M11102122 數位相機實務 HW

Author: M11102122 陳煜翔

Date: 2022/10/20

[TOC]

Introduction

In this task, we will use RGGB CFA raw image to arbitary interpolation algorithm to fill the missing color, and calculate the MSE.

In this case, I use the bilinear algorithm be my interpolation algorithm.

Code Review

```
from PIL import Image, ImageOps
import numpy as np
import cv2
import os

def get_concat_h(im1, im2):
    dst = Image.new('RGB', (im1.width + im2.width, im1.height))
    dst.paste(im1, (0, 0))
    dst.paste(im2, (im1.width, 0))
    return dst

def mse(image_0, image_1):
    MSE_VALUE = np.mean(np.power((image_0 - image_1), 2))
    print("\nMSE Value: {}".format(MSE_VALUE))
    return MSE_VALUE

FILE_PATH = r"C:\Users\Skyler\Desktop\Course\Digital Camera Theory and
Practice\HW1"
FILE_NAME = "test_o.jpg"
FILE_TEST = "test.JPG"

color = {"B":0, "G":1, "R":2}

# image_o = cv2.imread(os.path.join(FILE_PATH, FILE_NAME)) # Read Image from file
image_o = cv2.imread(os.path.join(FILE_PATH, FILE_NAME),
cv2.IMREAD_GRAYSCALE).astype(np.uint8) # Read Image from file
image_PIL = ImageOps.grayscale(Image.open(os.path.join(FILE_PATH, FILE_NAME)))
image_PIL = np.array(image_PIL)

shape_tuple = list(image_o.shape)
shape_tuple.append(3)
shape_tuple = tuple(shape_tuple)
```

```
color_image_cvt = np.zeros(shape_tuple).astype(np.uint8) # Get image_shape
# image_o = cv2.cvtColor(image_o, cv2.COLOR_BGR2RGB) # Convert IMAGE to gray scale



for i in range(image_o.shape[0]):
    for j in range(image_o.shape[1]):
        if(i % 2 == 0): # Even row
            if(j % 2 == 0): # Even column
                color_image_cvt[i][j][color["B"]] = int(image_o[i][j]) # B
            else: # Odd Column
                color_image_cvt[i][j][color["G"]] = int(image_o[i][j]) # G
        else: # Odd row
            if(j % 2 == 0):
                color_image_cvt[i][j][color["G"]] = int(image_o[i][j]) # G
            else:
                color_image_cvt[i][j][color["R"]] = int(image_o[i][j]) # R

cv2.imwrite(os.path.join(FILE_PATH, "RAW_CVT.png"), color_image_cvt)


color_image_bilinear = color_image_cvt.copy()
for i in range(image_o.shape[0]): # Bilinear Interpolation
    for j in range(image_o.shape[1]):

        if(((i > 0) & (j > 0)) & ((i < (image_o.shape[0] - 1)) & (j <
(image_o.shape[1] - 1)))):

            if(i % 2 == 0): # Even row
                if(j % 2 == 0): # Even column -> B
                    color_image_bilinear[i][j][color["G"]] =
(int(color_image_cvt[i][j-1][color["G"]]) + int(color_image_cvt[i][j+1]
[color["G"]]) + int(color_image_cvt[i-1][j][color["G"]]) +
int(color_image_cvt[i+1][j][color["G"]])) / 4 # G
                    color_image_bilinear[i][j][color["R"]] =
(int(color_image_cvt[i-1][j-1][color["R"]]) + int(color_image_cvt[i-1][j+1]
[color["R"]]) + int(color_image_cvt[i+1][j-1][color["R"]]) +
int(color_image_cvt[i+1][j+1][color["R"]])) / 4 # R
                else: # Odd Column -> G
                    color_image_bilinear[i][j][color["B"]] =
(int(color_image_cvt[i][j-1][color["B"]]) + int(color_image_cvt[i][j+1]
[color["B"]])) / 2  # B
                    color_image_bilinear[i][j][color["R"]] =
(int(color_image_cvt[i-1][j][color["R"]]) + int(color_image_cvt[i+1][j]
[color["R"]])) / 2 # R
            else: # Odd row
                if(j % 2 == 0): # Even column -> G
                    color_image_bilinear[i][j][color["B"]] =
(int(color_image_cvt[i-1][j][color["B"]]) + int(color_image_cvt[i+1][j]
[color["B"]])) / 2  # B
                    color_image_bilinear[i][j][color["R"]] =
(int(color_image_cvt[i][j-1][color["R"]]) + int(color_image_cvt[i][j+1]
[color["R"]])) / 2 # R
                else: # Odd Column -> R
```

```
                    color_image_bilinear[i][j][color["G"]] =
(int(color_image_cvt[i][j-1][color["G"]]) + int(color_image_cvt[i][j+1]
[color["G"]]) + int(color_image_cvt[i-1][j][color["G"]]) +
int(color_image_cvt[i+1][j][color["G"]])) / 4 # G
                    color_image_bilinear[i][j][color["B"]] =
(int(color_image_cvt[i-1][j-1][color["B"]]) + int(color_image_cvt[i-1][j+1]
[color["B"]]) + int(color_image_cvt[i+1][j-1][color["B"]]) +
int(color_image_cvt[i+1][j+1][color["B"]])) / 4 # R

cv2.imwrite(os.path.join(FILE_PATH, "RAW_CVT_BILINEAR_RAW.png"),
color_image_bilinear)

color_image_test = cv2.imread(os.path.join(FILE_PATH, FILE_TEST))

MSE_VALUE = mse(color_image_test, color_image_bilinear)

color_image_bilinear = Image.fromarray(cv2.cvtColor(color_image_bilinear,
cv2.COLOR_BGR2RGB))
color_image_test = Image.fromarray(cv2.cvtColor(color_image_test,
cv2.COLOR_BGR2RGB))
color_image_cvt = Image.fromarray(cv2.cvtColor(color_image_cvt,
cv2.COLOR_BGR2RGB))

image_cat = get_concat_h(color_image_test, color_image_cvt)
image_cat = get_concat_h(image_cat, color_image_bilinear)

image_cat.save(os.path.join(FILE_PATH, "result_cat.png"))
```

Experiment Result