CSC566

LAB 6
Name: Amir Hafizi Bin Musa
Student ID: 2024745815
Group: A4CDCS2306A

CLASSICAL AND DEEP LEARNING SEGMENTATION METHODS

PART1: CLASSICAL IMAGE SEGMENTATION METHODS

Image used in this PART 1 section for classical segmentation methods are:

1. cameraman.tif
2. coins.png
3. peppers.png
4. rice.png
5. saturn.png

Segmentation methods used are:

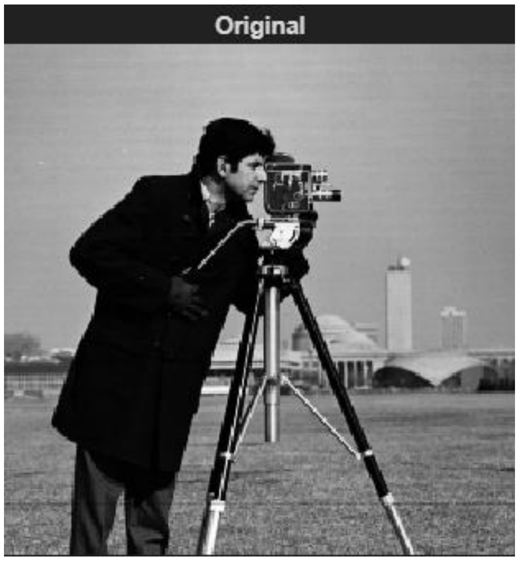
A. Adaptive Thresholding (Built-in)
B. Nick Thresholding (Custom)
C. Bradley Thresholding (Custom)
D. Region Growing (Using grayconnected)
E. Edge Detection (Sobel & Canny)

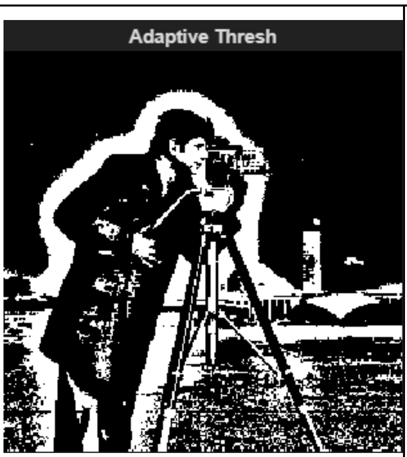
Each of the image result will be displayed in a table with the following columns:

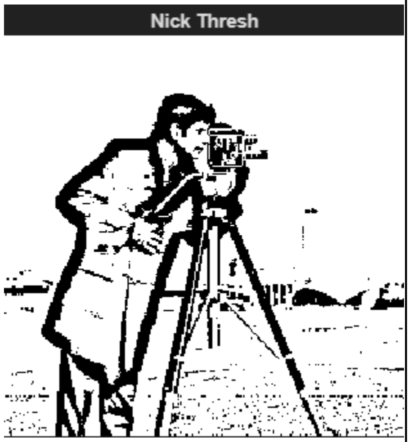
- Original
- Adaptive Thresholding
- Nick Thresholding
- Bradley Thresholding
- Region Growing
- Edge Method 1 (Sobel)
- Edge Method 2 (Canny)

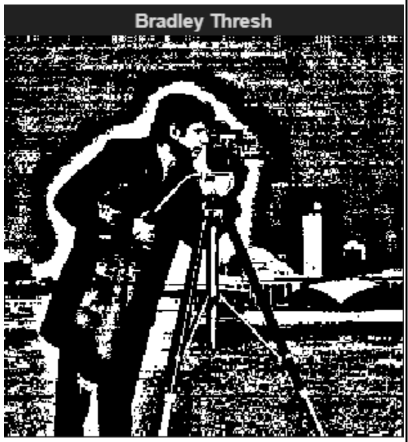
Classical Image Segmentation Result:

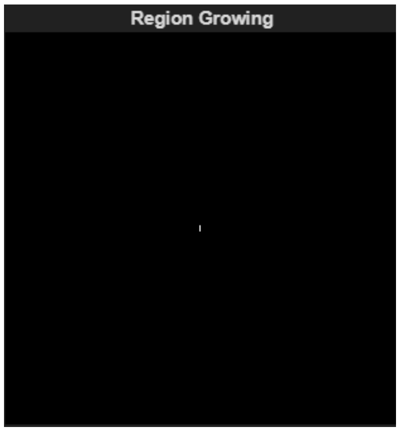
1. **cameraman.tif**

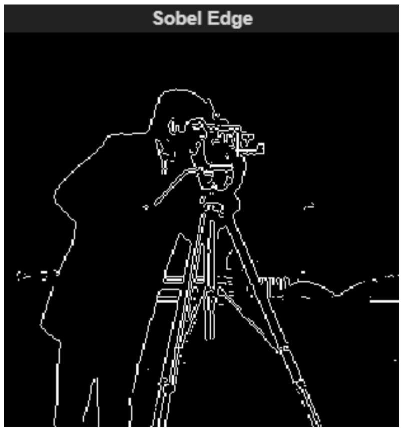


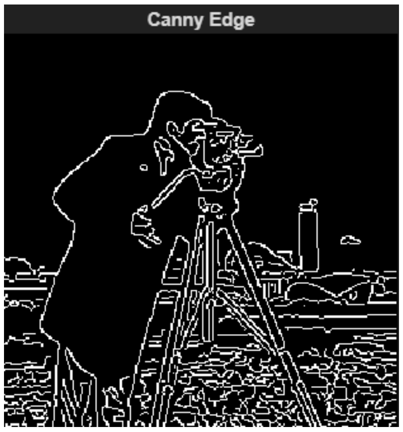




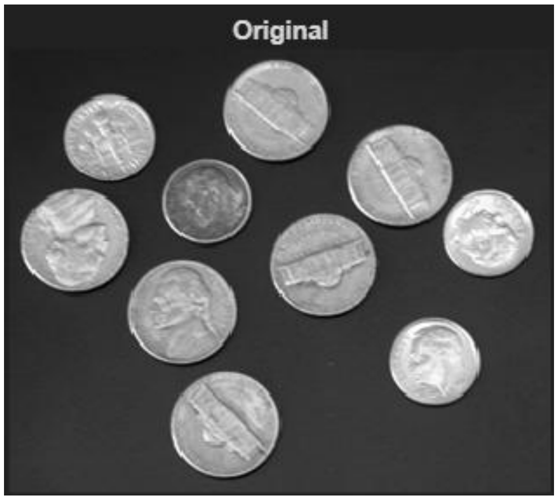


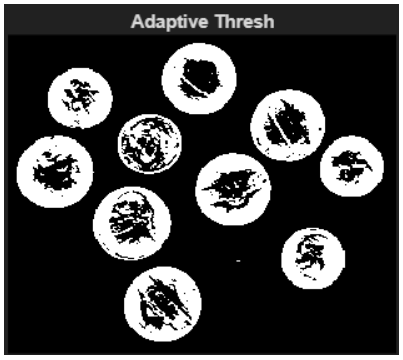


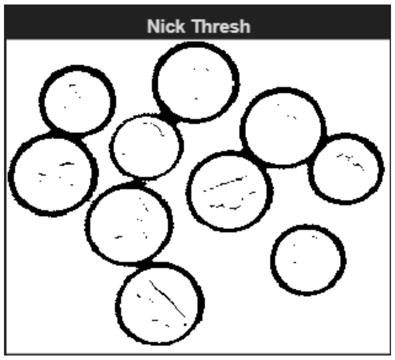


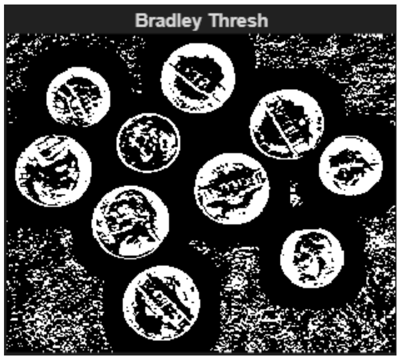


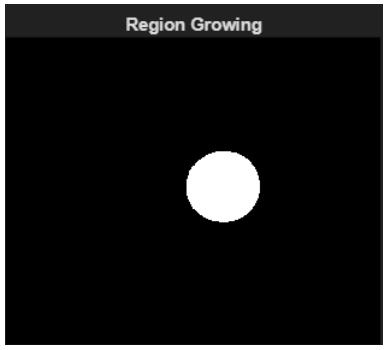
2. coins.png

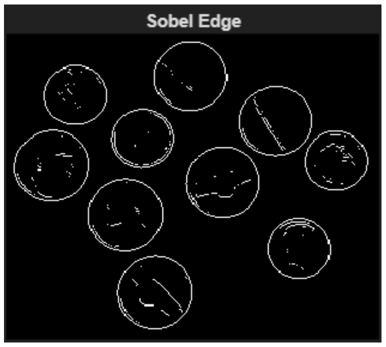

Original

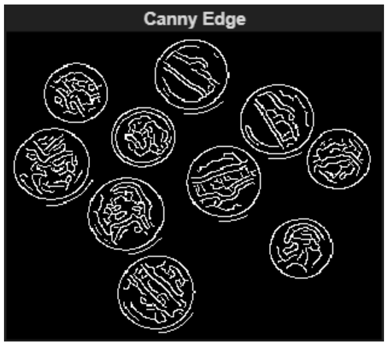

Adaptive Thresh


Nick Thresh

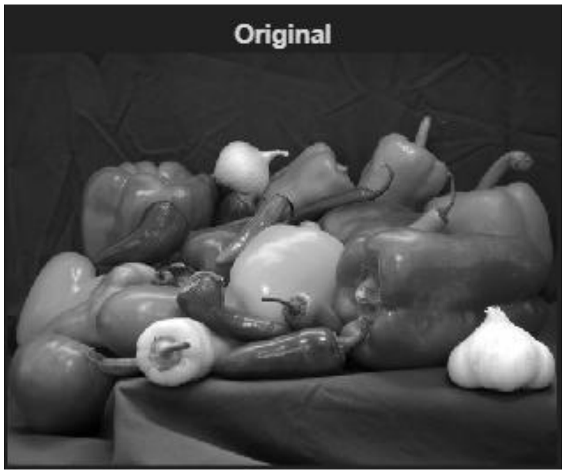

Bradley Thresh

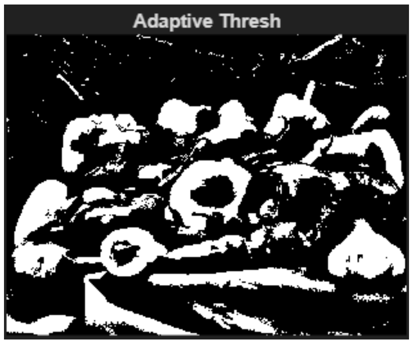

Region Growing

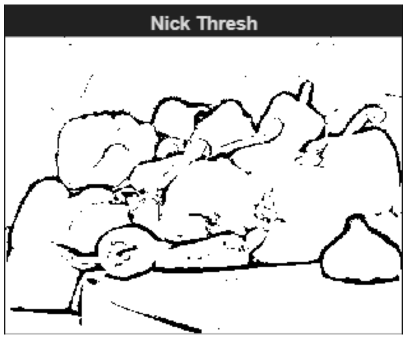

Sobel Edge

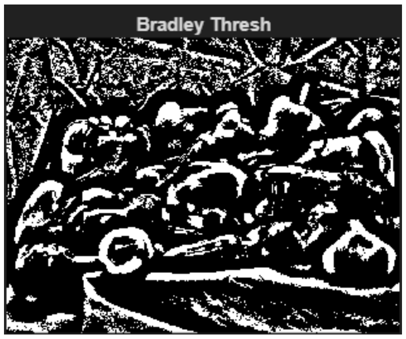

Canny Edge

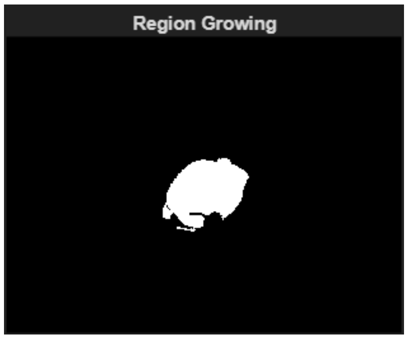
3. peppers.png

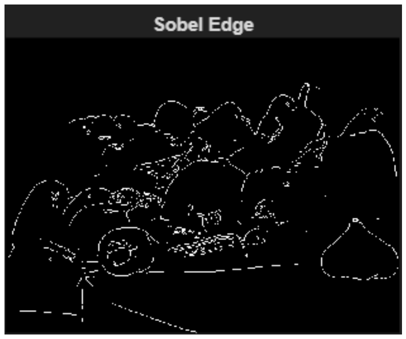


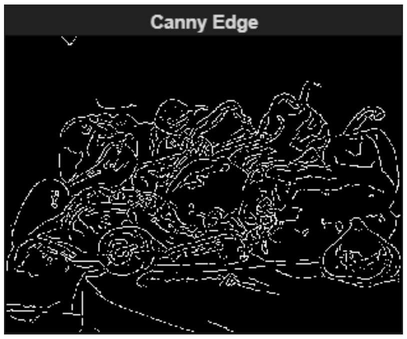




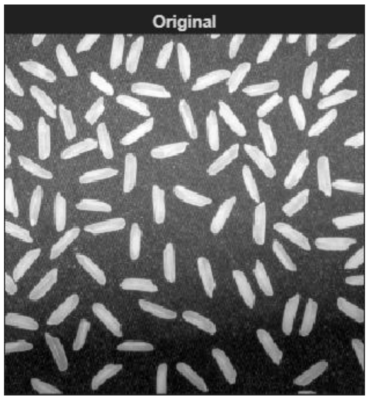
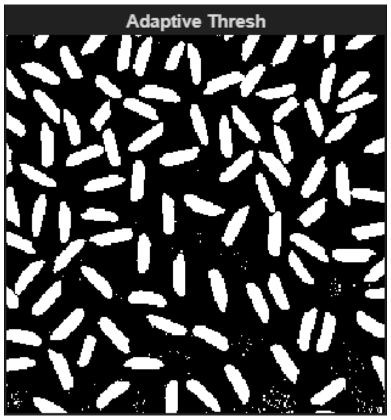
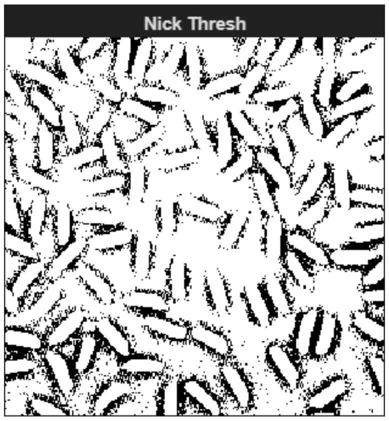
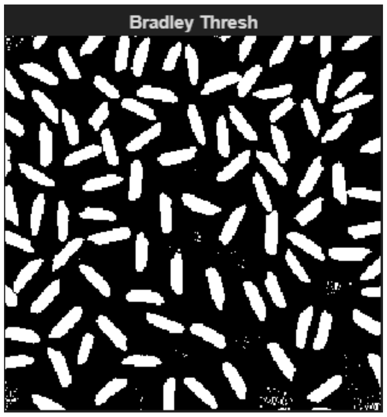
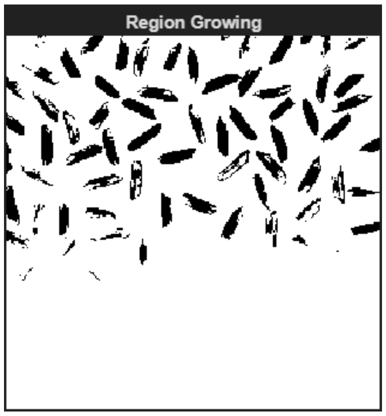
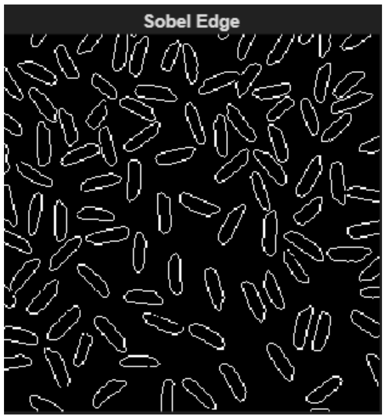
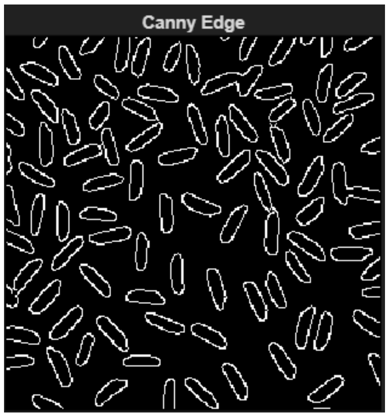




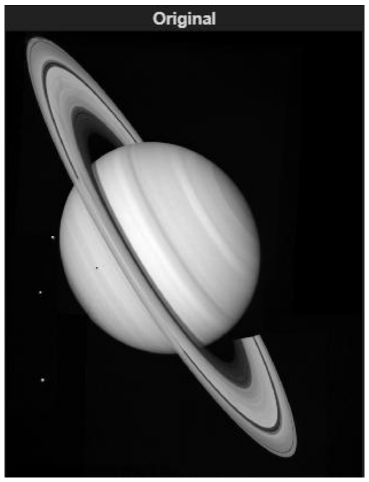
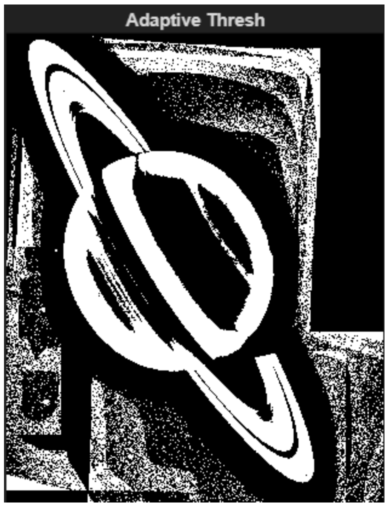
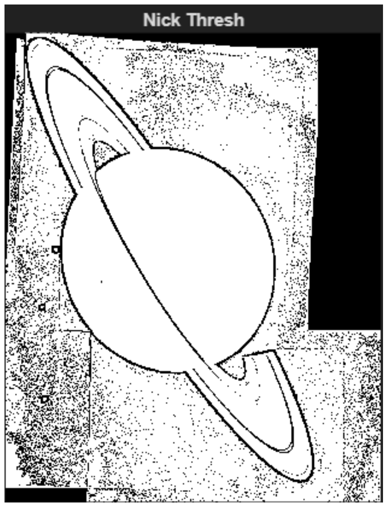
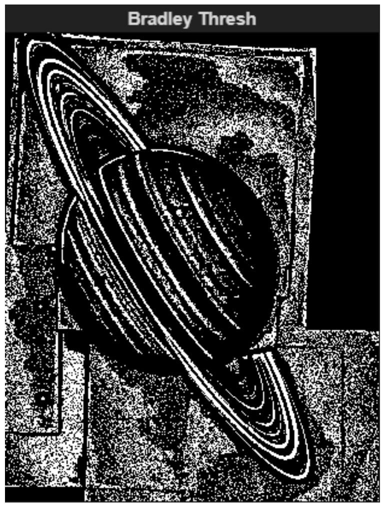
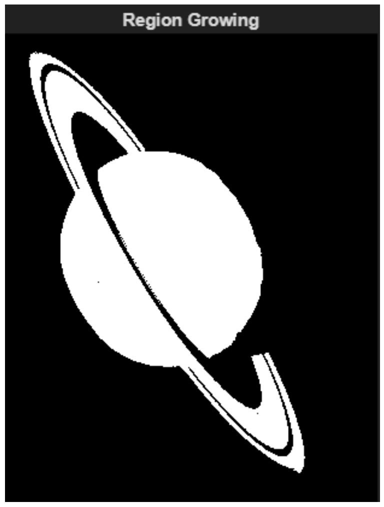
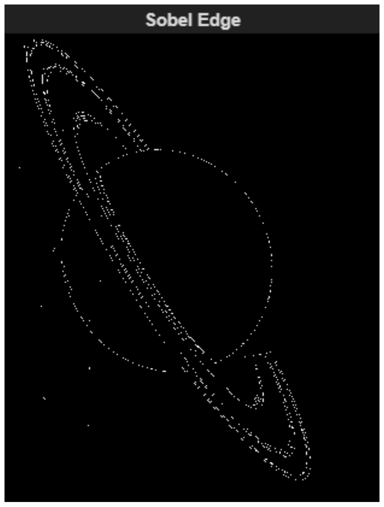
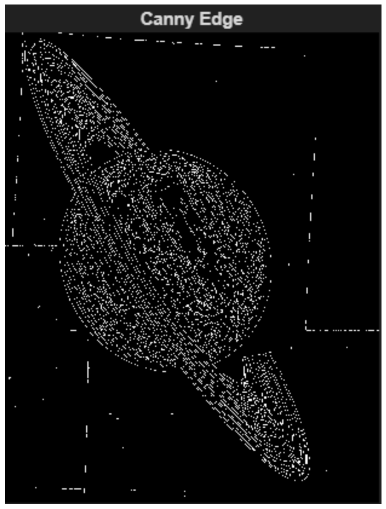




4. rice.png

	 Original	
 Adaptive Thresh	 Nick Thresh	 Bradley Thresh
 Region Growing	 Sobel Edge	 Canny Edge

5. saturn.png

	Original	
Adaptive Thresh	Nick Thresh	Bradley Thresh
Region Growing	Sobel Edge	Canny Edge

Discussion:

cameraman.tif:

The **best method** for cameramen.tif is **Adaptive Thresholding**. This is because of it handles local lighting variations which effectively managed to separate the cameraman from light background. **Why does certain method struggles?** This is because methods like Region Growing are experiencing under-segment because the seed point's intensity did not match its neighbors within the specified tolerance, preventing the region from expanding. And finaly the most **suitable method for this image type is** Adaptive Thresholding, because it is the best for consistent segmentation under uneven lighting.

coins.png:

The **best method** for coins.png is **Adaptive Thresholding**. This method able to isolates the coins from the background really well. Certain methods like Region Growing **struggles to handle this** because it merges adjacent coins if their intensities are similar. Method like Sobel Edge Detection **also struggle** as it produces fragmented edges. Because of this, Adaptive Thresholding become the **most suitable** method for simple object isolation.

peppers.png:

Based on the result, the **best method** for this peppers.png image is Canny Edge Detection as it robustly detects pepper boundaries despite the complex texture. We can easily see that method like Adaptive Thresholding **struggles** as it merges overlapping peppers due to the similar intensities. Region Growing also fails as the “central seed” grows into background due to texture variations. Thus, making the Canny Edge Detection the **most suitable** for boundary extraction, which is adaptive for initial segmentation.

rice.png:

After testing on the rice.png image, Bradley Thresholding gives out the **best output** as it produces the cleanest binary mask with well-defined and solid white rice against black background. Other method like Nick Thresholding **fails significantly** due to hypersensitivity to background texture and noise, resulting in over-segmented image. Also, Region Growing **perform poorly** due to uneven lighting. Hence, Bradley Thresholding becoming the **most suitable** method for this as it maintains the clear object separation. Canny and Sobel only provide outlines rather than filled regions of rice.

saturn.png:

Region Growing by far provided the **best result** as it creates a perfectly clean segmentation of planet and its rings, able to greatly ignore the background entirely. In comparison with other methods like Adaptive, Nick and Bradley Thresholding, these methods **struggle a lot** with dark background, these methods over-amplify the subtle noise in the black space, treating it as texture and create significant artifacts. Therefore, Region Growing is the **most suitable** method as the object, Saturn is a high-contrast, distinct and spatially connected object which allows the algorithm to easily group bright pixels together while ignoring the background.

PART2: SEMANTIC SEGMENTATION (DEEP LEARNING)

Using the DeepLabV3+, ResNet18 model downloaded from MathWorks as my MATLAB 2025b is the cracked version.

deeplabv3plus.mat 58.52 MB

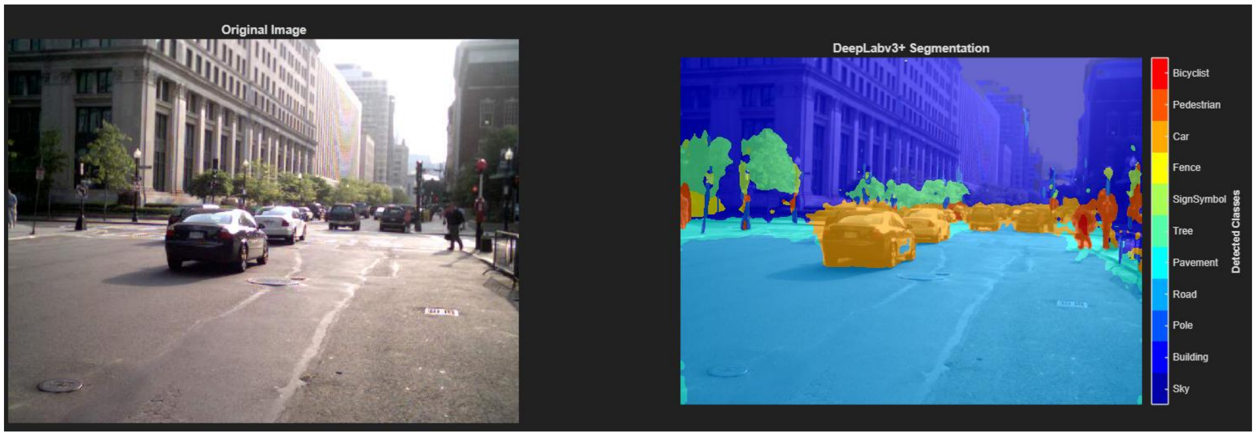
Model link:
https://www.mathworks.com/supportfiles/vision/data/deeplabv3plusResnet18CamVid_v2.zip
Using the MATLAB built-in **street2.jpg** image as the test image, streetScene.png is not available in this MATLAB 2025b version.

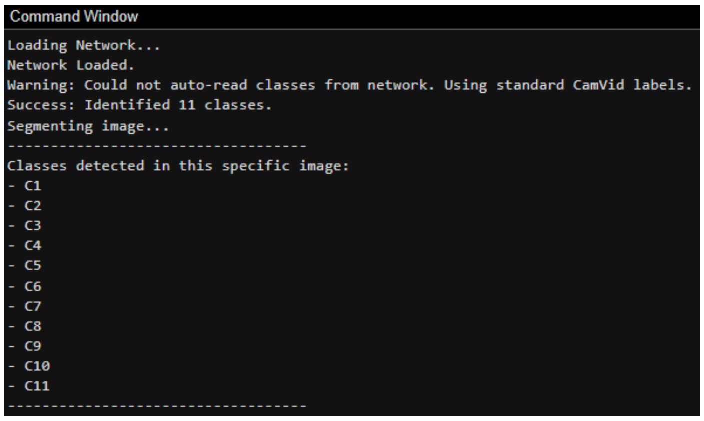
Steps in creating the PART2 image segmentation:

1. Step 1: Load MATLAB's pretrained DeepLabv3+ network
2. Choose a test image (MATLAB built-in DeepLearning Toolbox image, street2.jpg will be used)
3. Perform semantic segmentation
4. Display detected classes
5. Create a side-by-side comparison figure

Semantic Image Segmentation Result:

Side-by-side comparison:

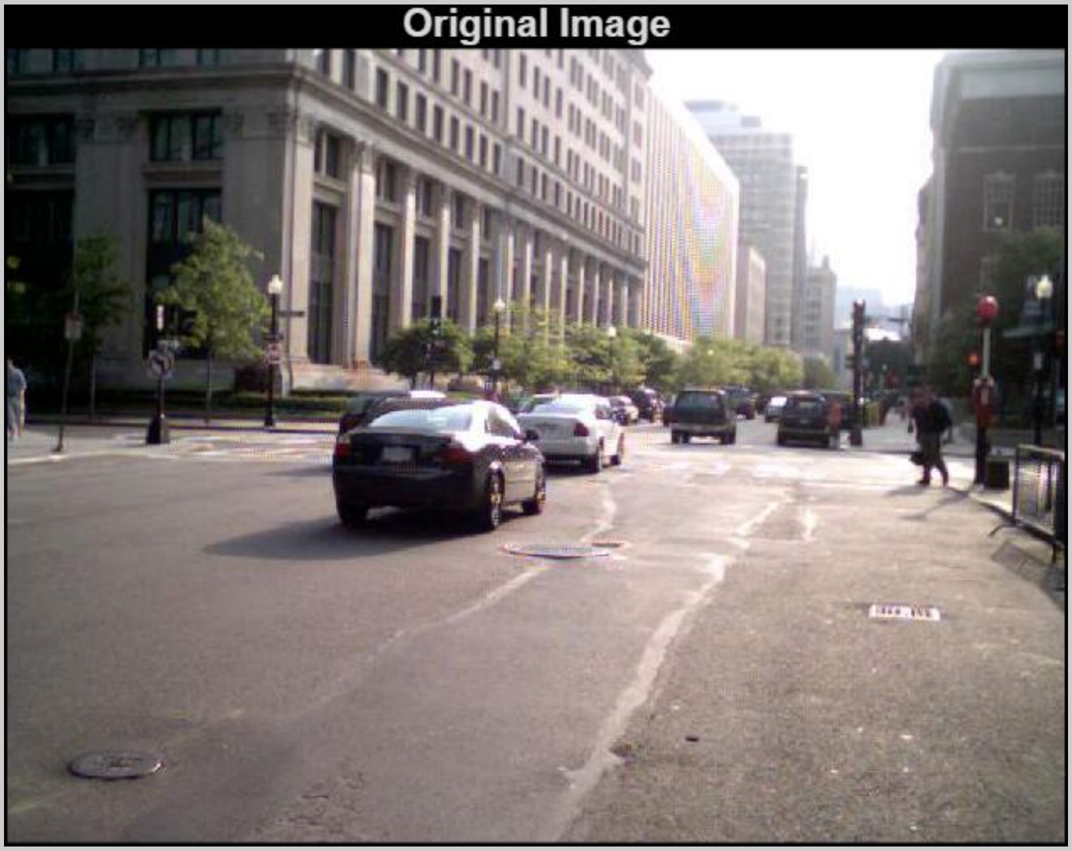
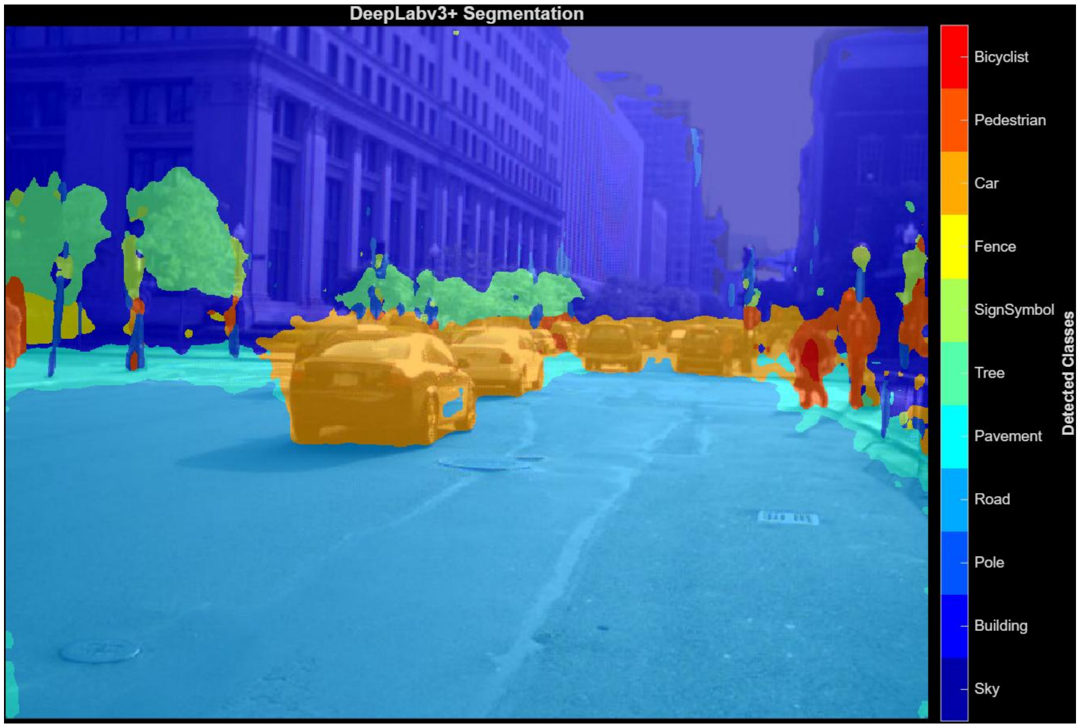




Command Window

```
Loading Network...
Network Loaded.
Warning: Could not auto-read classes from network. Using standard CamVid labels.
Success: Identified 11 classes.
Segmenting image...
-----------------------------------
Classes detected in this specific image:
- C1
- C2
- C3
- C4
- C5
- C6
- C7
- C8
- C9
- C10
- C11
-----------------------------------
```

Clearer Image view (Original & DeepLabV3+ Result):

Semantic Segmentation Result (Original & DeepLabV3+)



Discussion:

Based on the result, the DeepLabv3+ model **did a solid job identifying** the main parts of the scene. For example, the road (cyan), buildings (dark blue), trees (green), and the car in front (orange). It even told the gray road apart from the gray buildings, which something simple brightness-based methods would mess up. But **it's not perfect**. Small, far-away objects like **pedestrians and background cars look blurry and lose detail,** which makes the model made a **mistake on pedestrian as a bicyclist**. The tree edges also get rough where they meet the sky. This really shows the difference from older techniques. Methods like **Thresholding** or **Region Growing** just group pixels by color or brightness, which would wrongly lump the similar gray road and buildings together. Semantic segmentation, on the other hand, uses AI to understand context and shape, giving every pixel a meaningful label.

Short Reflection on Comparing Classical VS Deep Learning Segmentation:

Classical techniques (like Thresholding) are simple filters that just look at pixel brightness. They're fast and work great for clean, high-contrast images (like rice grains on a black background). But they don't actually *understand* what they see. They'd lump a gray road and gray building together because the colors match. They also get tripped up by noise, shadows, or busy textures, and you have to constantly tweak settings.

Deep learning (like DeepLabv3+) teaches a computer to see. It learns from thousands of examples and gets context. It correctly split the road from the building because it knows roads are flat and horizontal while buildings are vertical (even if both are gray). It's more demanding (needs powerful hardware and huge datasets), but it's far more robust. Instead of just saying "object" or "no object," it tells you exactly what it sees: "car," "tree," "pedestrian."

Feature	Classical Methods (Thresholding/Edges)	Deep Learning (Semantic Segmentation)
Basis	Math (Brightness, Contrast, Gradient)	Data (Learned Patterns & Features)
Knowledge	Binary: Foreground vs. Background	Semantic: Car, Road, Sky, Person
Best For	Controlled, simple scenes (Microscopy, Industry)	Uncontrolled, complex scenes (Self-driving cars, Medical AI)
Weakness	Sensitive to noise and lighting shadows	Requires powerful GPU & training data
Lab Example	Counting Rice / Saturn Rings	Identifying Cars in a City Street