Instituto Tecnológico de Aeronáutica - ITA
Inteligência Artificial para Robótica Móvel - CT-213
Aluno: Danilo de Farias Matos

Relatório do Laboratório 9 - Detecção de Objetos

1. Breve Explicação em Alto Nível da Implementação

Comecei a implementação com a recriação das camadas ausentes da rede para espelhar a referência da tabela 1, após isso foi a vez de implementar o pré-processamento das imagens para adaptá-las ao formato adequado de entrada da rede.

Na sequência, fiz a implementação da função “process_yolo_output” que criou basicamente duas arrays para armazenamento dos candidatos da bola e das traves. Após percorrer todos elementos da grid (15x20) e armazenar os dados dos candidatos em duas arrays diferentes, fiz um sort para ordenar da maior probabilidade de ocorrência para a menor, extraí as features e fiz os cálculos necessários da posição e bounding box da maior ocorrência apenas para a detecção da bola e das duas maiores para detecção das barras e passei como return da função.

1.1. Sumário do Modelo

Model: "ITA_YOLO"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 120, 160, 3)	0	-
conv_1 (Conv2D)	(None, 120, 160, 8)	216	input_layer[0][0]
norm_1 (BatchNormalizatio…	(None, 120, 160, 8)	32	conv_1[0][0]
leaky_relu_1 (LeakyReLU)	(None, 120, 160, 8)	0	norm_1[0][0]
conv_2 (Conv2D)	(None, 120, 160, 8)	576	leaky_relu_1[0][…
norm_2 (BatchNormalizatio…	(None, 120, 160, 8)	32	conv_2[0][0]
leaky_relu_2 (LeakyReLU)	(None, 120, 160, 8)	0	norm_2[0][0]
conv_3 (Conv2D)	(None, 120, 160, 16)	1,152	leaky_relu_2[0][…
norm_3 (BatchNormalizatio…	(None, 120, 160, 16)	64	conv_3[0][0]
leaky_relu_3 (LeakyReLU)	(None, 120, 160, 16)	0	norm_3[0][0]
max_pool_3 (MaxPooling2D)	(None, 60, 80, 16)	0	leaky_relu_3[0][…
conv_4 (Conv2D)	(None, 60, 80, 32)	4,608	max_pool_3[0][0]

conv_4 (Conv2D)	(None, 60, 80, 32)	4,608	max_pool_3[0][0]
norm_4 (BatchNormalizatio…	(None, 60, 80, 32)	128	conv_4[0][0]
leaky_relu_4 (LeakyReLU)	(None, 60, 80, 32)	0	norm_4[0][0]
max_pool_4 (MaxPooling2D)	(None, 30, 40, 32)	0	leaky_relu_4[0][…
conv_5 (Conv2D)	(None, 30, 40, 64)	18,432	max_pool_4[0][0]
norm_5 (BatchNormalizatio…	(None, 30, 40, 64)	256	conv_5[0][0]
leaky_relu_5 (LeakyReLU)	(None, 30, 40, 64)	0	norm_5[0][0]
max_pool_5 (MaxPooling2D)	(None, 15, 20, 64)	0	leaky_relu_5[0][…
conv_6 (Conv2D)	(None, 15, 20, 64)	36,864	max_pool_5[0][0]
norm_6 (BatchNormalizatio…	(None, 15, 20, 64)	256	conv_6[0][0]
leaky_relu_6 (LeakyReLU)	(None, 15, 20, 64)	0	norm_6[0][0]
max_pool_6 (MaxPooling2D)	(None, 15, 20, 64)	0	leaky_relu_6[0][…
conv_7 (Conv2D)	(None, 15, 20, 128)	73,728	max_pool_6[0][0]
norm_7 (BatchNormalizatio…	(None, 15, 20, 128)	512	conv_7[0][0]

norm_7 (BatchNormalizatio…	(None, 15, 20, 128)	512	conv_7[0][0]
leaky_relu_7 (LeakyReLU)	(None, 15, 20, 128)	0	norm_7[0][0]
conv_skip (Conv2D)	(None, 15, 20, 128)	8,192	max_pool_6[0][0]
conv_8 (Conv2D)	(None, 15, 20, 256)	294,912	leaky_relu_7[0][…
norm_skip (BatchNormalizatio…	(None, 15, 20, 128)	512	conv_skip[0][0]
norm_8 (BatchNormalizatio…	(None, 15, 20, 256)	1,024	conv_8[0][0]
leaky_relu_skip (LeakyReLU)	(None, 15, 20, 128)	0	norm_skip[0][0]
leaky_relu_8 (LeakyReLU)	(None, 15, 20, 256)	0	norm_8[0][0]
concat (Concatenate)	(None, 15, 20, 384)	0	leaky_relu_skip[… leaky_relu_8[0][…
conv_9 (Conv2D)	(None, 15, 20, 10)	3,850	concat[0][0]

Total params: 445,346 (1.70 MB)
Trainable params: 443,938 (1.69 MB)
Non-trainable params: 1,408 (5.50 KB)

2. Figuras Comprovando Funcionamento do Código

2.1. Detecção de Objetos com YOLO

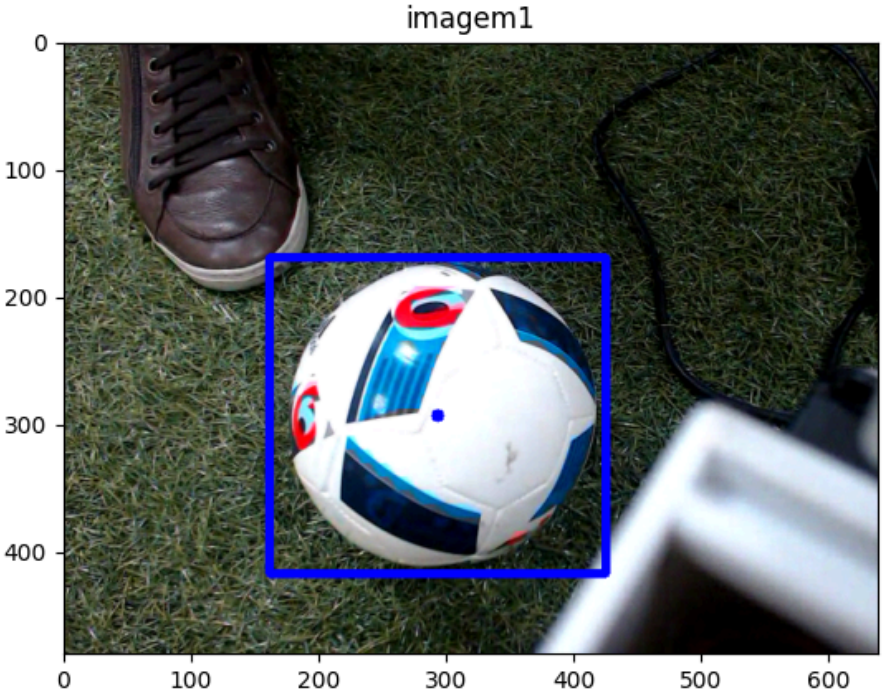
imagem1
0
100
200
300
400
0
100
200
300
400
500
600

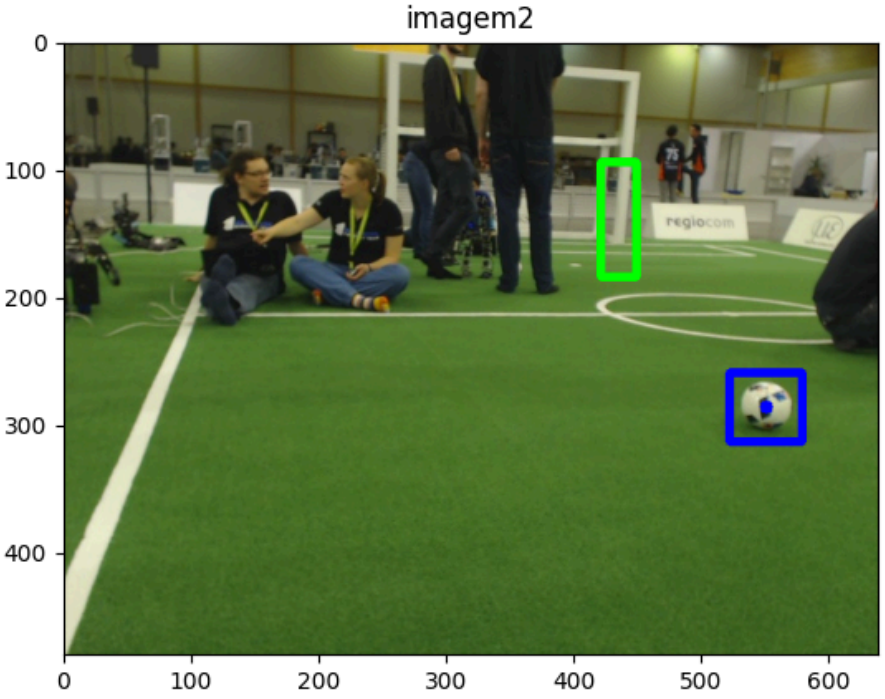
imagem2
0
100
200
300
400
0
100
200
300
400
500
600

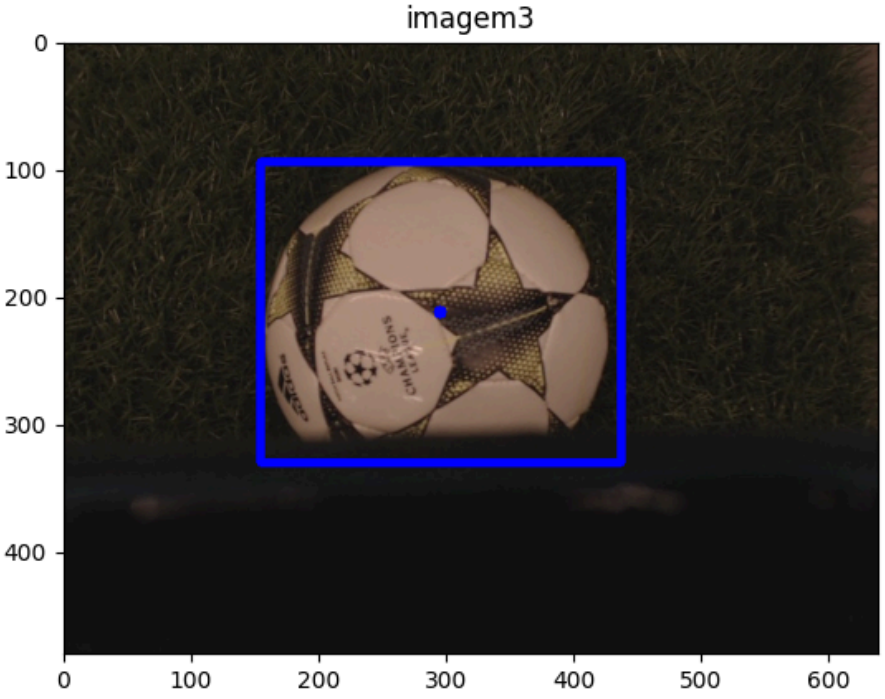
imagem3
0
100
200
300
400
0
100
200
300
400
500
600

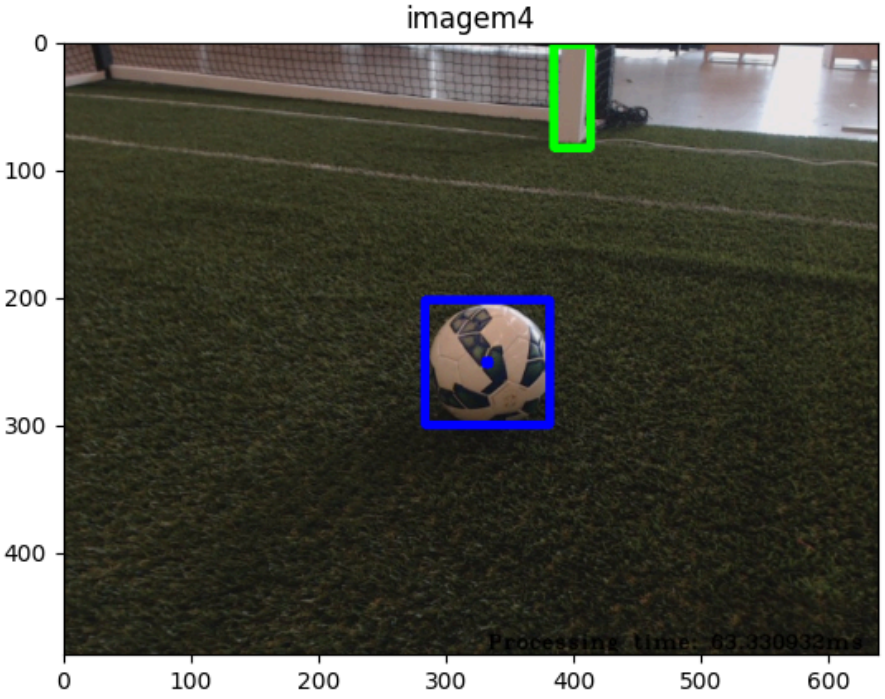
imagem4
0
100
200
300
400
0
100
200
300
400
500
600
Processing time: 63.330932ms

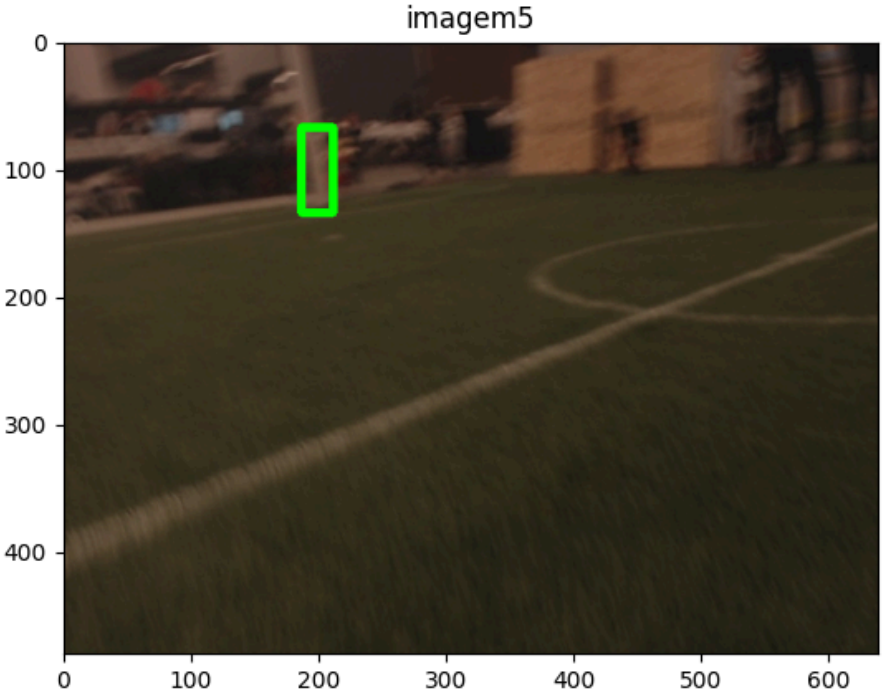
imagem5
0
100
200
300
400
0
100
200
300
400
500
600

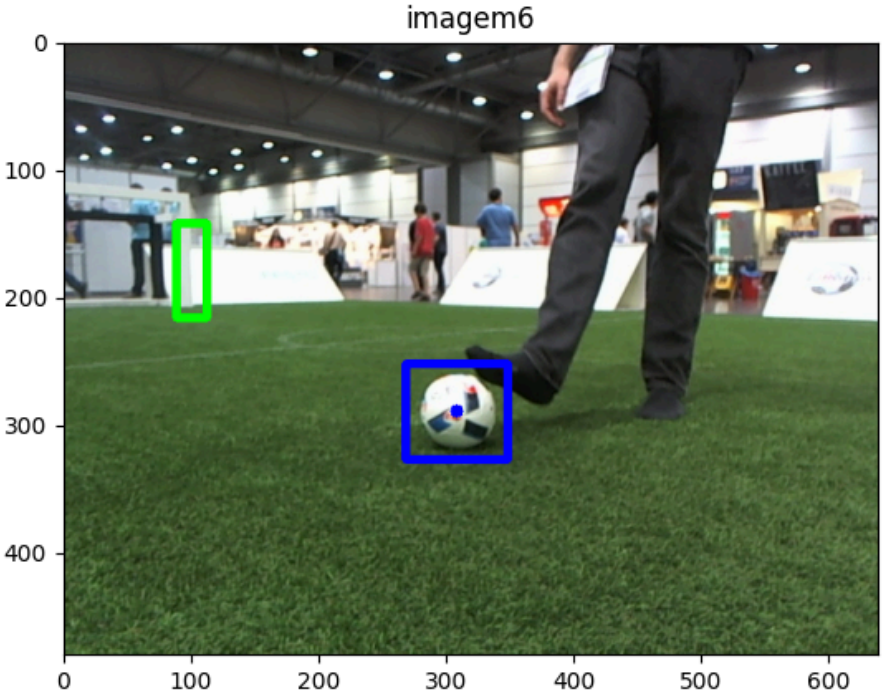
imagem6
0
100
200
300
400
0
100
200
300
400
500
600

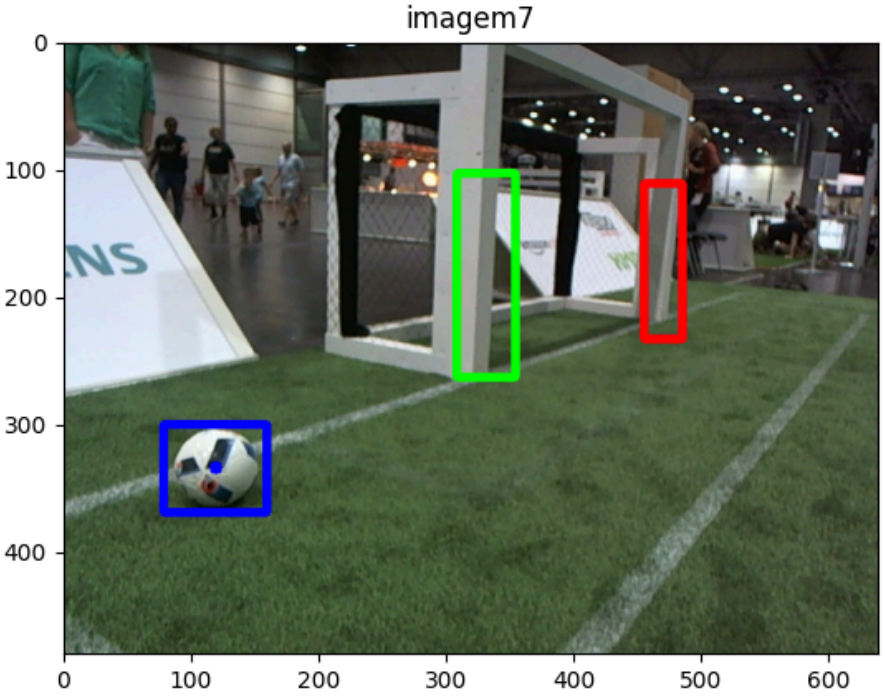
imagem7
0
100
200
300
400
0
100
200
300
400
500
600

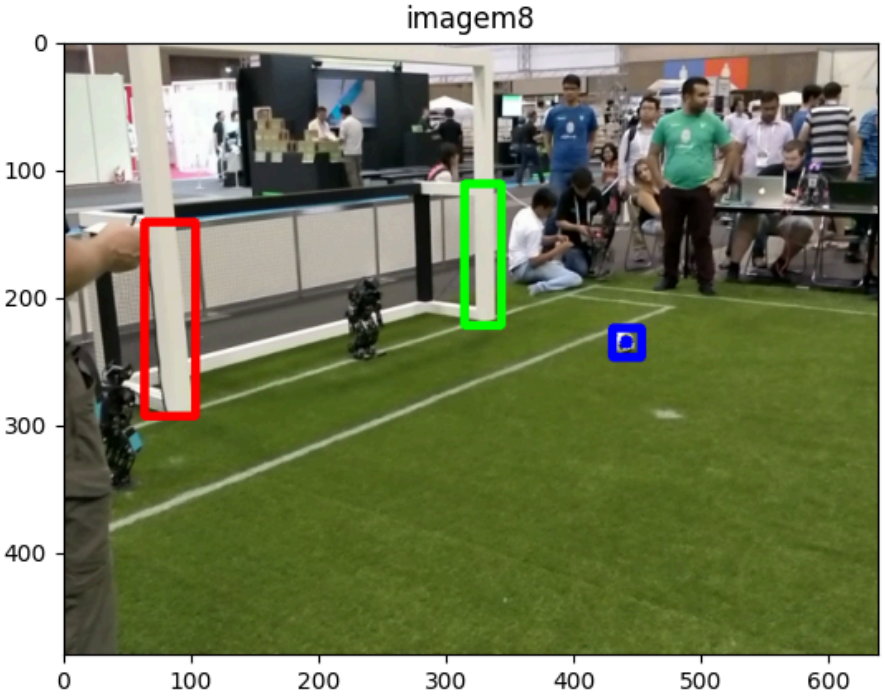
imagem8
0
100
200
300
400
0
100
200
300
400
500
600

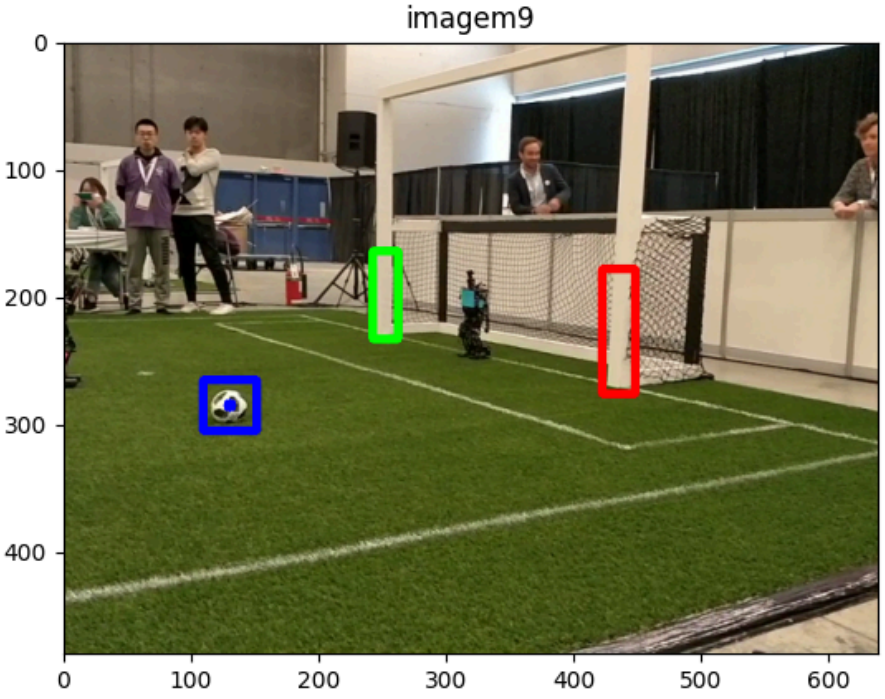
imagem9
0
100
200
300
400
0
100
200
300
400
500
600

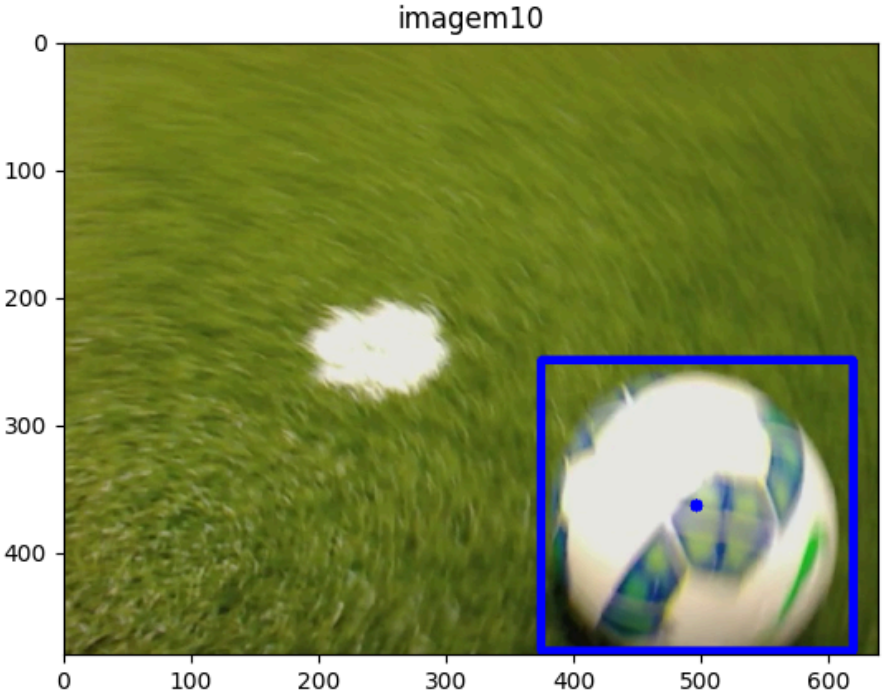
imagem10
0
100
200
300
400
0
100
200
300
400
500
600

3. Discussão

Observei que a rede possui grande capacidade de detecção da bola e das traves mesmo quando a imagem não está muito bem definida (Imagem 5 e 10 por exemplo). Outra coisa importante que observei é que a velocidade para inferência (pelo menos nas imagens estáticas) foi bem alta, o código inteiro demorou 7 segundos para produzir o resultado final com as 10 imagens - O tempo médio de inferência foi de 0.1309 segundos.