2. Трансформації (Transformations)

1.Making your own linear filter (Створення власних лінійних фільтрів!)

код:

```
import cv2
import numpy as np
img=cv2.imread("/home/rodion/yuliia0/aboba/cv/trans/dog.jpg")
cv2.imshow ("original", img)

ker = np.ones((3, 3), dtype=np.float32)
ker /= (3 * 3)
dst = cv2.filter2D (img, -1, ker)
cv2.imshow ("ker 3", dst)

ker = np.ones((7, 7), dtype=np.float32)
ker /= (7 * 7)
dst = cv2.filter2D (img, -1, ker)
cv2.imshow ("ker 7", dst)

ker = np.ones((11, 11), dtype=np.float32)
ker /= (11 * 11)
dst = cv2.filter2D (img, -1, ker)
cv2.imshow ("ker 11", dst)

cv2.waitKey (0)
```

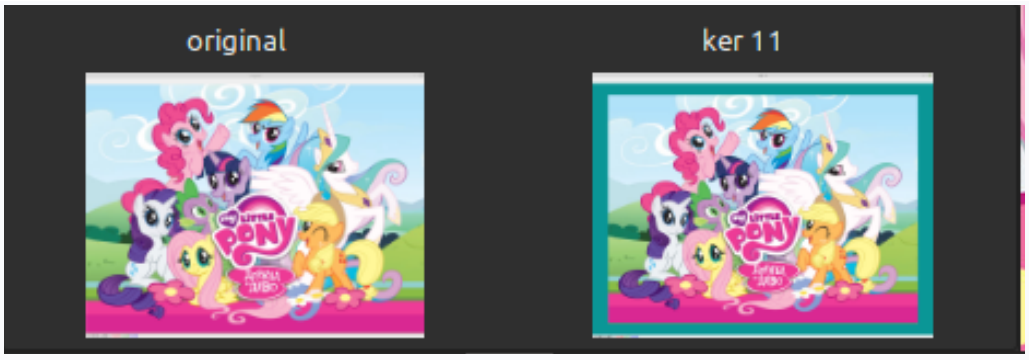
результат:

2. Додавання рамок до зображень

код:

```
import cv2

img=cv2.imread("/home/rodion/yuliia0/aboba/cv/trans/my.png")
cv2.imshow ("original", img)

bottom =top = int(0.05 * img.shape[0])
right =left = int(0.05 * img.shape[1])
dst = cv2.copyMakeBorder(img, top, bottom, left,
right,cv2.BORDER_CONSTANT, None, [150,150,10])
cv2.imshow("ker 11", dst)

dst = cv2.copyMakeBorder(img, top, bottom, left,
right,cv2.BORDER_REPLICATE, None, [100,150,150])
cv2.imshow("ker 12", dst)

cv2.waitKey(0)
```

результат:



(не влізало зображення і аби було видно всю рамку кинула так)

3. Похідні Sobel (Sobel Derivatives)

код:

```
import cv2

img=cv2.imread("/home/rodion/yuliia0/aboba/cv/trans/my.jpg")

img =cv2.GaussianBlur(img, (3, 3),0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
x = cv2.Sobel(img, cv2.CV_8U, 1, 0, 3, 0, cv2.BORDER_REPLICATE)
y = cv2.Sobel(img, cv2.CV_8U, 0, 1, 3, 0, cv2.BORDER_REPLICATE)
grad = cv2.addWeighted(x, 0.5, y, 0.5, 0)
cv2.imshow("2", grad)
cv2.waitKey(0)
```

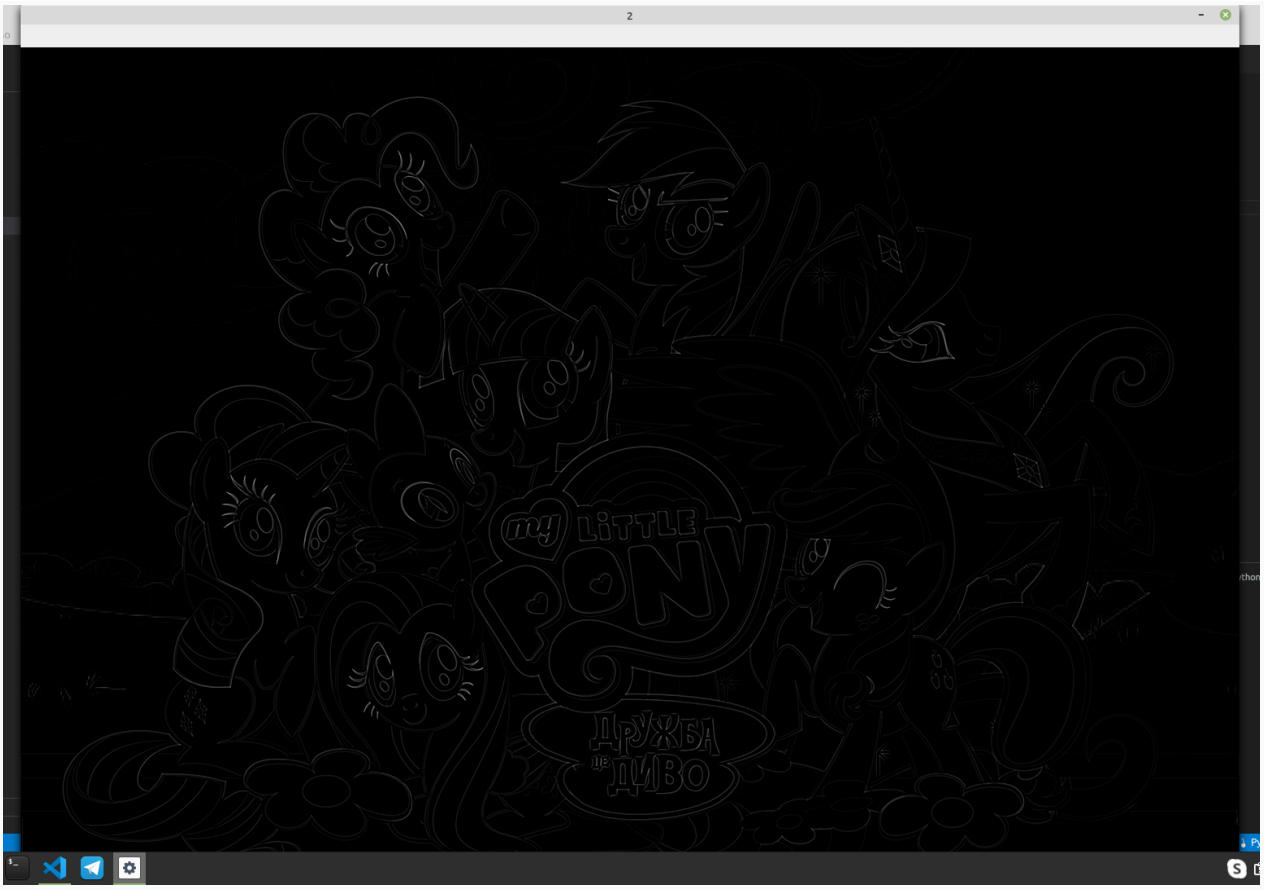
результат:

4. Оператор Лапласа (Laplace Operator)

код:

```
import cv2

img=cv2.imread("/home/rodion/yuliia0/aboba/cv/trans/my.jpg")

img =cv2.GaussianBlur(img, (3, 3),0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

dst = cv2.Laplacian(img, cv2.CV_8U, 3)

dst = cv2.convertScaleAbs(dst)
cv2.imshow("2", dst)
cv2.waitKey(0)
```

Результат:



5. Canny Edge Detector

код:

```
import cv2
img=cv2.imread("/home/rodion/yuliia0/aboba/cv/trans/my.jpg")

img =cv2.GaussianBlur(img, (3, 3),0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
det = cv2.Canny(img, 5, 5*3, 3)

cv2.imshow("2", det)
cv2.waitKey(0)
```

Результат:

6.Трансформація лінії Хафа (Hough Line Transform)

код:

```
import cv2
import numpy as np
import math
img=cv2.imread("/home/rodion/yuliia0/aboba/cv/trans/my.jpg")
cv2.imshow("original", img)
img = cv2.GaussianBlur(img, (3, 3),0)
img=g = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
img = cv2.Canny(img, 50, 200, 3)
```

```
lines = cv2.HoughLines(img, 1, np.pi / 180, 150, None)

for i in range(0, len(lines)):
a = math.cos(lines[i][0][1])
b = math.sin(lines[i][0][1])
x0 = a * lines[i][0][0]
y0 = b * lines[i][0][1]
pt1 = (int(x0 + 1000*(-b)), int(y0 + 1000*(a)))
pt2 = (int(x0 - 1000*(-b)), int(y0 - 1000*(a)))

cv2.line (g, pt1, pt2, (0,0,255), 3, cv2.LINE_AA)
linesP = cv2.HoughLinesP(img, 1, np.pi / 180, 50, None, 50, 10)
for i in range(0, len(linesP)):
l = linesP[i][0]
cv2.line (g, (l[0], l[1]), (l[2], l[3]), (0,0,0), 3, cv2.LINE_AA)
cv2.imshow("Standard Hough Line Transform", g)
cv2.imshow("Probabilistic Line Transform", g)

cv2.waitKey(0)
```

результат:


Standard Hough Line Transform
my LITTLE PONY
ДРУЖБА ЦЕ ДИВО

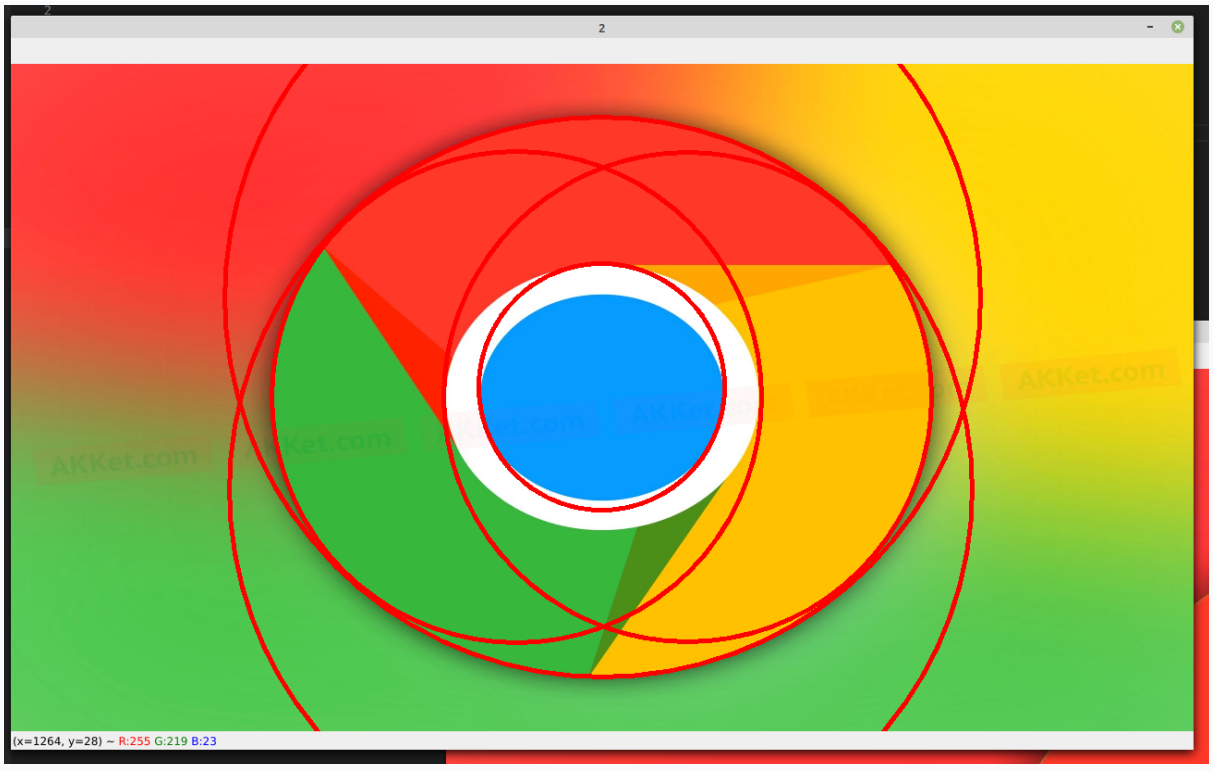

Probabilistic Line Transform
my LITTLE PONY
ДРУЖБА ЦЕ ДИВО

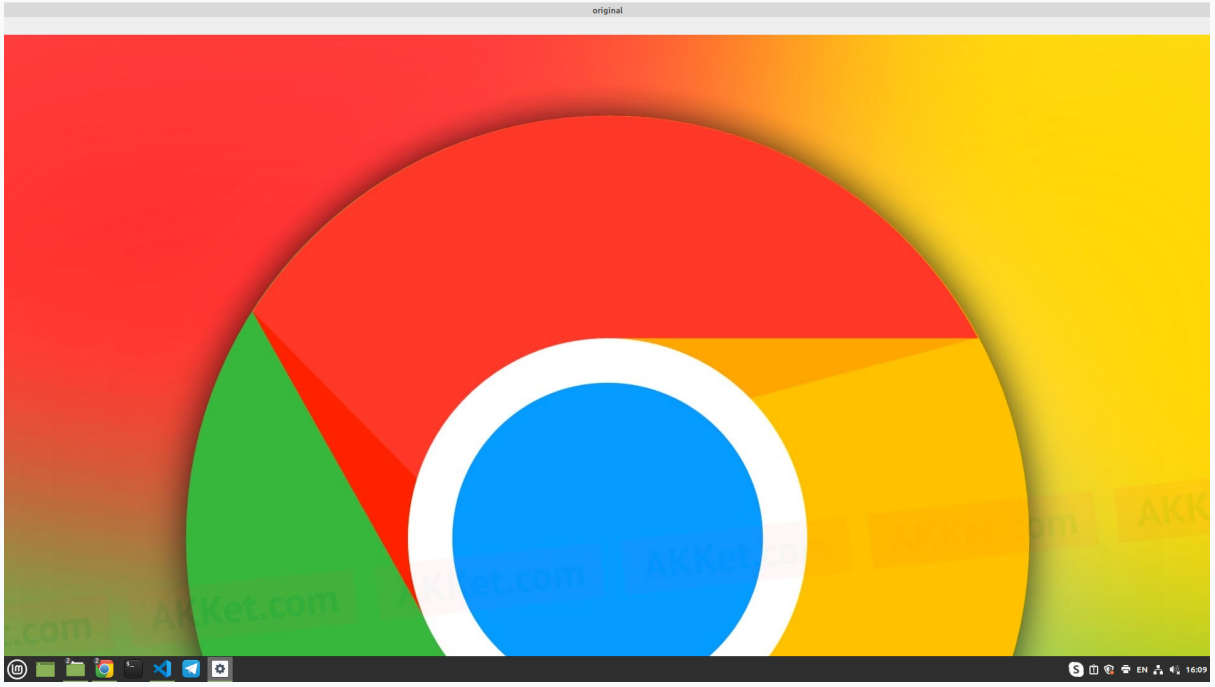
7. Перетворення кола Хафа (Hough Circle Transform)

Код:

```
import cv2

img=g=cv2.imread("/home/rodion/yuliia0/aboba/cv/trans/tr.jpg")
cv2.imshow("original", img)

img = cv2.GaussianBlur(img, (3, 3),0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
img = cv2.resize(img, (1280, 720))
g = cv2.resize(g, (1280, 720))
rows = img.shape[0]
circles = cv2.HoughCircles(img, cv2.HOUGH_GRADIENT, 1, rows / 8,
param1=100, param2=30, minRadius=1, maxRadius=30000)

for i in circles[0, :]:
cv2.circle(g, (int(i[0]), int(i[1])), int(i[2]), (0, 0, 255), 3)

cv2.imshow("2", g)
cv2.waitKey(0)
```

Результат:





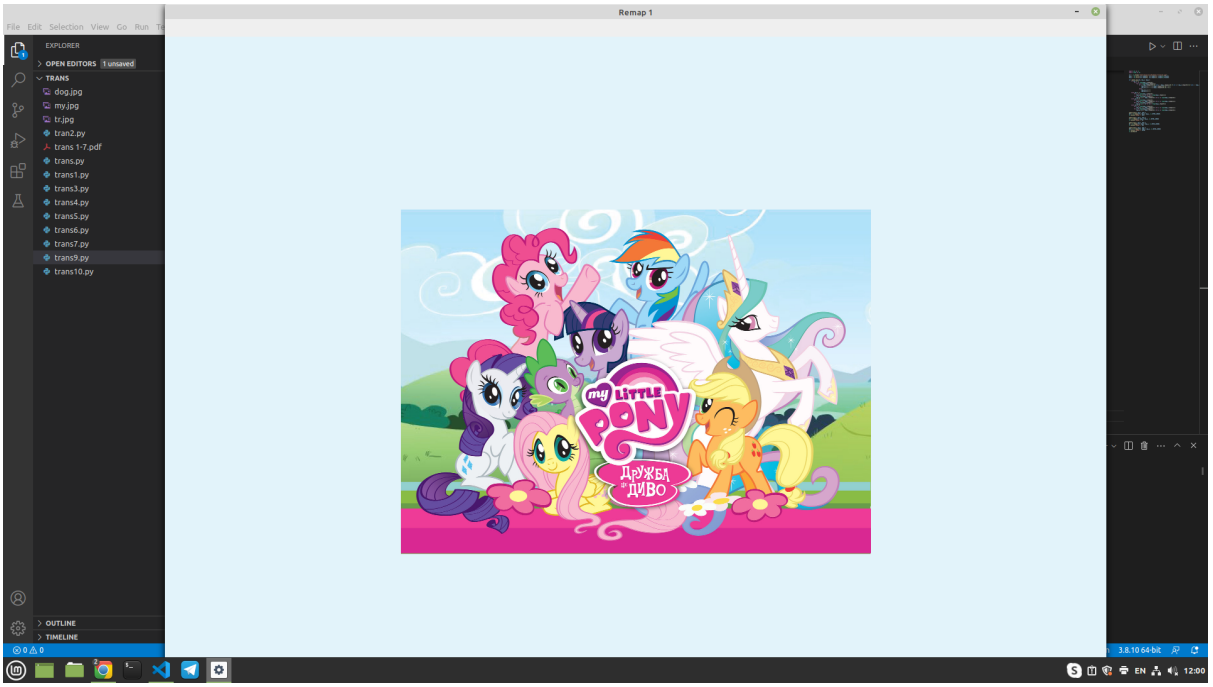
9.Перепризначення

код:

```
import cv2 as cv
import numpy as np

src = cv.imread("/home/rodion/yuliia0/aboba/cv/trans/my.jpg")
map_x = np.zeros((src.shape[0], src.shape[1]), dtype=np.float32)
map_y = np.zeros((src.shape[0], src.shape[1]), dtype=np.float32)

def update_map(ind, map_x, map_y):
if ind == 0:
for i in range(map_x.shape[0]):
for j in range(map_x.shape[1]):
if j > map_x.shape[1]*0.25 and j < map_x.shape[1]*0.75 and i >
map_x.shape[0]*0.25 and i < map_x.shape[0]*0.75:
map_x[i,j] = 2 * (j-map_x.shape[1]*0.25) + 0.5
map_y[i,j] = 2 * (i-map_y.shape[0]*0.25) + 0.5
else:
map_x[i,j] = 0
map_y[i,j] = 0
elif ind == 1:
for i in range(map_x.shape[0]):
map_x[i,:] = [x for x in range(map_x.shape[1])]
for j in range(map_y.shape[1]):
map_y[:,j] = [map_y.shape[0]-y for y in range(map_y.shape[0])]
elif ind == 2:
for i in range(map_x.shape[0]):
map_x[i,:] = [map_x.shape[1]-x for x in range(map_x.shape[1])]
for j in range(map_y.shape[1]):
```

```
map_y[:,j] = [y for y in range(map_y.shape[0])]
elif ind == 3:
for i in range(map_x.shape[0]):
map_x[i,:] = [map_x.shape[1]-x for x in range(map_x.shape[1])]
for j in range(map_y.shape[1]):
map_y[:,j] = [map_y.shape[0]-y for y in range(map_y.shape[0])]

update_map(0, map_x, map_y)
dst = cv.remap(src, map_x, map_y, cv.INTER_LINEAR)
cv.imshow("Remap 1", dst)

update_map(1, map_x, map_y)
d = cv.remap(src, map_x, map_y, cv.INTER_LINEAR)
cv.imshow("Remap 2", d)

update_map(2, map_x, map_y)
ds = cv.remap(src, map_x, map_y, cv.INTER_LINEAR)
cv.imshow("Remap 3", ds)

update_map(3, map_x, map_y)
dstq = cv.remap(src, map_x, map_y, cv.INTER_LINEAR)
cv.imshow("Remap 4", dstq)
cv.waitKey(0)
```

результат:


Remap 1
EXPLORER
OPEN EDITORS 1 unsaved
TRANS
dog.jpg
my.jpg
tr.jpg
tran2.py
trans 1-7.pdf
trans.py
trans1.py
trans3.py
trans4.py
trans5.py
trans6.py
trans7.py
trans9.py
trans10.py
OUTLINE
TIMELINE
my LITTLE PONY
ДРУЖБА Е ЧУДО


Remap 2
EXPLORER
OPEN EDITORS 1 unsaved
TRANS
dog.jpg
my.jpg
tr.jpg
tran2.py
trans 1-7.pdf
trans.py
trans1.py
trans3.py
trans4.py
trans5.py
trans6.py
trans7.py
trans9.py
trans10.py
OUTLINE
TIMELINE
Run or Debug...

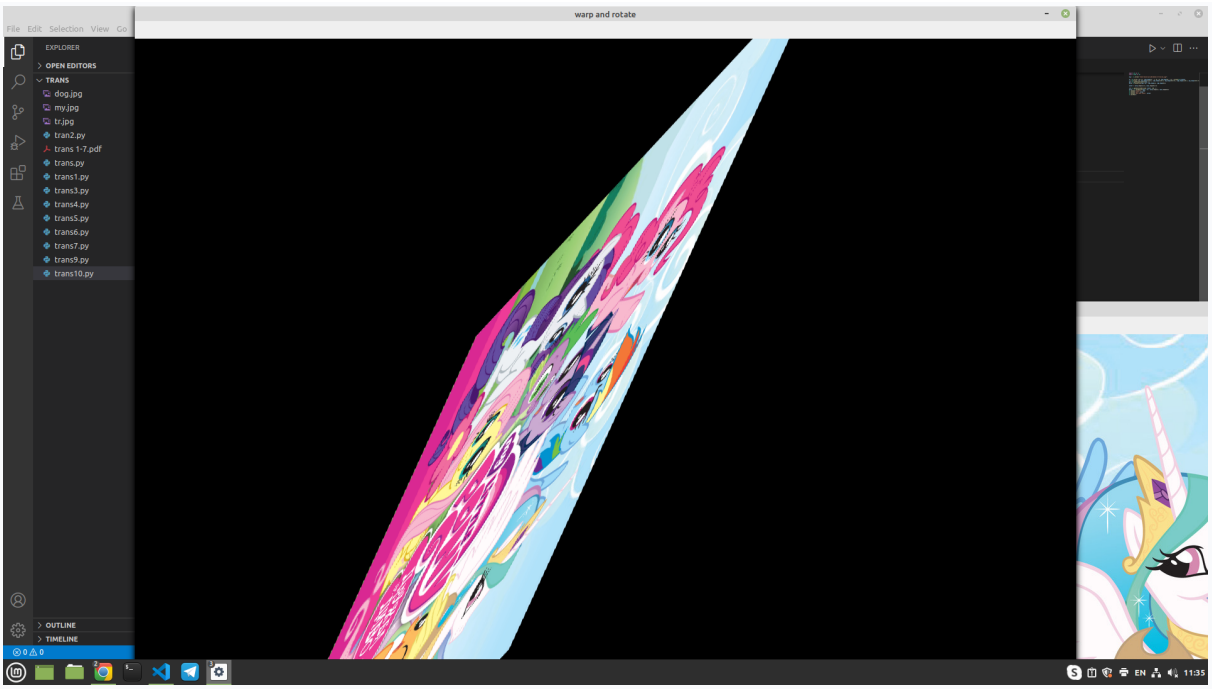
10. Афінні перетворення

код:

```
import cv2 as cv
import numpy as np

img = cv.imread("/home/rodion/yuliia0/aboba/cv/trans/my.jpg")
```

```
im = np.array( [[0, 0], [img.shape[1] - 1, 0], [0, img.shape[0] - 1]]
).astype(np.float32)
i = np.array( [[0, img.shape[1]*0.3], [img.shape[1]*0.8, img.shape[0]*0.5],
[img.shape[1]*0.5, img.shape[0]*0.7]] ).astype(np.float32)
war= cv.getAffineTransform(im, i)
warp= cv.warpAffine(img, war, (img.shape[1], img.shape[0]))

center = (warp.shape[1]//2, warp.shape[0]//2)

rot= cv.getRotationMatrix2D( center, 250, 1 )
warpr = cv.warpAffine(warp, rot, (warp.shape[1], warp.shape[0]))
cv.imshow('original', img)
cv.imshow('warp', warp)
cv.imshow('warp and rotate', warpr)
cv.waitKey()
```

результат:


warp and rotate
EXPLORER
OPEN EDITORS
TRANS
dog.jpg
my.jpg
tr.jpg
tran2.py
trans 1-7.pdf
trans.py
trans1.py
trans3.py
trans4.py
trans5.py
trans6.py
trans7.py
trans9.py
trans10.py
OUTLINE
TIMELINE
EN
11:35