MATLAB EXPO 2019

Pixels to Features to Models

Object Detection and Image Segmentation

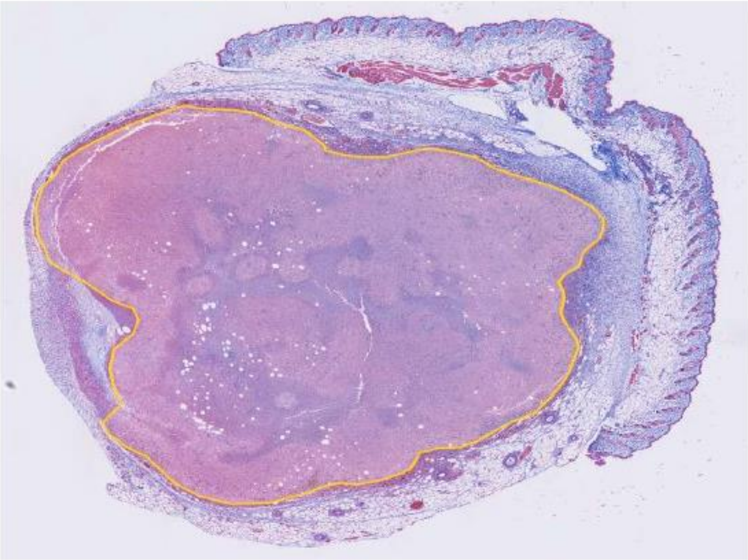
Matt Elliott

Agenda

- Introduction
  - Applications
  - Computer vision tasks
  - Choosing an approach
- Examples
  - ‘Traditional’ image processing
  - Deep learning
- Getting started

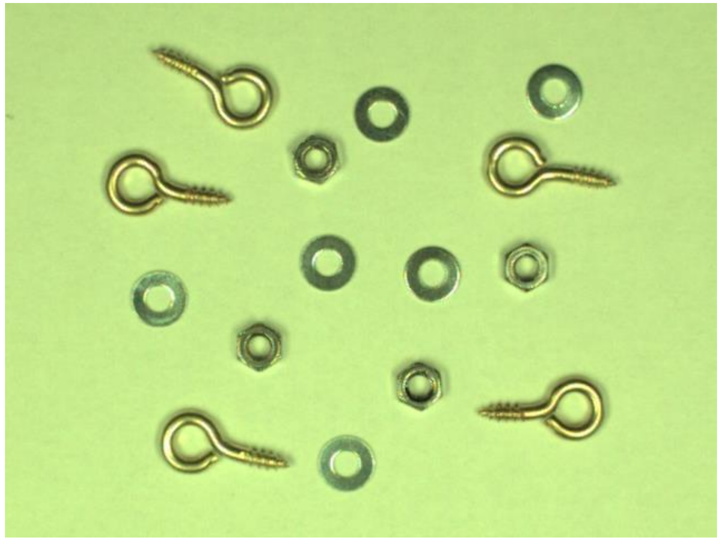
Computer Vision Tasks

Automated Driving



Medical Imaging

Manufacturing



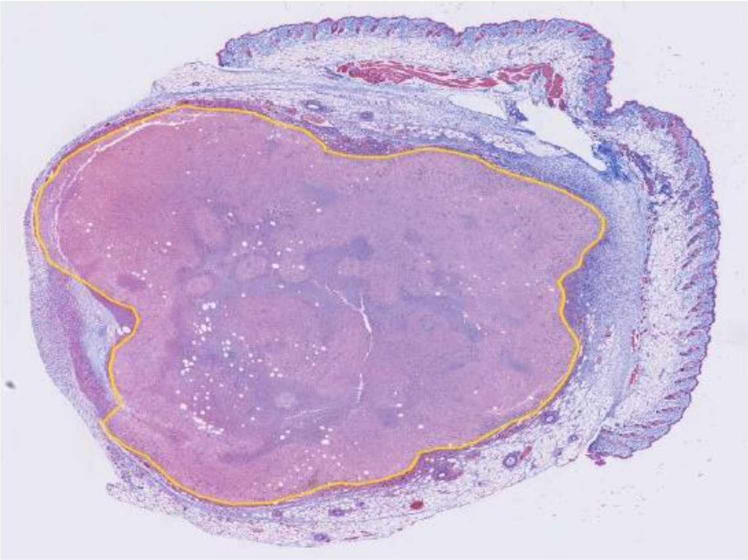
Computer Vision Tasks

Where are the cars?

Where can I drive?

How many parts?

Are they damaged?

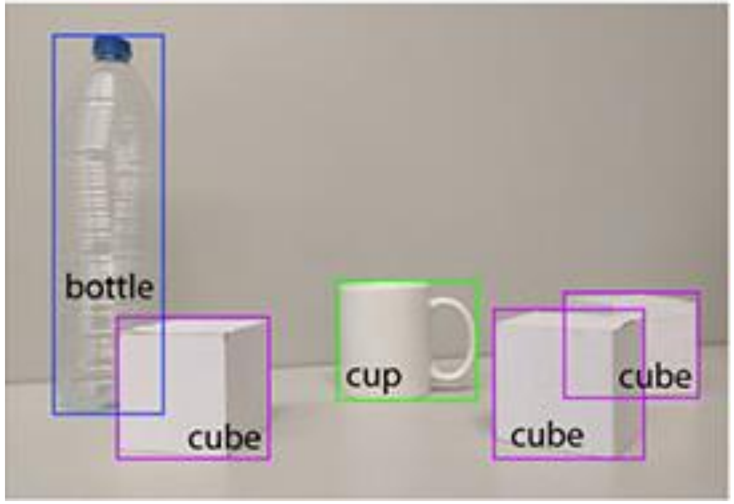


Is this a tumour?

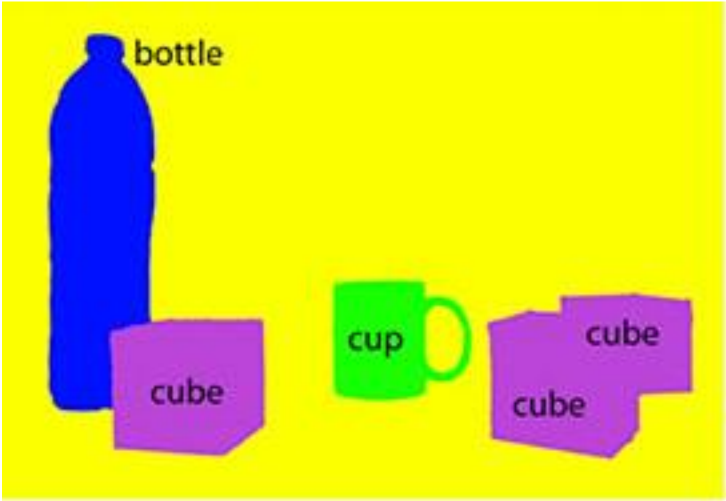
How large is it?

Computer Vision Tasks

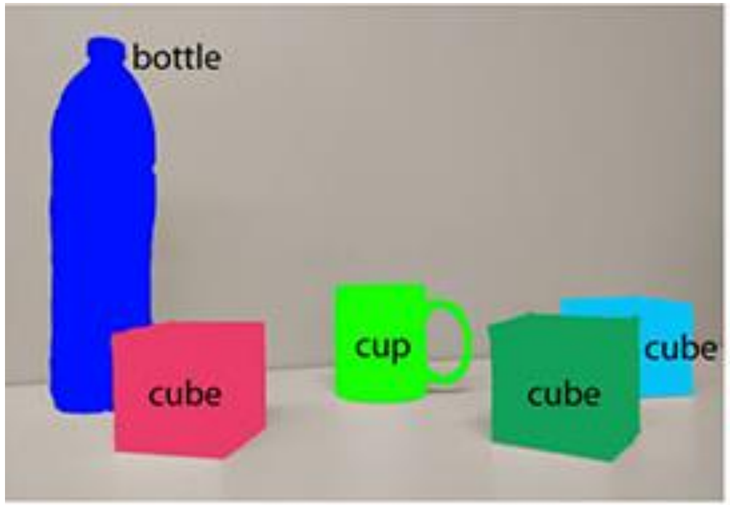
Image classification



Object detection



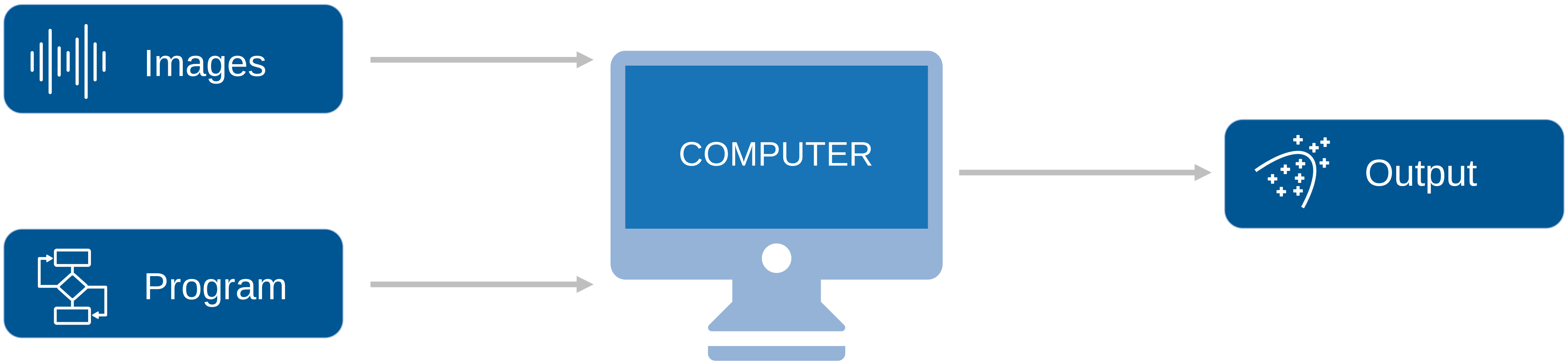
Semantic segmentation



Instance segmentation

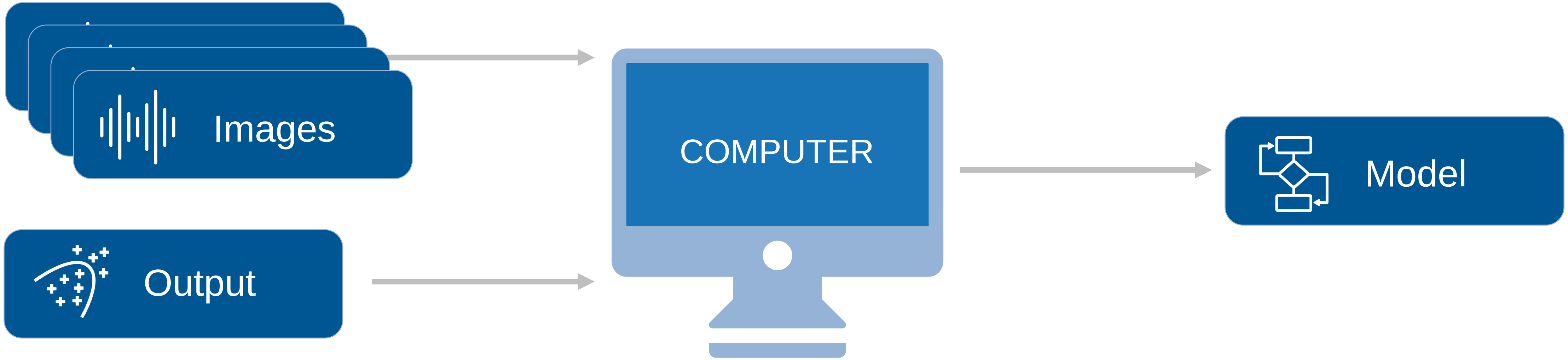
Adapted from arXiv:1704.06857

Two approaches to computer vision



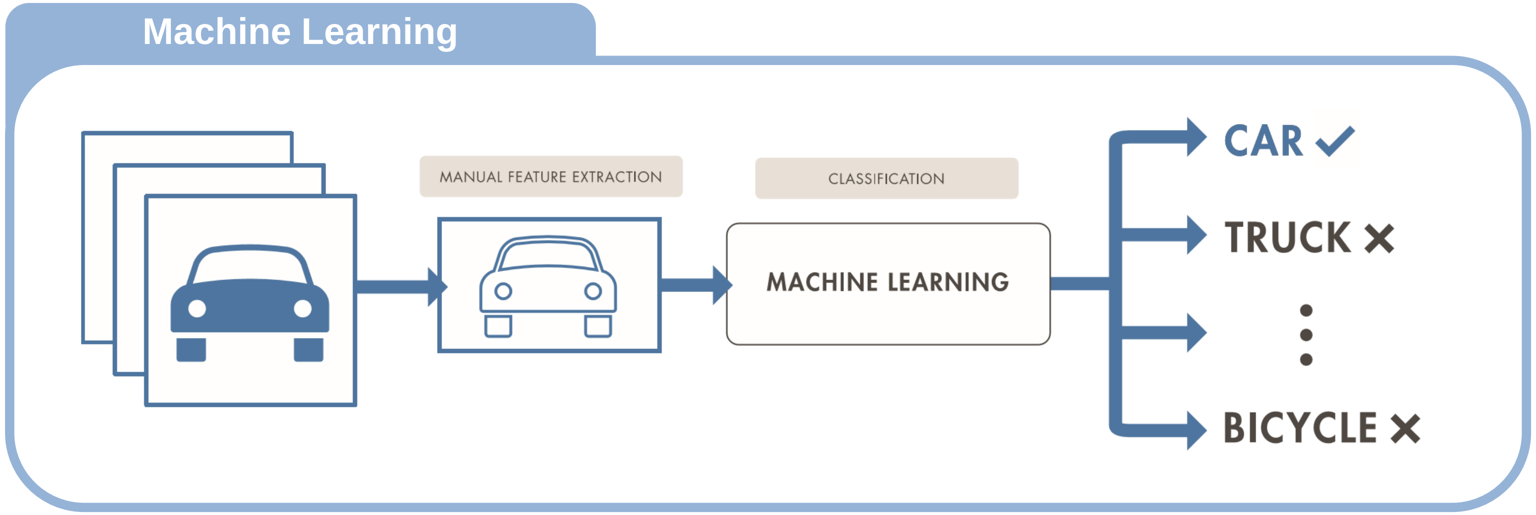
‘Traditional’ Image Processing

Two approaches to computer vision

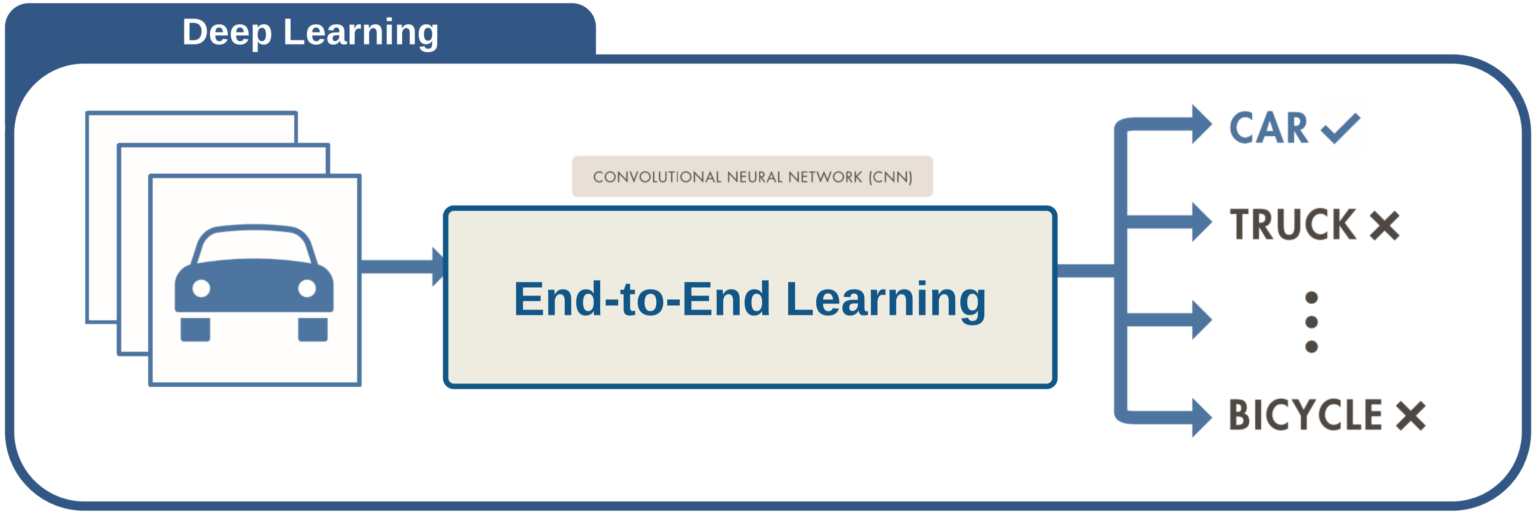


Machine Learning

Machine Learning v Deep Learning



Machine Learning learns tasks using manually extracted **features**



Deep Learning learns both features and tasks directly from data

Examples

Two examples to demonstrate these approaches:

1. Traditional image processing for segmentation
2. Deep learning for object detection

Example 1: Part Inspection

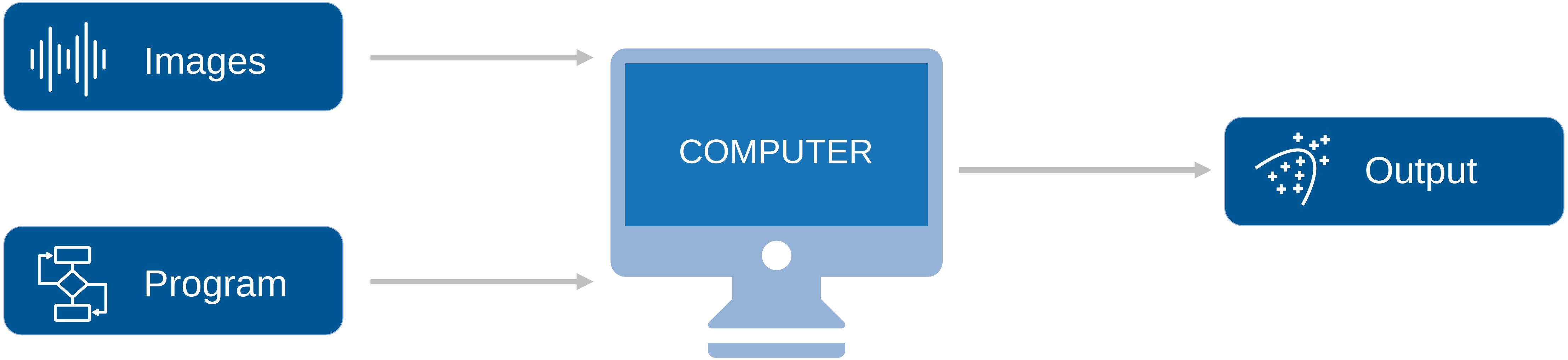
Challenge:

- Find all of the items in the image
- Classify them - hook, nut or washer

Data

- Small number of images, unlabelled
- Taken from fixed position, controlled lighting

Two approaches to computer vision



'Traditional' Image Processing

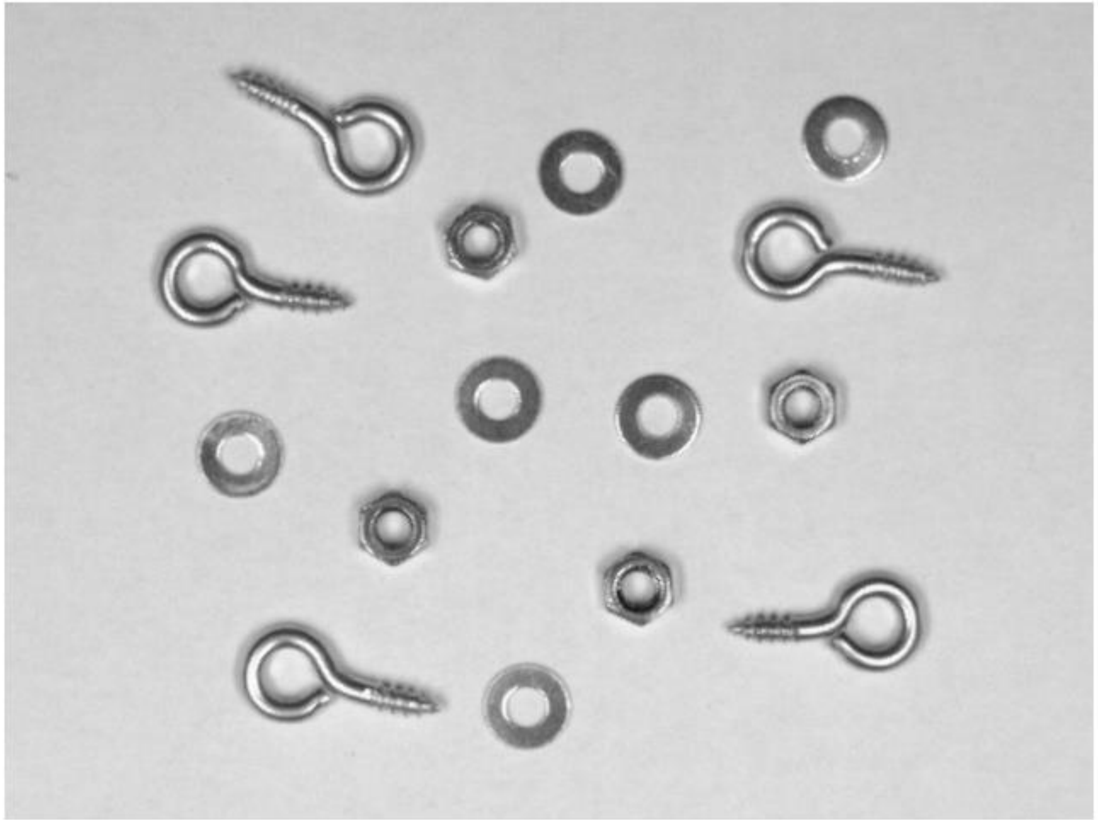
Load Image

Import image

```
I = imread('nutsAndBolts.png');
```

Convert to grayscale

```
Igray = rgb2gray(I);
imshow(Igray)
```



Segmentation

- Next we want to segment the image

- Two routes:
  - Writing MATLAB code
  - Apps (and then generating code)

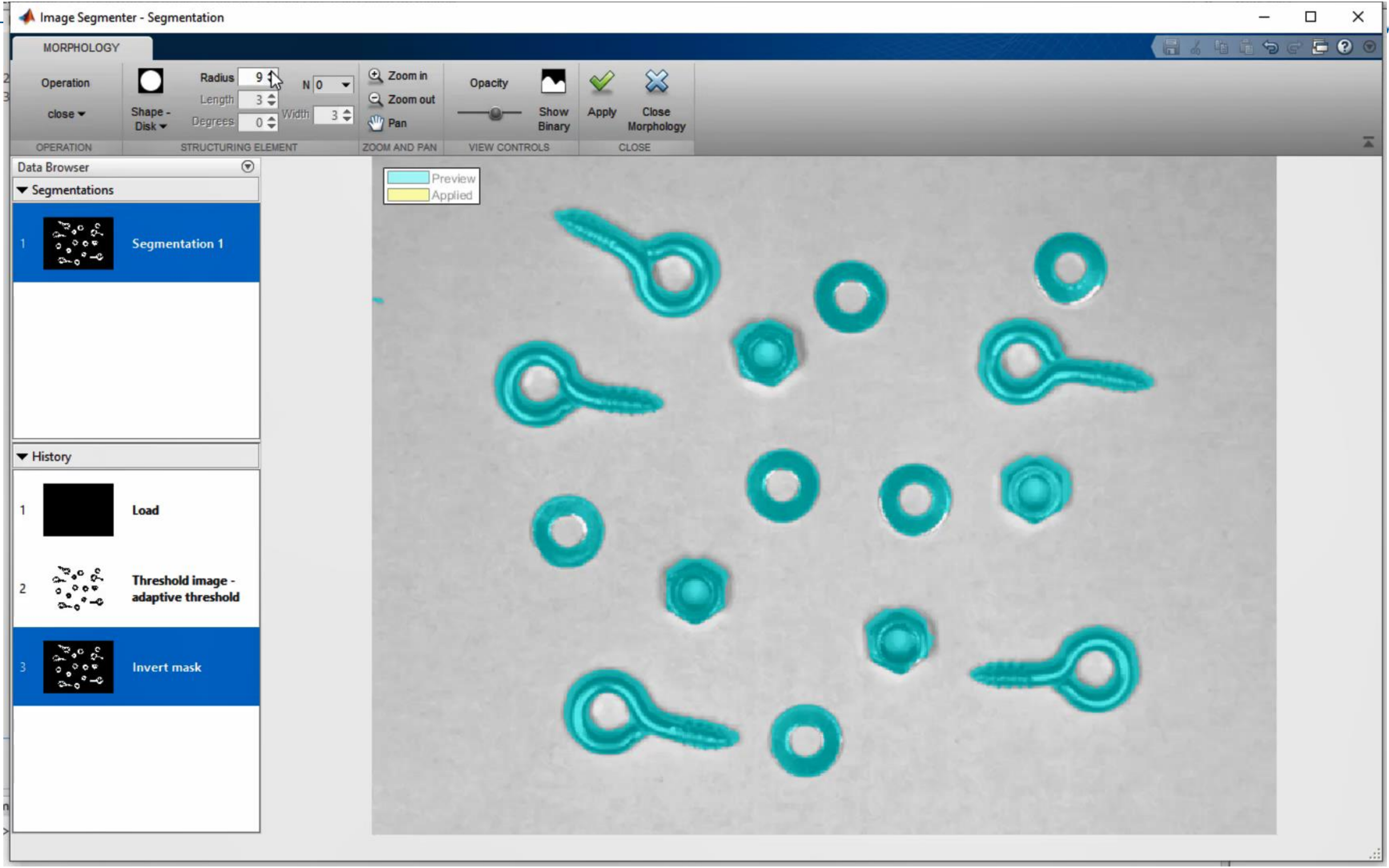
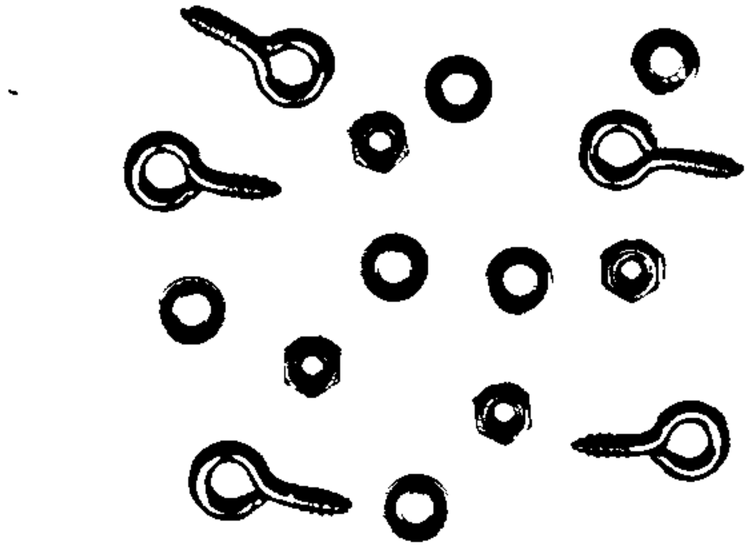


Image Segmenter - Segmentation
MORPHOLOGY
Operation
close
Shape - Disk
Radius 9
Length 3
Degrees 0
N 0
Width 3
Zoom in
Zoom out
Pan
Opacity
Show Binary
Apply
Close Morphology
OPERATION
STRUCTURING ELEMENT
ZOOM AND PAN
VIEW CONTROLS
CLOSE
Data Browser
Segmentations
1 Segmentation 1
History
1 Load
2 Threshold image - adaptive threshold
3 Invert mask
Preview
Applied

In code

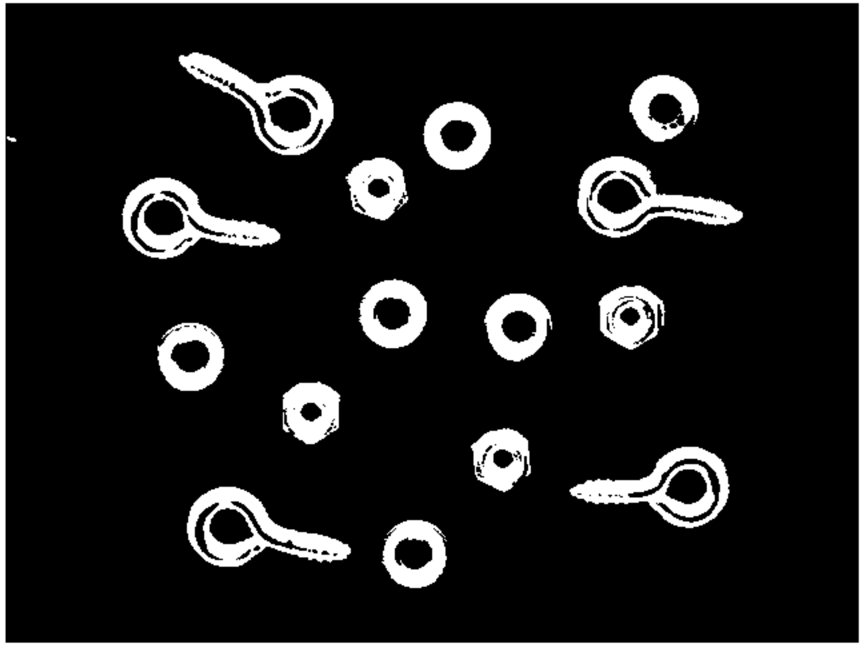
```
BW = imbinarize(Igray, 'adaptive', 'Sensitivity', 0.52,...
    'ForegroundPolarity', 'dark');
```



In code

```
BW = imbinarize(Igray, 'adaptive', 'Sensitivity', 0.52,...
    'ForegroundPolarity', 'dark');
```

```
BW = imcomplement(BW);
```

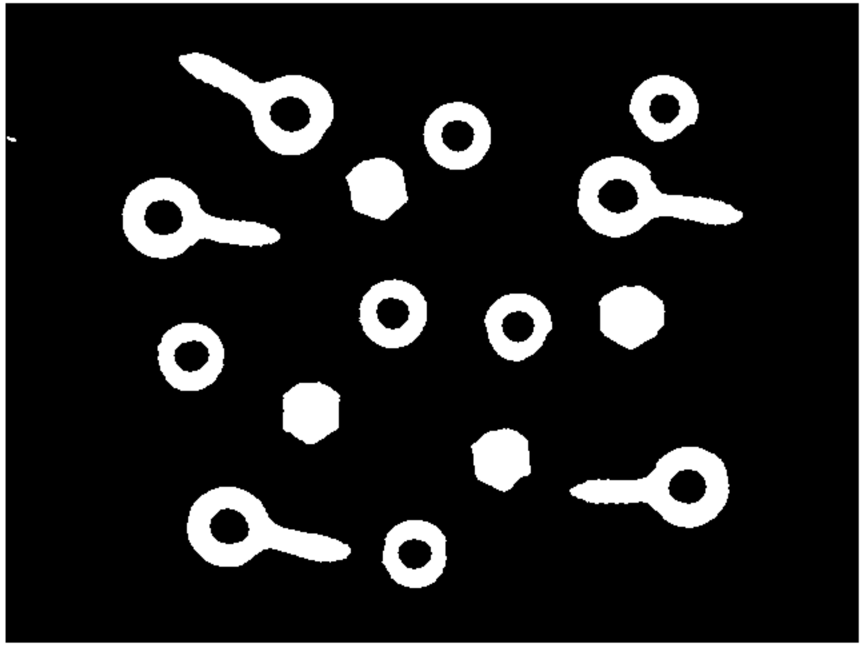


In code

```
BW = imbinarize(Igray, 'adaptive', 'Sensitivity', 0.52,...
    'ForegroundPolarity', 'dark');
```

```
BW = imcomplement(BW);
```

```
radius = 9;
decomposition = 0;
se = strel('disk', radius, decomposition);
BW = imclose(BW, se);
```



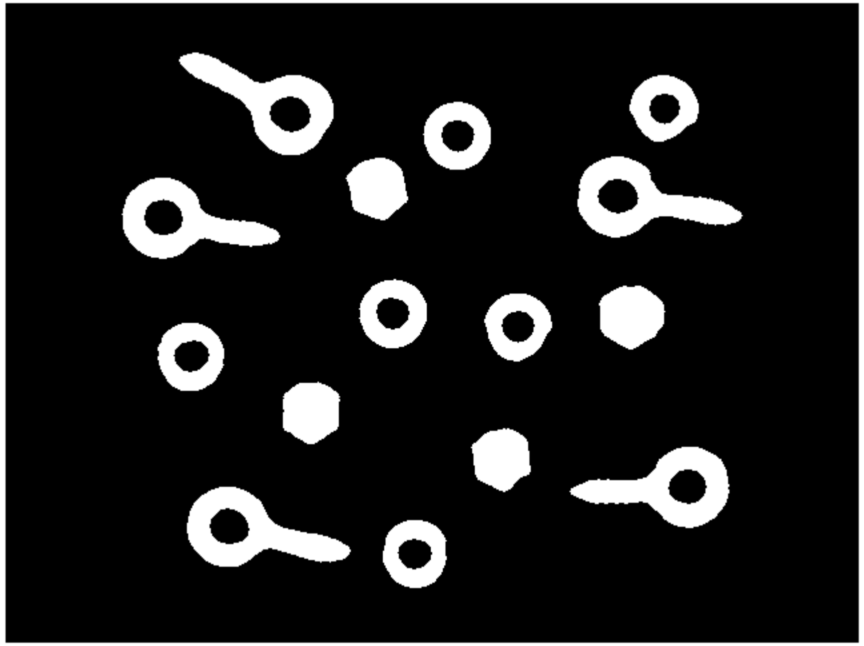
In code

```
BW = imbinarize(Igray, 'adaptive', 'Sensitivity', 0.52,...
    'ForegroundPolarity', 'dark');
```

```
BW = imcomplement(BW);
```

```
radius = 9;
decomposition = 0;
se = strel('disk', radius, decomposition);
BW = imclose(BW, se);
```

```
se2 = strel('disk', 3, 0);
BW = imopen(BW, se2);
```



In code

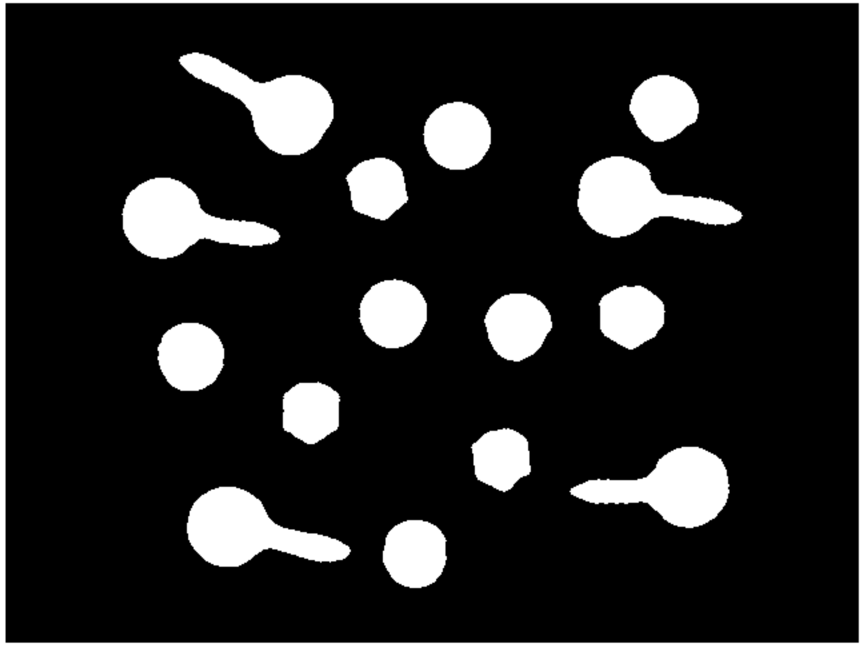
```
BW = imbinarize(Igray, 'adaptive', 'Sensitivity', 0.52,...
    'ForegroundPolarity', 'dark');
```

```
BW = imcomplement(BW);
```

```
radius = 9;
decomposition = 0;
se = strel('disk', radius, decomposition);
BW = imclose(BW, se);
```

```
se2 = strel('disk', 3, 0);
BW = imopen(BW, se2);
```

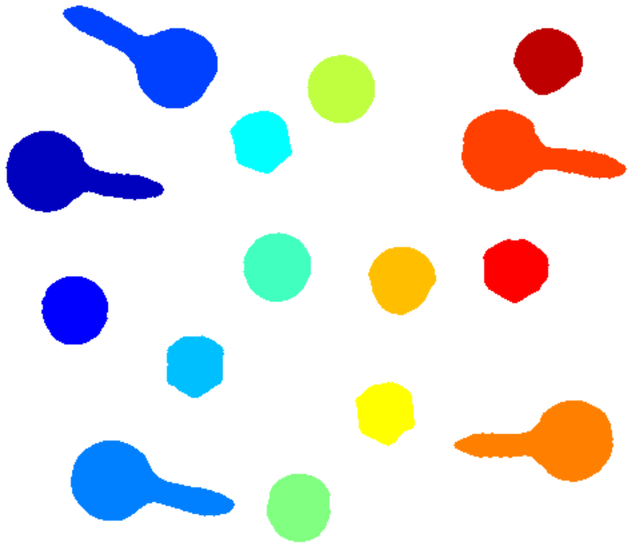
```
BW = imfill(BW, 'holes');
```



Classification

Going to classify the parts based on their area

```
[regions, numPixelRegions] = bwlabel(BW);

imshow(label2rgb(regions))
```

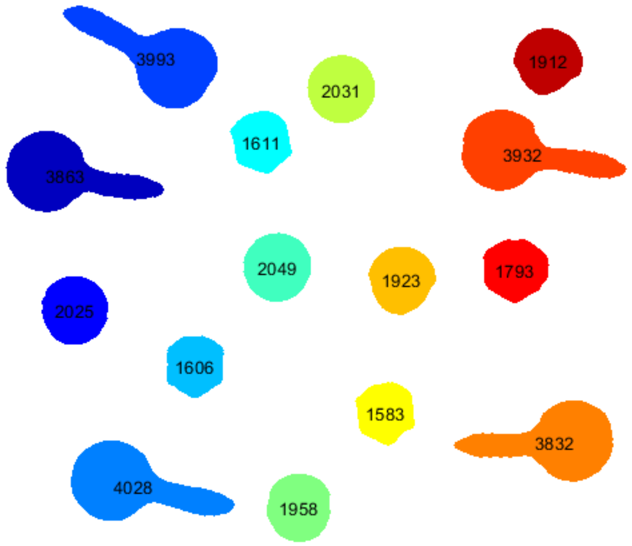


Classification

Going to classify the parts based on their area

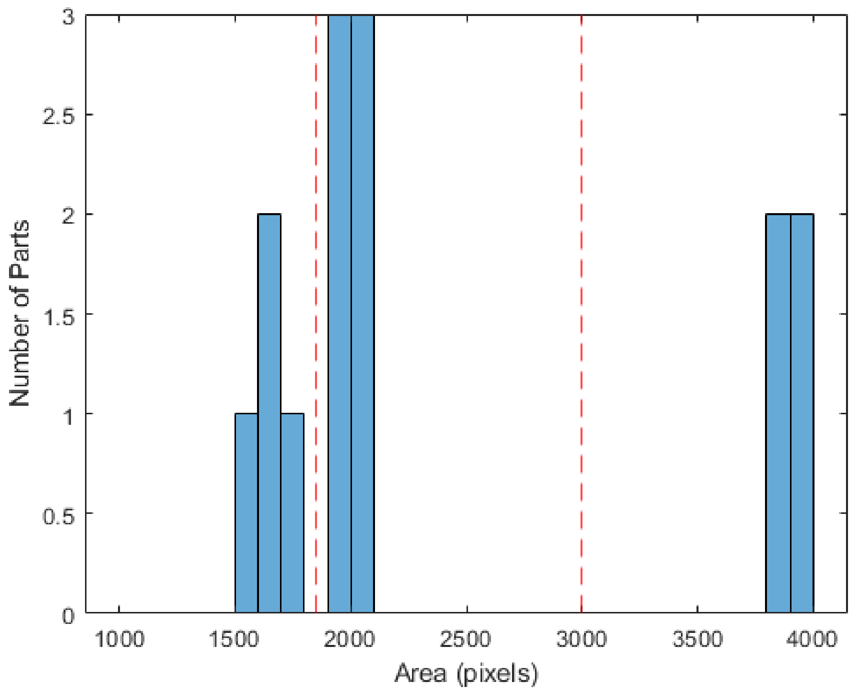
```
[regions, numPixelRegions] = bwlabel(BW);

imshow(label2rgb(regions))
```

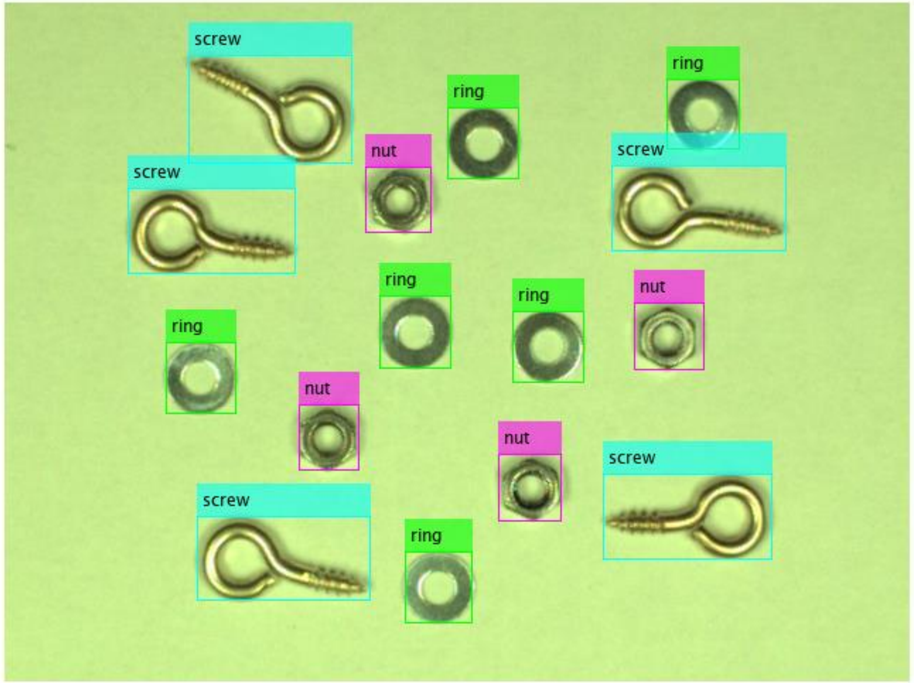
```
stats = regionprops(regions, 'all');

for k=1:length(stats)
    text(stats(k).Centroid(1),stats(k).Centroid(2),...
        sprintf('%04d',stats(k).Area),'Hor','Center','Vert','middle')
end
```



Classification

```
histogram(Area, 1000:100:4000)
xlabel('Area (pixels)')
ylabel('Number of Parts')
```

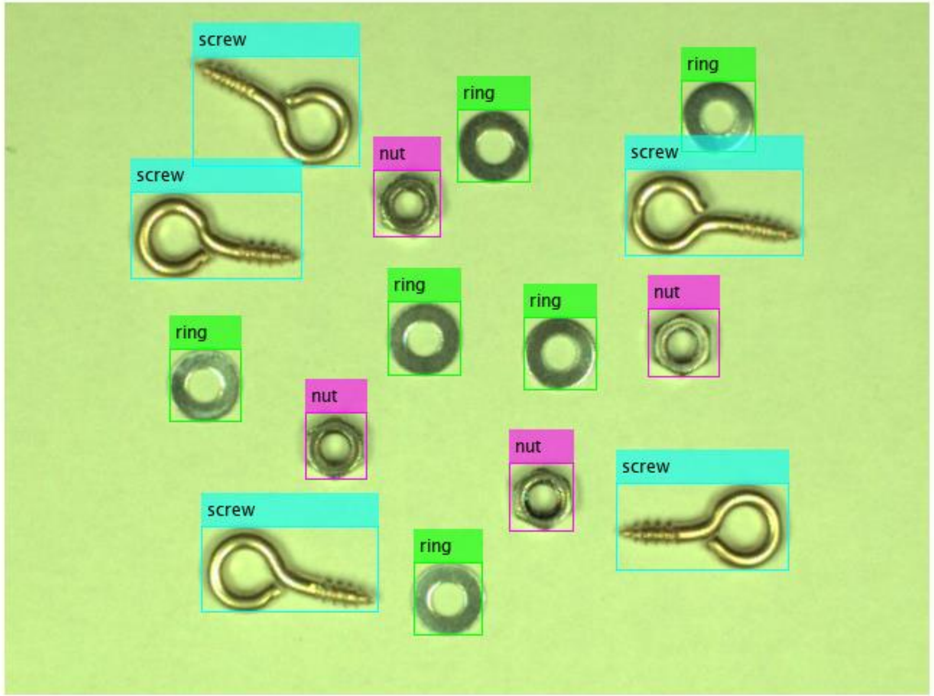





```
minArea = [1300 1900 3200];
maxArea = [1800 2200 4100];
partNames = {'nut', 'ring', 'screw'};
partColors = {'magenta'; 'green'; 'cyan'};
Iparts = I;
for k=1:3
    idx = Area > minArea(k) & Area < maxArea(k);
    Iparts = insertObjectAnnotation(Iparts,...
        'rectangle', vertcat(stats(idx).BoundingBox),...
        partNames{k}, 'Color', partColors{k});
end

imshow(Iparts)
```

Example 1: Part Inspection

- ‘Basic’ image processing can solve this problem well
- Single feature (area) can be used to classify
- Fast, and east to interpret



MATLAB provides:

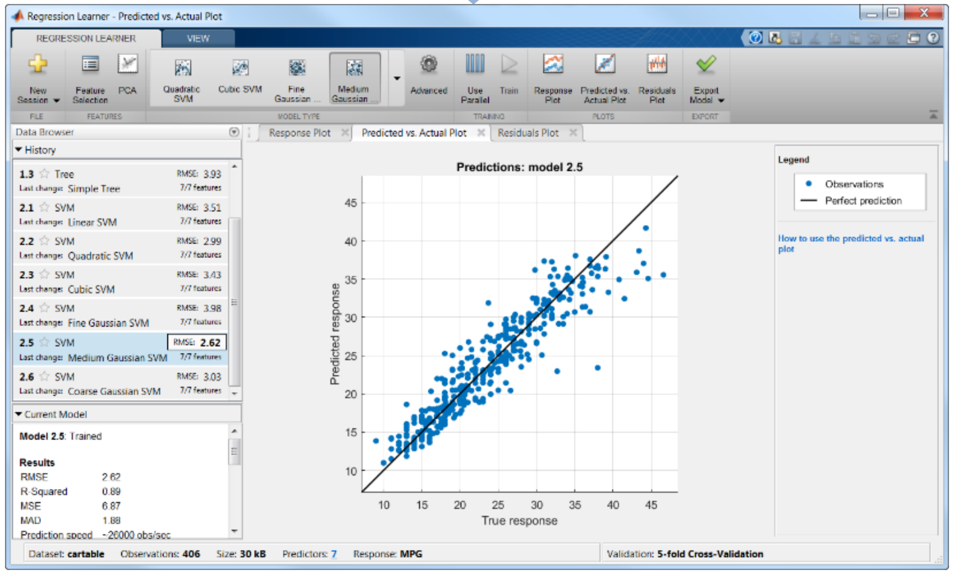
- High-level functions to chain together
- Apps to get started/learn functions
- Simple route to deployment

Adding more features

```
stats = regionprops(labeled, 'all');
```

Fields	Area	Centroid	BoundingBox	SubarrayIdx	MajorAxisLength	MinorAxisLength	Eccentricity	Orientation	ConvexHull
1	3863	[132.2923,164.4468]	[87.5000,131.5000,119,61]	1x2 cell	116.5162	53.5542	0.8881	-10.3597	73x2 double
2	2025	[139.4973,265.8198]	[114.5000,240.5000,50,51]	1x2 cell	51.8754	49.7679	0.2821	-84.8851	106x2 double
3	3993	[200.6604,76.1117]	[130.5000,37.5000,116,77]	1x2 cell	124.0825	53.3775	0.9027	-27.6098	97x2 double
4	4028	[183.4546,397.1611]	[136.5000,363.5000,123,60]	1x2 cell	124.5493	52.5141	0.9068	-12.8307	105x2 double
5	1606	[229.6227,307.1874]	[208.5000,284.5000,43,47]	1x2 cell	46.2989	44.4903	0.2768	73.9331	59x2 double
6	1611	[279.4333,139.3470]	[255.5000,116.5000,47,47]	1x2 cell	46.6591	44.2790	0.3153	-68.7663	55x2 double
7	2049	[291.3538,233.5915]	[265.5000,207.5000,51,52]	1x2 cell	51.7232	50.4794	0.2180	68.1632	97x2 double
8	1958	[307.6772,413.6563]	[283.5000,388.5000,48,51]	1x2 cell	51.8747	48.1504	0.3721	87.0232	93x2 double
9	2031	[339.4141,100.0768]	[313.5000,74.5000,51,51]	1x2 cell	51.1270	50.6080	0.1421	-86.0604	103x2 double
10	1583	[372.4839,342.3834]	[349.5000,319.5000,45,48]	1x2 cell	46.1031	44.1006	0.2915	76.6042	48x2 double
11	1923	[384.6121,242.8211]	[359.5000,218.5000,51,51]	1x2 cell	50.9729	48.2922	0.3200	66.2905	91x2 double
12	3832	[499.2584,364.4616]	[423.5000,333.5000,120,61]	1x2 cell	115.1116	53.6028	0.8850	3.4973	68x2 double
13	3932	[475.4209,148.3939]	[429.5000,115.5000,124,61]	1x2 cell	124.1413	52.9517	0.9045	-9.2141	79x2 double
14	1793	[469.8059,235.2711]	[445.5000,212.5000,50,48]	1x2 cell	49.6185	46.2617	0.3616	4.9631	60x2 double
15	1912	[494.3285,78.7374]	[468.5000,54.5000,52,50]	1x2 cell	50.5903	48.4191	0.2898	17.5298	94x2 double

- More complex classifications will require more features
- More features leads to a more complicated model
  - Machine Learning
- Other ways to extract features
  - e.g Visual bag of words



Example 2: Deep Learning

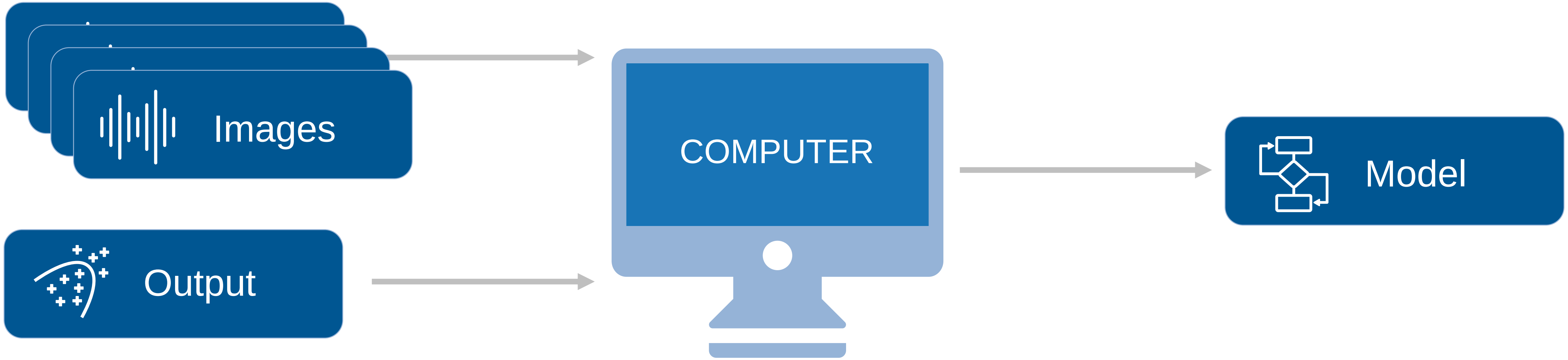
Challenge:

- Build an object detector to find cars

Data:

- Many images, each containing one or more cars
- Large variations in angle, lighting etc
- Labelled with bounding boxes (hopefully)

Two approaches to computer vision



Machine Learning

Deep Learning Workflow

Prepare Data

Data access and preprocessing

Ground truth labeling

Train Model

Model design, Hyperparameter tuning

Model exchange across frameworks

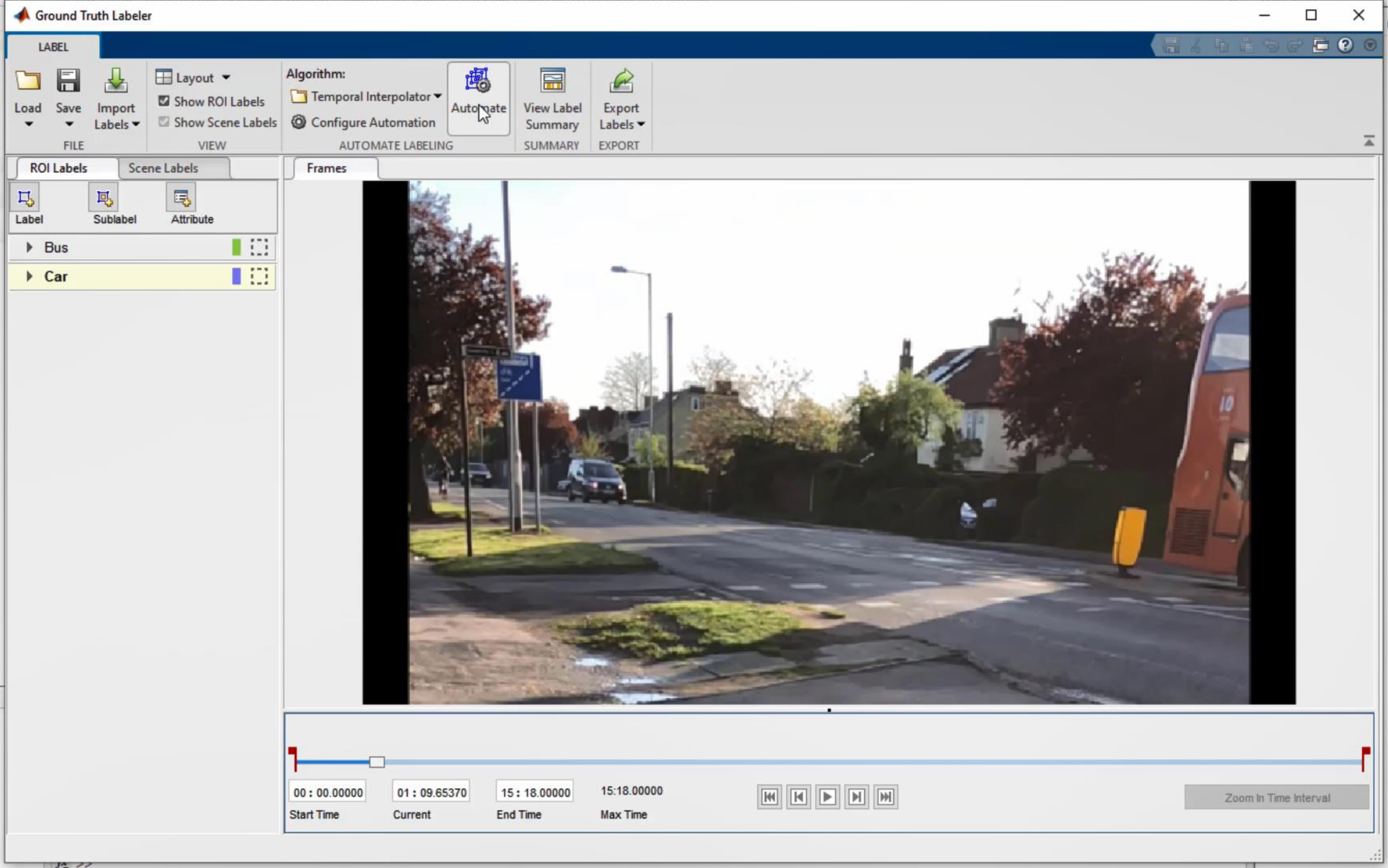
Hardware-accelerated training

Deploy

Multiplatform code generation (CPU, GPU)

Edge deployment

Enterprise Deployment


Ground Truth Labeler
LABEL
Load
Save
Import Labels
FILE
Layout
Show ROI Labels
Show Scene Labels
VIEW
Algorithm:
Temporal Interpolator
Configure Automation
Automate
AUTOMATE LABELING
View Label Summary
SUMMARY
Export Labels
EXPORT
ROI Labels
Scene Labels
Label
Sublabel
Attribute
Bus
Car
Frames
00 : 00.00000
Start Time
01 : 09.65370
Current
15 : 18.00000
End Time
15:18.00000
Max Time
Zoom In Time Interval

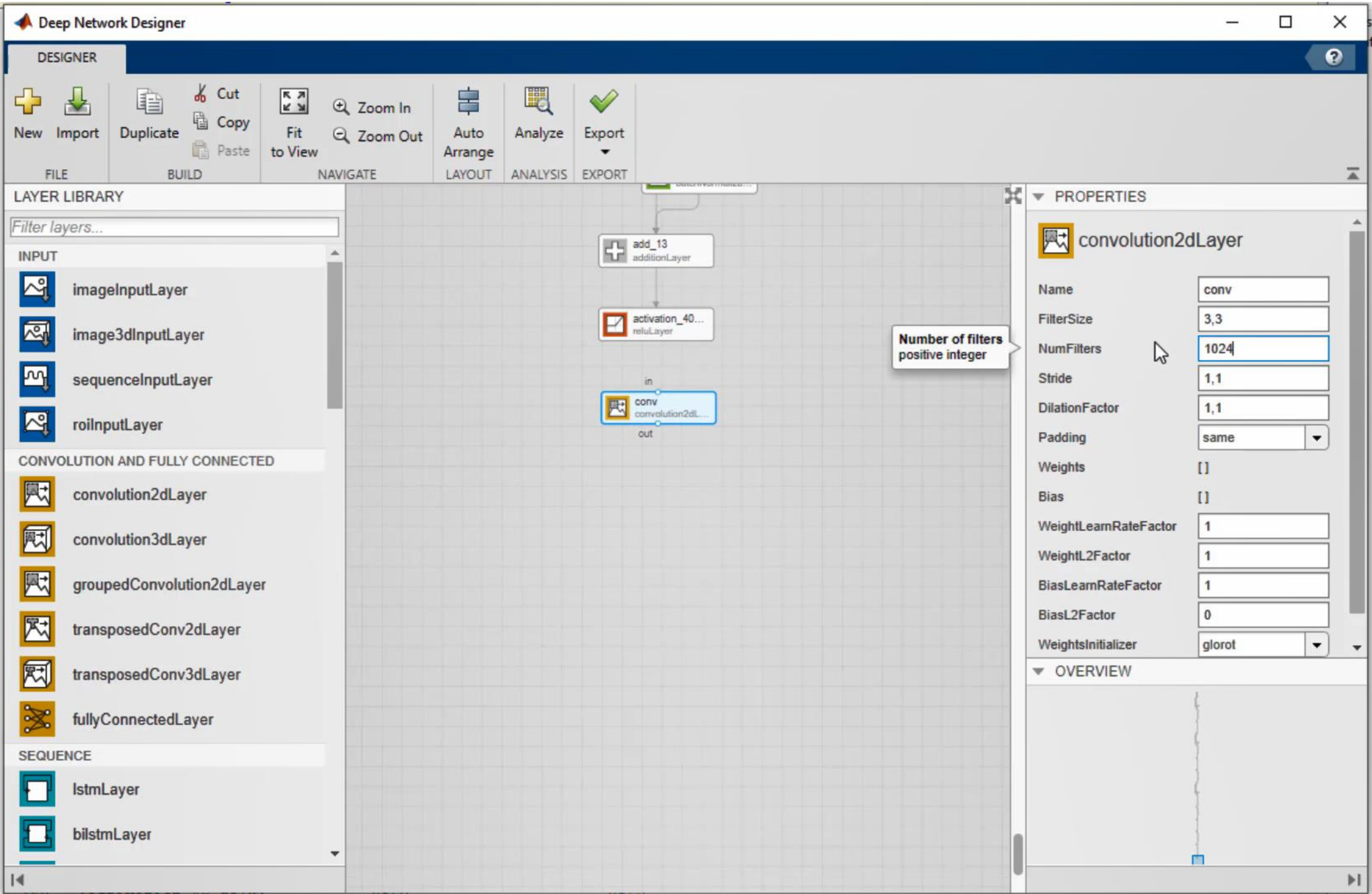
Prepare Data

- Dedicated MATLAB apps for automating and simplifying the labelling process
- Split data in training and test sets
- Datastore objects to manage collections of data

```
imdsTrain = imageDatastore(trainingDataTbl{:,'imageFilename'});
bldsTrain = boxLabelDatastore(trainingDataTbl(:,'vehicle'));

trainingData = combine(imdsTrain,bldsTrain);
```

Train Model

- Using YOLOv2 model architecture
  - Start-of-the-art object detector
  - Capable of running on real time video
  - Documented example
- Two stages:
  - Feature extraction layers – use a pretrained research network
  - Detector layers – build ourselves
- Use Deep Network Designer to build the network graphically


Deep Network Designer
DESIGNER
New
Import
Duplicate
Cut
Copy
Paste
Fit to View
Zoom In
Zoom Out
Auto Arrange
Analyze
Export
FILE
BUILD
NAVIGATE
LAYOUT
ANALYSIS
EXPORT
LAYER LIBRARY
Filter layers...
INPUT
imageInputLayer
image3dInputLayer
sequenceInputLayer
roiInputLayer
CONVOLUTION AND FULLY CONNECTED
convolution2dLayer
convolution3dLayer
groupedConvolution2dLayer
transposedConv2dLayer
transposedConv3dLayer
fullyConnectedLayer
SEQUENCE
lstmLayer
bilstmLayer
add_13
additionLayer
activation_40...
reluLayer
in
conv
convolution2dL...
out
Number of filters
positive integer
PROPERTIES
convolution2dLayer
Name
conv
FilterSize
3,3
NumFilters
1024
Stride
1,1
DilationFactor
1,1
Padding
same
Weights
[]
Bias
[]
WeightLearnRateFactor
1
WeightL2Factor
1
BiasLearnRateFactor
1
BiasL2Factor
0
WeightsInitializer
glorot
OVERVIEW

Train Model

Next define training options

```
options = trainingOptions('sgdm', ...
    'MiniBatchSize', 16, ....
    'InitialLearnRate',1e-3, ...
    'MaxEpochs',20);
```

And train the model

```
detector = trainYOLOv2ObjectDetector(trainingData, lgraph, options);
```

Testing

Evaluate model performance on validation images

```
I = imread(testDataTbl.imageFilename{1});

% Run the detector.
[bboxes,scores] = detect(detector,I);

% Annotate detections in the image.
I = insertObjectAnnotation(I,'rectangle',bboxes,scores);
imshow(I)
```

Testing

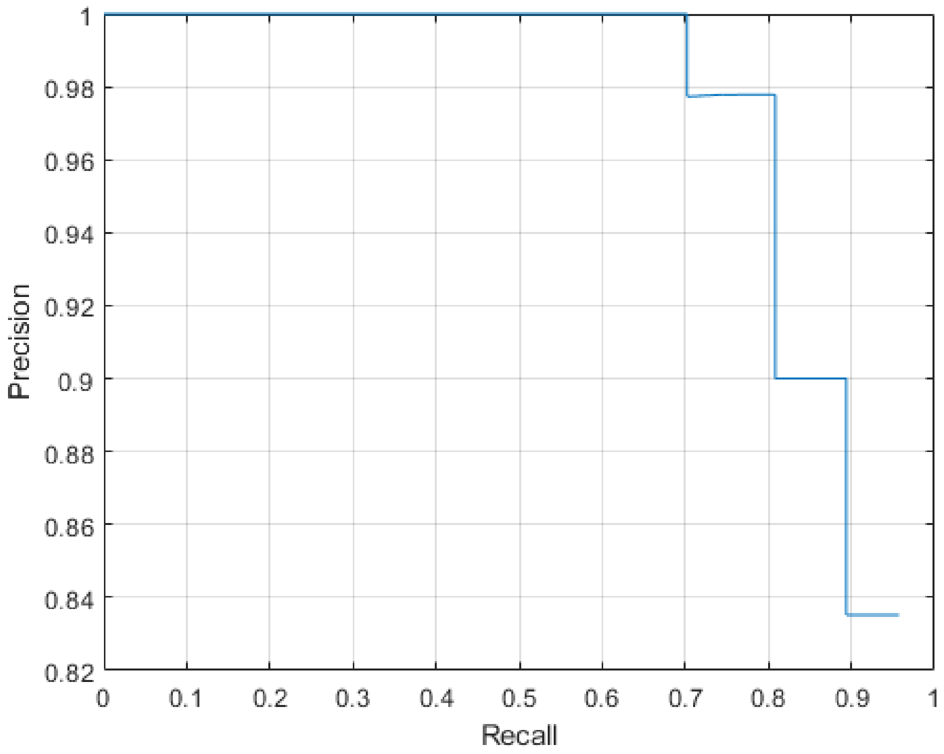
Evaluate model performance across training and test sets:

- Recall – what proportion of the cars do I detect?
- Precision – of the detections I make, what proportion are correct

Look out for:

- Underfitting – performance poor on training and test data
- Overfitting – good performance on training data, poor on test data

Iterate to improve the model



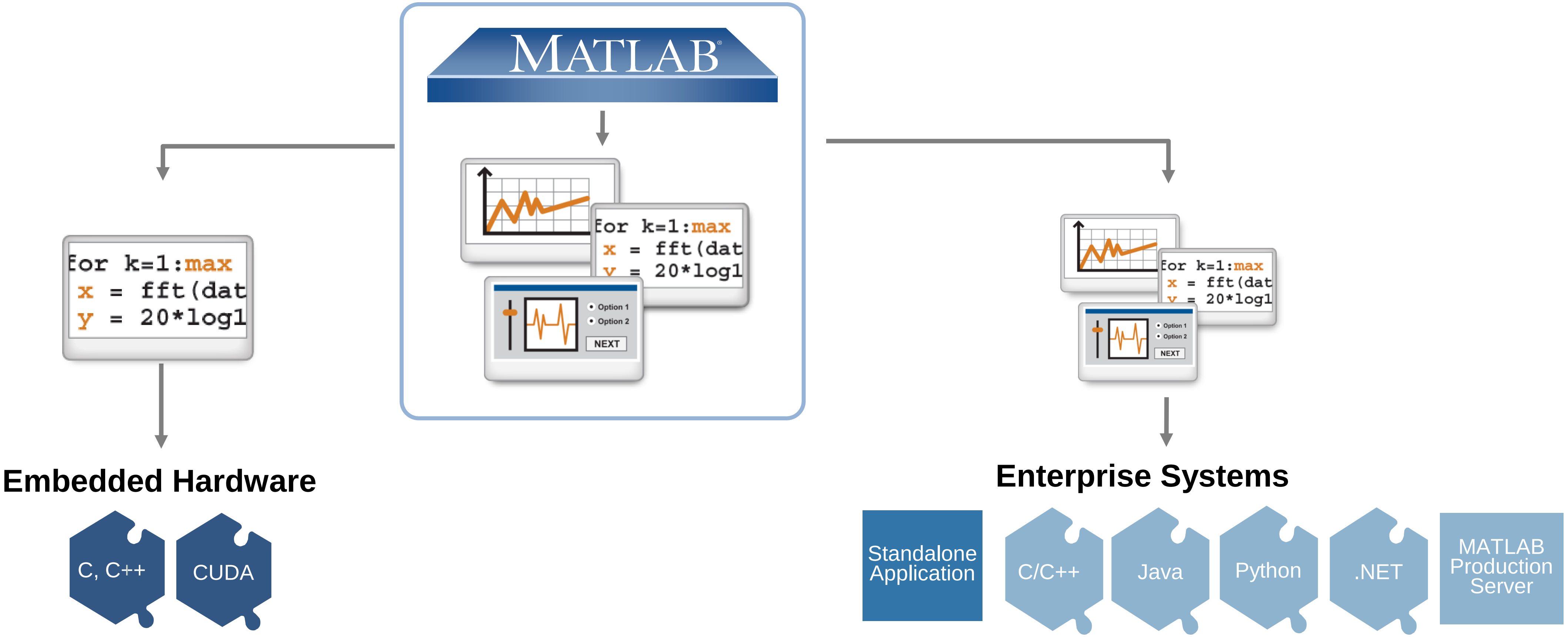
Example 2: Deep Learning

- Deep learning a good fit because of variation in the data
- Learns both a feature representation and a detection model

MATLAB provides:

- Graphical tools for labelling and network design
- Pretrained models to build on top of

Deploying Algorithms

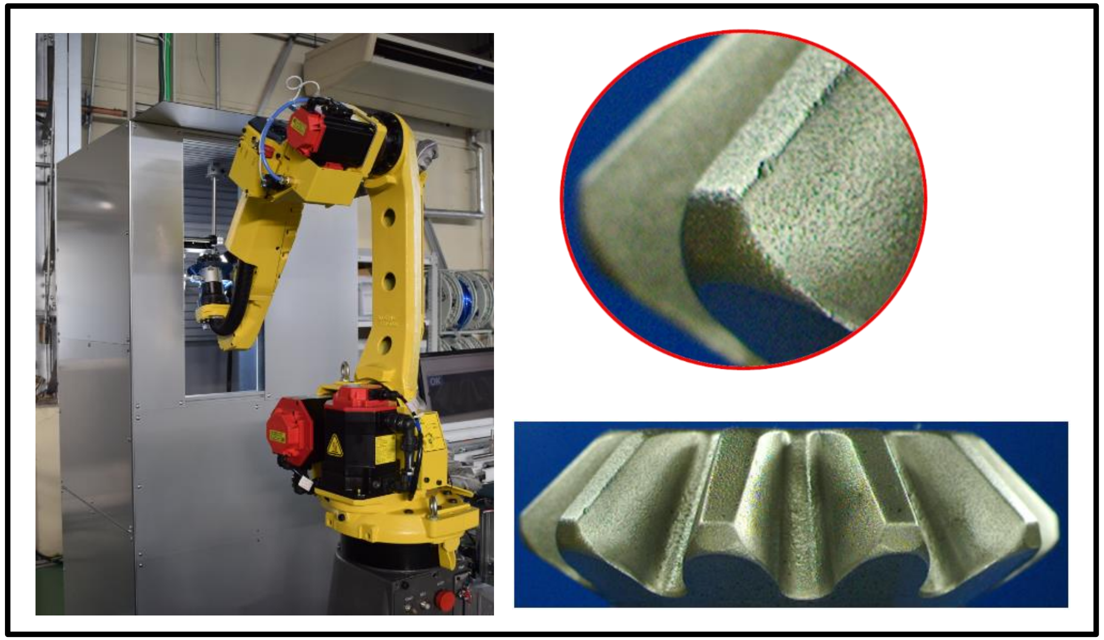


MATLAB Coder
GPU Coder

MATLAB Compiler
MATLAB Production Server

Musashi Seimitsu Industry Co.,Ltd.

Detect Abnormalities in Automotive Parts



Automated visual inspection of 1.3 million bevel gear per month

MATLAB use in project:

- Preprocessing of captured images
- Image labelling and annotation
- Deep learning based analysis
  - Various transfer learning methods (Combinations of CNN models, Classifiers)
  - Estimation of defect area using Class Activation Map
  - Abnormality/defect classification
- Deployment to NVIDIA Jetson using GPU Coder

Summary

- Segmentation and object detection form the basis of many common computer vision tasks
- Select image processing or machine learning approaches based on specifics of your problem

- MATLAB supports full workflow for both routes:
  - Easy data management
  - Apps to get started
  - Robust implementations of mathematical methods
  - Visualisations tools
  - Deployment to enterprise and embedded systems
  - Wide range of examples to adapt to your projects

What Next?

- Deep Learning Onramp
- Other talks
  - AI techniques for Signal, Time-series and Text Data
  - Automated Driving System Design
- Demo stands
  - Deep Learning and Reinforcement Learning
  - Driverless – Science Museum exhibition stand
- Doc examples
- Application Engineer support