Automated Driving
with MATLAB® and Simulink®

Marco Roggero
How can you use MATLAB and Simulink to develop automated driving algorithms?

- Perception
- Control
- Planning
Examples of how you can use MATLAB and Simulink to develop automated driving algorithms

- Deep learning
- Sensor models & model predictive control
- Sensor fusion with live data
- Path planning

Perception | Control | Planning
How can you use MATLAB and Simulink to develop perception algorithms?

- Perception
  - Deep learning
  - Sensor fusion with live data

- Control
  - Sensor models & model predictive control

- Planning
  - Path planning
Automated Driving System Toolbox introduced:
Ground Truth Labeling App to label video data
Automate labeling lanes with Ground Truth Labeler

Run automation algorithm
Specify attributes and sublabels in Ground Truth Labeler App
Automate labeling pixels with Ground Truth Labeler
Learn how to train a deep learning network using this example

- Train free space detection network using deep learning
  
  *Computer Vision System Toolbox™*
Load and plot training images

% Create datastore for images
imds = imageDatastore(imgDir);
I = readimage(imds, 1);
I = histeq(I);
imshow(I)
Load and overlay pixel labels

% Load pixel labels
classes = ["Sky"; "Building";...
"Pole"; "Road"; "Pavement"; "Tree";...
"SignSymbol"; "Fence"; "Car";...
"Pedestrian"; "Bicyclist"];
pxds = pixelLabelDatastore(...
    labelDir,classes,labelIDs);

% Display labeled image
C = readimage(pxds, 1);
cmap = camvidColorMap;
B = labeloverlay(I,C,'ColorMap',cmap);
imshow(B)

pixelLabelDatastore manages large collections of pixel labels
Visualize distribution of labeled pixels

```matlab
% Visualize label count by class
tbl = countEachLabel(pxds)

frequency = tbl.PixelCount / ... sum(tbl.PixelCount);

bar(1:numel(classes),frequency)
xticks(1:numel(classes))
xticklabels(tbl.Name)
xtickangle(45)
ylabel('Frequency')
```

Labeled pixels in this set are imbalanced
Create and visualize baseline network

% Create SegNet architecture
lgraph = segnetLayers(...
    imageSize, numClasses,...
    'vgg16');

% Display network structure
plot(lgraph)
title('Complete Layer Graph')

% Display last layers
plot(lgraph); ylim([0 9.5])
title('Last 9 Layers Graph')
Add weighted layer to compensate for imbalanced data set

```matlab
% Create weighted layer
pxLayer = pixelClassificationLayer(...
    'Name','weightedLabels', 'ClassNames',tbl.Name,...
    'ClassWeights',classWeights)
```

Last network layer
Add weighted layer to compensate for imbalanced data set

% Create weighted layer
pxLayer = pixelClassificationLayer(...
    'Name','weightedLabels', 'ClassNames',tbl.Name,...
    'ClassWeights',classWeights)

% Replace layer
lgraph = removeLayers(lgraph, 'pixelLabels');
lgraph = addLayers(lgraph, pxLayer);
lgraph = connectLayers(lgraph,...
    'softmax', 'weightedLabels');

% Display network structure
plot(lgraph); ylim([0 9.5])
title('Replaced Layers Graph')
Augment images to expand training set

```matlab
augmenter = imageDataAugmenter(...
    'RandXReflection', true,...
    'RandRotation', [-30 30],..., % degrees
    'RandXTranslation',[10 10],..., % pixels
    'RandYTranslation',[10 10]); % pixels

datasource = pixelLabelImageSource(...
    imdsTrain,... % Image datastore
    pxdsTrain,... % Pixel datastore
    'DataAugmentation',augmenter)
```
Deep learning on CPU, GPU, multi-GPU and clusters

```matlab
options = trainingOptions('sgdm', ... 'InitialLearnRate', 1e-3, ... 'MaxEpochs', 100, ... 'MiniBatchSize', 4, ... 'Shuffle', 'every-epoch', ... 'VerboseFrequency', 2, ... 'ExecutionEnvironment', ['auto']);
```

- Single CPU: 'auto'
- Single GPU: 'auto'
- Multiple GPUs: 'multi-gpu'
- On-prem server with GPUs: 'parallel'
- Cloud GPUs (AWS, Azure, etc.): 'parallel'
Train network and view progress

```
[net, info] = trainNetwork(datasource, lgraph, options);
```
Evaluate trained network on image

% Plot actual results
I = read(imdsTest);
actual = semanticseg(I, net);
B = labeloverlay(I, ...
   actual,...
    'Colormap', cmap,...
    'Transparency',0.4);
imshow(B)
pixelLabelColorbar(cmap, classes);
title('Actual')
Visually compare actual with original labeled results

% Plot expected results
% using original labels
expected = read(pxdsTest);
E = labeloverlay(I,...
   expected,...
   'Colormap', cmap,...
   'Transparency',0.4);
imshow(E)
title('Expected');
Visually compare actual with original labeled results

```matlab
% Plot differences
imshowpair(...
    uint8(actual),...
    uint8(expected));
title('Difference');
```
Assess similarity using intersection-over-union (IoU) metric

\[
iou = \text{jaccard}(\text{actual}, \ldots, \text{expected});
\]

\[
\text{table(\text{classes}, \text{iou})}
\]

\[
\text{ans} = 11 \times 2 \text{ table}
\]

<table>
<thead>
<tr>
<th>classes</th>
<th>iou</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Sky&quot;</td>
<td>0.92659</td>
</tr>
<tr>
<td>&quot;Building&quot;</td>
<td>0.7987</td>
</tr>
<tr>
<td>&quot;Pole&quot;</td>
<td>0.16978</td>
</tr>
<tr>
<td>&quot;Road&quot;</td>
<td>0.95177</td>
</tr>
<tr>
<td>&quot;Pavement&quot;</td>
<td>0.41877</td>
</tr>
<tr>
<td>&quot;Tree&quot;</td>
<td>0.43401</td>
</tr>
<tr>
<td>&quot;SignSymbol&quot;</td>
<td>0.32509</td>
</tr>
<tr>
<td>&quot;Fence&quot;</td>
<td>0.492</td>
</tr>
<tr>
<td>&quot;Car&quot;</td>
<td>0.068756</td>
</tr>
<tr>
<td>&quot;Pedestrian&quot;</td>
<td>0</td>
</tr>
<tr>
<td>&quot;Bicyclist&quot;</td>
<td>0</td>
</tr>
</tbody>
</table>
pxdsResults = ...

\texttt{semanticseg}(... \\
\texttt{imdsTest,net,... \\
'WriteLocation',tempdir,... \\
'Verbose',false);}

metrics = ...

\texttt{evaluateSemanticSegmentation}(... \\
pxdsResults,pxdsTest,... \\
'Verbose',false);}

metrics.ClassMetrics

\textbf{Evaluation metrics of network}

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>IoU</th>
<th>MeanBFScore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sky</td>
<td>0.93544</td>
<td>0.89279</td>
<td>0.88239</td>
</tr>
<tr>
<td>Building</td>
<td>0.79978</td>
<td>0.75543</td>
<td>0.59861</td>
</tr>
<tr>
<td>Pole</td>
<td>0.73166</td>
<td>0.18361</td>
<td>0.51426</td>
</tr>
<tr>
<td>Road</td>
<td>0.93644</td>
<td>0.90663</td>
<td>0.7086</td>
</tr>
<tr>
<td>Pavement</td>
<td>0.90624</td>
<td>0.72932</td>
<td>0.70585</td>
</tr>
<tr>
<td>Tree</td>
<td>0.86587</td>
<td>0.73694</td>
<td>0.67097</td>
</tr>
<tr>
<td>SignSymbol</td>
<td>0.76118</td>
<td>0.35339</td>
<td>0.44175</td>
</tr>
<tr>
<td>Fence</td>
<td>0.83258</td>
<td>0.49648</td>
<td>0.50265</td>
</tr>
<tr>
<td>Car</td>
<td>0.90961</td>
<td>0.75263</td>
<td>0.64837</td>
</tr>
<tr>
<td>Pedestrian</td>
<td>0.83751</td>
<td>0.35409</td>
<td>0.46796</td>
</tr>
<tr>
<td>Bicyclist</td>
<td>0.84156</td>
<td>0.5472</td>
<td>0.46933</td>
</tr>
</tbody>
</table>
Distribution of labels in data affects intersection-over-union (IoU)

Underrepresented classes such as Pedestrian and Bicyclist are not segmented as well as classes such as Sky and Road.
% Save network to MAT file
save('SegNet.mat', 'net')

function out = segnet_predict(in) %#codegen
persistent mynet;
if isempty(mynet)
    mynet = coder.loadDeepLearningNetwork('SegNet.mat');
end
out = predict(mynet, in);

% Generate CUDA code
cfg = coder.config('lib');
cfg.TargetLang = 'C++';
codegen -config cfg segnet_predict -args
{ones(360, 480, 3, 'uint8')} -report
Free Space Detection Using Semantic Segmentation
Learn more about developing deep learning perception algorithms with these examples

- **Train free space detection network** using deep learning
  *Computer Vision System Toolbox™*

- **Generate CUDA® code** to execute directed acyclic graph network on an NVIDIA GPU
  *GPU Coder™*

- **Add semantic segmentation automation algorithm to** Ground Truth Labeler App
  *Automated Driving System Toolbox™*
Learn about developing lidar perception algorithms with these examples

- **Read Velodyne** files
  - `velodyneFileReader`
  - Automated Driving System Toolbox™

- **Register** point clouds with Normal Distributions Transform
  - `pcregisterndt`
  - Computer Vision System Toolbox™

- **Segment** lidar point cloud
  - `segmentLidarData`
  - Automated Driving System Toolbox™
How can you use MATLAB and Simulink to develop perception algorithms?

- Deep learning
- Sensor fusion with live data
- Perception
- Control
- Planning
- Sensor models & model predictive control
- Path planning
- Control Planning
Automated Driving System Toolbox introduced: Multi-object tracker to develop sensor fusion algorithms

- Assigns detections to tracks
- Creates new tracks
- Updates existing tracks
- Removes old tracks

- Predicts and updates state of track
- Supports linear, extended, and unscented Kalman filters
Automated Driving System Toolbox introduced examples to:

Develop sensor fusion algorithms with recorded data

- Design multi-object tracker based on logged vehicle data
- Generate C/C++ code from algorithm which includes a multi-object tracker
How can I test my sensor fusion algorithm with live data?
Test forward collision warning algorithm with live data from vehicle

- Radar
- Camera
- IMU
- CAN
- CAN Tx
- CAN FD
- CAN Rx
- TCP/IP
- Ethernet
- Vision Object
- Radar Object
- Lane
- Vehicle Speed
- Yaw Rate
- Video frame
- FCW application
  - Read sensor data stream and video stream
  - FCW algorithm
  - Visualization
Test forward collision warning algorithm with live data from “surrogate” vehicle
Send and live CAN FD and TCP/IP data
Receive live CAN FD and TCP/IP data
Generate C/C++ code for algorithm
Stream live CAN FD and TCP/IP data into compiled algorithm code

Change “interpreted” to “software in the loop” arrow toward end of video

Algorithm uses 1’s of msec in software-in-the-loop mode
Learn about developing sensor fusion algorithms with live data using this example

- **Stream CAN FD** data to prototype algorithms on your laptop

  Vehicle Network Toolbox™
How can you use MATLAB and Simulink to develop control algorithms?

Deep learning

Perception

Control

Sensor models & model predictive control

Sensor fusion with live data

Planning

Path planning
Automated Driving System Toolbox introduced examples to:

Synthesize detections to test sensor fusion algorithms

- Synthesize radar detections with probabilistic impairments
- Synthesize vision detections with probabilistic impairments
- Synthesize scenario to test multi-object tracker
Simulate closed loop system with radar/vision detections, sensor fusion, and model-predictive control.
Synthesize detections to test sensor fusion and model-predictive controller
How can MPC be applied to lane keeping control?

References
1. For $E_{lateral}$ (0)
2. For $E_{yaw}$ (0)

Measured outputs
1. Lateral deviation ($E_{lateral}$)
2. Relative yaw angle ($E_{yaw}$)

Manipulated variable
1. Steering angle ($u$)

Measured disturbance
1. Previewed curvature

minimize: $|E_{lateral}|^2 + |E_{yaw}|^2$
subject to: $u_{min} \leq u \leq u_{max}$
Vision Detection Generator models lane detection sensor

![Diagram of Vision Detection Generator](image.png)

**Block Parameters: Vision Detection Generator**

- **Sensor Identification**
  - Unique Identifier of sensor: 1

- **Types of detections generated by sensor:**
  - Lanes and objects

- **Required interval between sensor updates (s):**

- **Required Interval between lane detection updates (s):**
  - Lanes and occlusion

**Sensor Extrinsics**

- **Sensor's (x,y) position (m):** [1.9, 0]
- **Sensor's height (m):** 1.1
- **Yaw angle of sensor mounted on ego vehicle (deg):** 0
- **Pitch angle of sensor mounted on ego vehicle (deg):** 1
- **Roll angle of sensor mounted on ego vehicle (deg):** 0

**Source code**
Create highway double curve with drivingScenario

- Driver waypoints simulate distraction at curvature changes
Simulate distracted driver
Simulate lane keep assist at distraction events
Compare distracted and assisted results

- Detect lane departure and maintain lane during distraction
Detect departure based on lateral offset to lane boundary

Max Safe Lateral Distance from Lane Boundary
Simulate lane following by increasing minimum safe distance
Explore lane following results

- Vehicle stays within lane boundaries
Graphically edit scenarios with Driving Scenario Designer
Explore what is required to follow high curvature paths.
Learn about synthesizing sensor detections to develop control algorithms with these examples

- **Simulate and generate C++** for model-predictive control and sensor fusion algorithms
- **Simulate and generate C++** for model-predictive control with lane detections
- **Edit roads, cuboid actors, and sensors** with Driving Scenario Designer App

```
drivingScenarioDesigner
```

Generate Synthetic Detections from an Interactive Driving Scenario
Learn about modeling vehicle dynamics to develop control algorithms with these examples

- **Simulate vehicle dynamics** for closed loop design
  Vehicle Dynamics Blockset™

- **Co-simulate with Unreal Engine** and to set actor positions get camera image
  Vehicle Dynamics Blockset™
How can you use MATLAB and Simulink to develop planning algorithms?

- Deep learning
- Sensor models & model predictive control
- Perception
- Control
- Sensor fusion with live data
- Path planning
Robotics System Toolbox introduced: Connectivity with the ROS ecosystem

- Communicate via ROS to integrate with externally authored ROS components
- Communication with Gazebo to visualize and simulated system
- Follow path for differential drive robot with ROS based simulator
We are investing in design and simulation of path planning for automobiles

Motion planning:
Plan path to next waypoint (RRT*)

Rapidly-exploring Random Tree (RRT*)
Learn about developing path planning algorithms with these examples

- **Plan path** for automobile given pre-defined map
  Automated Driving System Toolbox™

- **Plot map tiles** using World Street Map (Esri)
  Automated Driving System Toolbox™

- **Simulate V2X communication** to assess channel throughput
  LTE System Toolbox™
Examples of how you can use MATLAB and Simulink to develop automated driving algorithms

Perception

- Deep learning
- Sensor fusion with live data

Control

- Sensor models & model predictive control

Planning

- Path planning
MathWorks can help you customize MATLAB and Simulink for your automated driving application

- Web based ground truth labeling
  - Consulting project with Caterpillar
  - [2017 MathWorks Automotive Conference](#)

- Lidar ground truth labeling
  - Joint customer presentation
  - [2018 MathWorks Automotive Conference](#)

- Lidar sensor model for Unreal Engine
  - Joint paper with Ford
  - SAE Paper 2017-01-0107
How can we help you can use MATLAB and Simulink to develop automated driving algorithms?