BCSE417P Fall 2024-25 Lab Assignments-3

Laleeth Adithya S 21BAI1863

August 16, 2024

By turning in this assignment, I agree and declare that all of this is my own work.

Write an assignment on Spatio- Temporal segmentation task given below, following the same Template as Assignment 3.

Task Description:

1. **Load Video**:
   - Load the provided video file.
2. **Frame Extraction**:
   - Extract individual frames from the video.
3. **Spatio-Temporal Segmentation**:
   - Perform segmentation on each frame using a technique like color thresholding or edge detection.
   - Track the segmented objects across frames to observe changes in motion and shape.
   - Identify the regions that remain consistent over time (foreground vs. background segmentation).
4. **Scene Cut Detection**:
   - Use pixel-based comparison or histogram differences between consecutive frames to detect abrupt changes (hard cuts).
   - Detect gradual scene transitions (Soft cuts) by analyzing frame-to-frame intensity changes over time.
5. **Mark Scene Cuts**:
   - Highlight the frames where scene cuts are detected.
   - Create a summary displaying the detected scene boundaries.
6. **Result Visualization**:

- Display frames where scene cuts are identified and show segmentation results for selected frames.

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

Step 1: Load Video

- Import necessary libraries (e.g., OpenCV, numpy, matplotlib).
- Load the video file using OpenCV's cv2.VideoCapture function.
- Checking if the video is loaded successfully and retrieve video properties like frame count, frame width, and height.

```python
import cv2

# Load video
video_path = r"C:\Users\DELL\Downloads\Video2.mp4"
cap = cv2.VideoCapture(video_path)

# Verify video load
if not cap.isOpened():
    print("Error: Could not open video file.")
else:
    frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    fps = cap.get(cv2.CAP_PROP_FPS)
    width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
    print(f"Video loaded: {frame_count} frames, {fps} FPS, {width}x{height}")
```

[14] Python

```
Video loaded: 1194 frames, 25.0 FPS, 3840x2160
```

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

Step 2: Frame Extraction

- Loop through each frame in the video.
- Extract frames and store them in a list for processing.

```python
frames = []
while True:
    ret, frame = cap.read()
    if not ret:
        break
    frames.append(frame)
cap.release()
print(f"Extracted {len(frames)} frames from the video.")
```

[15] Python

```
Extracted 1194 frames from the video.
```

```python
frames
```

[18] Python

```
[array([[[ 83, 117, 112],
         [ 81, 115, 110],
         [ 79, 113, 108],
         ...,
         [ 65,  75,  72],
         [ 65,  75,  72],
         [ 65,  75,  72]],

        [[ 83, 117, 112],
```

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

```
         [ 70, 112, 107],
         ...,
         [ 70,  80,  77],
         [ 70,  80,  77],
         [ 70,  80,  77]],

        ...,
 ...
         ...,
         [ 73,  82,  76],
         [ 73,  82,  76],
         [ 73,  82,  76]]], dtype=uint8),
 ...]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# Display sample frames
plt.figure(figsize=(10, 5))
for i in range(5):  # Display first 5 frames as examples
    plt.subplot(1, 5, i+1)
    plt.imshow(cv2.cvtColor(frames[i], cv2.COLOR_BGR2RGB))
    plt.title(f"Frame {i}")
    plt.axis('off')
plt.show()
```

[19] Python

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

Step 3: Spatio-Temporal Segmentation

- Applying color thresholding or edge detection for segmenting each frame.
- Identifying foreground and background regions based on frame differences.
- Track segmented objects across frames by using bounding boxes or contour detection.

```python
segmented_frames = []
for frame in frames:
    # Convert to grayscale
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Apply thresholding
    _, thresh = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)

    # Store segmented frame
    segmented_frames.append(thresh)
```

[3] Python

```python
segmented_frames
```

[10] Python

```
[array([[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]], dtype=uint8),
 array([[0, 0, 0, ..., 0, 0, 0],
```

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

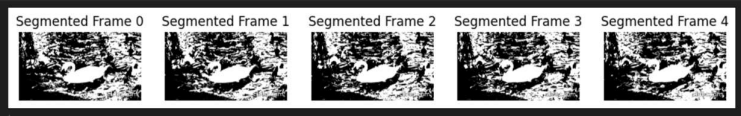
+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

```
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]], dtype=uint8),
 array([[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
...
        ...,
        [  0,   0,   0, ...,   0,   0,   0],
        [  0,   0,   0, ...,   0,   0,   0],
        [  0,   0,   0, ...,   0,   0,   0]], dtype=uint8),
 ...]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# Display segmented frames
plt.figure(figsize=(12, 5))
for i in range(5):  # Display the first 5 segmented frames
    plt.subplot(1, 5, i+1)
    plt.imshow(segmented_frames[i], cmap='gray')
    plt.title(f"Segmented Frame {i}")
    plt.axis('off')
plt.show()
```

[23] Python



21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

Step 4: Track Consistent Regions Across Frames

- For each frame, compare segmented regions to track motion and shape changes.
- Identify stable areas that represent the background, while varying regions signify foreground elements.

```python
consistent_regions = []
for i in range(1, len(segmented_frames)):
    # Calculate frame difference to track motion
    frame_diff = cv2.absdiff(segmented_frames[i-1], segmented_frames[i])

    # Mark regions with no motion as background
    background = cv2.bitwise_not(frame_diff)
    consistent_regions.append(background)
```

[4] Python

```python
consistent_regions
```

[9] Python

```
[array([[255, 255, 255, ..., 255, 255, 255],
        [255, 255, 255, ..., 255, 255, 255],
        [255, 255, 255, ..., 255, 255, 255],
        ...,
        [255, 255, 255, ..., 255, 255, 255],
        [255, 255, 255, ..., 255, 255, 255],
        [255, 255, 255, ..., 255, 255, 255]], dtype=uint8),
 array([[255, 255, 255, ..., 255, 255, 255],
        [255, 255, 255, ..., 255, 255, 255],
        [255, 255, 255, ..., 255, 255, 255],
        ...,
        [255, 255, 255, ..., 255, 255, 255],
```

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

```
array([[255, 255, 255, ...,   0,   0,   0],
       [255, 255, 255, ...,   0,   0,   0],
       [255, 255, 255, ...,   0,   0,   0],
       ...,
...
       ...,
       [  0,   0,   0, ..., 255, 255, 255],
       [  0,   0,   0, ..., 255, 255, 255],
       [  0,   0,   0, ..., 255, 255, 255]], dtype=uint8),
 ...]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# Display tracked consistent regions
plt.figure(figsize=(12, 5))
for i in range(5):  # Display every 5th consistent region frame
    plt.subplot(1, 5, i+1)
    plt.imshow(consistent_regions[i], cmap='gray')
    plt.title(f"Consistency Frame {i}")
    plt.axis('off')
plt.show()
```

[22] Python

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

Step 5: Scene Cut Detection

- Use pixel-based or histogram-based comparisons to detect scene cuts.
- Calculate pixel or intensity differences between consecutive frames to identify hard cuts.
- For soft cuts, analyze gradual changes over multiple frames.

```python
scene_cuts = []
for i in range(1, len(frames)):
    # Calculate histogram difference
    hist1 = cv2.calcHist([frames[i-1]], [0], None, [256], [0, 256])
    hist2 = cv2.calcHist([frames[i]], [0], None, [256], [0, 256])

    # Use correlation or Euclidean distance for comparison
    hist_diff = cv2.compareHist(hist1, hist2, cv2.HISTCMP_CORREL)
    if hist_diff < 0.7:  # Threshold for hard cut
        scene_cuts.append(i)
```

[5] Python

```python
scene_cuts
```

[8] Python

```
[106,
 425,
 426,
 1179,
 1184,
 1185,
 1186,
 1187,
```

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

```
 426,
 1179,
 1184,
 1185,
 1186,
 1187,
 1188,
 1189,
 1190,
 1191,
 1192,
 1193]
```

```python
# Display frames where scene cuts are detected
plt.figure(figsize=(14, 5))
for j, cut in enumerate(scene_cuts[:5]):  # Display first 5 cuts
    plt.subplot(1, 5, j+1)
    plt.imshow(cv2.cvtColor(frames[cut], cv2.COLOR_BGR2RGB))
    plt.title(f"Scene Cut at Frame {cut}")
    plt.axis('off')
plt.show()
```

[26] Python

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

Step 6: Mark Scene Cuts

- Highlight detected frames where scene cuts are detected.
- Create a summary displaying detected scene boundaries.

```python
scene_cut_summary = [frames[i] for i in scene_cuts]
print(f"Scene cuts detected at frames: {scene_cuts}")
```

[6] Python

```
Scene cuts detected at frames: [106, 425, 426, 1179, 1184, 1185, 1186, 1187, 1188, 1189, 1190, 1191, 1192, 1193]
```

```python
# Summary of detected scene cuts and segmentation
summary_frames = scene_cuts[:5]  # Show first 5 scene cuts for visualization

plt.figure(figsize=(15, 6))
for i, idx in enumerate(summary_frames):
    plt.subplot(2, len(summary_frames), i + 1)
    plt.imshow(cv2.cvtColor(frames[idx], cv2.COLOR_BGR2RGB))
    plt.title(f"Scene Cut at Frame {idx}")
    plt.axis('off')

    plt.subplot(2, len(summary_frames), i + 1 + len(summary_frames))
    plt.imshow(segmented_frames[idx], cmap='gray')
    plt.title(f"Segmented Frame {idx}")
    plt.axis('off')
plt.show()
```

[27] Python

21BAI1863_MV_Lab-3.ipynb

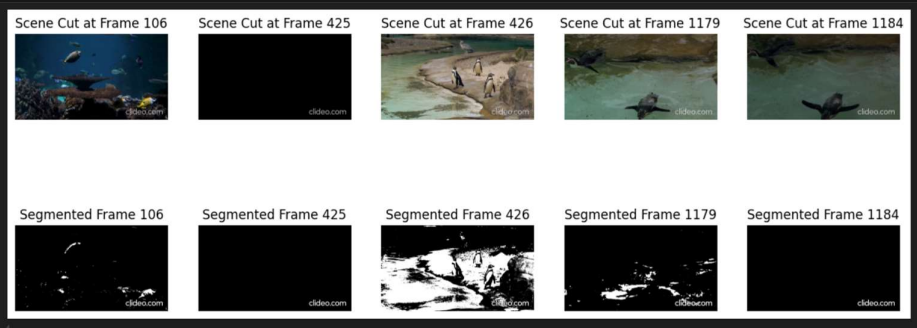
C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

```python
    plt.axis('off')

    plt.subplot(2, len(summary_frames), i + 1 + len(summary_frames))
    plt.imshow(segmented_frames[idx], cmap='gray')
    plt.title(f"Segmented Frame {idx}")
    plt.axis('off')
plt.show()
```

[27] Python



21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

Step 7: Result Visualization

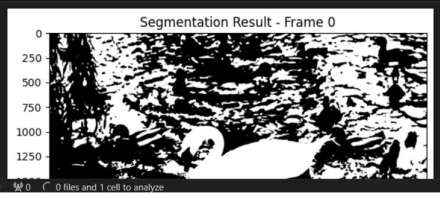
- Display frames with scene cuts and segmentation results for selected frames.
- Use matplotlib or OpenCV's imshow to visualize segmentation and scene-cut detection results.

```python
import matplotlib.pyplot as plt

# Display segmentation result
for i in range(0, len(segmented_frames), 50):  # Display every 50th frame for brevity
    plt.imshow(segmented_frames[i], cmap='gray')
    plt.title(f'Segmentation Result - Frame {i}')
    plt.show()

# Display frames with scene cuts
for cut in scene_cuts:
    plt.imshow(frames[cut])
    plt.title(f'Scene Cut Detected - Frame {cut}')
    plt.show()
```

[7] Python

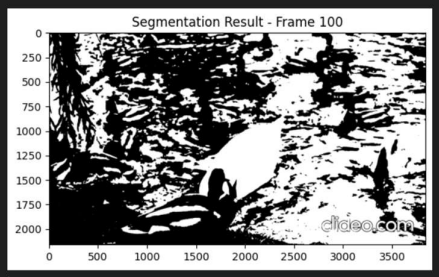


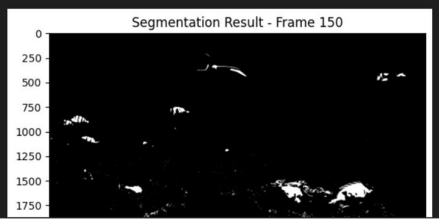
0 files and 1 cell to analyze Cell 1 of 23 Go Live

21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

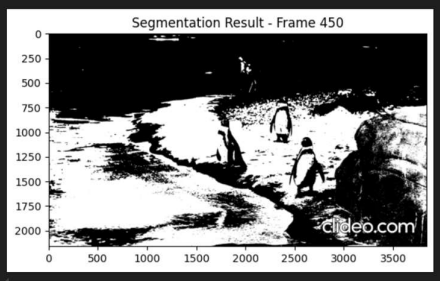


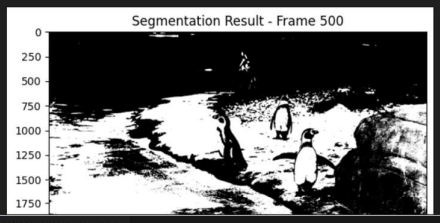


21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel

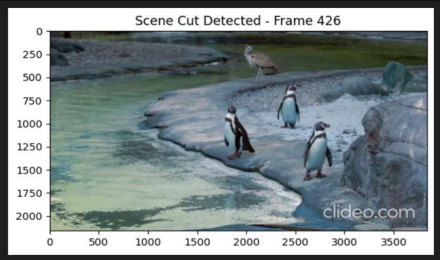


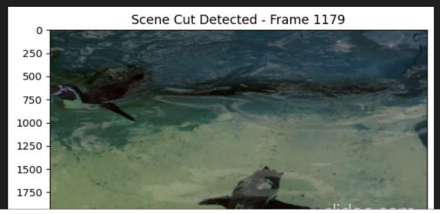


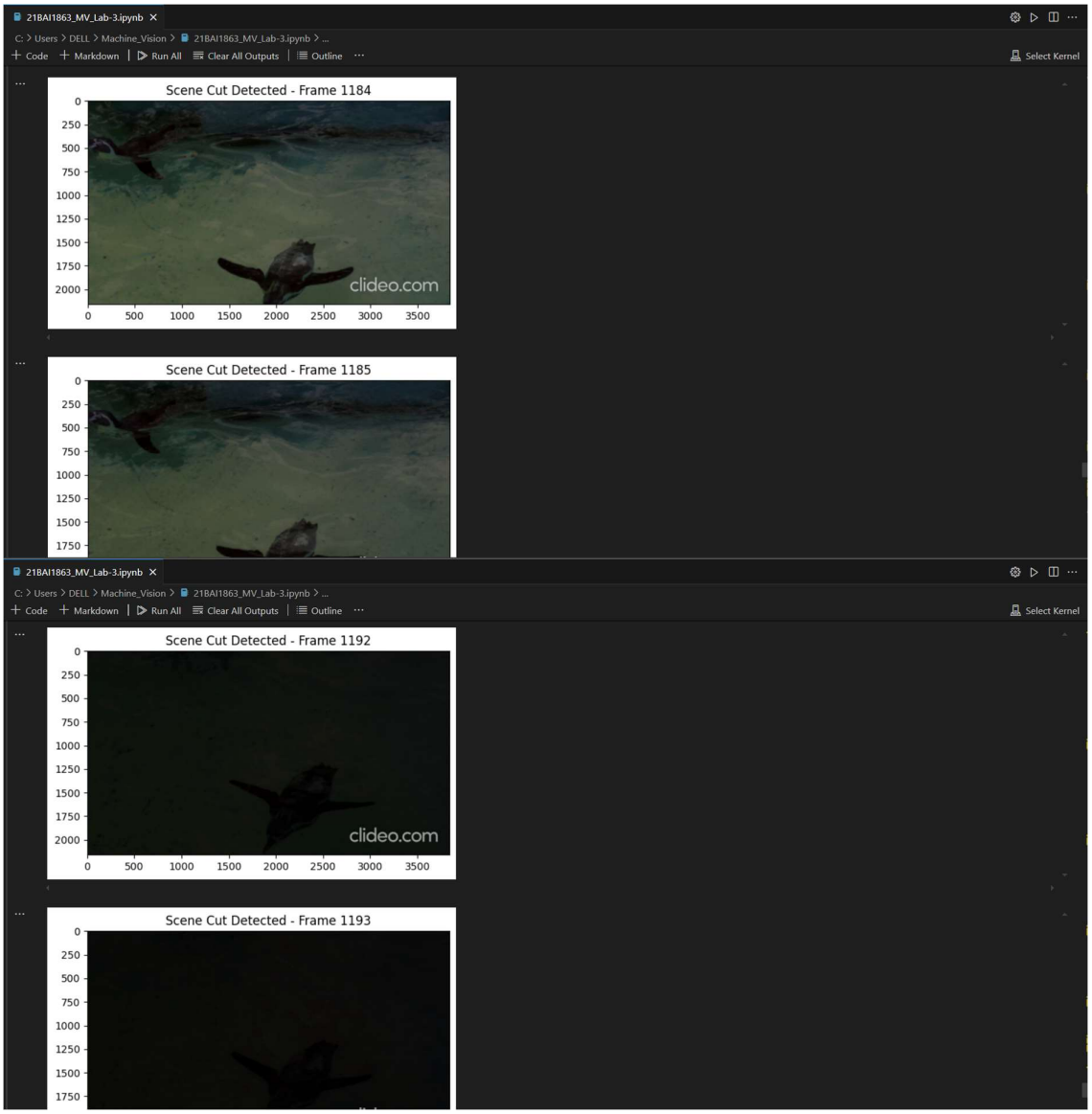
21BAI1863_MV_Lab-3.ipynb

C: > Users > DELL > Machine_Vision > 21BAI1863_MV_Lab-3.ipynb > ...

+ Code + Markdown | Run All Clear All Outputs | Outline ··· Select Kernel







Video Link:
https://drive.google.com/file/d/1uvuOtWbUOkJM89zp3Zgq5mjmLfH-QhvV/view?usp=sharing

Github Link:
https://github.com/Adithya007-Dev/Machine_Vision/tree/main/Lab_3