# Image Processing and Analysis

# Assignment 0: Create your own dataset of labeled images

Prepared by Prof. Oge Marques, PhD, Professor of Engineering and Computer Science at   
Florida Atlantic University - <http://www.eng.fau.edu/directory/faculty/marques/>

**Goal:** Learn how to use MATLAB® for labeling images and creating your own dataset for training and testing image segmentation solutions.

**Learning objectives:**

* Learn how to label your own images using MATLAB Image Labeler
* Learn how the image labeling process fits within a typical semantic image segmentation workflow in MATLAB

**Instructions:**

You might want to start by watching the 8-min *Semantic Segmentation Overview* video at: <https://www.mathworks.com/videos/semantic-segmentation-overview-1510858047780.html>

In that video, the semantic segmentation (using deep learning) workflow is described in five main steps (Figure 1):

A picture containing bird

Description automatically generated

*Figure 1: Image semantic segmentation workflow.*

This document focuses on *Step 1: Label your data*.   
We will break it down into several steps, as described in the Procedure below.

**Procedure:**

1. **Define your problem**.
   * What is the context?
   * How do images look like? Where do they come from?
   * What are the relevant portions within the images? How should they be labeled?
   * What is the goal of applying semantic segmentation techniques to those images?
   * What will the semantic segmentation results be used for?
   * How can you measure the quality of the solution?
2. **Collect useful images** and organize them in a folder.
3. **Prepare a list of labels** (pen-and-paper is fine if you only have a few labels, otherwise use a text file or spreadsheet) that will be used to label selected regions of interest (ROIs) within each image.
4. **Start MATLAB**.
5. **Open the Image Labeler app** by double-clicking on its icon (under the APPS tab) (Figure 2).

A screenshot of a cell phone

Description automatically generated

*Figure 2: Image Labeler semantic segmentation workflow.*

1. Click on the top-left button to **start a new session** (Figure 3).
2. **Load images** (either from a folder in your computer or from the MATLAB workspace).

A screenshot of a cell phone

Description automatically generated

*Figure 3: Image Labeler App (partial) screenshot: useful buttons for steps 6 and 7.*

1. From your list of labels, **add the ROI labels**, one at a time. (Optionally, you can import labels from the workspace or from a .mat file).   
   Note: for semantic image segmentation tasks, you will probably choose the *Pixel label* option. The other two options (*Rectangle label* and *Line label*) are useful for other types of computer vision tasks (e.g., object detection and tracking) [2].

A screenshot of a cell phone

Description automatically generated

*Figure 4: Image Labeler App (partial) screenshot: examples of ROI labels. The names were chosen by me, based on the context of the task (segmentation of desktop scenes into their most common objects, see examples of image in Figure 5). The colors were picked automatically by MATLAB (but could be overridden) and the gray symbol to the right of the color indicates that they are Pixel Labels.*

|  |  |  |
| --- | --- | --- |
| *A desk with a computer mouse and keyboard  Description automatically generated* | *A desktop computer sitting on top of a table  Description automatically generated* | *A desk with a computer monitor sitting on top of a table  Description automatically generated* |

*Figure 5: Examples of images for semantic segmentation of desktop scenes into their most common objects.*

1. For each image, **draw the segmentation mask** in a way as to cover every pixel belonging to the ROI, as precisely as you can.

Hint: use zooming, retouching (brush, erase), and (smart) polygon tools, if appropriate.

1. Depending on the segmentation task, you might want to use **an algorithm to help automate the process**. MATLAB Image Labeler has built-in automation algorithms (e.g., pedestrian detector) and also allows you to add your own custom automation algorithm [1].
2. **Export the labels to the MATLAB workspace**. This will trigger the creation of a collection of mask images (in the *PixelLabelData* folder) and a ground truth MAT-file (see figure below).

A screenshot of a cell phone

Description automatically generated

1. From this point on, you can implement your segmentation workflow in MATLAB, i.e., steps 2 through 5 from Figure 1).
2. (OPTIONAL) Explore other MATLAB tools for Ground Truth labeling for computer vision deep learning applications:
   * *Ground Truth Labeler* app
   * *Video Labeler* app
3. (OPTIONAL) Prepare a **report**, with all relevant steps, screenshots, and – most importantly – your insights and lessons learned.

**References**

* **Videos**
  + *How to Label Data for Deep Learning* (focuses on the *Video Labeler* app):  
    <https://youtu.be/V2e0cygY9Vg>
  + *Labeling Ground Truth for Object Detection* (focuses on the *Ground Truth Labeler* app):   
    <https://youtu.be/ow_B_30WU1s>
* **MATLAB documentation**
  + *Image Labeler* app:<https://www.mathworks.com/help/vision/ref/imagelabeler-app.html>
  + *Ground Truth Labeler* app:<https://www.mathworks.com/help/driving/ref/groundtruthlabeler-app.html>
  + *Video Labeler* app:<https://www.mathworks.com/help/vision/ref/videolabeler-app.html>

1. <https://www.mathworks.com/help/vision/ug/choose-a-labeling-app.html>
2. <https://www.mathworks.com/help/vision/ug/get-started-with-the-image-labeler.html>

# Notes for instructors

1. When explaining the first step in the Procedure, an example could be:
   1. What is the context?   
      Self-driving cars.
   2. How do images look like?   
      Scenes of streets and roads.   
      Where do they come from?   
      Dashboard cameras, publicly available datasets (e.g., CamVid), etc.
   3. What are the relevant portions within the images?   
      Pedestrians, road, traffic lights, trees, sidewalk, etc. (see figure below).   
      How should they be labeled?   
      They should be labeled in accordance to some agreed-upon terminology and a definition of the number of relevant classes to train and test the algorithm on.   
      For example, one could ask: 'TrafficSign' or 'SignSymbol'? Should a 'StopSign' be its own category/label?
   4. What is the goal of applying semantic segmentation techniques to those images?  
      To classify each pixel in the image according to the category of the ROI that it belongs to.
   5. What will the semantic segmentation results be used for?

In this case, to train, validate, and test self-driving vehicles' abilities to pay attention to the portion of the scene that is relevant, and (re)act accordingly.

* 1. How can you measure the quality of the solution?  
     By comparing the results of the semantic segmentation algorithm against pixel-level ground truth, such as the one available in some datasets, or created by ourselves using the tools/steps described in this document and associated screencast.

