W pierwszej części ćwiczenia należy zapoznać się z algorytmem transformacji Hough, (zob. https://en.wikipedia.org/wiki/Hough_transform , https://pl.wikipedia.org/wiki/Transformacja_Hougha).
Należy zapoznać się z pracą pt. „Use of the Hough Transform to Detect Lines and Curves in Pictures", 1972, zob. folder ./Hough-Line-Detection/HoughTransformPaper.pdf.
Następnie należy zapoznać się ze skryptem find_hough_lines.py do detekcji linii na obrazach.
W oparciu o skrypt find_hough_lines.py należy przygotować notebook find_hough_lines.ipynb w którym należy zamieścić obrazy ze znalezionymi liniami dla obrazów imgs/ex1.png, imgs/ex2.png, imgs/ex3.png, dla różnych parametrów (dyskretyzacji) rho-theta.
Należy przedstawić także wyniki uzyskane w oparciu o implementację transformaty Hough z biblioteki OpenCV (cv2).
Przykładowe wywołanie:

```
# Use canny edge detection
edges = cv2.Canny(gray,50,150,apertureSize=3)

# Apply HoughLinesP method
lines_list =[]
lines = cv2.HoughLinesP(
        edges,              # Input edge image
        1,                  # Distance resolution in pixels
        np.pi/180,          # Angle resolution in radians
        threshold=100,      # Min number of votes for valid line
        minLineLength=5,    # Min allowed length of line
        maxLineGap=10       # Max allowed gap between line for joining them
        )

# Iterate over points
for points in lines:
     # Extracted points nested in the list
   x1,y1,x2,y2=points[0]
   # Draw the lines joing the points
   # On the original image
   cv2.line(image,(x1,y1),(x2,y2),(0,255,0),2)
   # Maintain a simples lookup list for points
   lines_list.append([(x1,y1),(x2,y2)])
```

Wyniki uzyskane w oparciu o cv2.HoughLinesP należy zamieścić w notebook find_hough_lines.ipynb .
W oparciu o find_hough_lines.ipynb należy wygenerować find_hough_lines.pdf

W drugiej części ćwiczenia zaleźy zapoznać się algorytmem RANSAC (https://en.wikipedia.org/wiki/Random_sample_consensus).
Należy przestudiować skrypt zamieszczony na wspomnianej stronie, a także skrypt RANSAC_devel.py.
Następnie należy zapoznać się z implementacją algorytmu RANSAC z biblioteki scikit-image:
https://scikit-image.org/docs/stable/auto_examples/transform/plot_ransac.html
W oparciu o skrypt RANSAC_devel.py należy przygotować notebook RANSAC_devel.ipynb w którym należy także zamieścic wyniki uzyskane przez RANSAC z biblioteki scikit-image.
W oparciu o RANSAC_devel.ipynb należy wygenerować RANSAC_devel.pdf

W trzeciej części ćwiczenia należy wstępnie zapoznać się z siecią moveNet do estymacji pozy człowieka, zob. folder moveNet.

W czwartej części ćwiczenia należy wstępnie zapoznać się z metodami detekcji twarzy, a w szczególności anonimizacji twarzy, zob. skrypty w folderze: faceBluring.