Image Formation and Features

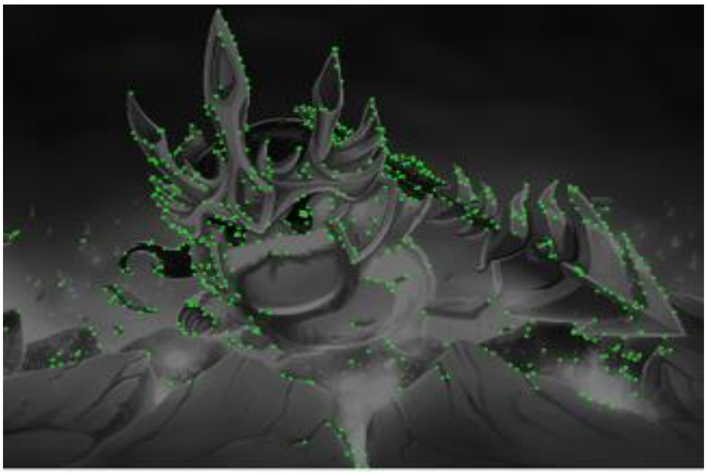
CS655000 Computer Vision Homework 1

Brief

- **Due: Wed, 10/16, 23:59**
- Use **Python** to complete the homework.
- If you encounter any problem, let's discuss on iLMS instead of email.

Part 1. Harris Corner Detection

With the Harris corner detector described in slides (p.79), mark the detected corners on the image.



A. **Functions:**
   a. **gaussian_smooth()**: filter images with Gaussian blur.
   b. **sobel_edge_detection()**: apply the Sobel filters to the blurred images and compute the magnitude and direction of gradient. (You should eliminate weak gradients by proper threshold.)
   c. **structure_tensor()**: use the gradient magnitude above to compute the structure tensor (second-moment matrix).
   d. **nms()**: perform non-maximal suppression on the results above along with appropriate threshold for corner detection.

B. **Results:**
   a. Original image
      i. Gaussian smooth results: σ=5 and kernel size=5 and 10 **(2 images)**
      ii. Sobel edge detection results
         1. magnitude of gradient (Gaussian kernel size=5 and 10) **(2 images)**
         2. direction of gradient (Gaussian kernel size=5 and 10) **(2 images)** (You can choose arbitrary color map to display)
      iii. Structure tensor + NMS results (Gaussian kernel size=10)
         1. window size = 3x3 **(1 image)**
         2. window size = 30x30 **(1 image)**

b. Final results of rotating (by 30°) original images **(1 image)**
c. Final results of scaling (to 0.5x) original images **(1 image)**

C. Report:

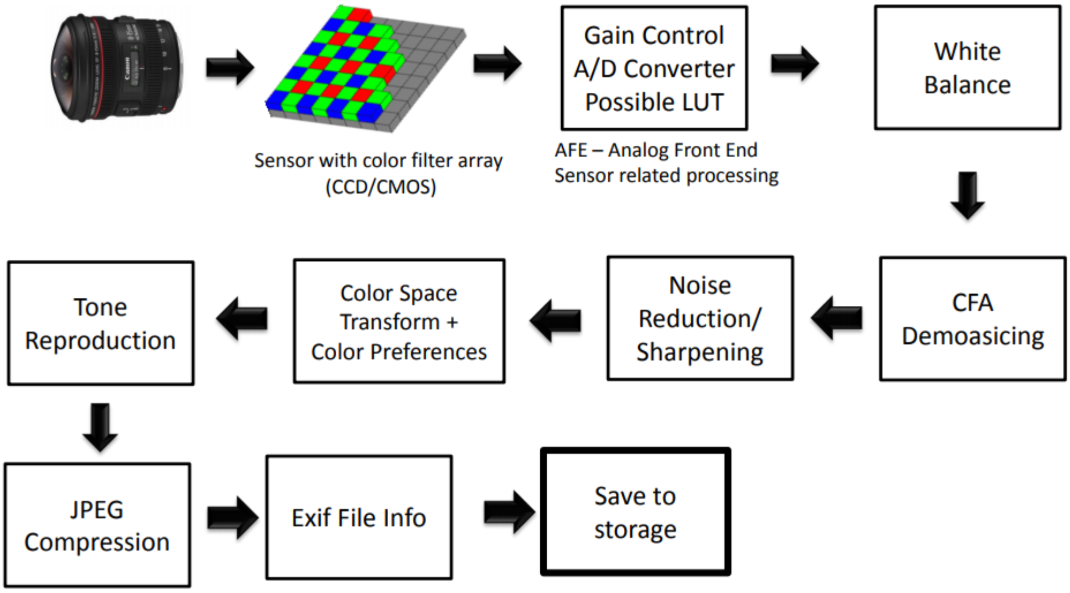
a. Discuss the results of blurred images and detected edges between different kernel sizes of Gaussian filter.
b. Discuss the difference between 3x3 and 30x30 window sizes of structure tensor.
c. Discuss the effect of non-maximal suppression.
d. Discuss the results of rotated and scaled image. Is Harris detector rotation-invariant or scale-invariant? Explain the reason.

D. Notice:

a. You should **NOT** use any functions which can get the result directly in each steps. *(cv2.Sobel, cv2.Laplacian, cv2.cornerHarris, skimg.feature.local_binary_pattern, etc.)*
b. Your code should display and output image results mentioned above.
c. You should provide a **README** file about your execution instructions.

Part 2. Image Sensing Pipeline (ISP)

Image sensing pipeline is a significant process in camera, and our goal is to write a simplified version.



A. Functions

a. **color_correction()**: apply the color correction matrix (CCM) into original image.
b. **generate_wb_mask()**: for generating mask, apply the given red and blue value into appropriate position according to different Bayer patterns.
c. **mosaic()**: discard values of other two channels according to different Bayer patterns.

B. Report

a. Discuss different treatments of different Bayer patterns when
   i. applying white balance mask into original image.
   ii. doing mosaic algorithm.

b. Show the image results of each step as **p.13-14** in hw1_tutorial.pdf.

c. In recent AI de-noising methods, in order to generate paired data for training, we will add synthetic noise to clean image on RAW domain instead of RGB domain. Explain the reason.

C. Notice

a. **All Python packages are allowed to use**.

b. **DO NOT** correct any function names of sample code.

c. We will calculate PSNR of 10 testing image with corresponding parameters to judge the final score.

d. Don't submit any image to iLMS in this part.

D. Install useful Python packages

```
pip install scipy
pip install opencv-python
pip install scikit-image
```

Rubric

- **+40 pts: Harris corner detection results**
- **+40 pts: ISP results**
- **+20 pts: Report**
- **-20 pts for each day after deadline**

Submission

hw1_{Student-ID}.zip

1. hw1_1 **(Folder)**
   - all Python (.py) files
   - results **(Folder)**: contain all image results
   - README
2. hw1_2 **(Folder)**
   - all Python (.py) files
3. hw1_{Student-ID}.pdf