load C:\Users\amogh\Documents\Matlab\MATLAB.mat

if ~isfolder(fullfile("TrainingDataCustomMultiDetect"))

mkdir TrainingDataCustomMultiDetect

end

trainingData1 = objectDetectorTrainingData(gTruth,'SamplingFactor',1,...

'WriteLocation','TrainingDataCustomMultiDetect');%yellow\_cone and blue\_cone

trainingData = trainingData1;

trainingData(1:4,:)

rng(0);

shuffledIndices = randperm(height(trainingData));

idx = floor(0.8 \* length(shuffledIndices));

trainingIdx = 1:idx;

testIdx = trainingIdx(end)+1 : length(shuffledIndices);

trainingDataSet = trainingData(shuffledIndices(trainingIdx),:);

testDataSet = trainingData(shuffledIndices(testIdx),:);

inputSize = [416 416 3];

inputLayer = imageInputLayer(inputSize, 'Name','input','Normalization','none');

filterSize = [3,3];

middleLayer = [

convolution2dLayer(filterSize, 16, 'Padding', 1,'Name','conv\_1',...

'WeightsInitializer','narrow-normal')

batchNormalizationLayer('Name','BN1')

reluLayer('Name','relu\_1')

maxPooling2dLayer(2, 'Stride',2,'Name','maxpool1')

convolution2dLayer(filterSize, 32, 'Padding', 1,'Name', 'conv\_2',...

'WeightsInitializer','narrow-normal')

batchNormalizationLayer('Name','BN2')

reluLayer('Name','relu\_2')

maxPooling2dLayer(2, 'Stride',2,'Name','maxpool2')

convolution2dLayer(filterSize, 64, 'Padding', 1,'Name', 'conv\_3',...

'WeightsInitializer','narrow-normal')

batchNormalizationLayer('Name','BN3')

reluLayer('Name','relu\_3')

maxPooling2dLayer(2, 'Stride',2,'Name','maxpool3')

convolution2dLayer(filterSize, 128, 'Padding', 1,'Name', 'conv\_4',...

'WeightsInitializer','narrow-normal')

batchNormalizationLayer('Name','BN4')

reluLayer('Name','relu\_4')

];

lgraph = layerGraph([inputLayer; middleLayer]);

numClasses = size(trainingData,2)-1;

trainingData11 = boxLabelDatastore(trainingData(:,2:end));

numAnchors =10;

[anchorBoxLayer,meanIoU] = estimateAnchorBoxes(trainingData11,numAnchors);

lgraph = yolov2Layers(inputSize,numClasses,anchorBoxLayer,lgraph,'relu\_4');

% deepNetworkDesigner

% analyzeNetwork(lgraph)

doTraining = "false";

% setting this flag to true will build & train a YOLOv2 detector

% false will load a pre-trained network

options = trainingOptions('sgdm',...

'InitialLearnRate',0.001, ...

'Verbose',true,...

'MiniBatchSize',16,...

'MaxEpochs',100,...

'Shuffle','every-epoch',...

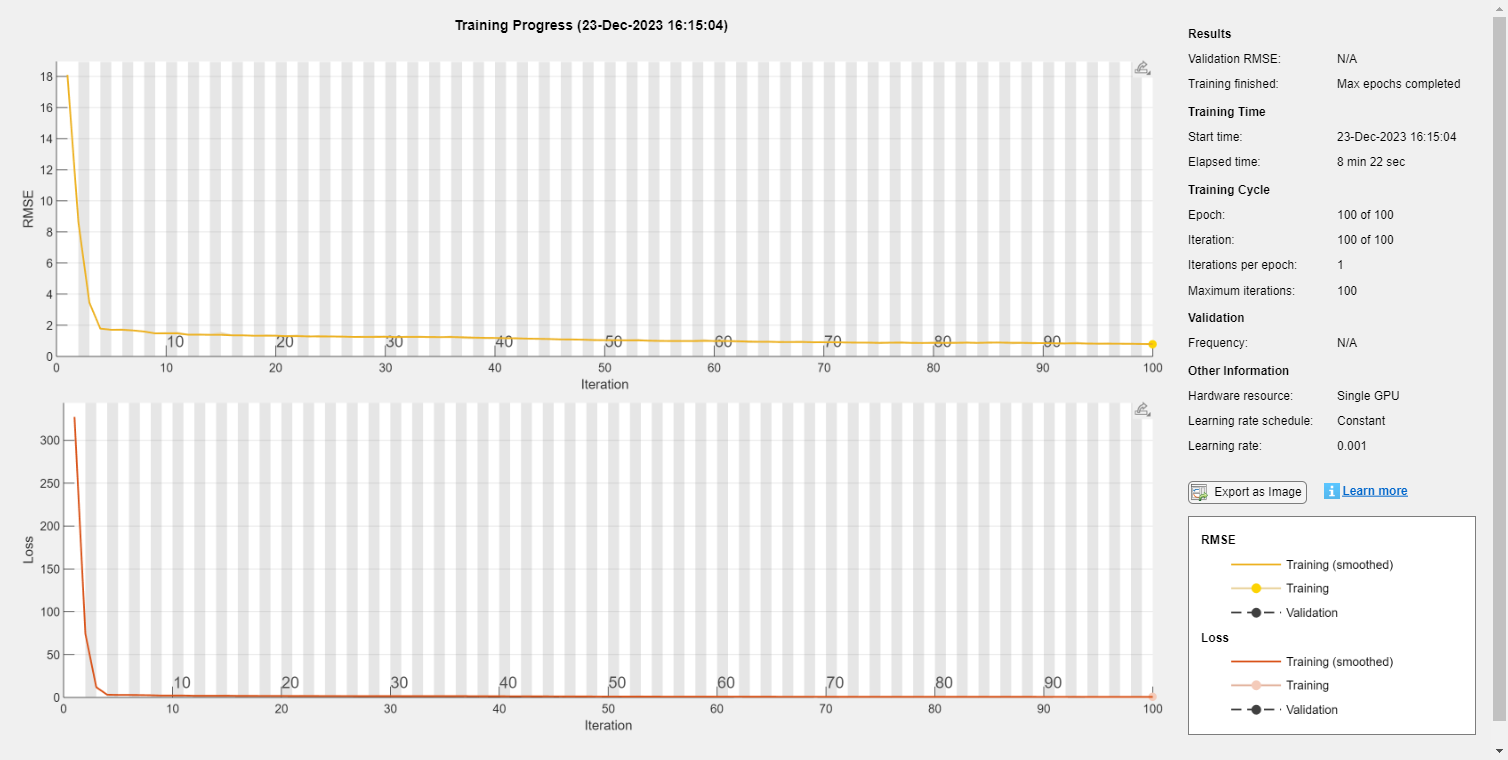
'VerboseFrequency',50,...

'ExecutionEnvironment','auto',...

'Plots',"training-progress");

[yolov2ConeDetector, info] = trainYOLOv2ObjectDetector(trainingDataSet,lgraph,options);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Training a YOLO v2 Object Detector for the following object classes:  
  
\* blue\_cone  
\* yellow\_cone  
  
Checking training data...done.  
Training on single GPU.  
|========================================================================================|  
| Epoch | Iteration | Time Elapsed | Mini-batch | Mini-batch | Base Learning |  
| | | (hh:mm:ss) | RMSE | Loss | Rate |  
|========================================================================================|  
| 1 | 1 | 00:00:18 | 18.10 | 327.4 | 0.0010 |  
| 50 | 50 | 00:04:20 | 1.05 | 1.1 | 0.0010 |  
| 100 | 100 | 00:08:22 | 0.78 | 0.6 | 0.0010 |  
|========================================================================================|  
Training finished: Max epochs completed.



Detector training complete.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

load ('matlab.mat');

results = table('Size',[height(testDataSet) 3],...

'VariableTypes',{'cell','cell','cell'},...

'VariableNames',{'Boxes','Scores','Labels'});

depVideoPlayer = vision.DeployableVideoPlayer;

for i = 1:height(testDataSet)

% Read the image

I = imread(testDataSet.imageFilename{i});

% Run the detector

[bboxes,scores,labels] = detect(yolov2ConeDetector,I);

if ~isempty(bboxes)

I = insertObjectAnnotation(I,'Rectangle',bboxes,cellstr(labels));

depVideoPlayer(I);

pause(0.1);

end

% Collect the results in the results table

results.Boxes{i} = floor(bboxes);

results.Scores{i} = (scores);

results.Labels{i} = (labels);

end

threshold = 0.7;

[ap,recall,precision] = evaluateDetectionPrecision(results, testDataSet(:,2:end),threshold);

[am,fppi,missRate] = evaluateDetectionMissRate(results, testDataSet(:,2:end),threshold);

figure

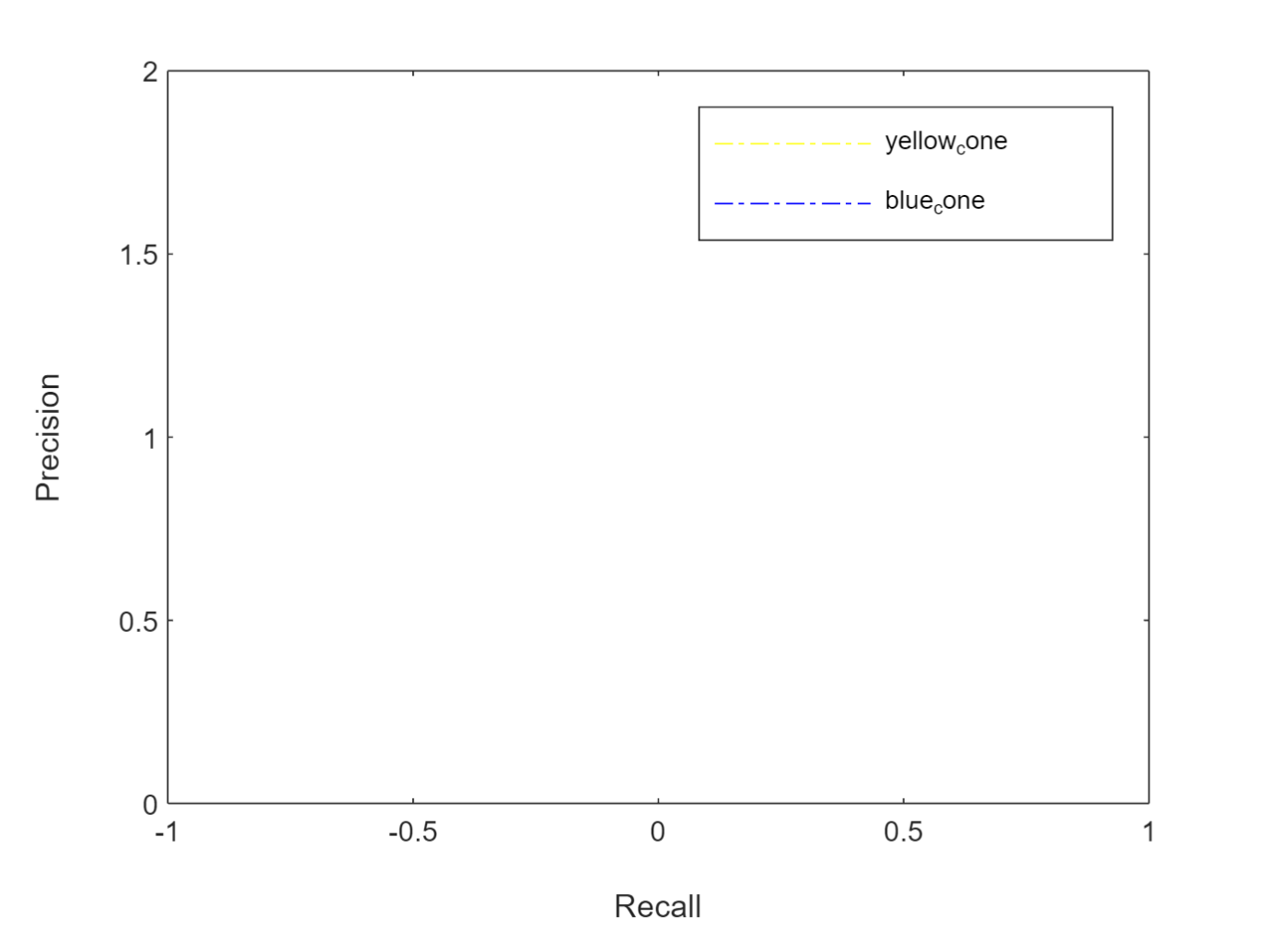
plot(recall{1,1},precision{1,1},'y-.',recall{2,1},precision{2,1},'b-.')

hold on;

xlabel('Recall')

ylabel('Precision')

legend('yellow\_cone','blue\_cone');



figure

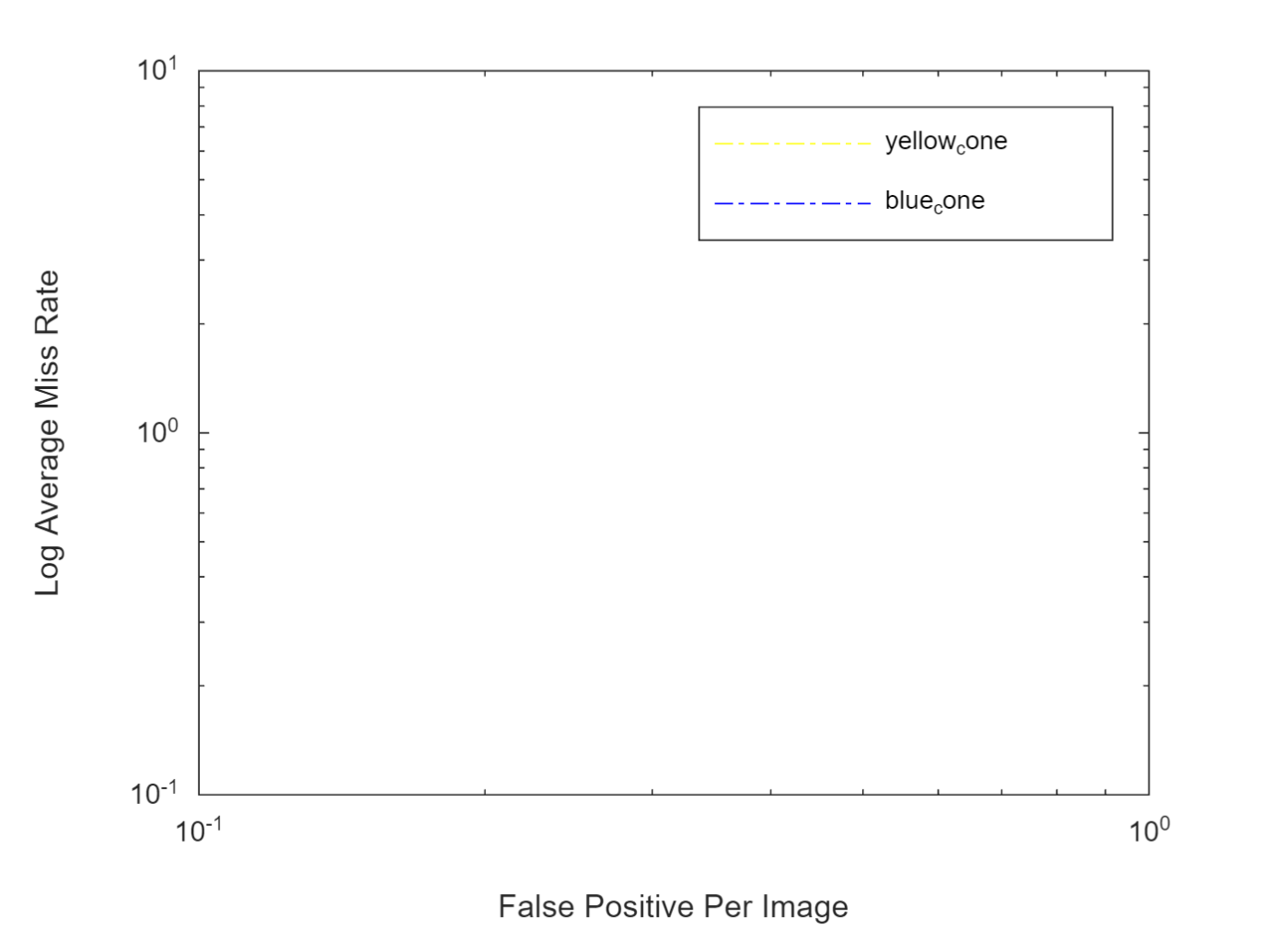
loglog(fppi{1,1},missRate{1,1},'y-.',fppi{2,1},missRate{2,1},'b-.')

hold on;

xlabel('False Positive Per Image')

ylabel('Log Average Miss Rate')

legend('yellow\_cone','blue\_cone');



vidReader = VideoReader('trail gt.mp4')

vidReader =

VideoReader with properties:  
  
 General Properties:  
 Name: 'trail gT.mp4'  
 Path: 'C:\Users\amogh\OneDrive\Documents\MATLAB'  
 Duration: 35.5340  
 CurrentTime: 0  
 NumFrames: 1066  
  
 Video Properties:  
 Width: 1920  
 Height: 1080  
 FrameRate: 30  
 BitsPerPixel: 24  
 VideoFormat: 'RGB24'

vidPlayer = vision.DeployableVideoPlayer;

i=1

i = 1

results = struct('Boxes',[],'Scores',[]);

% vidObj = VideoWriter('trail gt\_output.mp4');

% open(vidObj)

while(hasFrame(vidReader))

% GET DATA

I = readFrame(vidReader);

% PROCESS

end