Facial Recognition and Deep Fakes - Shiyuan Liu, Yuhan Zhang

Contents:

Introduction:

In this project, we experimented with different sensors, and filters by changing parameters to explore their effect on the performance of the facial detection system. We first used ISETCam AI Camera Designer (Fig1) to create pictures, this editing process simulates how the camera captures real-world objects and stores the image. Next, edited images are forwarded into pre-trained neural networks in a way of transfer learning. The neural network we used for the classification task in this project is SqueezeNet. We modified the output layer of the original neural network and trained the model to detect if there's a face in the image or not. The pipeline of our system is shown in Fig2.

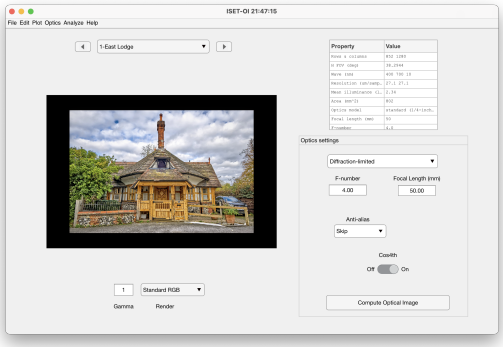

Fig1: User interface of the ISETCam.

Fig2: Pipline of our system.

We focused on two different sensors: the monochrome sensor and Bayer RGB sensor. We also changed the value of read noise, exposure time, and pixel size to learn their effects on the system. The performance of the system is evaluated by classification accuracy.

Background:

Facial Recognition:

Facial recognition is to capture the human face from a digital image against a database of faces. The basis for facial recognition is to determine if there's a face in the image. The accuracy of a facial recognition system is greatly influenced by the quality and parameters of input images. This is because the neural network (NN) needs to extract features (such as two eyes, nose, etc) from image, and based on these features, the fully connected layer in NN computes the probability that an image contains a human face.

Bayer filter:

Bayer RGB filter is a microfilmer overlay which enables photosensors to record light wavelength information. This filter uses a pattern with four pixels in each block, in which two of them are green pixels, one for red pixel and another one for blue pixel. As the combination of four pixels in a Bayer pixel block corresponds to the RGB value of a single pixel in the display, the four pixels see the same point in the visual field. The reason why we include two green pixels, one red pixel and one blue pixel in each block is to better match human eyes which is more sensitive to green light.

Demosaicing algorithm is used for Bayer filter whose raw output is a mosaic of red, green and blue pixels of different intensity. One of the commonly used demosaicing algorithms is to interpolate color value of neighboring pixels with the same color.

Monochrome filter:

A monochrome color filter array, also known as a black and white filter array, is a type of color filter used in digital cameras and image sensors. It is used to capture black and white images by filtering out all colors except for shades of gray.

The color filter array works by placing a series of color filters over the pixels on the image sensor. Each pixel is covered by a filter that allows light of a particular color to pass through to the sensor. In a monochrome color filter array, all of the filters are the same color, typically red, green, or blue. This means that the image sensor is only able to capture light in shades of that color, resulting in a black and white image.

Monochrome color filter arrays are often used in specialized applications such as infrared photography or scientific imaging, where color is not important and a high level of sensitivity to light is required. They can also be used to create a more artistic or timeless look in black and white photography.

Read Noise:

Read noise is one of the main noise source in a camera. In a digital camera, photons are converted to voltage signals and the fluctuation of electronic signals introduces read noise into the system due to the imperfection of physical electronic devices. The value of read noise influences the performance of a camera, and it reveals camera's ability to detect weak signals. A smaller read noise usually enables users to capture weaker signals.

Exposure:

Exposure time refers to the length of time that a camera's sensor is exposed to light when taking a photograph. The longer the exposure time, the more light the sensor is able to capture, which can result in a brighter image or a higher luminance. Conversely, a shorter exposure time will result in a darker image or lower luminance.

Pixel Size:

Resolution of an image or PPI (pixel per inch) is determined by the pixel size. When the size of image is fixed, image with smaller pixel size has a higher resolution. Because each pixel in the image only takes a single value, an image with more pixels is likely to contain more detail. Image resolution has an impact on the accuracy of facial recognition neural network model.

Results:

Dataset:

In our project, we created a dataset that contains 2000 images, in which 1000 contain the human face and 1000 do not contain the face. For the subset that contains faces, we included images of humans of different age groups, genders, and races. For the subset without human faces, we used images containing animal faces (eg. dogs, cats, wildlife animals, etc) and sceneries. Besides, 70% of images are used as the training set, and 30% of images are used as the validation set.

Read Noise:

Based on the experiment results, for both Bayer and monochrome sensor cameras, when the read noise in the sensor increases, the quality of the captured image becomes worse and the network's accuracy decreases, as shown in Fig3. A lower read noise is desirable because it results in a more sensitive sensor, and enables to show smaller contrast changes in the image.

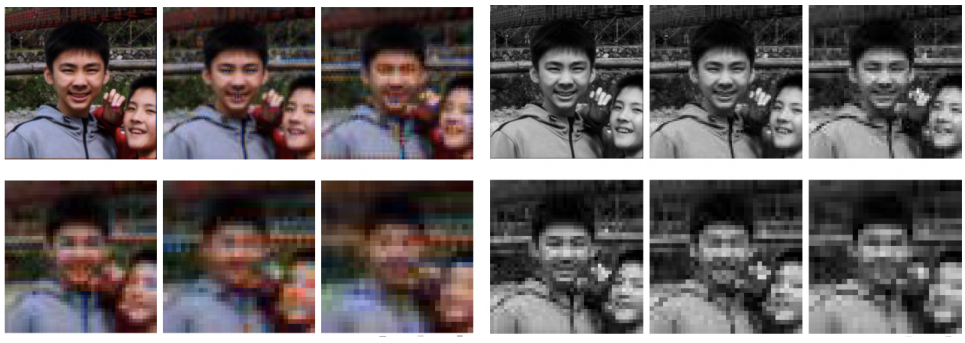

Fig3: Left: Images captured by Bayer RGB sensor, with read noise from 1mv to 0.5V. Right: Images captured by monochrome sensor, with read noise from 1mv to 0.5V.

Table1 and Fig4 show the accuracy of the face detection system in terms of read noise. Based on the results, we can conclude that using images captured by sensor with smaller read noise achieve better performance in a face detection system.

Table1: Face detection accuracy under different read noise.

Read Noise (V)	0	0.1	0.2	0.3	0.4	0.5
Bayer RGB Accuracy (%)	98.33	90.33	83.67	76.00	72.33	55.50
Monochrome Accuracy (%)	98.33	94.33	86.67	72.50	77.63	73.50

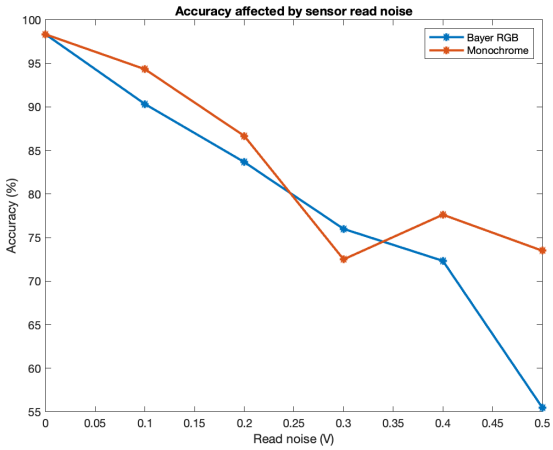

Fig4: Face detection accuracy in terms of read noise value.

Based on the results, it can be observed that the monochrome sensor achieves slightly better performance than Bayer RGB sensor at same read noise level. One of the reasons is that unlike color sensors, monochrome sensors does not need demosaicing process which blurs the edges of images. This means that pictures obtained from monochrome sensor has a relatively better resolution compared with Bayer one.

Exposure:

When the exposure time varies from 10ms to 50ms, output images through the camera with Bayer RGB sensor and monochrome sensor respectively are shown in Fig5. We can see that under this condition, with longer exposure time, facial detection accuracy decreases in monochrome case while increases in Bayer RGB case, as shown in Fig6.

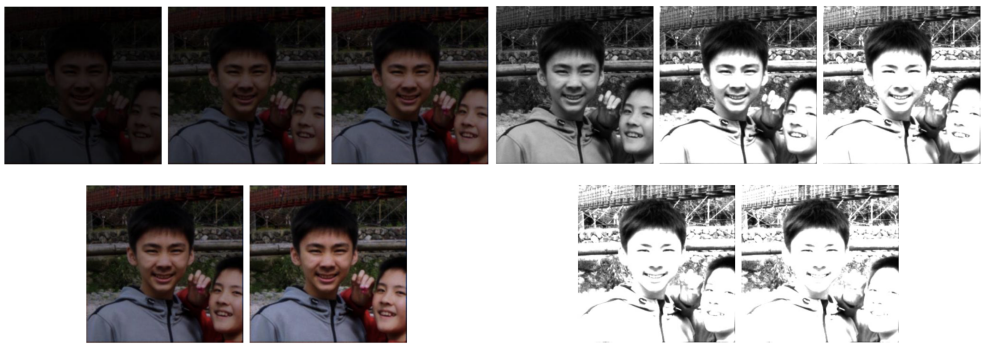

Fig5: Left: Images captured by Bayer RGB sensor, with exposure time ranging from 10ms to 50ms. Right: Images captured by monochrome sensor, with exposure time ranging from 10ms to 50ms.

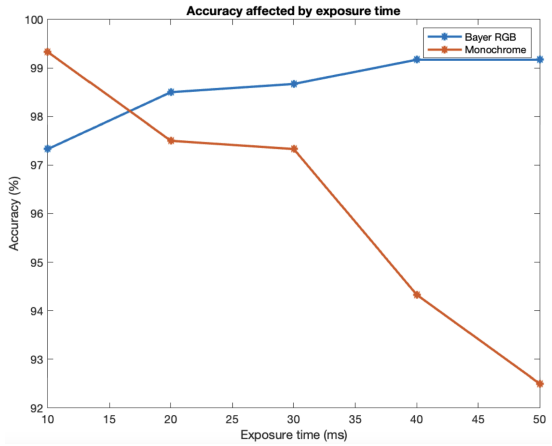

Fig6: Face detection accuracy in terms of exposure time

Above phenomenon is explained by different characteristics of two color filter arrays. Monochrome sensor is sensitive to light of all wavelengths, while each pixel in Bayer RGB sensor is only sensitive to a specific color (red, green or blue) and therefore monochrome camera captures more light with same exposure time. When the exposure time is between 20ms and 50ms, output images from the monochrome camera are over exposed, and therefore the longer exposure time is, the worse facial detection accuracy is. However, this condition cannot make images through camera with Bayer RGB sensor over exposed, and therefore the longer exposure time is, the higher facial detection accuracy is.

To verify that if images are over exposed, camera with Bayer RGB sensor works the same way as monochrome situation does (longer exposure time results in lower accuracy), we increase exposure time to over 100ms. As shown in Fig 7, if exposure time increases from 100ms to 900ms, accuracy shows a decrease trend.

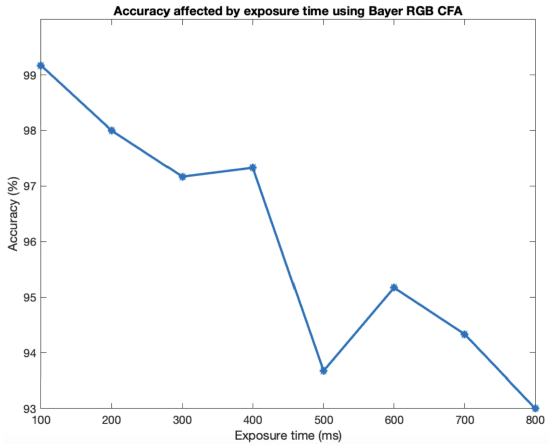

Fig7: Face detection accuracy with Bayer RGB sensor in terms of exposure time

Pixel size:

When we fix the absolute size of image (fixed length and width) and increase the pixel size, the total number of pixels in the image decreases. When there're fewer pixels per inch (PPI), the resolution of the image decreases, and we loss detail in the image. Images with increased pixel size is shown in Fig8.

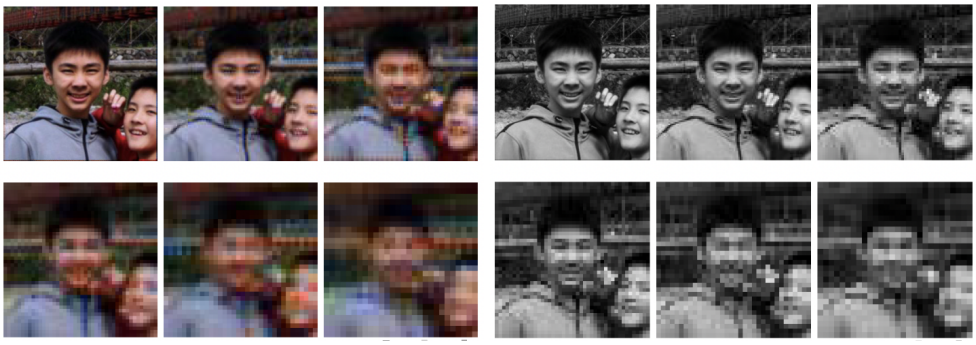

Fig8: Left: Images captured by Bayer RGB sensor, with pixel size ranging from $3 * 10^{-6}$ to $33 * 10^{-6}$ inch. Right: Images captured by monochrome sensor, with pixel size ranging from $3 * 10^{-6}$ to $33 * 10^{-6}$ inch.

Table2: Face detection accuracy using different pixel size.

Pixel size ($\times 10^{-6}$)	3	9	15	21	27	33
Bayer RGB Accuracy (%)	99.50	96.50	89.17	91.33	87.00	83.17
Monochrome Accuracy (%)	99.50	95.33	93.50	89.83	87.00	84.83

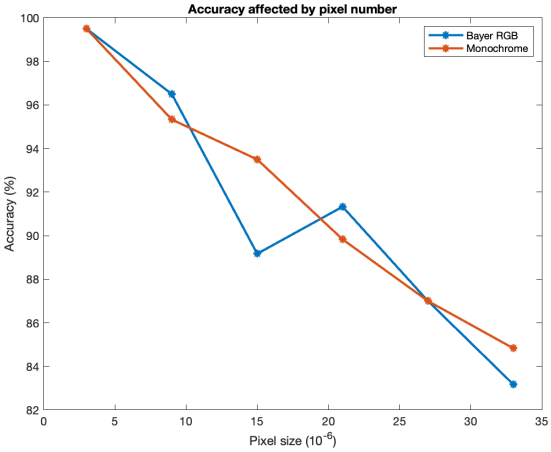

Fig9: Face detection accuracy in terms of pixel size value.

Based on Table2 and Fig9, we can conclude that with the increase of pixel size, the accuracy of face detection system drops. This is because in lower resolution image, some details and edge information are lost, it's more difficult for CNN model to extract effective features for classification.

There's a tradeoff between accuracy and resource needed for the system. The increase of pixel number (or PPI) in the image generally results in a greater image dimension and file size. It means we need larger storage to store these images. Besides, when the pixel number increases, the input data size of the CNN model becomes larger, which means that it requires more time and computational power to train the model. However, images with higher resolution / PPI is more likely to achieve better accuracy in facial detection task.

Conclusions:

In summary, from working on this project, we learned about the influence of several parameters of sensors and filters on image quality and accuracy of facial recognition system. Based on the experiment result, image quality improves when read noise is smaller. Exposure time can have an effect on the image quality either in a good or bad way. That is to say, under over exposed situation, the longer exposure time is, the worse images are, while under non-overexposed case, the longer exposure time is, the better images we can get. Furthermore, the increase in pixel size results in the decrease of pixel numbers in the image, lower the resolution of the image and performance of the system.

References:

ISETCam - https://github.com/ISET/isetcam

Resolution - https://guides.lib.umich.edu/c.php?g=282942&p=1885350#:~:text=Higher%20resolutions%20mean%20that%20there,visible%20like%20the%20image%20below

Resolution - https://smartframe.io/blog/what-is-image-resolution-everything-you-need-to-know/

Stanford PSYCH221 slides

Appendix:

Image processing code:

```
%% edit parameters in panel, save to camera folder

ieCameraDesigner

%% load camera design

%% create folder to save processed image

localDataStore = fullfile(isetRootPath, "local", "data");
if ~isfolder(localDataStore)
mkdir(localDataStore);
end
ourDataFolder = fullfile(localDataStore, 'proj_data/');

if ~isfolder(ourDataFolder)
error("%s isn't a valid folder", ourDataFolder);
end
ourIPDataFolder = strcat(fileparts(ourDataFolder),"_IP_mono_33_6"); % save processed image to this folder
if ~isfolder(ourIPDataFolder)
mkdir(ourIPDataFolder);
end

%% start image processing

imagesPerClass = 1000;

subFolders = dir(ourDataFolder);

for ii = 1:numel(subFolders)
ourFolder = subFolders(ii);
if ourFolder.name(1) ~= "."
ourIPDataSubFolder = fullfile(ourIPDataFolder, ourFolder.name);
fprintf(strcat("Processing: ", ourFolder.name))
expClassify('ipOutputFolder', ourIPDataSubFolder, 'classifier', '', ...
'imageFolder', fullfile(ourFolder.folder, ourFolder.name),...
'maxImages', imagesPerClass, 'progDialog', '', 'sensor', camera.sensor);
end
end
```

Transfer Learning code:

For transfer learning section, we used Matlab Open Deep Network Designer.

The implementations could be finished using UI by calling:

```
deepNetworkDesigner
```

The exported network code is:

transferLearning.mlx

Besides, for training options, we set initialLearnRate to 0.0001, ValidationFrequency to 10, MaxEpochs to 10, and MiniBatchSize to 32. For other parameters, we used the default value.