions are aligned or misaligned with ground truth and how the F1@τ metric reflects the segmentation accuracy. It shows a raw video frame before segmentation, the ground-truth segments, and the predicted segments annotated with TP, FP, and FN. At the bottom, the motion magnitude curve derived from RAFT optical flow magnitudes [8] highlights temporal variations that drive the segmentation.

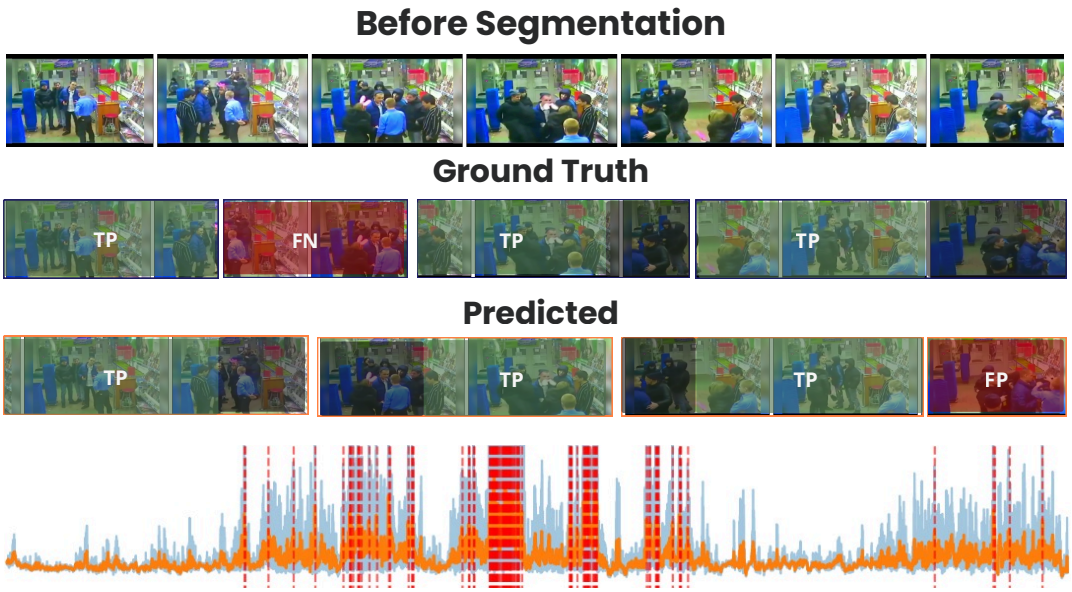


Figure 11: Example of our temporal segmentation. A raw video frame, ground-truth segments, and predicted segments labeled as TP, FP, and FN are shown. The bottom plot displays motion magnitude, illustrating how F1@τ is evaluated.

4.2 Stage 2: Clip-Level Caption Generation

The quality of the generated captions was evaluated using the EVQA (Evaluation via Visual Question Answering) framework[3] . In this approach, the pretrained CLIP (Contrastive Language–Image Pretraining) model was applied to measure the semantic alignment between video content and its corresponding textual description. For each segment, a fixed number of frames (8 frames per clip) was uniformly sampled across the duration of the video. Both the sampled frames and the candidate caption were encoded by CLIP. The resulting embeddings were compared through cosine similarity, and an aggregate score was calculated for every clip and caption pair.

$$\text{EVQAScore}(V, X) = \frac{\text{EVQAScore}(V, X)_c + \text{EVQAScore}(V, X)_f}{2}, \tag{3}$$

where $\text{EVQAScore}(V, X)_c$ measures the overall (coarse) video–caption similarity, and $\text{EVQAScore}(V, X)_f$ captures detailed object–keyword alignment. A higher EVQAScore indicates that the caption is more semantically consistent with the video content [3].
In our implementation, we adopt the maximum similarity (agg = max) across frames was adopted to capture the strongest alignment between the caption and any visual evidence present in the clip. This metric measures how well the generated captions align with the actual video content. After the first stage, For each segment, we compute an EVQA score using a vision–language model (CLIP) that quantifies the semantic similarity between the

clip and its caption. A higher score means the caption is more relevant to what happens in the video.
To evaluate the captions generated from different segmentation methods, captions from all segments were merged and ranked according to their EVQA scores, from highest to lowest. We then computed both the mean and median rank for each method. The mean rank provides an overall average position across all segments, while the median rank offers a robust measure that is less sensitive to outliers. Lower ranks correspond to better performance, as they indicate captions consistently appear higher in the EVQA ranking.

Table 4: Caption generation results.Captions are ranked by EVQA score, and both mean and median rank are reported (lower is better).

Method	Mean Rank	Median Rank
Ours	**29.5**	**29**
LoVR [1]	36	37.5

As shown in Table 4, our approach outperforms the baseline, achieving a mean rank of 29.5 and a median rank of 29, compared to 36 and 37.5 for LoVR. These results indicate that captions generated by our segmentation method are more ranked at the top according to EVQA, demonstrating stronger semantic alignment with the video content. The improvement in both mean and median rank highlight that the combination of motion-based segmentation then the clip-level caption generation produces higher-quality outputs.

5 Reflections on Challenges

One of the major challenges faced during this project was the lack of large, annotated datasets for long-duration surveillance videos. Only the UCA dataset was available with sufficient temporal and textual annotations, which limited the diversity of evaluation scenarios. This constraint highlighted a broader issue in the research community, where surveillance-specific benchmarks remain scarce compared to general video datasets.

On the technical side, a significant portion of our effort was spent on experimenting with multiple segmentation approaches to determine the most effective strategy. While traditional tools like PySceneDetect were fast, they failed to capture subtle motion patterns, requiring us to test and refine motion-based segmentation using RAFT with additional normalization and thresholding techniques. These experiments were time-consuming but crucial in achieving better results.

From a process perspective, the entire project was carried out within a single month, which placed strong constraints on planning, coordination, and iteration. Efficient task allocation within the team and continuous integration of feedback were key in overcoming these time limitations.

An unexpected finding was that small design choices—such as prompt engineering for captioning or the use of hysteresis thresholds for segmentation—had a disproportionately large impact on the system's overall effectiveness. These discoveries not only improved

our pipeline but also provided insights into the importance of fine-grained design decisions in video understanding systems.

6 Conclusion and Future Work

This project introduced an intelligent framework for surveillance video analysis, highlighting the importance of motion-based temporal segmentation in improving long video understanding. By replacing appearance-driven scene detection with RAFT-based optical flow segmentation, enhanced by diagonal normalization and hysteresis thresholding, we showed that precise segmentation yields semantically coherent activity-centered clips. These clips generate clearer and more relevant captions, which not only improve video comprehension but also strengthen retrieval performance by aligning queries with action-focused descriptions. Our results confirm that segmentation quality is a key factor cascading into improved captioning and retrieval outcomes.

For future work, we aim to further enhance the captioning and retrieval pipelines. On the captioning side, we plan to refine prompt engineering techniques and explore multimodal large language models to produce more accurate, activity-centered descriptions. On the retrieval side, we will investigate improved embedding alignment and cross-modal reasoning methods to better connect natural-language queries with video segments. These advancements are expected to improve the efficiency, accuracy, and scalability of automated surveillance video analysis systems.

References

[1] Q. Cai et al. *LoVR: A Benchmark for Long Video Retrieval in Multimodal Contexts*. arXiv preprint arXiv:2505.13928. [Online]. Available: http://arxiv.org/abs/2505.13928. May 2025.

[2] Guodong Ding, Fadime Sener, and Angela Yao. "Temporal action segmentation: An analysis of modern techniques". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[3] Hao Liang, Zirong Chen, and Wentao Zhang. "EVQAScore: Efficient Video Question Answering Data Evaluation". In: (Nov. 2024). DOI: 10.48550/arXiv.2411.06908.

[4] B. Liu et al. *SurveillanceVQA-589K: A Benchmark for Comprehensive Surveillance Video-Language Understanding with Large Models*. arXiv preprint arXiv:2505.12589. [Online]. Available: http://arxiv.org/abs/2505.12589. May 2025.

[5] Michael Polinowski. *Optical Flow Tracking with OpenCV*. https://mpolinowski.github.io/docs/IoT-and-Machine-Learning/ML/2021-12-10--opencv-optical-flow-tracking/2021-12-10/. Accessed: 2025-08-27. 2021.

[6] PySceneDetect Documentation. *Scene Detection Algorithms in PySceneDetect*. https://www.scenedetect.com/api/. 2025.

[7] Fitsum Reda et al. *flownet2-pytorch: Pytorch implementation of FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks*. https://github.com/NVIDIA/flownet2-pytorch. 2017.

[8] Zachary Teed and Jia Deng. *RAFT: Recurrent All-Pairs Field Transforms for Optical Flow*. 2020. arXiv: 2003.12039 [cs.CV]. URL: https://arxiv.org/abs/2003.12039,%20https://github.com/princeton-vl/RAFT.

[9] Tongtong Yuan et al. *Towards Surveillance Video-and-Language Understanding: New Dataset, Baselines, and Challenges*. 2023. arXiv: 2309.13925 [cs.CV]. URL: https://arxiv.org/abs/2309.13925.