02_CAMSHIFT

April 15, 2018

1 Assigment 2: CAMSHIFT

1.1 Paper

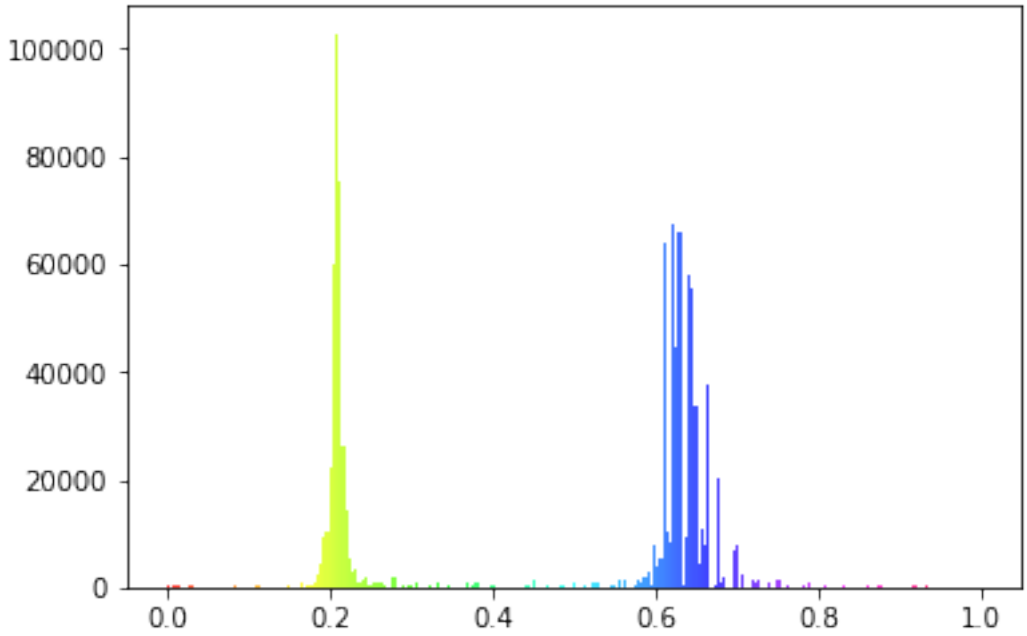
Lies das Paper "Bradski_etal_1998_camshift.pdf" im KVV (unter "Resources").

1.2 Histogramm berechnen

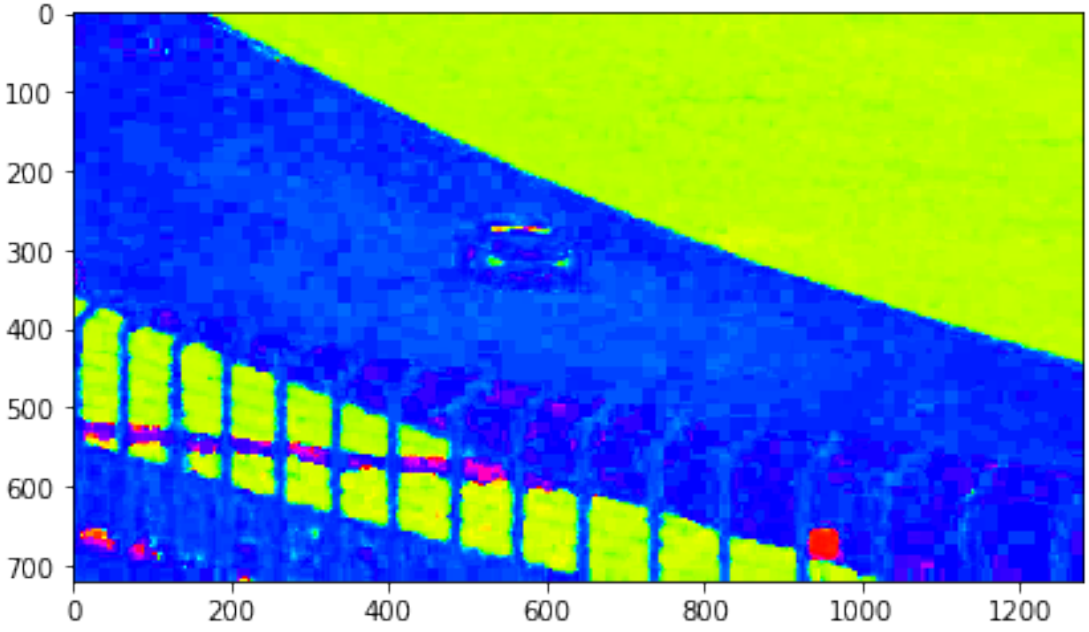
- Implementiere eine Funktion, die ein Farbhistogramm erstellt. Übergebe entweder ein Bild und ein ROI, oder das dem ROI unterliegende Bild.
- Hierzu ermögliche durch die Übergabe eines zweiten (bzw. dritten) Parameters die Zusammenfassung von Farbwerten in n Bins.
- Lade das Bild "images/racecar.png" und konvertiere das Bild in den HSV-Farbraum. Plotte den Hue-Kanal. (**RESULT**)

In [1]:
```
# dieser Code wurde als Musterlösung von Tobias Schülke zur Verfügung gestellt und von

%matplotlib inline
from skimage import io,color
import numpy as np
from matplotlib import pyplot as plt
import matplotlib.patches as patches
import os
import warnings; warnings.simplefilter('ignore')
import math
IMAGES_PER_ROW = 4

MIN_SATURATION_CAR = 0.2
MIN_VALUE_CAR = 0.5
MIN_SATURATION_TACO = 0.8
MIN_VALUE_TACO = 0.2

ROI_FRAME_MARGIN_CAR = 60
ROI_FRAME_MARGIN_TACO = 20

image = io.imread('images/racecar.png')
imageCar = image[260:350, 480:640]
```

```
# ...

def createColorHistogram(img, binCount = 256, out = plt):
    roi = color.rgb2hsv(img)
    invalid = 0 #ToDo
    #initialize list for plotting
    pixel = []
    #search and count pixel hues
    for x in range(roi.shape[0]):
        for y in range(roi.shape[1]):
            hue = roi[x][y][0]
            pixel.append(hue)

    #fig, axs = out.subplots()
    n, bins, patches = out.hist(pixel, binCount,  facecolor='g', alpha=0.75)

    bar_colors = []


    # prepare color list for the bucket colors
    for i in range(binCount):
        current_bin_color_hsv = np.array([[[i/float(binCount), 1, 1]]], dtype=np.float
        bar_color_rgb = color.hsv2rgb(current_bin_color_hsv)
        bar_color_rgb_norm = (bar_color_rgb[0][0][0],bar_color_rgb[0][0][1], bar_color
        bar_colors.append(bar_color_rgb_norm)

    # set bucket colors
    for thispatch, i in zip(patches, range(binCount)):
        thispatch.set_facecolor(bar_colors[i])
    return n, bins, patches


createColorHistogram(image)
plt.show()

img_hsv = color.rgb2hsv(image)
img_hsv[:, :, 1] = 1
img_hsv[:, :, 2] = 1
img_rgb = color.hsv2rgb(img_hsv)
io.imshow(img_rgb)
```

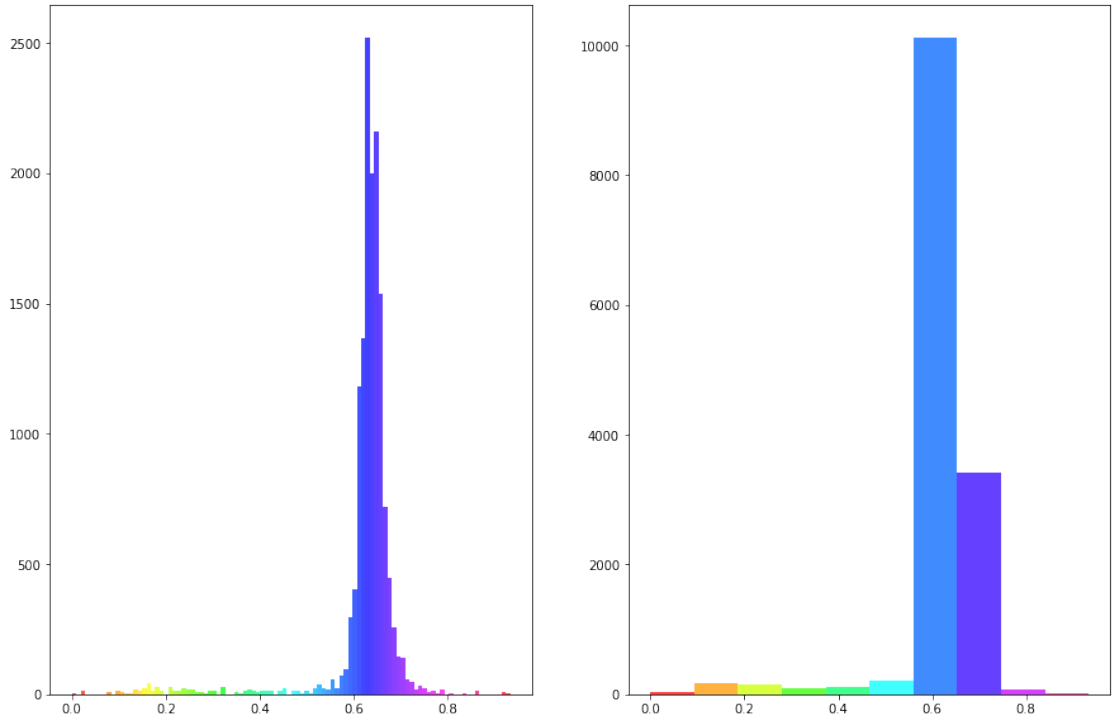



```
Out[1]: <matplotlib.image.AxesImage at 0x12713db9ef0>
```



- stelle das Histogramm über dem Hue-Kanal für das gesamte Bild und für den Ausschnitt (x,y) = (480, 260) bis (640, 350) dar. Variiere auch mal testweise die Zahl der Bins(**RESULT**)

```
In [2]: fig = plt.figure(figsize=(15, 10))
        ax1 = plt.subplot(1, 2, 1)
        ax2 = plt.subplot(1, 2, 2)

        histCar100 = createColorHistogram(imageCar, 100, ax1)

        histCar10 = createColorHistogram(imageCar, 10, ax2)

        plt.savefig('testbilder/car_100 and_10_bins.png')
```



1.3 Wahrscheinlichkeitsverteilung

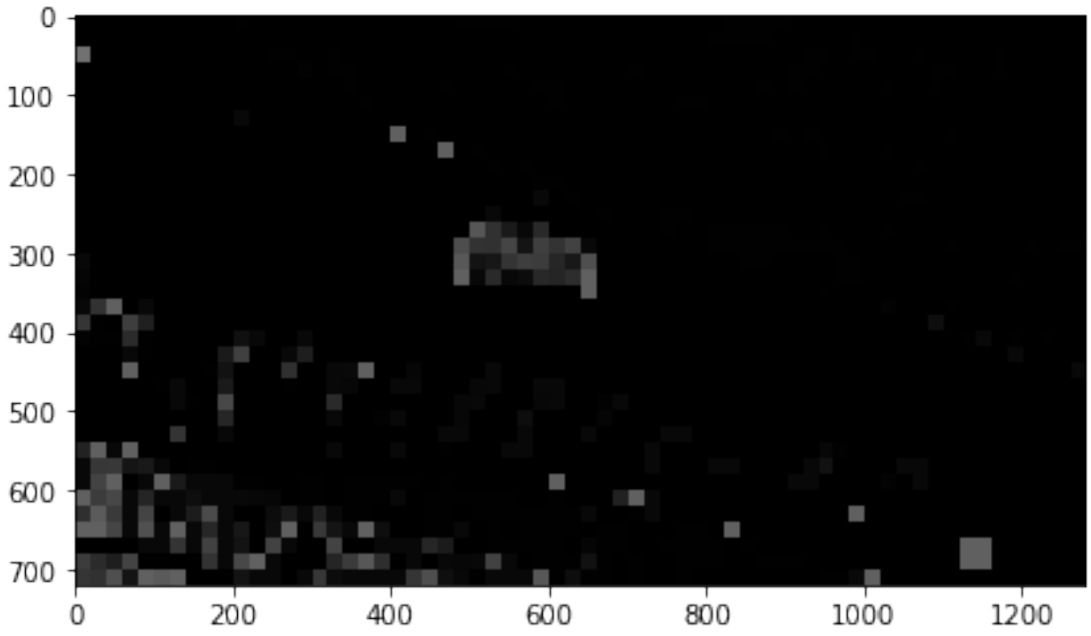
- implementiere die Methode aus der Vorlesung, die Dir - gegeben ein Hue-Histogramm - die Objekt-Wahrscheinlichkeitsverteilung für ein neues Bild berechnet.
- erzeuge das Histogramm des Autos aus dem Bild "racecar.png" und wende die neue Funktion auf das letzte frame des Videos (images/racecar/151.jpg) an (**RESULT**)

```
In [61]: def calculateProbFor(pixel_hsv, nHist, nbins, minSaturation, minValue):
             if (pixel_hsv[1] < minSaturation) or (pixel_hsv[2] < minValue):
                 return -1

             mass_of_histogramm = np.sum(nHist)
             current_hue = pixel_hsv[0]
             current_bin = math.floor(current_hue*nbins)
```

```
    number_of_counts = nHist[current_bin]
    relative_frequency = number_of_counts/mass_of_histogramm
    return relative_frequency

'''MIN_SATURATION_CAR = 0.2
MIN_VALUE_CAR = 0.5
MIN_SATURATION_TACO = 0.8
MIN_VALUE_TACO = 0.2'''
#Low and high thresholds are
#set off 10% of the maximum pixel value

# Tipp: in der Nacht sind alle Katzen grau ;)
def createProbDistribution(image, objectHist, nbins, minSaturation, minValue, searchBl
    probBlockSizeX, probBlockSizeY = searchBlock
    pixelPerProbBlock = probBlockSizeX*probBlockSizeY

    probDistribution = np.empty_like(image)
    #initialize probDistribution with color
    probDistribution[:, :, 0] = 200
    probDistribution[:, :, 1] = 0
    probDistribution[:, :, 2] = 255

    countBlocksX = math.floor(probDistribution.shape[1]/probBlockSizeX)
    countBlocksY = math.floor(probDistribution.shape[0]/probBlockSizeY)
    image_hsv = color.rgb2hsv(image)
    n, bins, patches = objectHist

    for currentBlockX in range(countBlocksX):
            x = currentBlockX * probBlockSizeX
            for currentBlockY in range(countBlocksY):
                y = currentBlockY * probBlockSizeY

                #iterate over area
                probPerArea = 0
                currentPixelPerBlock = pixelPerProbBlock
                for a in range(probBlockSizeX):
                    for b in range(probBlockSizeY):
                        current_pixel = image_hsv[y+b][x+a]
                        probability = calculateProbFor(current_pixel, n, nbins, minSat
                        if (probability < 0):
                            currentPixelPerBlock-=1
                        else:
                            probPerArea += probability

                try:
                    probPerArea = probPerArea/currentPixelPerBlock
                    probInGrey = np.array([probPerArea,probPerArea,probPerArea])*255
                except ZeroDivisionError: # if all pixels are invalid set the probabi
```

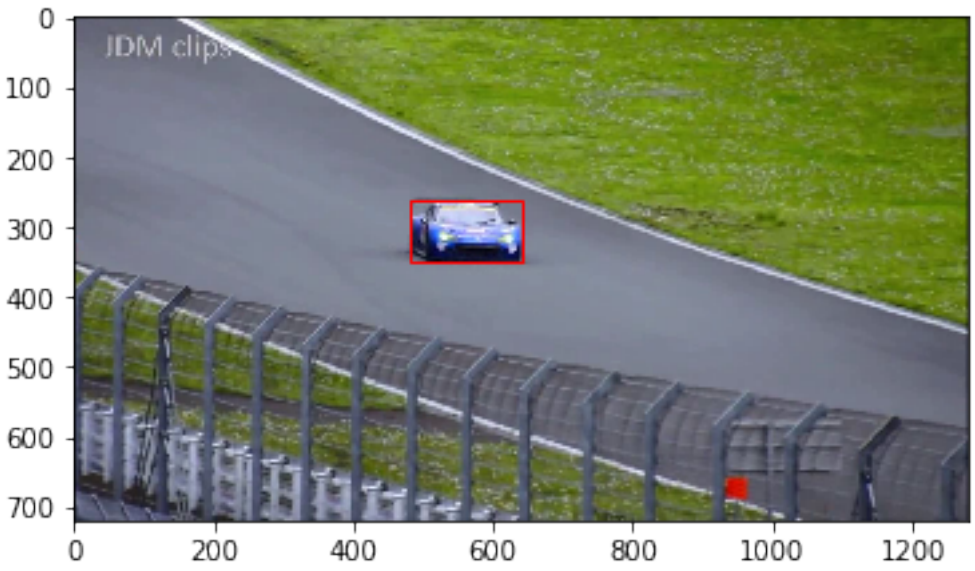
```
                probInGrey = np.array([0,0,0])

            #color area with probability
            probDistribution[y:y+probBlockSizeY, x:x+probBlockSizeX] = probInGrey
    return probDistribution

lastFrame = io.imread('images/racecar.png')

#lastFrame = io.imread('images/taco/001.jpg')
#face = lastFrame[160:280, 320:450]

nbins = 25 #20 to 80
#histogramFace = createColorHistogram(face, nbins, ax1)
histogramCar = createColorHistogram(imageCar, nbins, ax1)
searchBlock = (20, 20)

MIN_SATURATION_CAR = 0.2
MIN_VALUE_CAR = 0.5

probDistr = createProbDistribution(lastFrame, histogramCar, nbins, MIN_SATURATION_CAR

io.imshow(probDistr)
```

Out[61]: <matplotlib.image.AxesImage at 0x1271fc85390>



1.3.1 ROI auf Bild anzeigen

```
In [4]: def drawROI(image, x, y, width, height, out = plt):
            currentAxis = plt.gca()
            currentAxis.add_patch(
                patches.Rectangle(
                    (x, y),
                    width,
                    height,
                    fill=False,
                    edgecolor="red"
                )
            )

            out.imshow(image)

        drawROI(image, 480, 260, 160, 90)
```



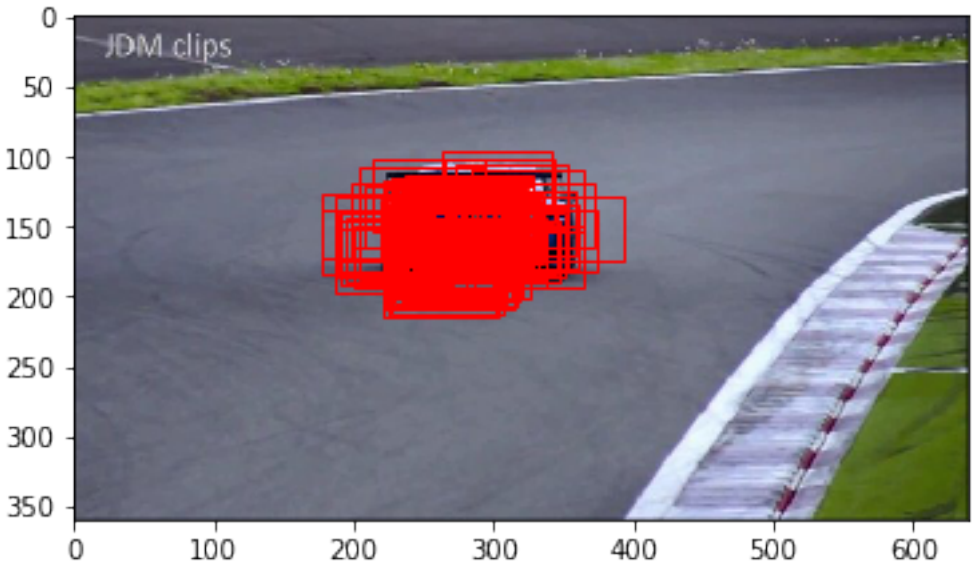
1.4 Exercise 1.2 - Mean Shift

- Implementiere die Verschiebung und des ROI wie in der Vorlesung beschrieben. Teste den Algorithmus auf den Bildfolgen “images/racecar/*.jpg" oder “images/taco/*.jpg“. Wähle das Tracking-Fenster geeignet (zur Reduktion der DateigröSSe habe ich die Bilder um Faktor 2 verkleinert, d.h. die ROI von oben muss entsprechend angepasst werden).
- Zeichne als Ausgabe die Trajektorie (die Bewegungsspur) der Objekte, wie durch CAMSHIFT zurückgegeben. (**RESULT**)

```
In [107]: from scipy import ndimage
          nbins = 25 #20 to 80
```

```
histogramCar = createColorHistogram(imageCar, nbins)
searchBlock = (10, 10)
#io.imshow(image[260:350, 480:640])
#oberer Rand:unterer Rand, linkerRand:rechter Rand

def meanShift(probImage, initialLeftCorner, initialWidth, initialHeight, out = plt):
    initX, initY = initialLeftCorner
    relativeCenter = (initialWidth/2, initialHeight/2)
    searchWindow = probImage[initY:initY+initialHeight,
                             initX: initX+initialWidth]
    lastLeftCorner = initialLeftCorner
    threshold = 4
    shiftAmount = threshold + 10
    #drawROI(probImage, lastLeftCorner[0], lastLeftCorner[1], initialWidth, initialH

    while shiftAmount > threshold:
        x,y,z = ndimage.measurements.center_of_mass(searchWindow)
        newRelativeCenter = (y, x) # https://github.com/scipy/scipy/issues/3040

        if not math.isnan(x) and not math.isnan(y):
            newShift = [new - old for new, old in zip(newRelativeCenter, relativeCent
        else:
            newShift = [0, 0]

        newLeftCorner = [int(round(old_corner+ shift)) for old_corner, shift in zip(

        shiftAmount = math.sqrt(newShift[0]**2 + newShift[1]**2)
        searchWindow = probImage[newLeftCorner[1]:
                                 newLeftCorner[1]+initialHeight,
                                 newLeftCorner[0]:
                                 newLeftCorner[0]+initialWidth]
        lastLeftCorner = newLeftCorner
    return (max(0,lastLeftCorner[0]), max(0, lastLeftCorner[1]))


def meanshift():
    lastFrame = io.imread('images/racecar/151.jpeg')
    sizeX = 80
    sizeY = 45
    fileNumber = []
    leftCorner = (240, 130)

    for i in range(10):
        fileNumber.append('00{0}'.format(str(i)))
    for i in range(11, 99):
        fileNumber.append('0{0}'.format(str(i)))
    for i in range(100, 152):
        fileNumber.append('{0}'.format(str(i)))
```

```
    for fileNr in fileNumber:
        file = 'images/racecar/{0}.jpeg'.format(fileNr)
        currentFrame = io.imread(file)
        probDistr = createProbDistribution(currentFrame, histogramCar, nbins, MIN_SA
        x, y = meanShift(probDistr, (240, 130), 80, 45)
        drawROI(lastFrame, x, y, 80, 45)

        #plt.gca().clear()
        #drawROI(currentFrame, x, y, 80, 45)
        #plt.savefig('results/meanshift_car/{0}'.format(fileNr)) # todo

meanshift()
```



1.5 Exercise 1.2 - CAMSHIFT

- erweitere Deinen Algorithmus um die Anpassung der GröSSe des ROI und das Finden der Objektorientierung
- führe den Algorithmus wieder auf eine der Bildfolgen aus und zeichne eine Ellipse auf das Bild, die die gefundenen Parameter repräsentiert (**RESULT**)

```
In [108]: nbins = 25 #20 to 80
          histogramCar = createColorHistogram(imageCar, nbins)
          searchBlock = (10, 10)

          def calculateWindowWidth(probImage):
              zerothMoment = np.sum(probImage[:, :, 0])
```

```
            try:
                maxIntensity = np.amax(probImage)
            except ValueError:
                maxIntensity = 0.1
            return round(2*math.sqrt(zerothMoment/max(maxIntensity, 0.1)))


        def camshiftFor(currentFrame,initialLeftCorner, initialWidth, initialHeight, out=plt
            probDistr = createProbDistribution(currentFrame, histogramCar, nbins, MIN_SATURA
            leftX,leftY = meanShift(probDistr, initialLeftCorner, initialWidth, initialHeigh

            searchWindow = probDistr[leftY:
                                     leftY+initialHeight,
                                     leftX:
                                     leftX+initialWidth]
            size = calculateWindowWidth(searchWindow)
            return leftX, leftY, int(round(size*1.4)), size


        def camshift():
            lastFrame = io.imread('images/racecar/151.jpeg')
            sizeX = 80
            sizeY = 45
            fileNames = []
            leftCorner = (240, 130)

            for i in range(10):
                fileNames.append('00{0}.jpeg'.format(str(i)))
            for i in range(11, 99):
                fileNames.append('0{0}.jpeg'.format(str(i)))
            for i in range(100, 152):
                fileNames.append('{0}.jpeg'.format(str(i)))


            for fileNr in fileNames:
                file = 'images/racecar/{0}'.format(fileNr)
                currentFrame = io.imread(file)
                leftX, leftY, sizeX, sizeY = camshiftFor(currentFrame,leftCorner, sizeX, siz
                leftCorner = (leftX, leftY)
                drawROI(lastFrame, leftX, leftY, sizeX, sizeY) # todo

                #plt.gca().clear()
                #drawROI(currentFrame, leftX, leftY, sizeX, sizeY)
                #plt.savefig('results/camshift_car/{0}'.format(fileNr)) # todo

        camshift()

leftX,leftY 238 131
```

```
initialHeight,initialWidth 45 80
leftX,leftY 227 111
initialHeight,initialWidth 79 111
leftX,leftY 234 107
initialHeight,initialWidth 82 115
leftX,leftY 235 112
initialHeight,initialWidth 81 113
leftX,leftY 225 109
initialHeight,initialWidth 88 123
leftX,leftY 228 110
initialHeight,initialWidth 88 123
leftX,leftY 229 115
initialHeight,initialWidth 87 122
leftX,leftY 235 115
initialHeight,initialWidth 76 106
leftX,leftY 227 112
initialHeight,initialWidth 94 132
leftX,leftY 227 119
initialHeight,initialWidth 87 122
leftX,leftY 227 121
initialHeight,initialWidth 84 118
leftX,leftY 223 122
initialHeight,initialWidth 86 120
leftX,leftY 212 129
initialHeight,initialWidth 91 127
leftX,leftY 220 135
initialHeight,initialWidth 84 118
leftX,leftY 220 138
initialHeight,initialWidth 96 134
leftX,leftY 221 139
initialHeight,initialWidth 91 127
leftX,leftY 219 137
initialHeight,initialWidth 94 132
leftX,leftY 218 137
initialHeight,initialWidth 94 132
leftX,leftY 216 133
initialHeight,initialWidth 97 136
leftX,leftY 225 140
initialHeight,initialWidth 89 125
leftX,leftY 210 136
initialHeight,initialWidth 101 141
leftX,leftY 207 135
initialHeight,initialWidth 102 143
leftX,leftY 211 139
initialHeight,initialWidth 104 146
leftX,leftY 197 139
initialHeight,initialWidth 107 150
leftX,leftY 182 135
```

```
initialHeight,initialWidth 117 164
leftX,leftY 171 133
initialHeight,initialWidth 121 169
leftX,leftY 179 143
initialHeight,initialWidth 114 160
leftX,leftY 174 145
initialHeight,initialWidth 105 147
leftX,leftY 170 139
initialHeight,initialWidth 124 174
leftX,leftY 180 133
initialHeight,initialWidth 116 162
leftX,leftY 184 132
initialHeight,initialWidth 120 168
leftX,leftY 188 128
initialHeight,initialWidth 119 167
leftX,leftY 200 128
initialHeight,initialWidth 120 168
leftX,leftY 190 124
initialHeight,initialWidth 114 160
leftX,leftY 187 111
initialHeight,initialWidth 127 178
leftX,leftY 186 99
initialHeight,initialWidth 127 178
leftX,leftY 173 94
initialHeight,initialWidth 137 192
leftX,leftY 189 85
initialHeight,initialWidth 134 188
leftX,leftY 201 83
initialHeight,initialWidth 135 189
leftX,leftY 202 86
initialHeight,initialWidth 130 182
leftX,leftY 214 88
initialHeight,initialWidth 131 183
leftX,leftY 215 82
initialHeight,initialWidth 134 188
leftX,leftY 222 90
initialHeight,initialWidth 138 193
leftX,leftY 236 85
initialHeight,initialWidth 139 195
leftX,leftY 226 75
initialHeight,initialWidth 162 227
leftX,leftY 219 77
initialHeight,initialWidth 170 238
leftX,leftY 218 76
initialHeight,initialWidth 171 239
leftX,leftY 233 95
initialHeight,initialWidth 179 251
leftX,leftY 226 99
```

```
initialHeight,initialWidth 189 265
leftX,leftY 205 135
initialHeight,initialWidth 221 309
leftX,leftY 168 110
initialHeight,initialWidth 265 371
leftX,leftY 137 104
initialHeight,initialWidth 286 400
leftX,leftY 150 104
initialHeight,initialWidth 305 427
leftX,leftY 162 98
initialHeight,initialWidth 312 437
leftX,leftY 152 112
initialHeight,initialWidth 300 420
leftX,leftY 147 108
initialHeight,initialWidth 312 437
leftX,leftY 118 107
initialHeight,initialWidth 323 452
leftX,leftY 93 93
initialHeight,initialWidth 344 482
leftX,leftY 45 83
initialHeight,initialWidth 360 504
leftX,leftY 38 78
initialHeight,initialWidth 359 503
leftX,leftY 97 54
initialHeight,initialWidth 366 512
leftX,leftY 88 76
initialHeight,initialWidth 305 427
leftX,leftY 54 76
initialHeight,initialWidth 305 427
leftX,leftY 22 71
initialHeight,initialWidth 305 427
leftX,leftY 0 47
initialHeight,initialWidth 323 452
leftX,leftY 0 34
initialHeight,initialWidth 335 469
leftX,leftY 29 27
initialHeight,initialWidth 331 463
leftX,leftY 57 17
initialHeight,initialWidth 347 486
leftX,leftY 46 5
initialHeight,initialWidth 360 504
leftX,leftY 48 3
initialHeight,initialWidth 366 512
leftX,leftY 33 2
initialHeight,initialWidth 380 532
leftX,leftY 19 5
initialHeight,initialWidth 380 532
leftX,leftY 11 14
```

```
initialHeight,initialWidth 369 517
leftX,leftY 0 16
initialHeight,initialWidth 369 517
leftX,leftY 0 16
initialHeight,initialWidth 354 496
leftX,leftY 8 18
initialHeight,initialWidth 347 486
leftX,leftY 2 1
initialHeight,initialWidth 344 482
leftX,leftY 16 12
initialHeight,initialWidth 318 445
leftX,leftY 0 27
initialHeight,initialWidth 304 426
leftX,leftY 0 29
initialHeight,initialWidth 322 451
leftX,leftY 0 31
initialHeight,initialWidth 331 463
leftX,leftY 0 28
initialHeight,initialWidth 344 482
leftX,leftY 0 25
initialHeight,initialWidth 338 473
leftX,leftY 0 18
initialHeight,initialWidth 343 480
leftX,leftY 0 29
initialHeight,initialWidth 340 476
leftX,leftY 0 24
initialHeight,initialWidth 349 489
leftX,leftY 0 23
initialHeight,initialWidth 347 486
leftX,leftY 0 31
initialHeight,initialWidth 325 455
leftX,leftY 0 38
initialHeight,initialWidth 216 302
leftX,leftY 17 10
initialHeight,initialWidth 145 203
leftX,leftY 87 94
initialHeight,initialWidth 96 134
leftX,leftY 8 117
initialHeight,initialWidth 139 195
leftX,leftY 27 139
initialHeight,initialWidth 161 225
leftX,leftY 26 139
initialHeight,initialWidth 163 228
leftX,leftY 0 147
initialHeight,initialWidth 137 192
leftX,leftY 0 148
initialHeight,initialWidth 147 206
leftX,leftY 0 153
```

```
initialHeight,initialWidth 136 190
leftX,leftY 0 160
initialHeight,initialWidth 136 190
leftX,leftY 25 176
initialHeight,initialWidth 146 204
leftX,leftY 28 193
initialHeight,initialWidth 137 192
leftX,leftY 1 196
initialHeight,initialWidth 141 197
leftX,leftY 0 176
initialHeight,initialWidth 155 217
leftX,leftY 0 158
initialHeight,initialWidth 162 227
leftX,leftY 0 136
initialHeight,initialWidth 158 221
leftX,leftY 0 120
initialHeight,initialWidth 158 221
leftX,leftY 0 114
initialHeight,initialWidth 162 227
leftX,leftY 0 120
initialHeight,initialWidth 168 235
leftX,leftY 0 122
initialHeight,initialWidth 167 234
leftX,leftY 0 111
initialHeight,initialWidth 177 248
leftX,leftY 0 107
initialHeight,initialWidth 177 248
leftX,leftY 18 91
initialHeight,initialWidth 178 249
leftX,leftY 40 64
initialHeight,initialWidth 198 277
leftX,leftY 57 64
initialHeight,initialWidth 210 294
leftX,leftY 65 65
initialHeight,initialWidth 203 284
leftX,leftY 89 72
initialHeight,initialWidth 204 286
leftX,leftY 114 73
initialHeight,initialWidth 201 281
leftX,leftY 115 74
initialHeight,initialWidth 206 288
leftX,leftY 220 69
initialHeight,initialWidth 218 305
leftX,leftY 214 58
initialHeight,initialWidth 232 325
leftX,leftY 210 53
initialHeight,initialWidth 234 328
leftX,leftY 200 53
```

```
initialHeight,initialWidth 232 325
leftX,leftY 222 55
initialHeight,initialWidth 228 319
leftX,leftY 241 56
initialHeight,initialWidth 221 309
leftX,leftY 243 56
initialHeight,initialWidth 225 315
leftX,leftY 241 69
initialHeight,initialWidth 215 301
leftX,leftY 260 76
initialHeight,initialWidth 203 284
leftX,leftY 256 82
initialHeight,initialWidth 195 273
leftX,leftY 229 91
initialHeight,initialWidth 194 272
leftX,leftY 218 93
initialHeight,initialWidth 186 260
leftX,leftY 211 100
initialHeight,initialWidth 181 253
leftX,leftY 204 109
initialHeight,initialWidth 169 237
leftX,leftY 224 118
initialHeight,initialWidth 152 213
leftX,leftY 205 120
initialHeight,initialWidth 145 203
leftX,leftY 205 117
initialHeight,initialWidth 155 217
leftX,leftY 205 107
initialHeight,initialWidth 151 211
leftX,leftY 208 99
initialHeight,initialWidth 154 216
leftX,leftY 205 94
initialHeight,initialWidth 145 203
leftX,leftY 203 79
initialHeight,initialWidth 153 214
leftX,leftY 214 75
initialHeight,initialWidth 148 207
leftX,leftY 220 72
initialHeight,initialWidth 145 203
leftX,leftY 214 57
initialHeight,initialWidth 150 210
leftX,leftY 215 59
initialHeight,initialWidth 144 202
leftX,leftY 219 56
initialHeight,initialWidth 144 202
leftX,leftY 197 60
initialHeight,initialWidth 150 210
leftX,leftY 195 62
```

```
initialHeight,initialWidth 138 193
leftX,leftY 181 61
initialHeight,initialWidth 145 203
leftX,leftY 171 61
initialHeight,initialWidth 148 207
leftX,leftY 175 70
initialHeight,initialWidth 140 196
leftX,leftY 181 69
initialHeight,initialWidth 134 188
leftX,leftY 190 66
initialHeight,initialWidth 137 192
```