```
import cv2
import math
import numpy as np
# capture frames from a camera
cap = cv2.VideoCapture(0)

# loop runs if capturing has been initialized.
while 1:

    # reads frames from a camera
    ret, img = cap.read()
    rows,cols,p = img.shape
    size = rows,cols,p
    y1 = np.zeros(size, dtype=np.uint8) # defining new image variable set to zero
for filling it up by four copies

         # the below loop form image in the quadrant of (x=0,y=0) taking one of the
four corresponding pixels from img
    for i in range(0,int(rows/2)):
        for j in range(0,int(cols/2)):
                # from the colorset bgr blue and red are equally valued ( calculated
by using all input bgr values)
            y1[i,j,0] = 0.21*img[2*i,2*j,2]+0.72*img[2*i,2*j,1]+0.07*img[2*i,2*j,0]
            y1[i,j,2] = 0.21*img[2*i,2*j,2]+0.72*img[2*i,2*j,1]+0.07*img[2*i,2*j,0]
            y1[i,j,1] =
int((0.21*img[2*i,2*j,2]+0.72*img[2*i,2*j,1]+0.07*img[2*i,2*j,0])/20)

    # the below loop form image in the quadrant of (x=rows-1,y=0) taking one of the
four corresponding pixels from img
    for i in range(int(rows/2),rows):
        for j in range(0,int(cols/2)):
        # from the colorset bgr green and red are equally valued ( calculated by
using all input bgr values)
            y1[i,j,1] =
0.21*img[2*(i-int(rows/2)),2*j,2]+0.72*img[2*(i-int(rows/2)),2*j,1]+0.07*img[2*(i-in
t(rows/2)),2*j,0]
            y1[i,j,2] =
0.21*img[2*(i-int(rows/2)),2*j,2]+0.72*img[2*(i-int(rows/2)),2*j,1]+0.07*img[2*(i-in
t(rows/2)),2*j,0]

# the below loop form image in the quadrant of (x=rows-1,y=cols-1) taking one of the
four corresponding pixels from img
    for i in range(int(rows/2),rows):
        for j in range(int(cols/2),cols):
        # from the colorset bgr blue,green and red are equally valued ( calculated
by using all input bgr values) making gray
            y1[i,j] =
0.21*img[2*(i-int(rows/2)),2*(j-int(cols/2)),2]+0.72*img[2*(i-int(rows/2)),2*(j-int(
cols/2)),1]+0.07*img[2*(i-int(rows/2)),2*(j-int(cols/2)),0]

# the below loop form image in the quadrant of (x=0,y=cols-1) taking one of the four
corresponding pixels from img
    for i in range(0,int(rows/2)):
        for j in range(int(cols/2),cols):
            # pixel values of same image is copied one from four of the pixels
            y1[i,j] = img[2*i,2*(j-int(cols/2))]

    cv2.imshow('img',y1)

    # Wait for Esc key to stop
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

# Close the window
cap.release()

# De-allocate any associated memory usage
cv2.destroyAllWindows()
```