A Multi-Method Approach to Scene Detection in Lecture Videos

Sunwoo Baek, Supia Park, Ashley Li, Enya Chen, Charitha Nannapaneni

Siebel School of Computing and Data Science, University of Illinois Urbana Champaign

CS STARS
STUDENT AMBASSADORS/RESEARCH SCHOLARS

INTRODUCTION

Lecture videos have a wide variety of forms with a varying mixture of static slides and dynamic annotations. This makes it challenging to extract keyframes that best represent the content. In our research, we are addressing the specific problem *given a lecture video, how do we extract its key frames to create a concise PDF equivalent*? Traditional methods like pixel similarity and OCR-based text extraction are not effective for distinguishing scene changes and gradual, non-discrete changes such as handwritten notes, highlights, and incremental annotations.

To address this, we explore a series of methods to complete the stated task. We utilize methods such as optical flow, masking, and subtraction in order to efficiently and accurately select representative frames from a lecture video.

OUR APPROACH

In our approach, a lecture video will go through a series of methods, refining the work from Angrave, Li, and Zhong, 2022, that is currently implemented in ClassTranscribe**.**

First, we receive an input video segmented into individual frames to match the required input format for ClassTranscribe. The ClassTranscribe methods are then applied to generate initial scene change predictions. Afterwards, we refine the results by addressing incorrect predictions caused by scrolling and annotations. Scrolling is detected and removed from the set of possible scene changes using optical flow to track content movement across frames. To handle annotations, we apply thresholding to generate a mask and then subtraction to quantify the information unique to each slide. Once this is performed across the video, frames with outlier values are marked as scene changes. Finally, a frame may be selected from each video segment to form series of representative frames concisely reflecting the key material. This process, illustrated in *Figure 3,* helps reduce the oversensitivity to scrolling and annotations found in previous work.

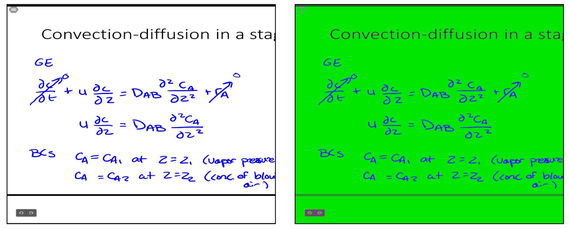
HANDLING ANNOTATIONS

Broadly, we handle annotations by first masking each frame in the lecture video and secondly "subtracting" each adjacent pair of frames. Ideally, the pixels remaining following subtraction are representative of the information unique to the frame and may be counted and used to help identify scene changes.

Masking

For our masking technique, we used thresholding to mask the segmented frames.

Selected frames undergo Gaussian blur filtering to reduce image noise. RGB values are discretized into five bins, reducing each frame to a palette of five distinct colors. A binary thresholding process is then applied to convert the image into a strictly black-and-white format (not grayscale). In this step, a fixed threshold value is used to determine whether each pixel is classified as black or white based on its brightness.



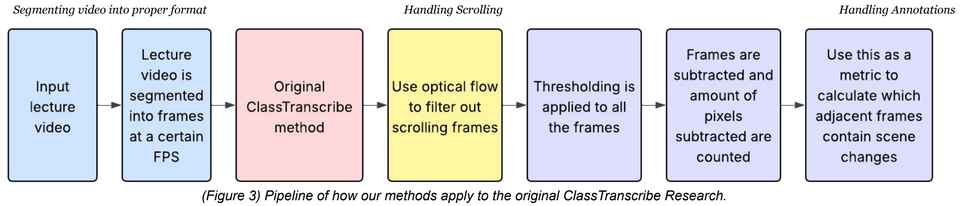
(Figure 1) Original screenshot from a lecture video vs. (Figure 2) masked version.

Subtraction

For our subtraction technique, we implemented an operation akin to exclusive-or: if the pixels at (x,y) are the same between two frames, the difference will have no pixel. Otherwise, the minuend's value is copied over to the difference. This may be summarized as the following formula:

$$diff(x,y) = \begin{cases} frame1(x,y) & \text{if } frame1(x,y) \\ & \&\& \ (frame1(x,y) \oplus frame2(x,y)) \\ 0 & \text{otherwise} \end{cases}$$

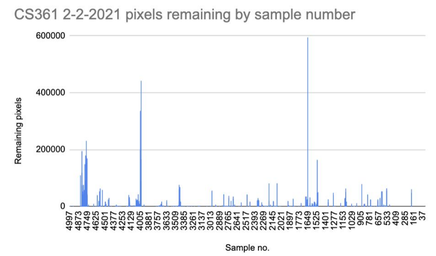
At the end, only frames following a scene change should have pixels remaining. This removes the false-positive mid-annotation frames for cleaner scene detection.



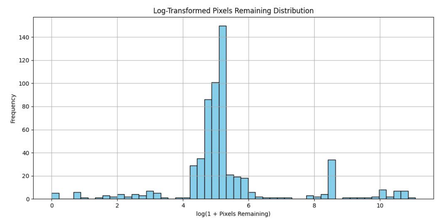
(Figure 3) Pipeline of how our methods apply to the original ClassTranscribe Research.

ANNOTATION RESULTS

After subtraction, the pixels remaining between 2 adjacent frames accurately shows where a scene change is located. We then apply min-max normalization and select frames above a certain threshold.



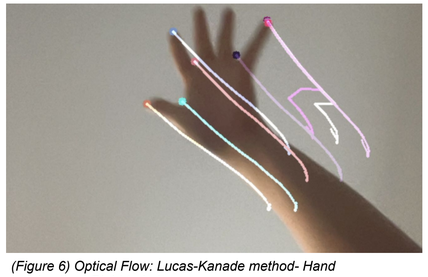
(Figure 4) Pixels remaining in subtracted adjacent frames in a CS361 lecture video.



(Figure 5) Data under logarithmic transformation to reduce skew and variance.

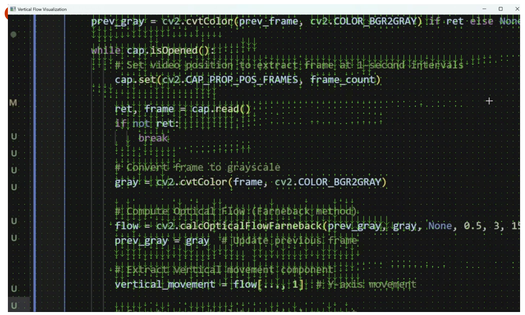
HANDLING SCROLLING

By analyzing motion vectors between consecutive frames, optical flow allows us to track scrolling movements rather than relying on absolute pixel differences. Our experiments with Lucas-Kanade and Dense Optical Flow show that this method effectively detects the gradual evolution of content. However, challenges remain in filtering out background movement artifacts and distinguishing intentional content changes from noise. This approach provides a complementary perspective to static frame comparisons, offering a motion-based strategy for scene detection.

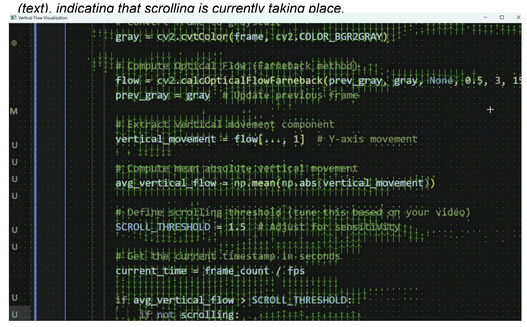


(Figure 6) Optical Flow: Lucas-Kanade method- Hand moving from Right to Left

SCROLLING RESULTS



(Figure 6 and 7) Dense Optical Flow method--motion vectors placed on video subject (text), indicating that scrolling is currently taking place.



FUTURE WORK

We plan to enhance all key components of our system: scroll detection, masking and subtraction, and mean shift. Once these improvements are in place, our next step is to integrate these methods to develop a robust scene detection pipeline *(Figure 3)*. Ultimately, we aim to apply this system to **ClassTranscribe**, our university's lecture video platform, to enable more accurate and efficient segmentation of lecture content.

REFERENCES

[1] OpenCV. Optical Flow. OpenCV Documentation. https://docs.opencv.org/4.x/d4/dee/tutorial_optical_flow.html

[2] Angrave, L., Li, J., and Zhong, N. 2022. Creating TikToks, Memes, Accessible Content, and Books from Engineering Videos? First Solve the Scene Detection Problem. In Proceedings of the 2022 ASEE Annual Conference & Exposition, Minneapolis, MN.