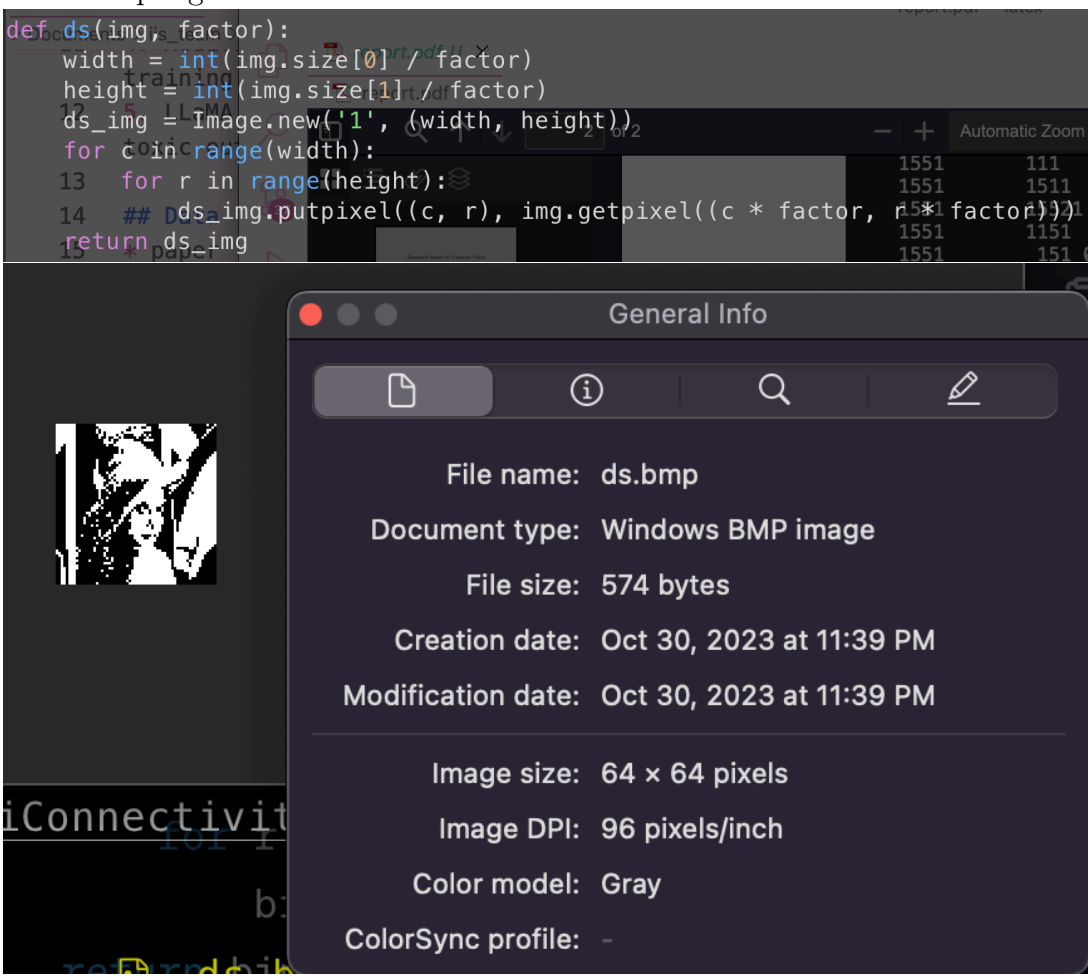
Homework Report for Computer Vision

Yu Xiang, Luo

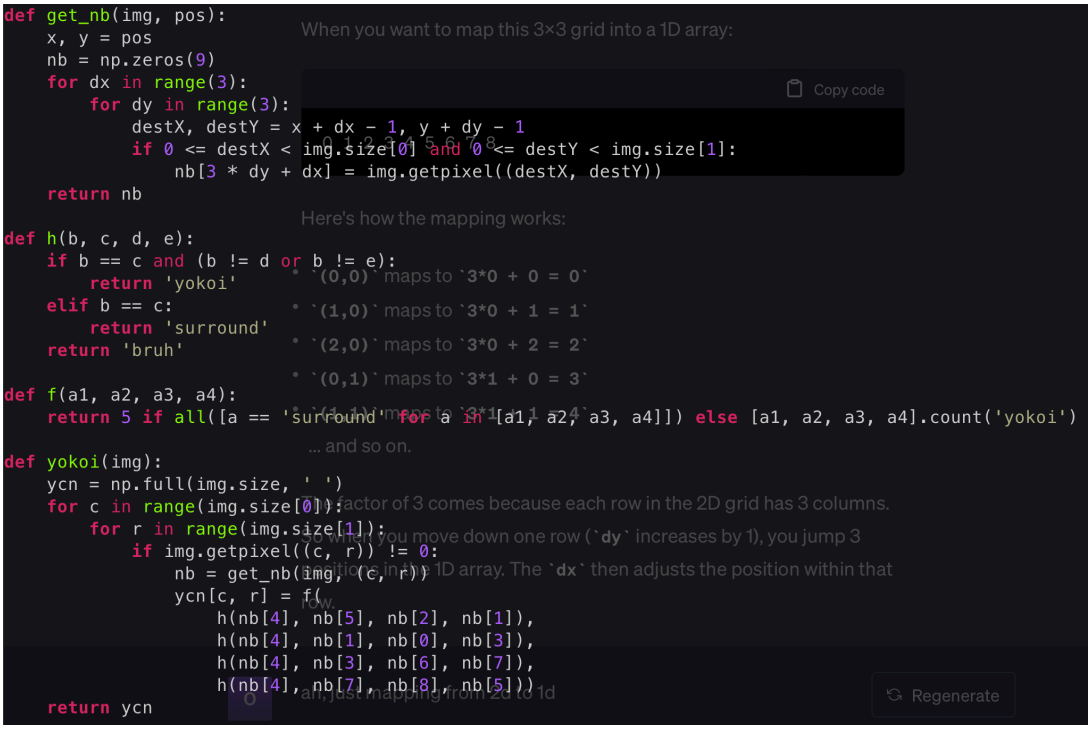
October 31, 2023

You can check this github for more information

(a) Downsampling Lena to 64 × 64:



(b) yokoi:
In this code, we need to record the neighbor pixel, then we check h function to see if two neighbors are the same, or only just one. In the end, we use f to aggregate the infomation.


def get_nb(img, pos):
x, y = pos
nb = np.zeros(9)
for dx in range(3):
for dy in range(3):
destX, destY = x + dx - 1, y + dy - 1
if 0 <= destX < img.size[0] and 0 <= destY < img.size[1]:
nb[3 * dy + dx] = img.getpixel((destX, destY))
return nb
def h(b, c, d, e):
if b == c and (b != d or b != e):
return 'yokoi'
elif b == c:
return 'surround'
return 'bruh'
def f(a1, a2, a3, a4):
return 5 if all([a == 'surround' for a in [a1, a2, a3, a4]]) else [a1, a2, a3, a4].count('yokoi')
def yokoi(img):
ycn = np.full(img.size, ' ')
for c in range(img.size[0]):
for r in range(img.size[1]):
if img.getpixel((c, r)) != 0:
nb = get_nb(img, (c, r))
ycn[c, r] = f(
h(nb[4], nb[5], nb[2], nb[1]),
h(nb[4], nb[1], nb[0], nb[3]),
h(nb[4], nb[3], nb[6], nb[7]),
h(nb[4], nb[7], nb[8], nb[5]))
return ycn
When you want to map this 3×3 grid into a 1D array:
Copy code
0 1 2 3 4 5 6 7 8
Here's how the mapping works:
(0,0) maps to 3*0 + 0 = 0
(1,0) maps to 3*0 + 1 = 1
(2,0) maps to 3*0 + 2 = 2
(0,1) maps to 3*1 + 0 = 3
... and so on.
The factor of 3 comes because each row in the 2D grid has 3 columns.
dy, when you move down one row (dy increases by 1), you jump 3
positions in the 1D array. The dx then adjusts the position within that
row.
Regenerate