CS 513 Assignment 1 Smear Detection

By: Theo Guidroz, A20426895
Hunter Negron, A20417545
Zhengcong Xiao, A20453141

Content

- Introduction
- Background/motivation
- approach/methodology
- Results
- Conclusion
- References

Introduction

Problem Statement:

An issue mapping companies often encounter is the presence of smears on the lens of the cameras taking pictures to create a street view. To avoid the unnecessary task of retaking pictures because of the presence of a smear, the team will work on a smear detection algorithm.

Introduction

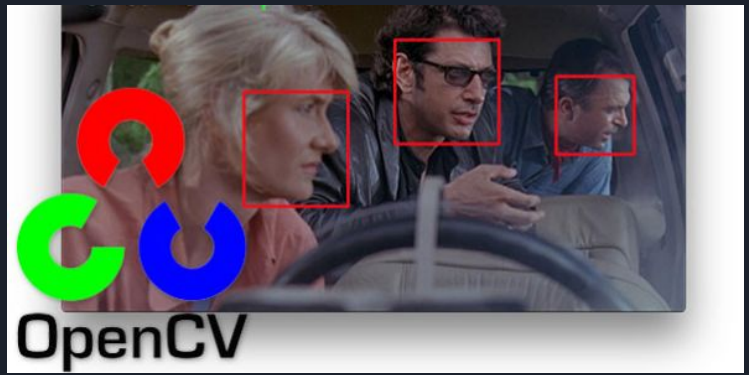
- Our goal is going to detect any smear appear on the camera lens from the giving street view images.
- This goal is going to improve the image quality, to avoid unnecessary dirt or occlusions appear on the lens.
- We resort to standard and well-documented image processing techniques in an attempt to isolate the dirt on the lens.

Background/motivation

1. The smear appears on the lens of camera by different way. It may cause a sharp focus that make the science experiment inaccurate. Therefore, clear the smear from the picture is important before we go to forward science experiment.
2. The intermediate layer between target and camera lens, which is smear, affects the image irradiance in two ways:
    a. Attenuation: the scene radiance is reduced.
    b. Intensification: the intermediate layer has the radiance which can impact the image sensor, either by scattering the light from other directions, or by reflecting the light from the surface of the layer. (Jinwei Gu, P1)

Approach/Methodology

1. OpenCV function used: OpenCV is a large open-source library for computer vision, machine learning and image processing. Through OpenCV, we can identify objects, faces, or handwriting.
2. Environment:
   a. Python3.8.7
   b. Numpy
   c. Pathlib
   d. Opencv-python
   e. glob2



https://www.pyimagesearch.com/2018/07/19/opencv-tutorial-a-guide-to-learn-opencv/

Approach/Methodology

To detect the smear from the given images, we use the following steps:

1. Looking through manually, to make sure where the smear is, and consider how to detect the smear part.
2. Try to read a few images and convert each of them into grayscale, which can help us to notice the potential smear.
3. Blur the image to remove harsh lines

Approach/Methodology

4. Using adaptive threshold (ADAPTIVE_THRESH_MEAN_C)) to further increase smear visibility, which will determines the threshold value of the adjacent pixels.

5. Using canny edge detection

6. Mask Check

7. Find the contours

8. Evaluate contour in contours to see if it a smear or not.

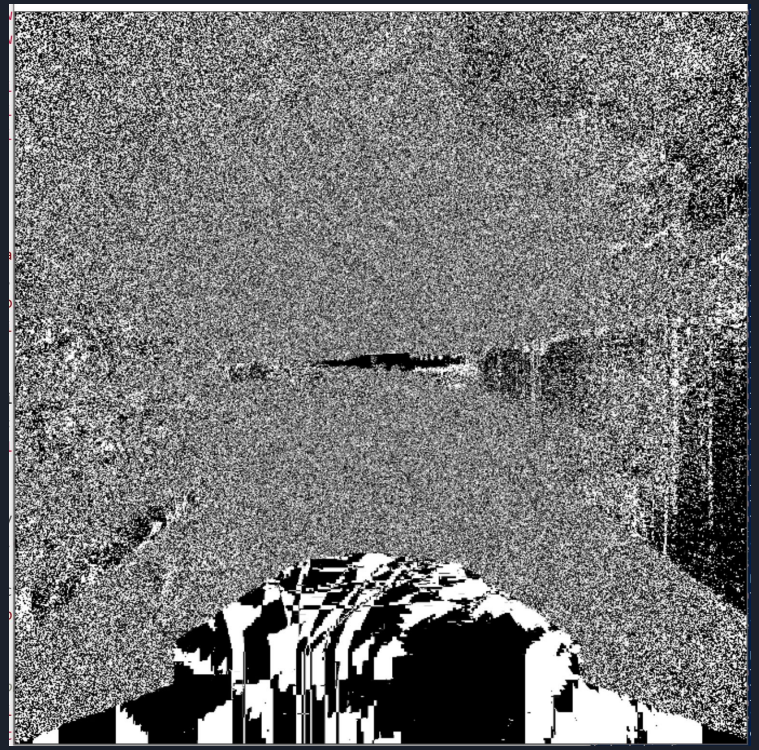
Approach #1: ID the wrong thing

Method

- Read 30 images of cam_0
- Convert to grayscale
- Convert to binary with threshold 120
- Invert (logical NOT)
- Logical XOR all 30 together

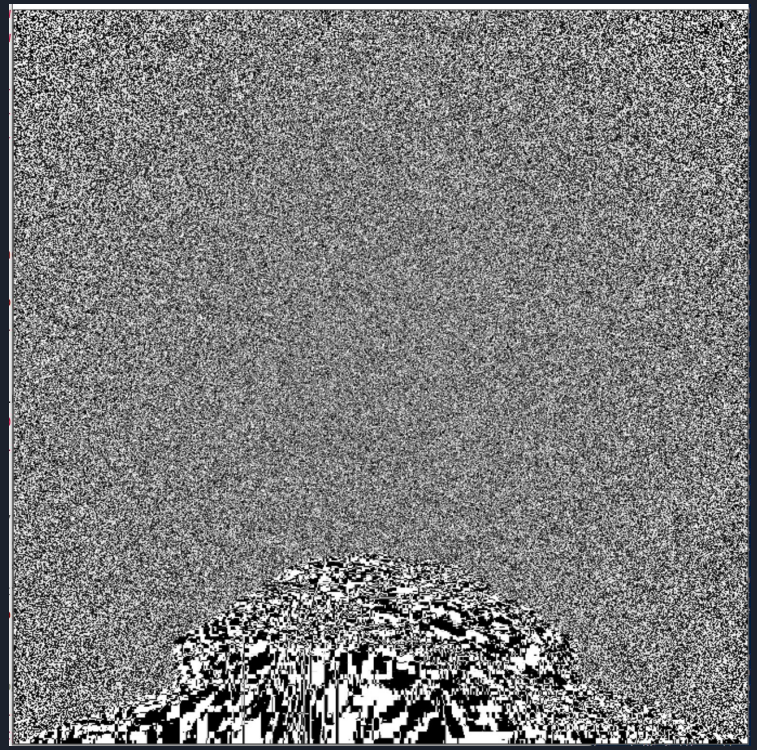
Result

- Exaggerates the blurred car
- Does not identify a smear on the lens



Approach #1: Limitations

- Same method on 200 images instead of 30
- The blur gets lost in the noise after too many images are compiled, indicating that this artifact is not a smear on the lens at all
- This method is not useful since it just turns into noise after a short time



Approach #2: ID the wrong thing, but better

Method

- Read a single image of cam_0
- Canny edge detection with parameters 100, 200
- Blur the result with parameters (100,100)
- Convert to binary with threshold 0

Result

- Clearly identifies the blurred car, but no camera smears

Approach #2: Limitations

- This method does not even work well on identifying the car blur
- Since it also identifies the sky, this only works on the blur in the tunnels
- It still does not correctly identify and lens smears, so this method is not useful at all

Approach #3: ID the right thing

Method

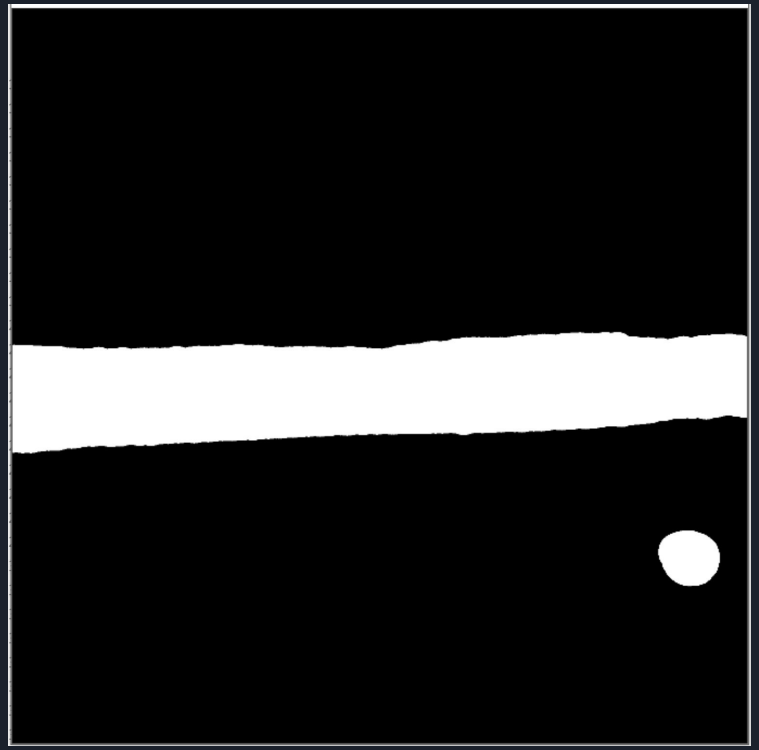
- Read the first N images of cam_3
- Add them together using cv.accumulate()
- Divide the result by N to get an average image
- Convert to grayscale

Results

- Set N = 5000
- The spot on cam_3 is now visible

Approach #3: Limitations

- In an attempt to get a mask of the smear, a binary threshold is performed, then a dilation to recover the original size of the smear.
- However, after much tweaking of parameters and sharpening techniques on the original and averaged images, the mask still includes the horizon line along with the smear.



Final Approach

Assumption:

1. The smear is between a range (200-100)
2. The smear is present on every photo in the folder

Method

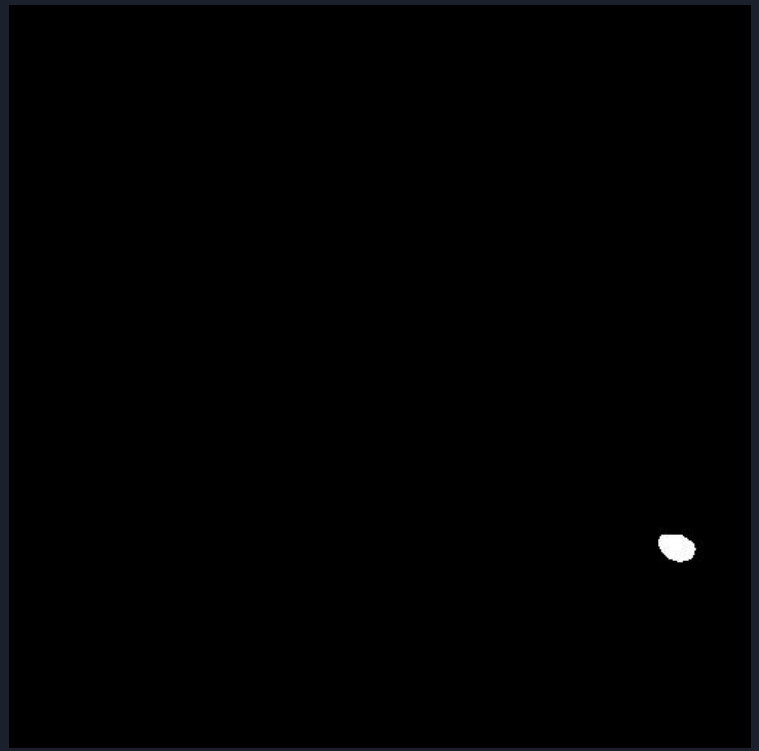
1. Read each image from the cam folder
2. Slightly blur and filter each image
3. Average all images together
4. Convert to grayscale

Here is the averaged grey scaled image for cam_3.

Final Approach

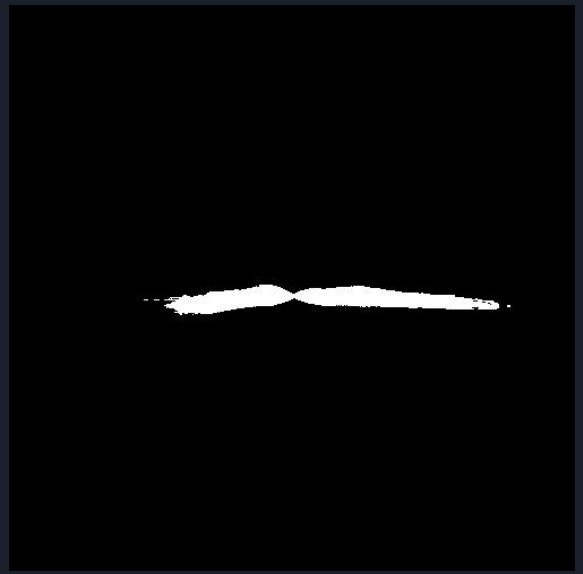
Method (continued)

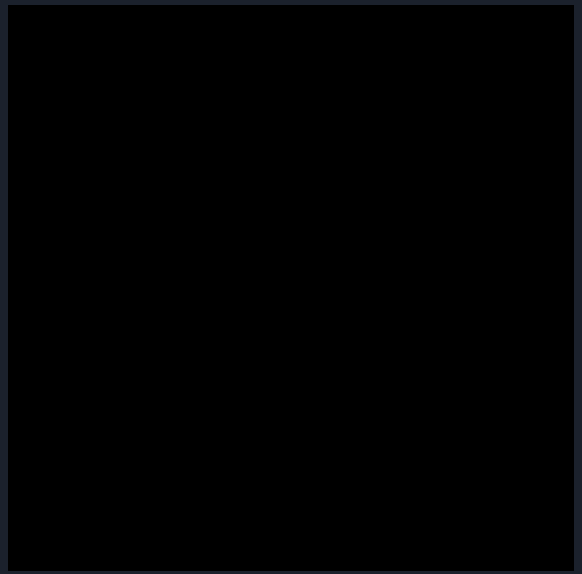
5. Convert to binary with Gaussian Threshold and invert
6. Use cv.findContours() to get smear candidates
7. Check each contour against size restrictions
8. If a smear is present, create a video using all the images of the cam folder and include a contour around the smear.



Here is the mask for cam_3 after the contour size is checked

Final Approach

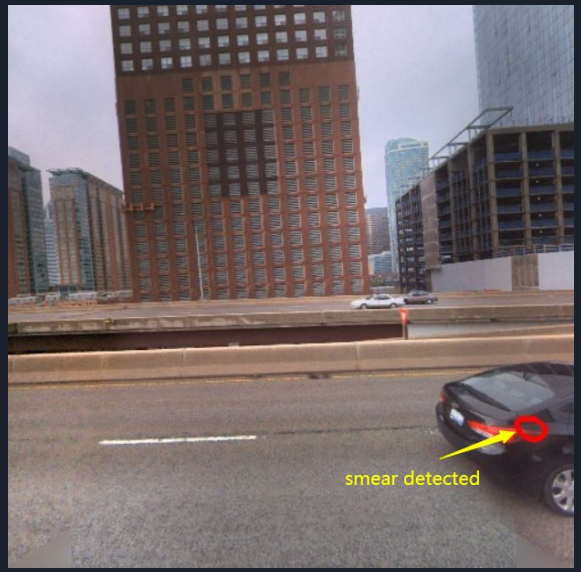




Here is the mask for cam_2 before and after the contour size is checked. Notice that before checking, the mask contains contours that are both too small (as residue from previous processing stages) and too large (perhaps due to other factors in the environment, such as the horizon or sky). These will be eliminated as candidates for lens smears. Therefore, no smears are detected for cam_2, as expected.

Results

Link: https://youtu.be/7uMu1pqE3sA



Here is the smear detected in cam_3

Conclusion

- Removing noise from images is necessary before we detect the smear.
- Converting an image to grayscale greatly improved intermediate and final results
- Averaging many images together is very useful to determine which pixels are constant. That is, those that would make good candidates for a lens smear
- Determine threshold value of the adjacent pixels to help ignore the large constant light or dark area of image, such as sky or tunnel.

Conclusion: Further Steps

- Going beyond the scope of the data provided here, it is important to consider the scenario when a smear is not present on the camera the whole time. In this case, we should have an algorithm to step through the images in the dataset to determine when a smear appears and disappears. For practical applications, there will be many more images, which makes temporary smears an almost certainty.
- Furthermore, our current method of averaging *all* the images together would be inefficient for larger datasets. In that case, one approach would be to break it into smaller chunks, each with a few hundred or few thousand frames, and detect the smears on each individual chunk.

Reference

J. Gu, R. Ramamoorthi, P.N. Belhumeur and S.K. Nayar, "Removing Image Artifacts Due to Dirty Camera Lenses and Thin Occluders," ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia), Dec. 2009.