BLG202E Numerical Methods in Comp. Eng.

Spring 2025 - Term Project

Due: May 20, 2025

By turning in this assignment, I agree by the ITU honor code and declare that all of this is my own work.

Important Notes

- Upload your solutions through **Ninova**. Homeworks sent via e-mail and late submissions **will not be accepted**.
- Please make sure that you write your **full name** and student identification **number** to **every file** you submit.
- Cheating is highly discouraged. It will be punished by a negative grade. Also disciplinary actions will be taken. Please do your homework on your own. Team work is not allowed. Pattern of your **solutions must belong only to you**.
- All codes and reports will be run through **plagiarism check**s. Please **do not copy any text or code** from other sources.
- If you have any questions, please contact with **İlknur Çelik** (celikil17@itu.edu.tr).
- Remember, there are only 10 types of people in the world – those who understand binary, and those who don't.

Project: Scene Change Detection Using Singular Value Decomposition (SVD)

Please read the following prerequisites.

- Install a Conda environment if you do not have it already.
- Install Jupyter Notebook (Optional).
- The project should be done in a Python script. You may use Jupyter Notebooks for data inspection and visual generation.
- Do not forget to format your code and leave comments for non-trivial sections.
- You are may use matplotlib, numpy / SciPy, json and pandas libraries.

1 Problem Description

In this project, you will implement a video shot boundary detection system based on Singular Value Decomposition (SVD) and pattern matching, following the method described in the paper ”Fast Video Shot Boundary Detection Based on SVD and Pattern Matching” [1].

The goal of this project is to analyze a given video, extract frames, compute the Singular Value Decomposition (SVD) for each frame, and identify keyframes where significant changes occur (i.e., scene changes). You will implement the numerical methods manually without using built-in library functions for SVD computation.

2 Instructions

You will read and analyze the paper by Lu and Shi[1] and re-implement the method using your own Singular Value Decomposition (SVD) function.

- You must write your own implementation of **SVD from scratch** (without using numpy.linalg.svd or other pre-built functions).
- You can use the OpenCV (cv2) library only for reading the video file and extracting frames.
- Other image processing functions (e.g., feature extraction) must be implemented manually.
- The goal is to detect shot boundaries in a video using SVD-based similarity analysis.

Report:

- Write a maximum of 3 pages report using IEEE Latex Template. State the problem, implementation details, available data and the experiments.

  Formatting Tips:

  - Use section headings (e.g., Introduction, Methods, Results, Discussion).
  - Label plots and tables clearly; caption them to highlight key findings.
  - Your report must adhere to the IEEE conference paper template in LaTeX, available at:
    `https://www.ieee.org/conferences/publishing/templates.html`
  - Ensure proper citation formatting using IEEE citation style.
  - Maintain consistency in notation and terminology throughout your document.
- Add visual and numeric results to your report.

Submission

- Submit a zip file that includes your Python code, figures and results until the deadline through Ninova.
- Upload your project report until the deadline through Ninova.

3 Inputs

- Video 1: You can download the first example video from here
- Video 2: You can download the second example video from here

You can work on these two videos. Grading will be done based on Video 1.

4 Outputs

- Print the detected shot transition frames. Save a text file listing the frame numbers where transitions occur. For example:

```
Detected Shot Transitions:
Frame 120
Frame 275
Frame 430
```

- Save the frames where the transitions occur as images (you can use JPG or PNG). The images can be either in color or grayscale.

- If the images are taken from the points where scene transitions occur, there should be no image from the last frame of the video. However, if images are generated for each distinct scene, the number of images should match the number of scenes in the video. Both implementation approaches are acceptable. Please specify which approach you have implemented in your report.
- You can examine example output images from here. Please note that the images here do not represent all possible outcomes. They are provided as examples only.

5 Implementation Details

Analyze Algorithm

Identify the main steps of the algorithm, focusing on:

- How frame differences are used to find candidate shot transitions.
- How SVD is applied to measure frame similarity.
- How pattern matching is used to refine the results.

Video Reading

- Use OpenCV (cv2.VideoCapture) to read a video file and extract individual frames.
- Convert frames to grayscale.
- Store the extracted frames in a list for further processing.
- You can review the OpenCV library.

Implement SVD

- Implement SVD from scratch (without using numpy.linalg.svd).
- Given a matrix A, your function should compute:

$$A = U\Sigma V^T \tag{1}$$

  - U and V are orthonormal matrices.
  - Σ is a diagonal matrix of singular values.
- Use Jacobi or QR decomposition to compute SVD.
- Extract the top singular values (largest ones) as feature representations for each frame.

Compute Frame Differences and Find Candidate Transitions

Compute Total Absolute Difference (TAD) between consecutive frames:

$$TAD(F_i, F_{i+1}) = \sum \mid F_i - F_{i+1} \mid \tag{2}$$

If $TAD(F_i, F_{i+1})$ exceeds a certain threshold, mark frame F_i as a candidate shot boundary.
Hint: The threshold can be determined experimentally.

Apply SVD

- Apply your SVD function to candidate frames.
- Extract the dominant singular values.
- Compute Euclidean distance between singular value vectors of consecutive frames
- If difference is above a predefined threshold, mark it as a confirmed shot boundary.

6 Assessment Criteria

The term project will be graded as follows:

- **80%: Implementation quality**
  - Implementation Correctness, Comparative Analysis, Visualization Clarity
- **20%: Report**
  - Organization, clarity, and adherence to guidelines.
  - Conciseness and completeness of the final write-up.

7 Notes

- You may use numpy for matrix operations (e.g. matrix multiplication, determinant).
- Please do not copy any code from your friends or the internet. Any kind of cheating will be graded negatively.
- You need to implement SVD from scratch. You can't use built-in libraries or methods like 'numpy.linalg.svd', 'scipy.linalg.svd', 'sklearn.decomposition' etc. Please define svd explicitly as a function and add as appendix to your report.

```
import numpy as np

def SVD(Matrix):
    ...
    return U, E, Vt
```

References

[1] Zhe-Ming Lu and Yong Shi. Fast video shot boundary detection based on svd and pattern matching. *IEEE Transactions on Image Processing*, 22(12):5136–5145, 2013.