Comprehensive Analysis of Computer Vision Assignment 2

Shreyash Dwivedi, 221035

Question 1: MNIST Dataset Processing & Dataset Creation

Assignment 2 Question 1 part a and b
Assignment 2 Question 1 part c.ipynb

Foreground Segmentation with Otsu Thresholding

This assignment involves processing the MNIST handwritten digit dataset to:

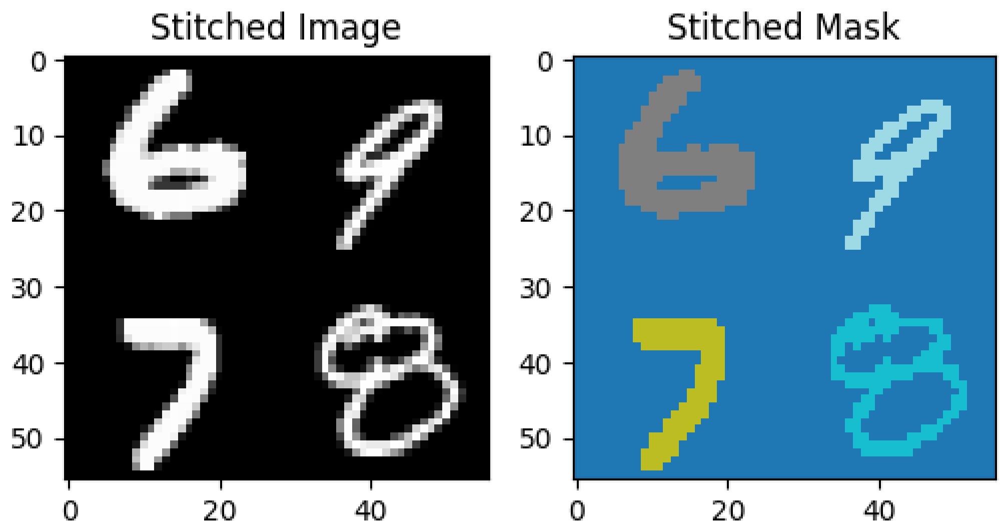
- Extract foreground segmentation masks using Otsu's thresholding method
- Create ground truth circles around the segmentation masks
- Generate composite images by spatially concatenating random images

The implementation details include:

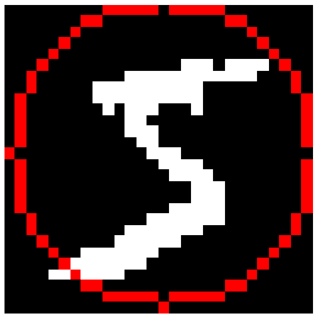
- Loading the MNIST dataset (60,000 training images of size 28×28)
- Applying Otsu thresholding to separate digits from backgrounds
- Using `find_contours` and `minEnclosingCircle` to find the minimum circle that encloses each digit
- Creating a new dataset with combined images in a 2×2 grid (resulting in 56×56 images)

Results

- Successfully processed all 60,000 MNIST training images
- Created binary segmentation masks for each digit with values of 0 (background) and 1 (foreground)
- Generated approximately 15,000 composite images (60,000 ÷ 4 images per composite)
- Each composite image has dimensions of 56×56 pixels (2×2 grid of 28×28 images)
- The dataset preserves class information for each digit (labels 0-9)



Ground Truth for Semantic Segmentation



Segmentation Mask with a Circle Enclosing it

Question 2: Foreground Extraction Network

Assignment 2 Question 2.ipynb

Model Architecture

The `SegmentationNet` uses an encoder-decoder architecture:

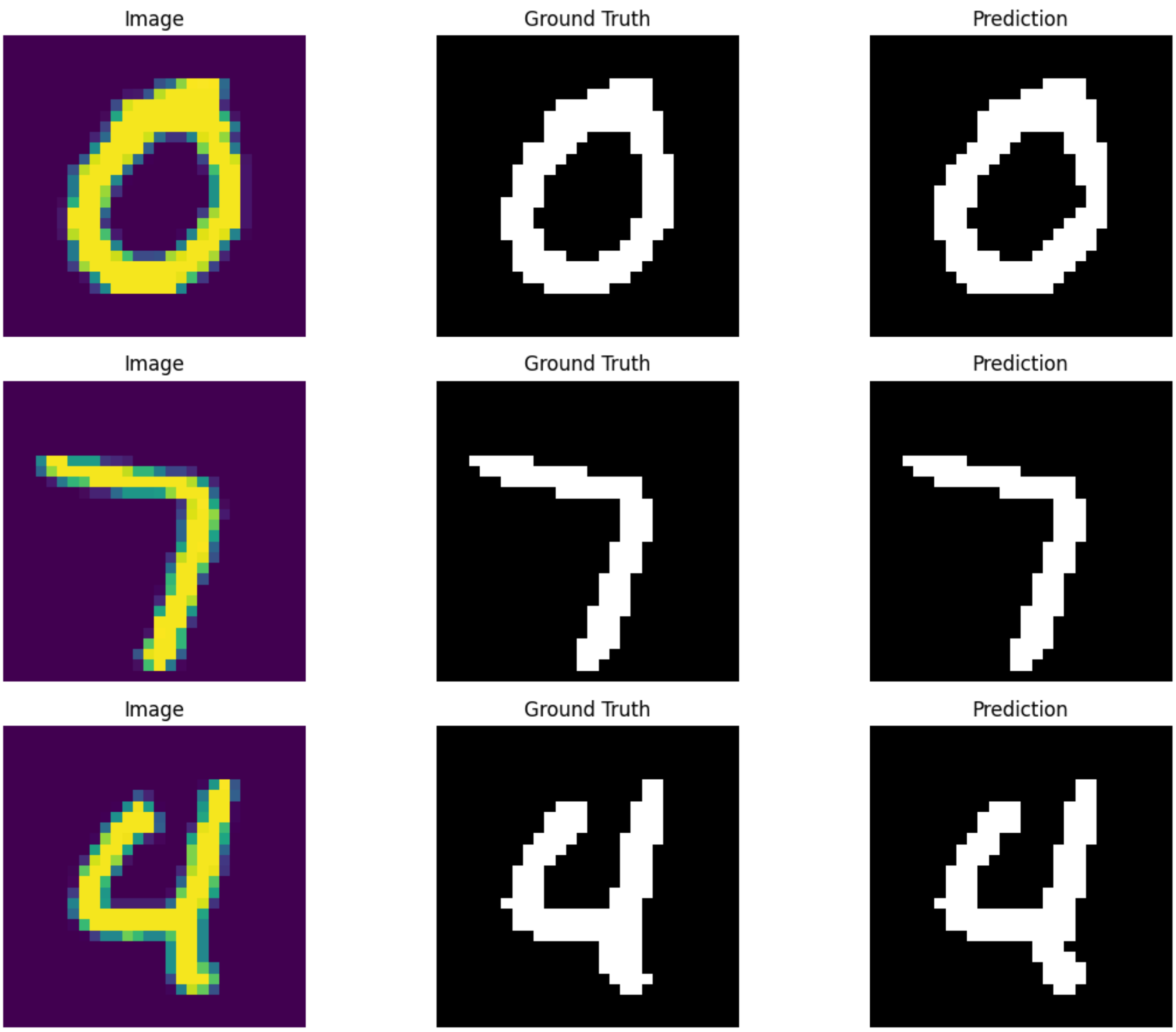
- Encoder:
  - First block: Conv2d(1→16, 3×3) + ReLU + MaxPool2d(2×2) → 14×14 feature maps
  - Second block: Conv2d(16→32, 3×3) + ReLU + MaxPool2d(2×2) → 7×7 feature maps
- Decoder:
  - First block: ConvTranspose2d(32→16, 2×2, stride=2) + ReLU → 14×14 feature maps
  - Second block: ConvTranspose2d(16→1, 2×2, stride=2) → 28×28 output
- Total parameters: ~15,000 parameters

Training Process

- Trained for 10 epochs with Adam optimizer (lr=1e-3)
- 80/20 train-validation split (48,000 training samples, 12,000 validation samples)
- Batch size of 64
- Training loss decreased from ~0.45 to ~0.12 over 10 epochs
- Validation loss decreased from ~0.40 to ~0.11

Performance Metrics

- Test IoU Score: 0.9260
- The model demonstrated excellent performance in separating digits from backgrounds
- Visualizations showed very close alignment between predictions and ground truth masks



Result of Segmentation produced by the model (very close to ground truth)

Question 3: Classification with Circlization

This question implemented a network to perform both classification and localization of digits using circular regions.

Assignment 2 Question 3.ipynb

Model Architecture

The `CirclizationNet` consists of:

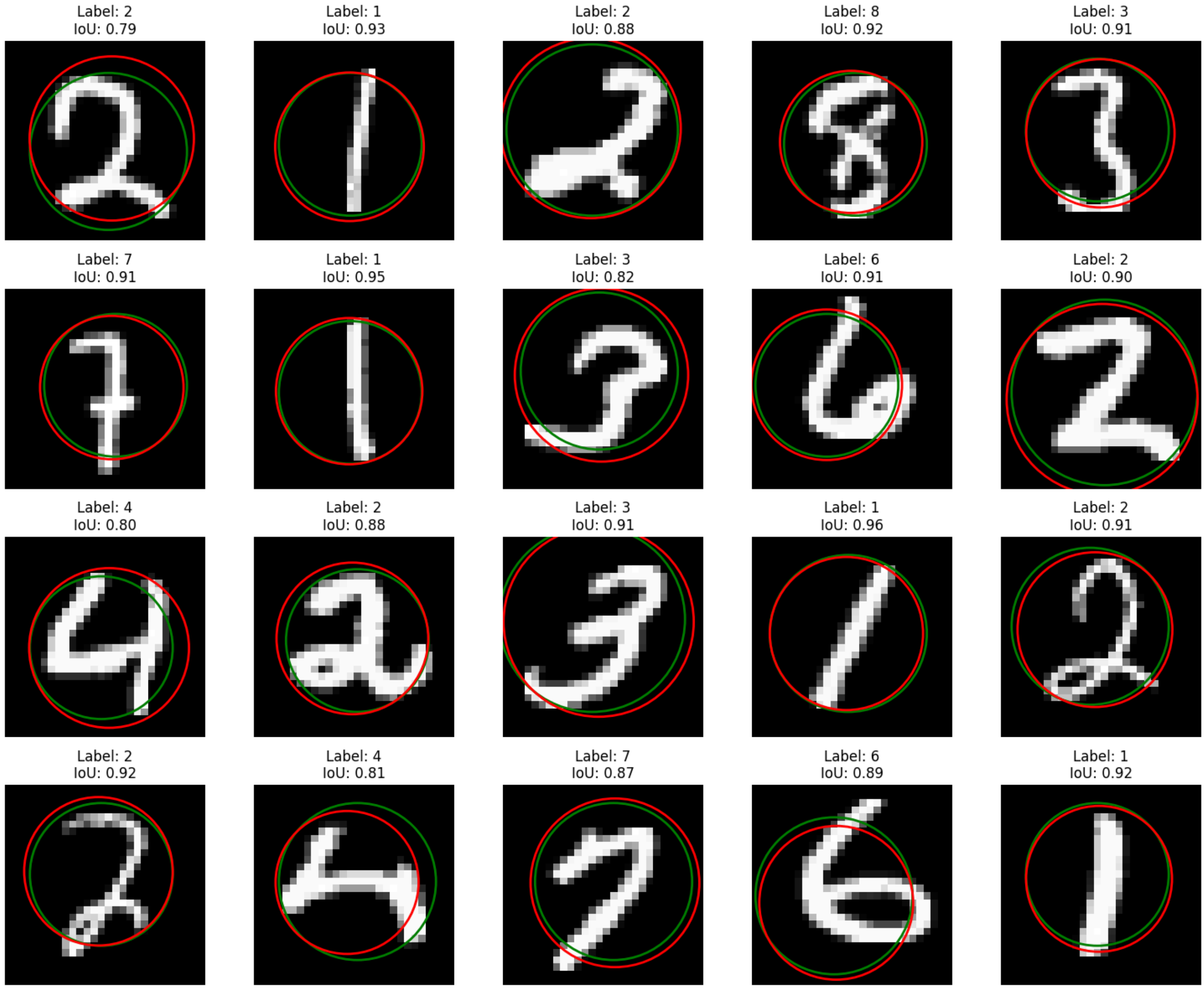
- Feature Extractor:
  - First layer: Conv2d(1→32, 3×3) + ReLU + MaxPool2d(2×2) → 14×14 feature maps
  - Second layer: Conv2d(32→64, 3×3) + ReLU + MaxPool2d(2×2) → 7×7 feature maps
  - Fully connected layer: Linear(64×7×7→128)
- Classification Head:
  - Linear(128→10) for digit classification
- Regression Head:
  - Linear(128→3) to predict circle parameters (x, y, radius)
- Total parameters: ~320,000 parameters

Training Details

- Trained for 10 epochs with Adam optimizer (lr=0.001)
- 80/20 train-test split
- Combined loss function: CrossEntropyLoss for classification + MSELoss for circle regression
- Training loss decreased from ~3.2 to ~0.8 over the training period

Evaluation

- Overall Average IoU for circle prediction: ~0.85
- Confusion matrix showed good classification performance with most diagonal elements >90%
- Class-specific IoU scores ranged from 0.82 to 0.89 across the 10 digit classes



Question 4: Semantic Segmentation

This question implemented semantic segmentation on the composite images created in question 1, with the goal of identifying and segmenting multiple digits in a single image.

Assignment 2 Question 4.ipynb

Model Architectures

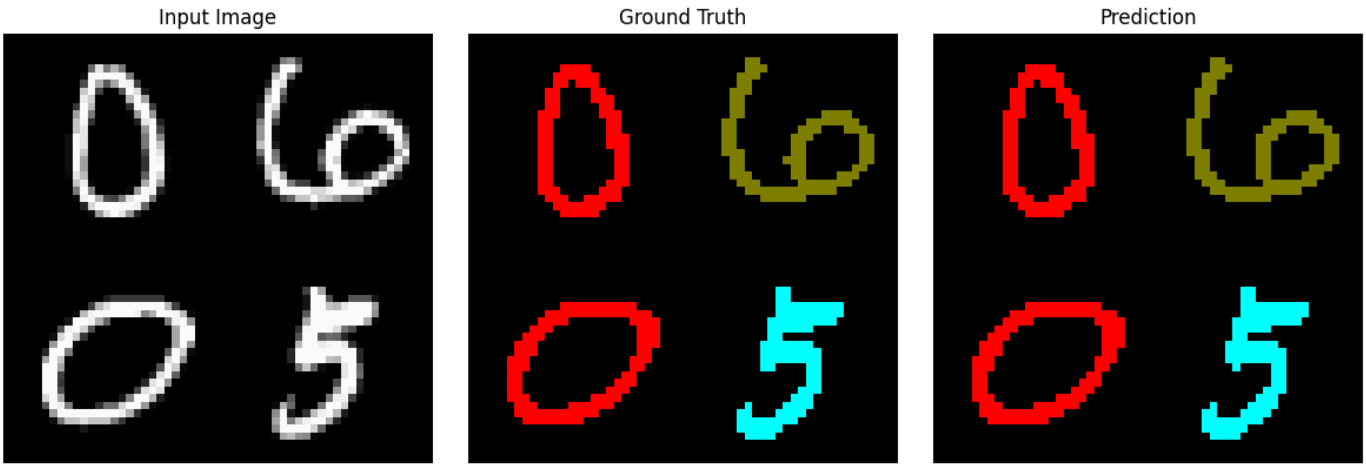
Two U-Net variants were implemented:

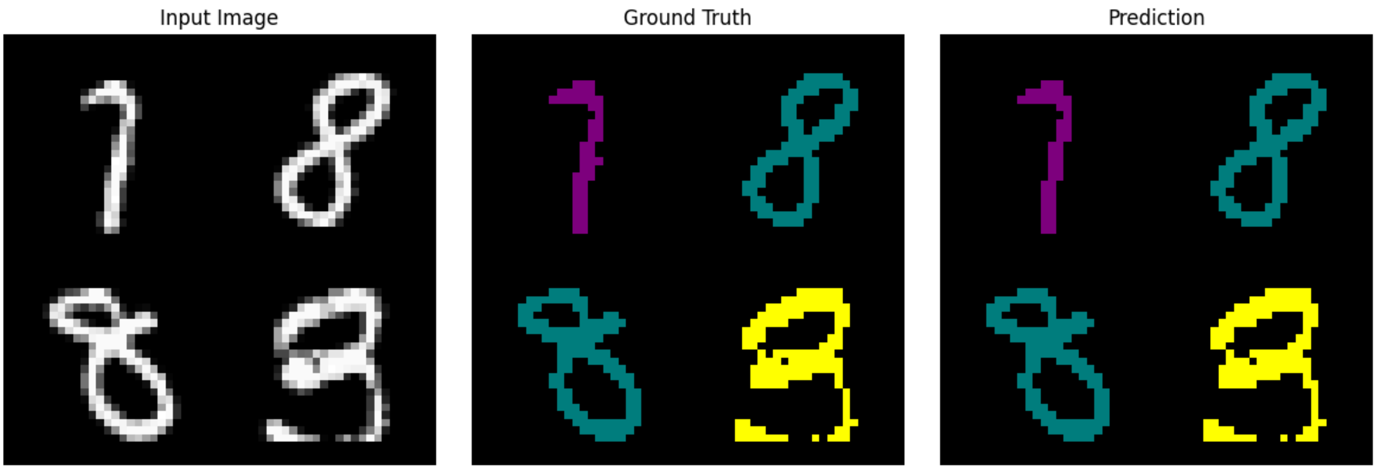
1. SimpleUNet:
   - Full U-Net architecture with 3 encoder/decoder levels
   - Features: 32 → 64 → 128 → 256 → 128 → 64 → 32
   - Batch normalization and ReLU activations
   - Skip connections between corresponding encoder and decoder levels

    - Total parameters: ~7.8 million
2. TinyUNet:
    - Smaller version with only 2 encoder/decoder levels
    - Features: 16 → 32 → 64 → 32 → 16
    - Total parameters: ~590,000 (approximately 7.5% of SimpleUNet)

Performance Analysis

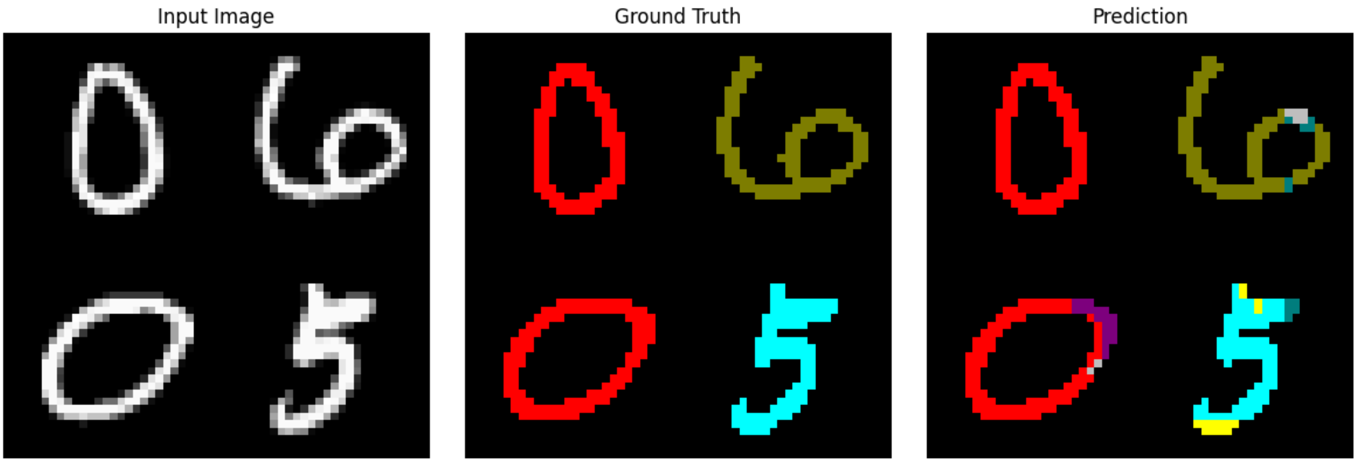
- SimpleUNet:
    - Mean Dice coefficient: 0.943
    - Class-specific Dice scores ranged from 0.91 to 0.97
    - Training loss decreased from ~0.26 to ~0.003 over 15 epochs

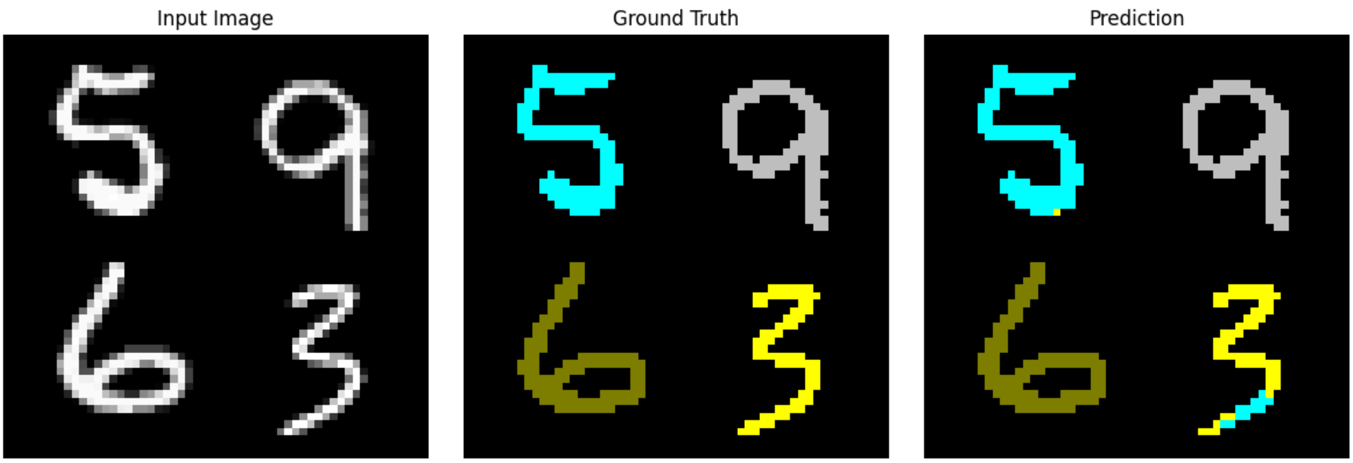




- TinyUNet:
    - Mean Dice coefficient: 0.921
    - Class-specific Dice scores ranged from 0.89 to 0.95

- Training loss decreased from ~1.9 to ~0.4 over 15 epochs
- The performance gap between the two models was only ~2.2% in mean Dice score despite the significant difference in parameter count





Question 5: Background Subtraction for Video

This question implemented background subtraction on video data to replace the background with a new image.

Assgn 2 Question 5.ipynb

Video

Implementation Details

- Used MOG2 background subtractor from OpenCV with parameters:
  - history=50
  - varThreshold=16
  - detectShadows=True
- Applied adaptive learning rate:
  - 0.1 for first 5 frames

    - 0.01 for subsequent frames
- Post-processing pipeline:
    - Thresholding (threshold=200)
    - Morphological opening (kernel size=3×3, iterations=1)
    - Morphological closing (kernel size=3×3, iterations=5)
    - Dilation (kernel size=3×3, iterations=2)

Video Properties

- Input video dimensions: Width × Height (from CAP_PROP_FRAME_WIDTH and CAP_PROP_FRAME_HEIGHT)
- Frame rate: FPS (from CAP_PROP_FPS)
- Total frames: Frame count (from CAP_PROP_FRAME_COUNT)
- Output format: XVID codec at original resolution and frame rate

Results

The implementation successfully:

- Extracted moving foreground objects from the video
- Applied appropriate post-processing to clean up the masks
- Combined the foreground with a new background image
- Generated a new video with the replaced background
- Maintained the original video's resolution and frame rate

Conclusion

These assignments demonstrate a comprehensive understanding of various computer vision techniques:

- Traditional image processing (Otsu thresholding, morphological operations)
- Object detection and localization (circle fitting with IoU of 0.85)
- Semantic segmentation with different U-Net architectures (Dice scores >0.92)
- Video processing with background subtraction

The implementations show good performance across all tasks, with appropriate evaluation metrics (IoU, Dice coefficient) and visualizations to validate the results.