# Detection of camera errors through analysis of sample images by machine learning

Hanoi University of Science and Technology
Computer Science Department


Author: Phan Xuan Duc
Mentor: PhD. Nguyen Thi Thu Huong

# Contents

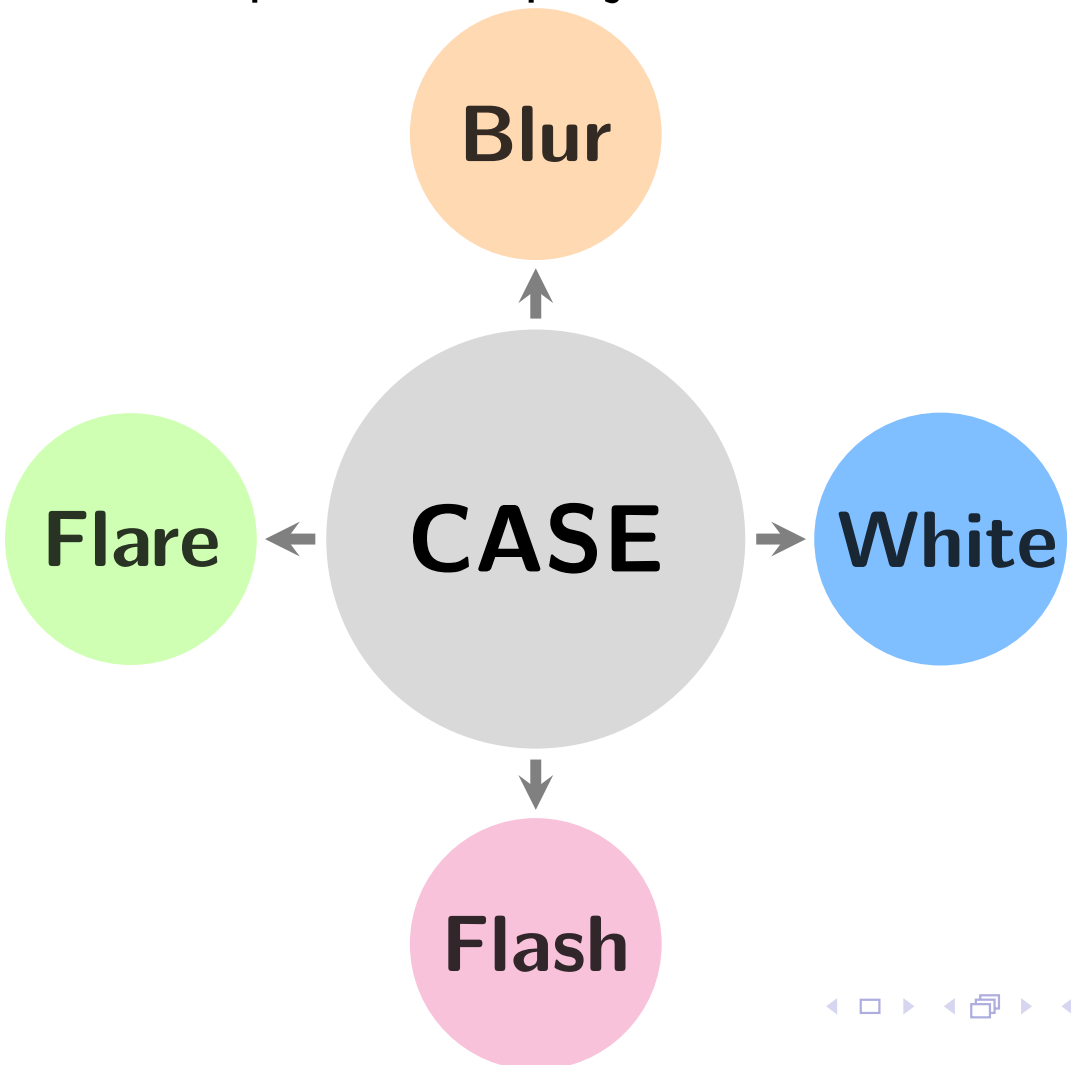
# Introduction

- In order to develop more in the 4.0 industrial, Samsung wants factories to aim for automated testing for its benefits.
- Camera quality testing is one of the important standard output processes for a phone.
- The phones will be checked under certain circumstances and conditions. The camera will take pictures and analyze.
- Example:

Figure 1: Check the quality of smartphone cameras

# Introduction

- Cases in the scope of this project:



# Introduction

- Satisfactory image is the image that identifies the color cells, can identify the numbers, the chart is not mixed together.

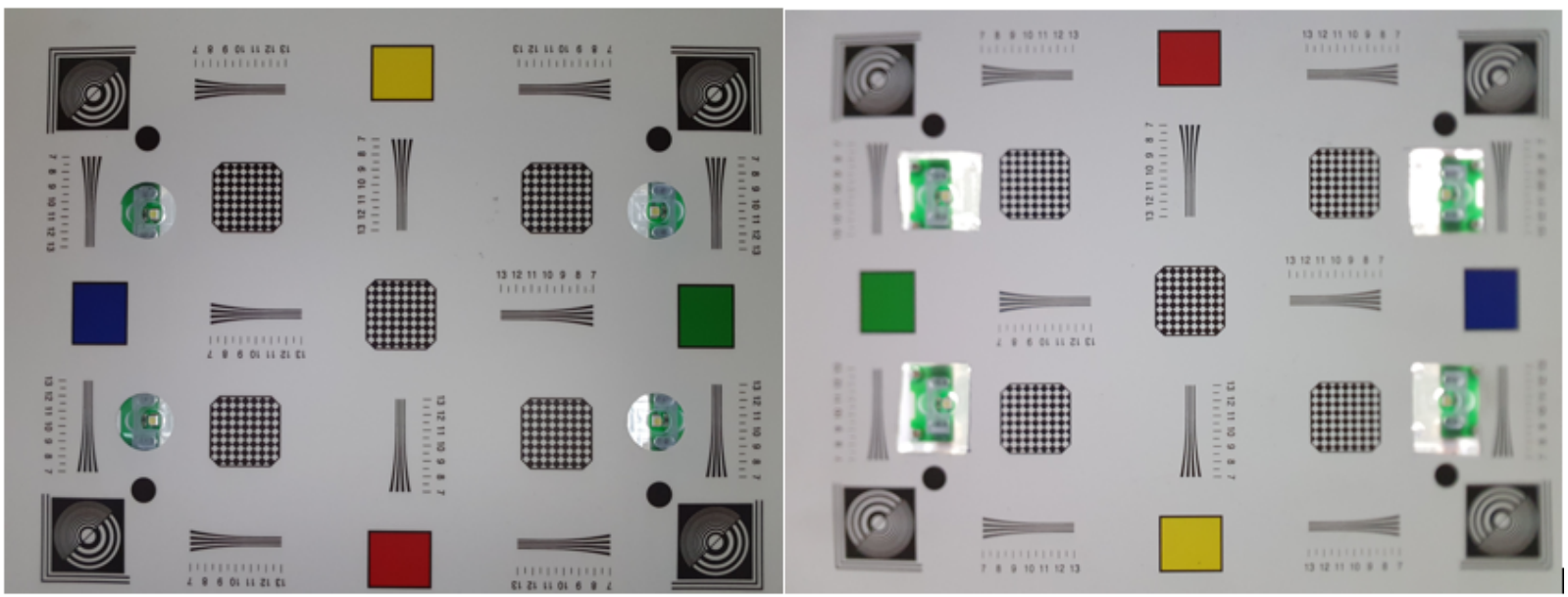



Figure 2: Example of case Blur.

# Introduction

- Satisfactory images with nearby areas of bright LEDs do not appear when flare rays appear in streaks. Error light trails are defined as twice as long as the radius of the bright led area.

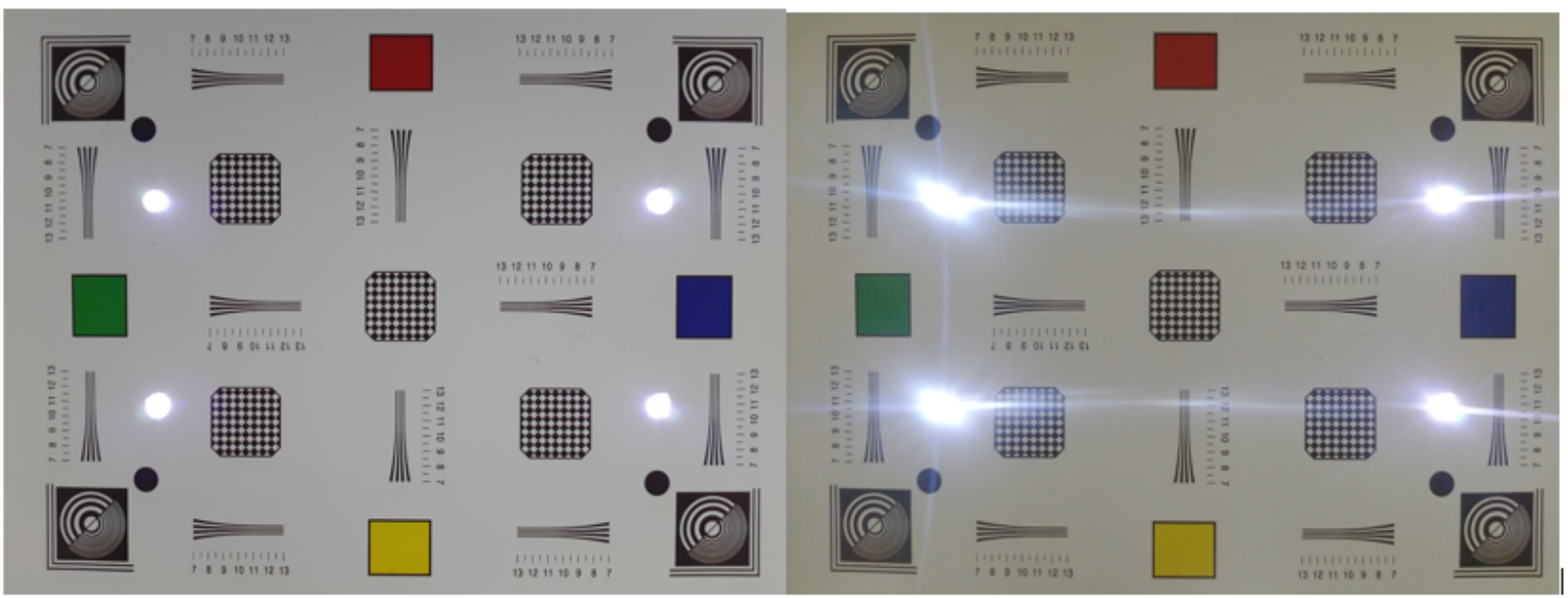



Figure 3: Example of case Flare.

# Introduction

- The central area of the image has a bright area (due to the flash), the brightness of the central area is brighter than 20 units compared to the border areas. For low-end phones, slow shutter speeds lead to photo burns, the center and adjacent areas have greater brightness than 220 units.


Figure 4: Example of case Flash.

# Introduction

- The average brightness of the right image$>$ 120 captures awareness on a white background.
- The value of the color channels is approximately the same if there is any color channel larger than the other 20 channels, the image is considered to be haunted color.

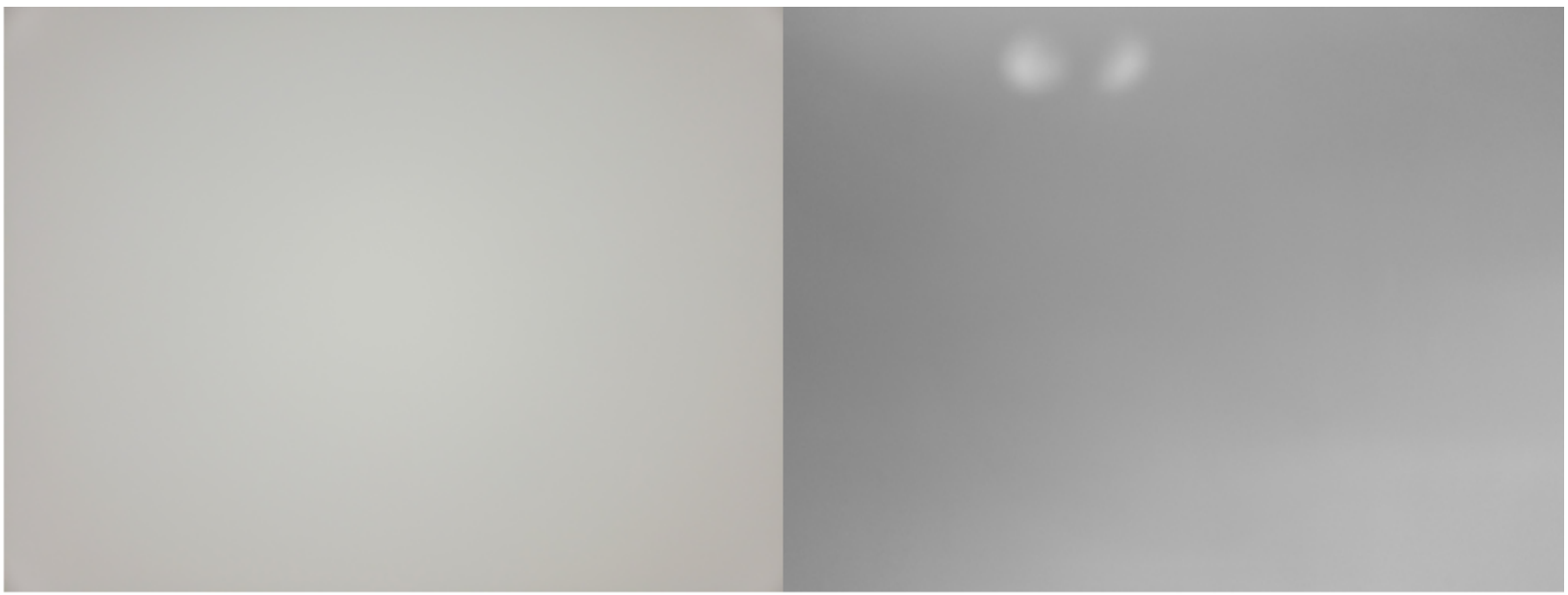



Figure 5: Example of case White.

# Introduction

- Traditional Method:
  - Take photos manually
  - Compare, evaluate with eyes

**Factors affecting the misclassification:**

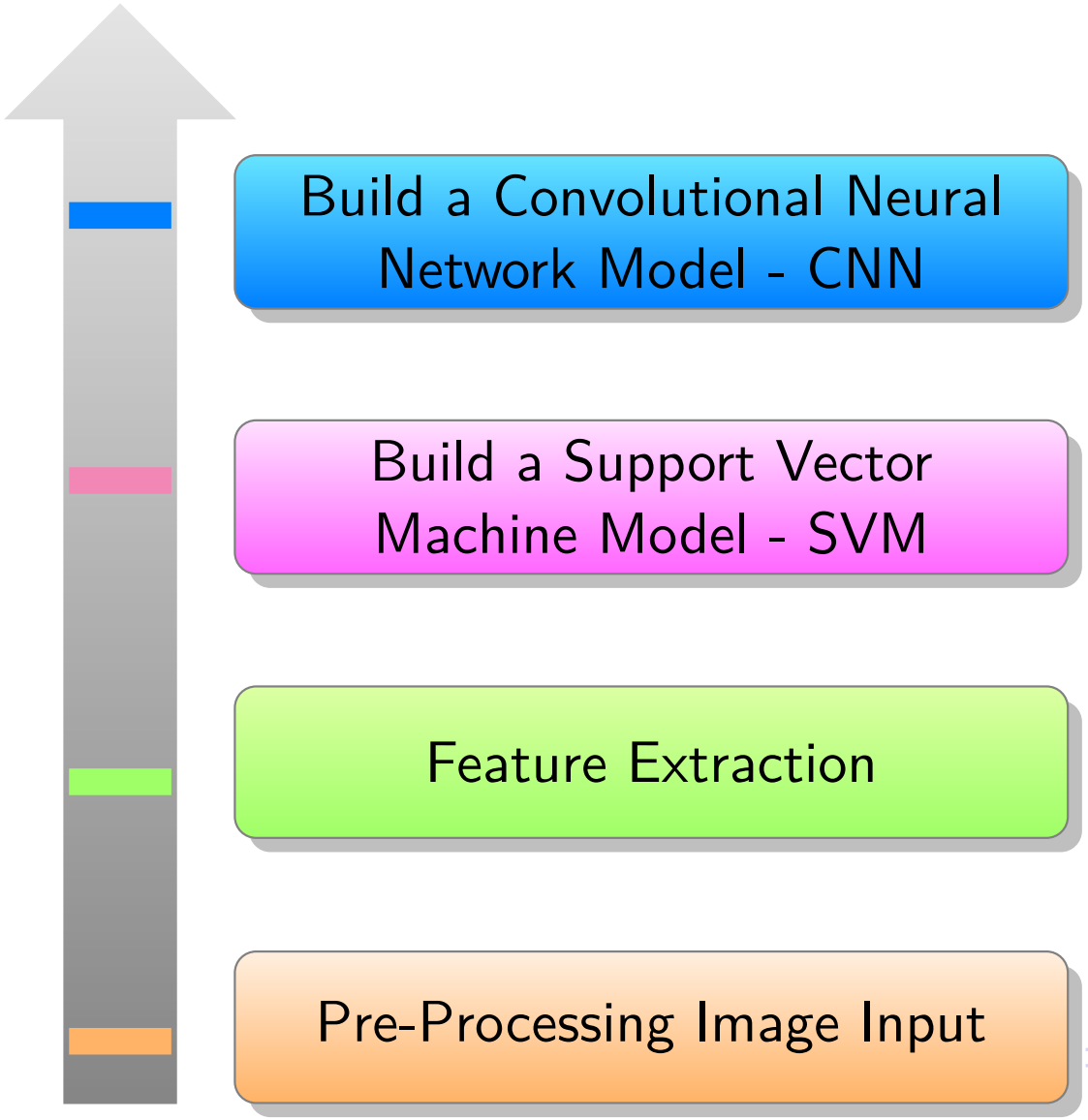
- The number of cameras to check is large
- Repetitive work
- The environment is not always ideal
- The camera standard is very high

# Introduction

## Problem

- Automatic classification of phones with camera errors and warnings for production staff, production managers to take measures to handle business

# The Proposed Method



# Data processing and Feature extraction

- Case Blur:

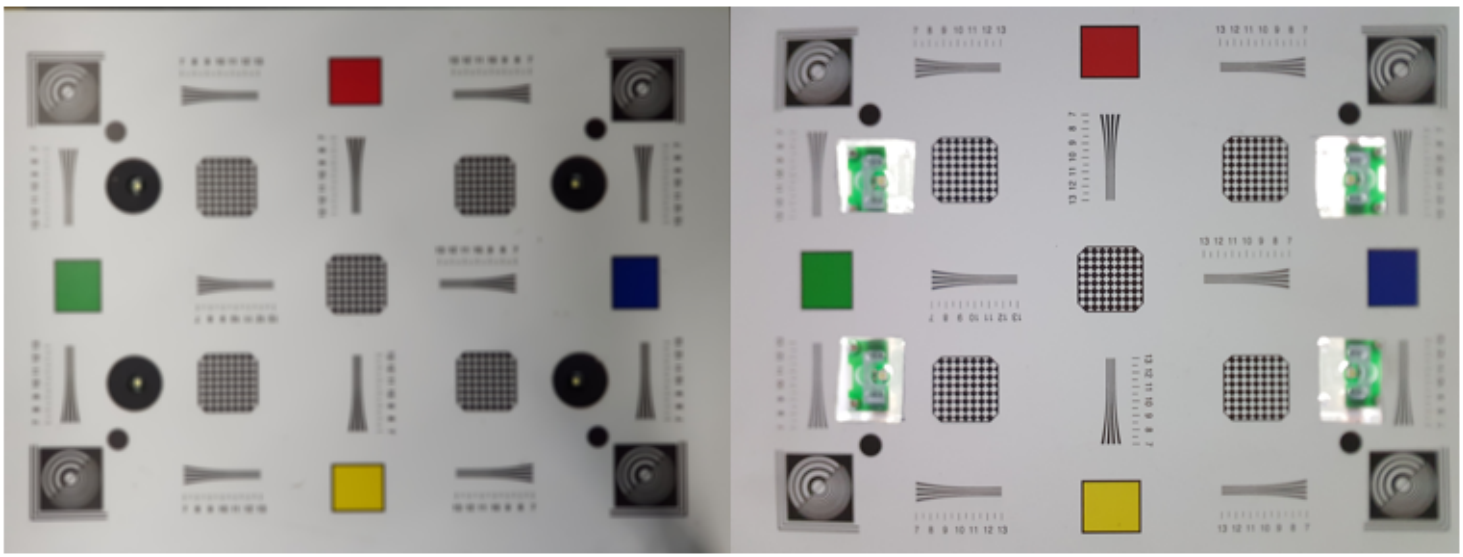



Figure 6: The image is blurred in case Blur.

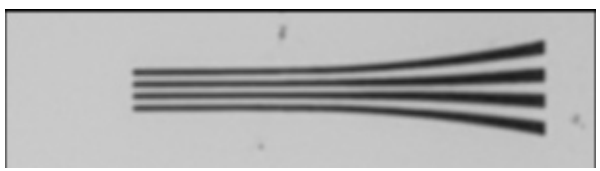


Figure 7: The area defines fuzzy characteristics.

# Data processing and Feature extraction

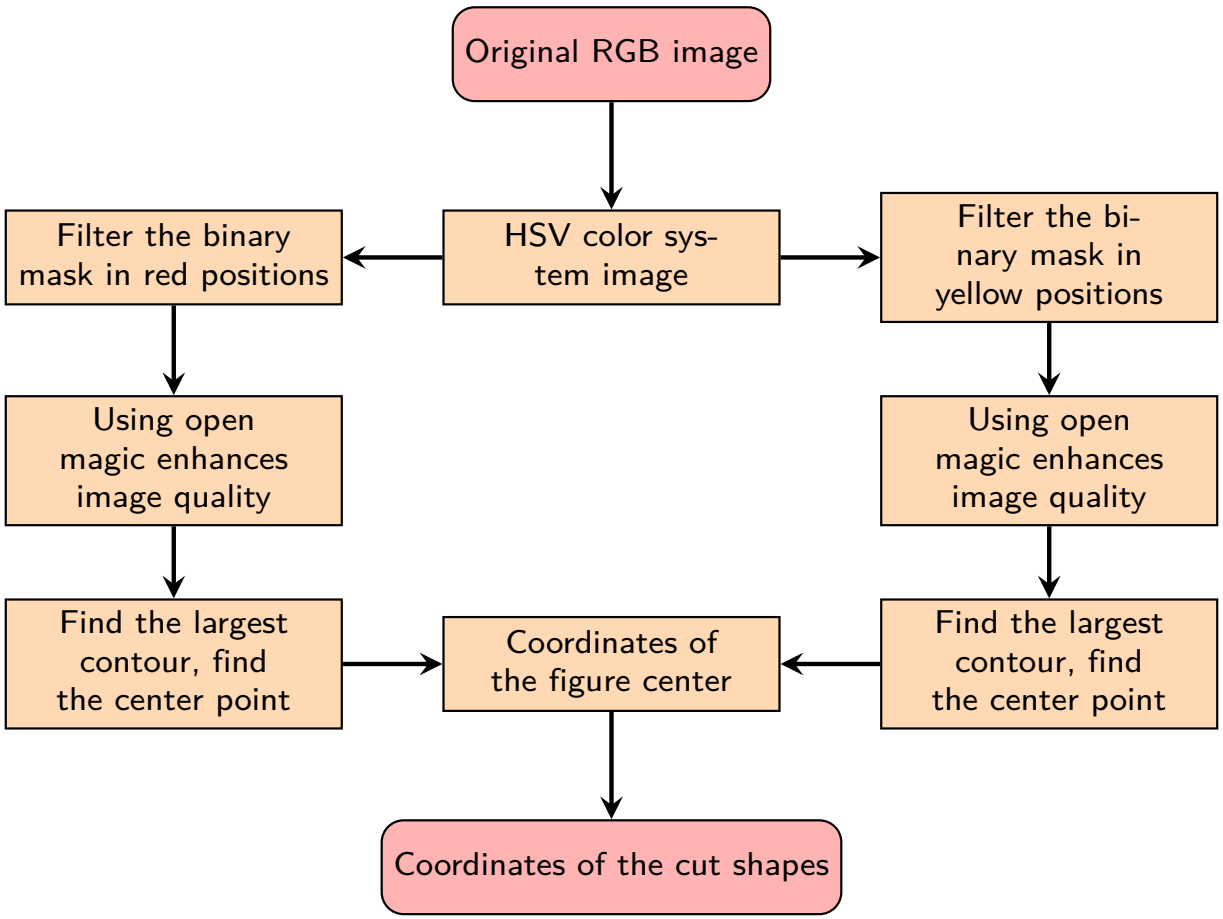



Figure 8: Process of cutting photo case Blur.

# Data processing and Feature extraction

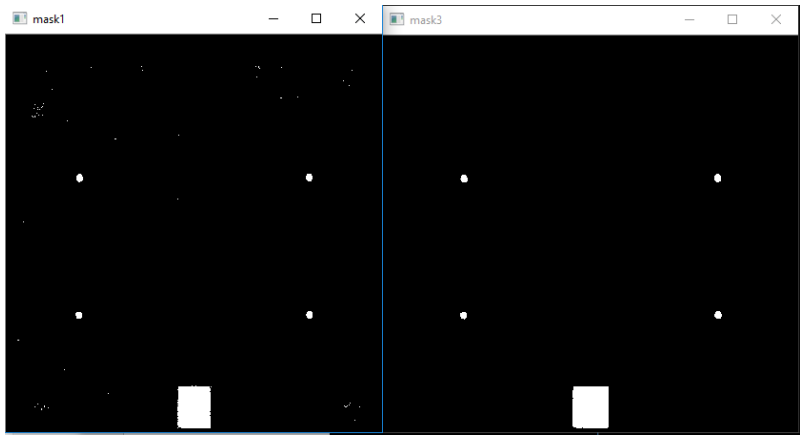



Figure 9: Use open noise reduction.

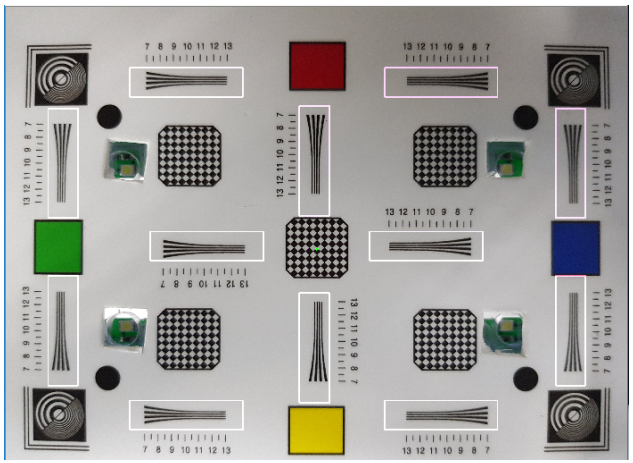


Figure 10: The position of the cut shapes obtained.

# Data processing and Feature extraction

- Solution:
  - Convert original image to gray image (Gray scale)
  - Multiply convolution with Laplace matrix
  - Find the variance of the image obtained

**Characteristics of use: (3 features)**

- The variance after multiplying with Laplace filter matrix
- The average gray level value of the image
- Gray level value with highest frequency (gray level value at background)

# Data processing and Feature extraction

- Case Flare:

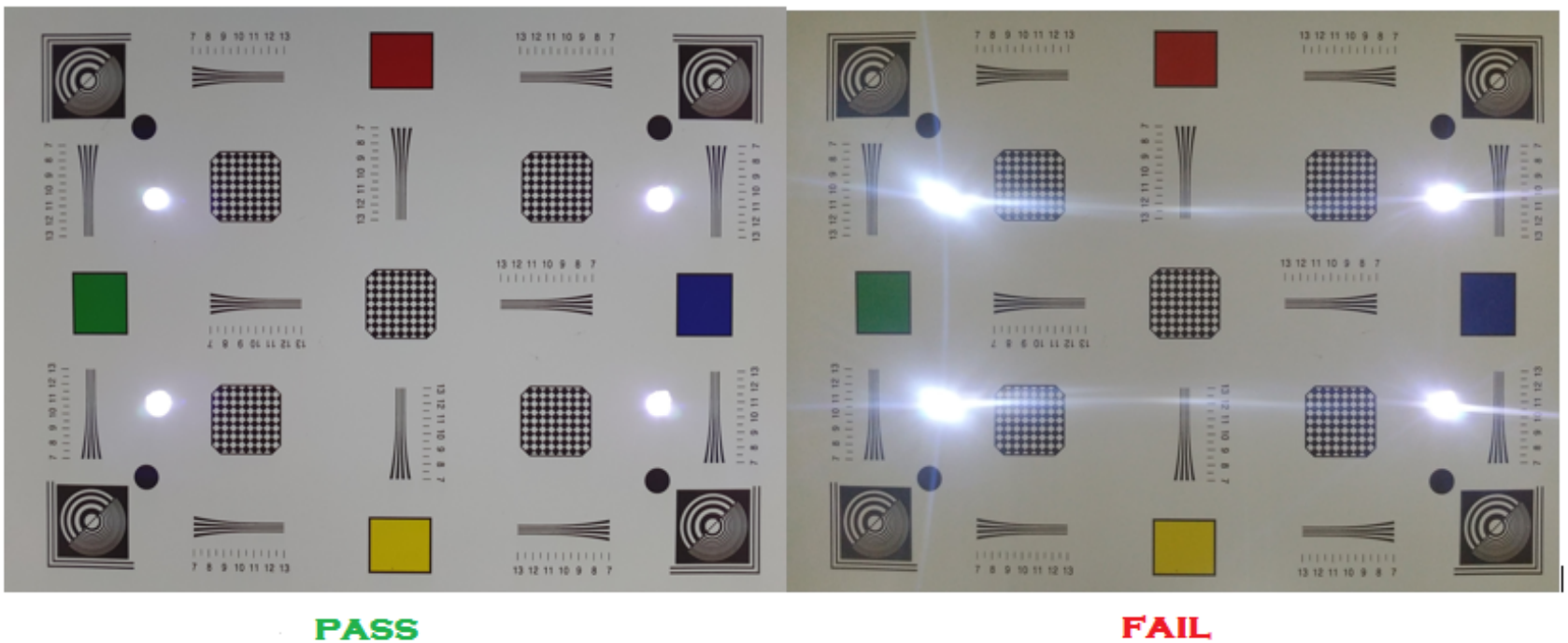



Figure 11: Picture of case Flare.

# Data processing and Feature extraction

Figure 12: Feature Identification Area.

- Perform image cutting case Flare:
  - Convert original image to gray image
  - Perform threshold separation, threshold: 250
  - Cut the image square around the center of the led

# Data processing and Feature extraction

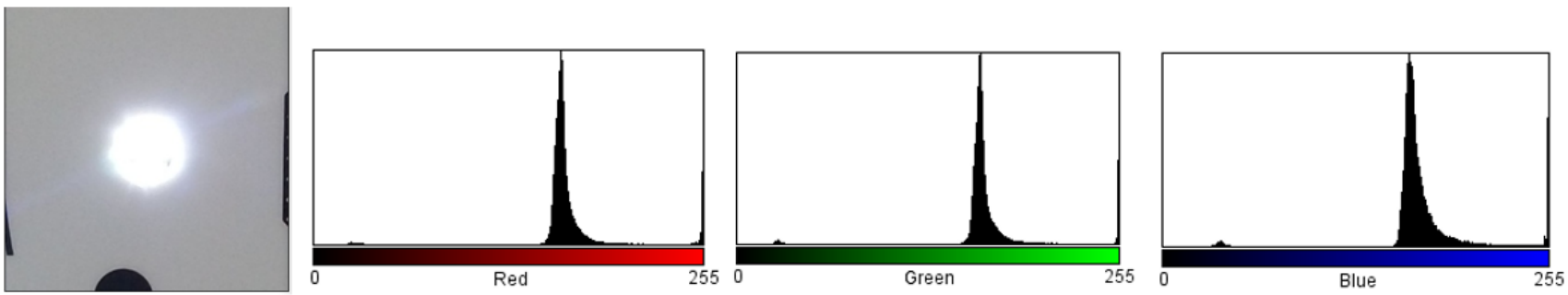



Figure 13: Histogram for 3 color channels.

- Implementation of threshold separation: (2 methods)
  - Separation of Fixed threshold
  - Separation of Adaptation threshold

# Data processing and Feature extraction

- Separation of Fixed threshold:
  - Use threshold from automatic threshold separation (Otsu algorithm) (a)
  - Separate threshold from gray image, threshold: 170 (b)
  - Separate the threshold from the Blue color channel, threshold: 170 (c)
  - Separate the threshold from HSV color channel image according to the range of brightness and saturation. S: [0, 51]; V: [190, 255] (d)

# Data processing and Feature extraction

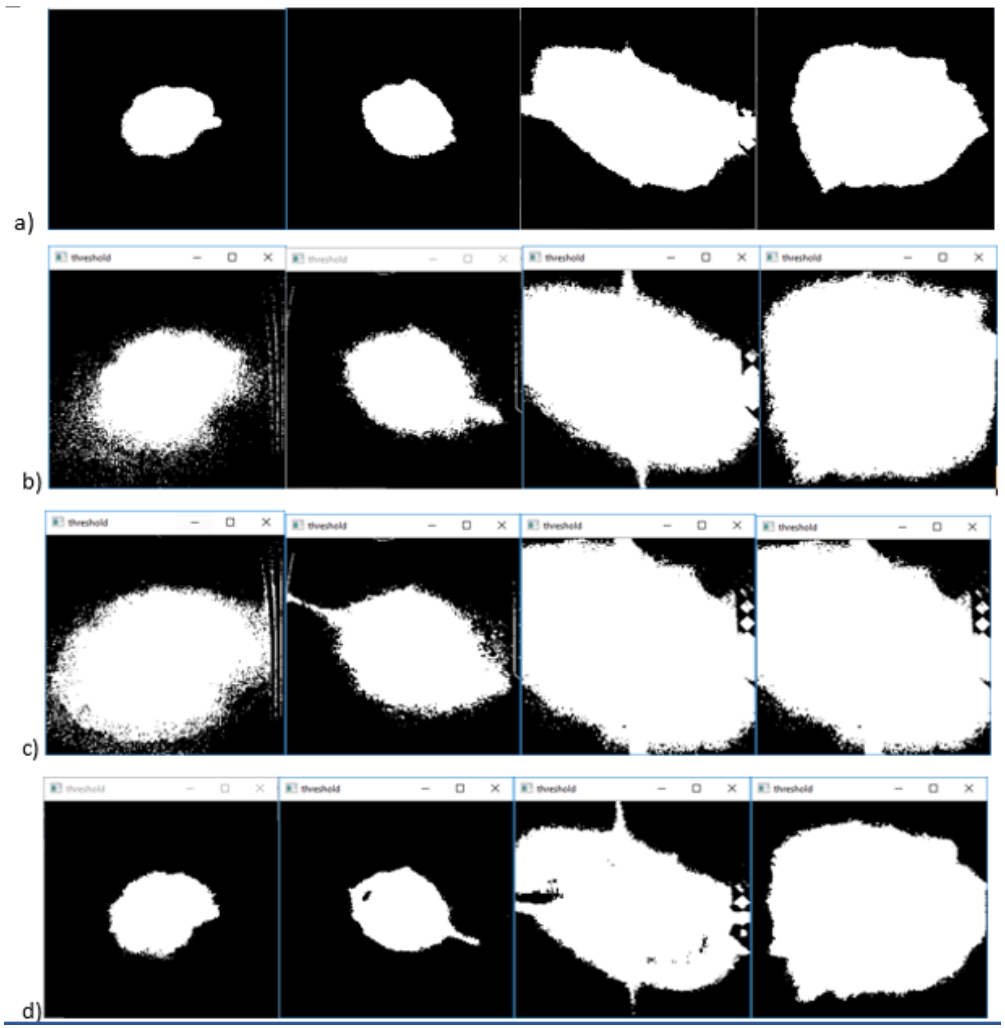



Figure 14: Test results of fixed threshold separation.

# Data processing and Feature extraction

- Separation of Adaptation threshold:

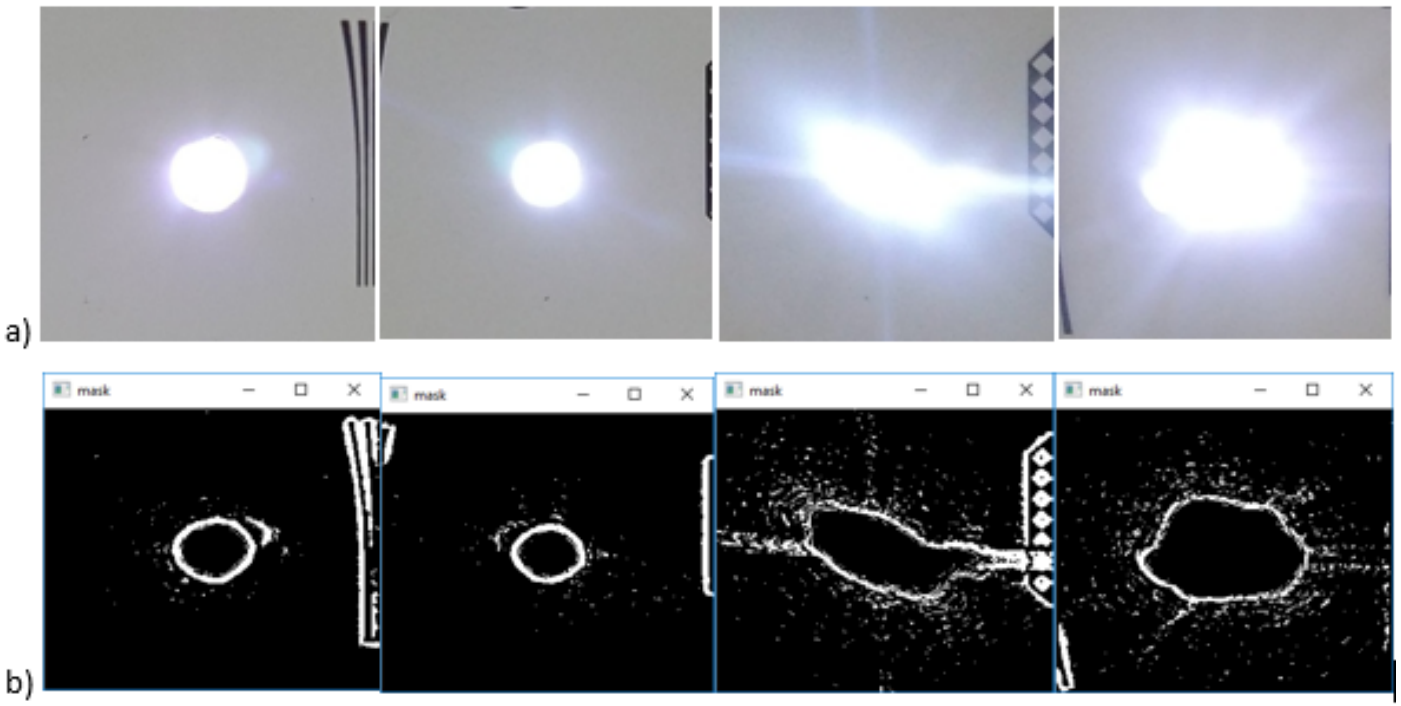



Figure 15: Test results of adaptive threshold separation.

# Data processing and Feature extraction

- Evaluate 2 threshold separation methods:

| | |
|---|---|
| **Fixed** | Blue color channel has the best ability to cover dim light |
| **Fixed** | Photos are very bright (due to the environment) good in certain situations |
| **Adaptive** | The general form of the ray area is best in the case |
| **Adaptive** | Small and faint light rays are not clearly visible |

# Data processing and Feature extraction

- Solution:
  - Creating a binary mask is the operator '**&**' of the mask after separating the appropriate threshold and the mask after separating the fixed threshold from the Blue channel
  - Reduce noise and smooth images
  - Find the contour around the light area with OpenCV's findContour() function

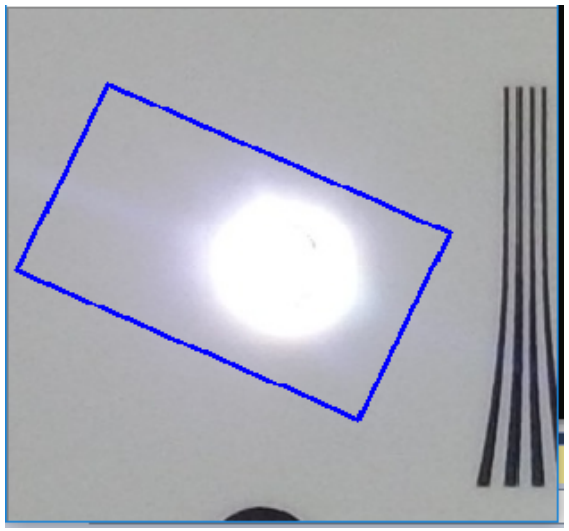


Figure 16: The rectangle rotates around the bright area.

- Lights and lamp sizes affect the features:
  - Extract featured on led light with 250 threshold.

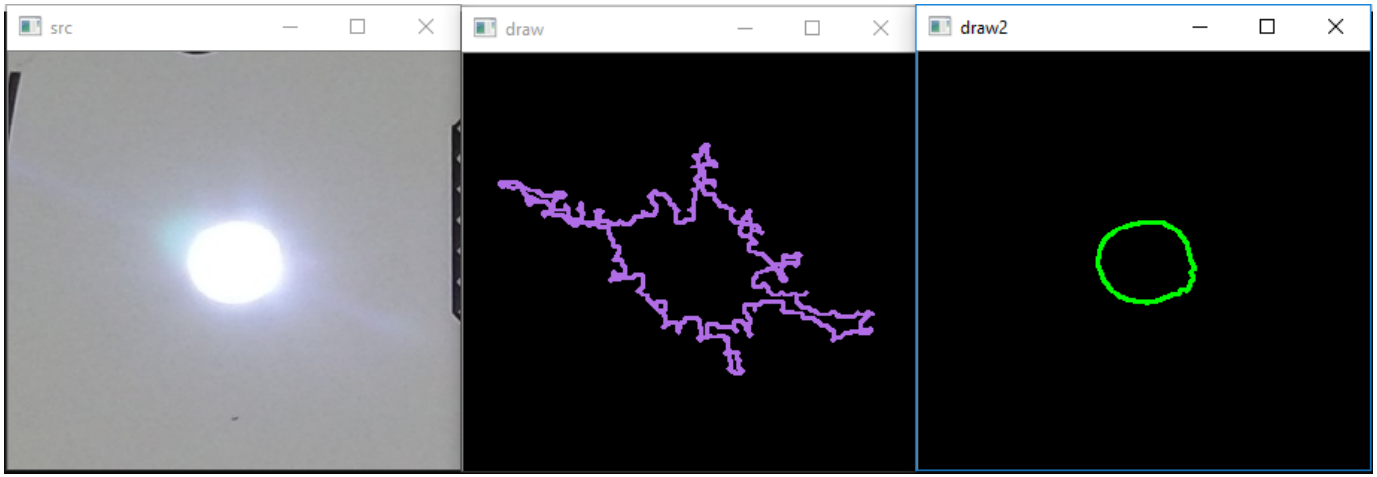



Figure 17: Two contours from two threshold filters.

## Usage features: (20 features)

- Area of contour, length, width, 7 invariant Hu-moment from led light and light flare
- Area of contour, length, width, 7 invariant Hu-moment from visible, high-intensity led light

# Data processing and Feature extraction

- Case Flash:

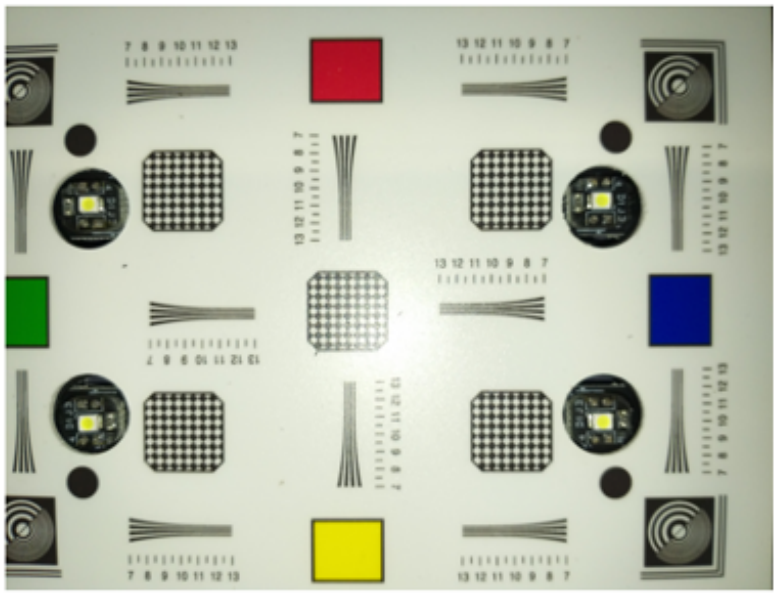




Figure 18: Two cases are satisfactory in case Flash.

# Data processing and Feature extraction

- Compare of light intensity between 5 regions:

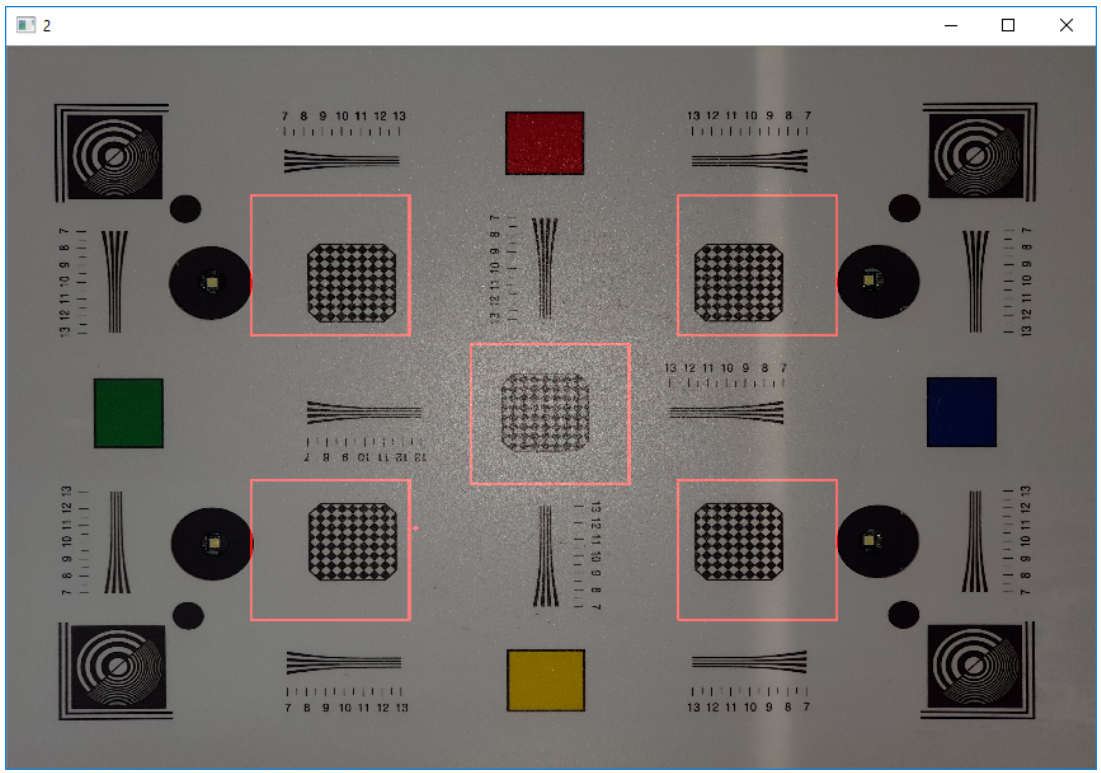



Figure 19: Photos of regions are considered in the case Flash.

# Data processing and Feature extraction

- In some cases, skewed images will affect the central position:

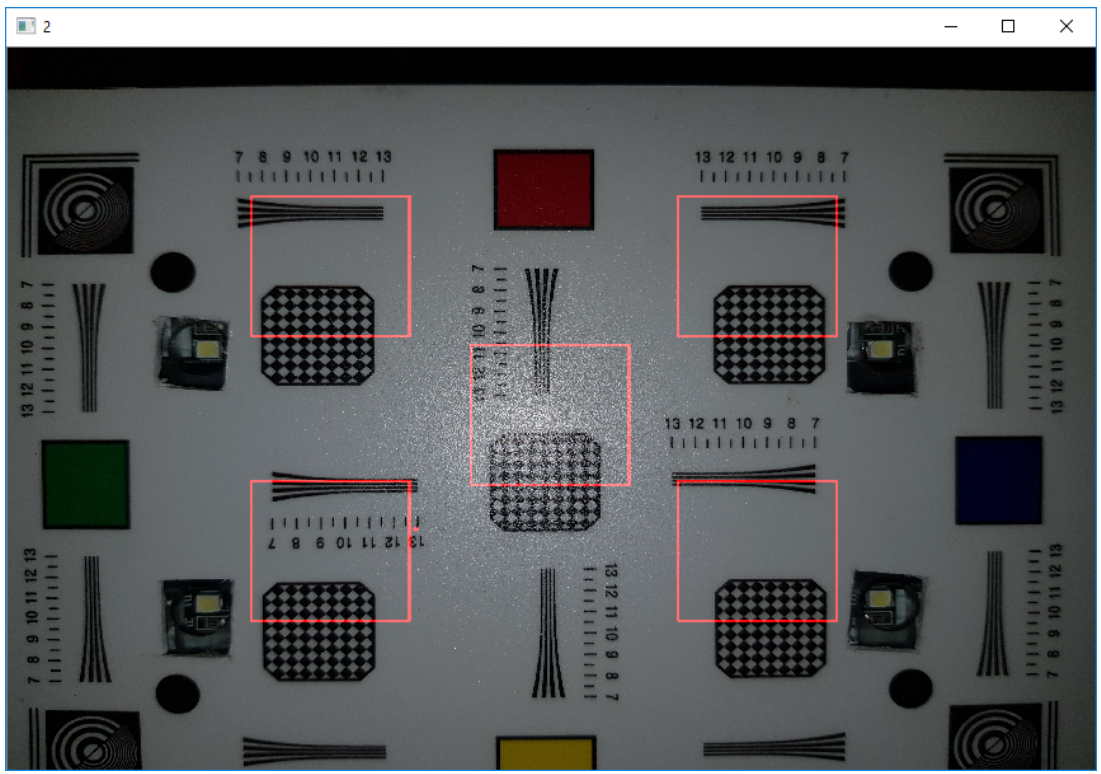



Figure 20: Photo set deflected in case Flash.

# Data processing and Feature extraction

- Solution:
  - Distinguish the black image from the background by automatic threshold separation (Otsu algorithm)
  - If the pixel is smaller than the threshold (the black part) will be set to 0, greater than the threshold (background) that remains the same value
  - The average value of the image will only be calculated by pixels other than 0

## Usage features: (10 features)

- Automatic threshold values in the center and 4 diagonal zones around
- The average value of the gray level of the background pixels in the center region and the 4 diagonal regions around

# Data processing and Feature extraction

- Case White:

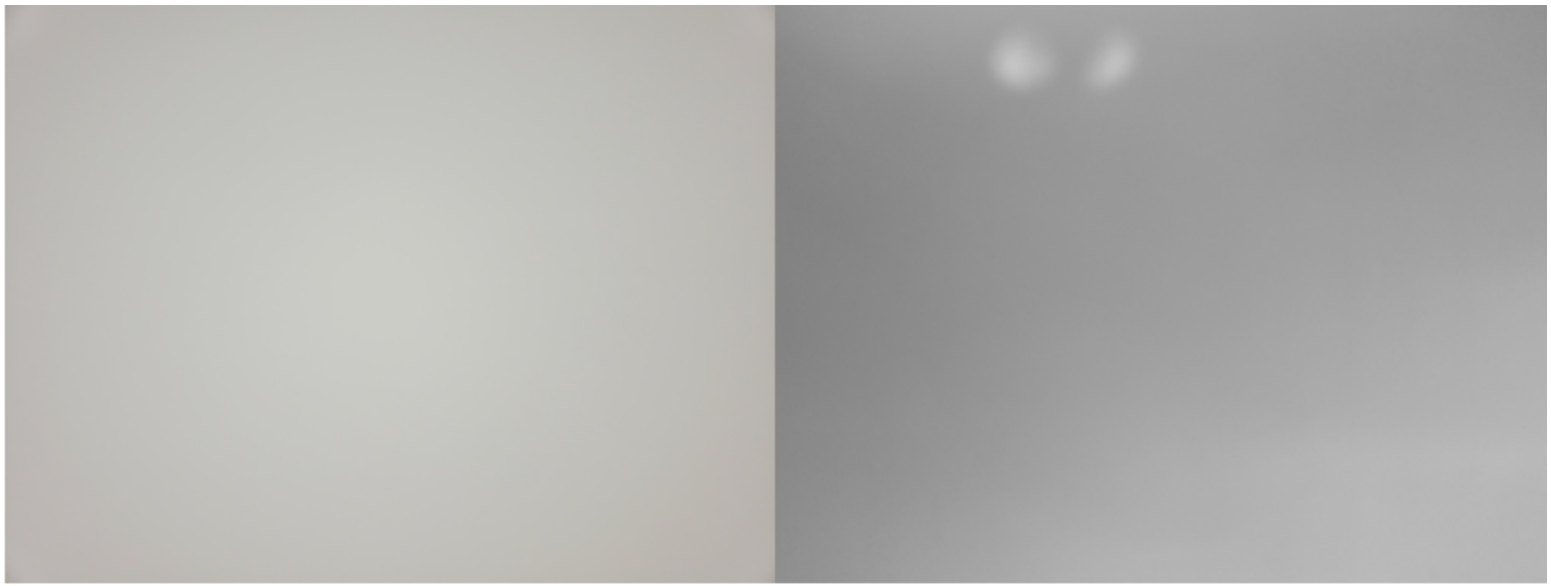


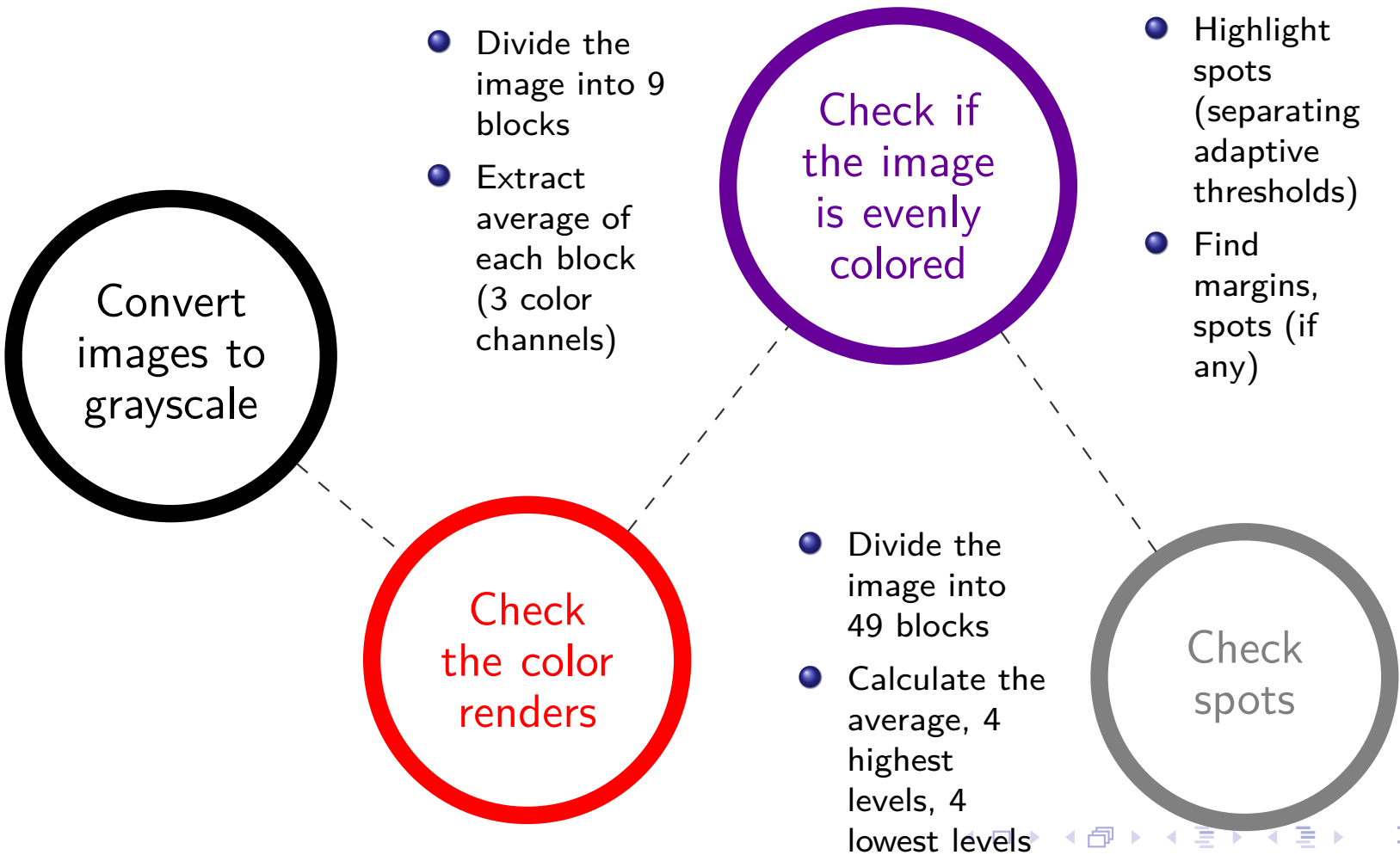

Figure 21: Picture of case White.

# Data processing and Feature extraction

- Satisfaction Conditions:
  - The average brightness of the image must be$>$ 120 units
  - The value of the color channels is approximately the same, the color channel is bigger than the other 20 channels, the image is considered to be haunted
  - Transparent white background without stains, light streaks

# Data processing and Feature extraction

- Solution:



# Data processing and Feature extraction

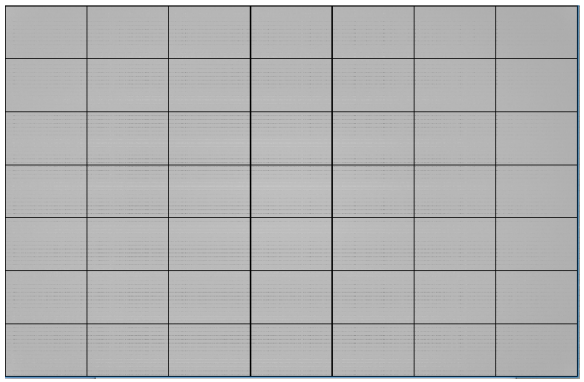


Figure 22: Images are divided into 49 blocks.

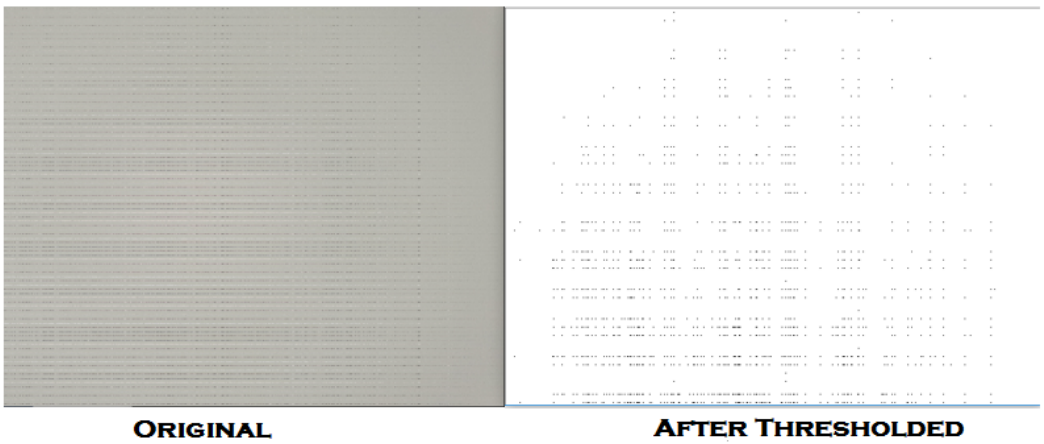



Figure 23: Image after threshold separation of case White.

# Data processing and Feature extraction

## Usage features: (469 features)

- Average value of each color channel on each block (9-division image)
- The average gray level value of the block, the highest 4 gray levels, the lowest 4 gray levels (49 blocks divided)
- Area of spots after separation of threshold

# Dataset

- Original image size:
  3264 x 2448 ; 4032 x 3024 ; 4608 x 3456 ; 5565 x 4248

| Case | Training Set NG | Training Set OK | Test Set NG | Test Set OK |
|---|---|---|---|---|
| Blur | 708 | 732 | 50 | 50 |
| Flare | 819 | 1721 | 50 | 50 |
| Flash | 78 | 225 | 50 | 53 |
| White | 117 | 101 | 61 | 50 |

Table 1: Training data set after cutting.

# Perform classification

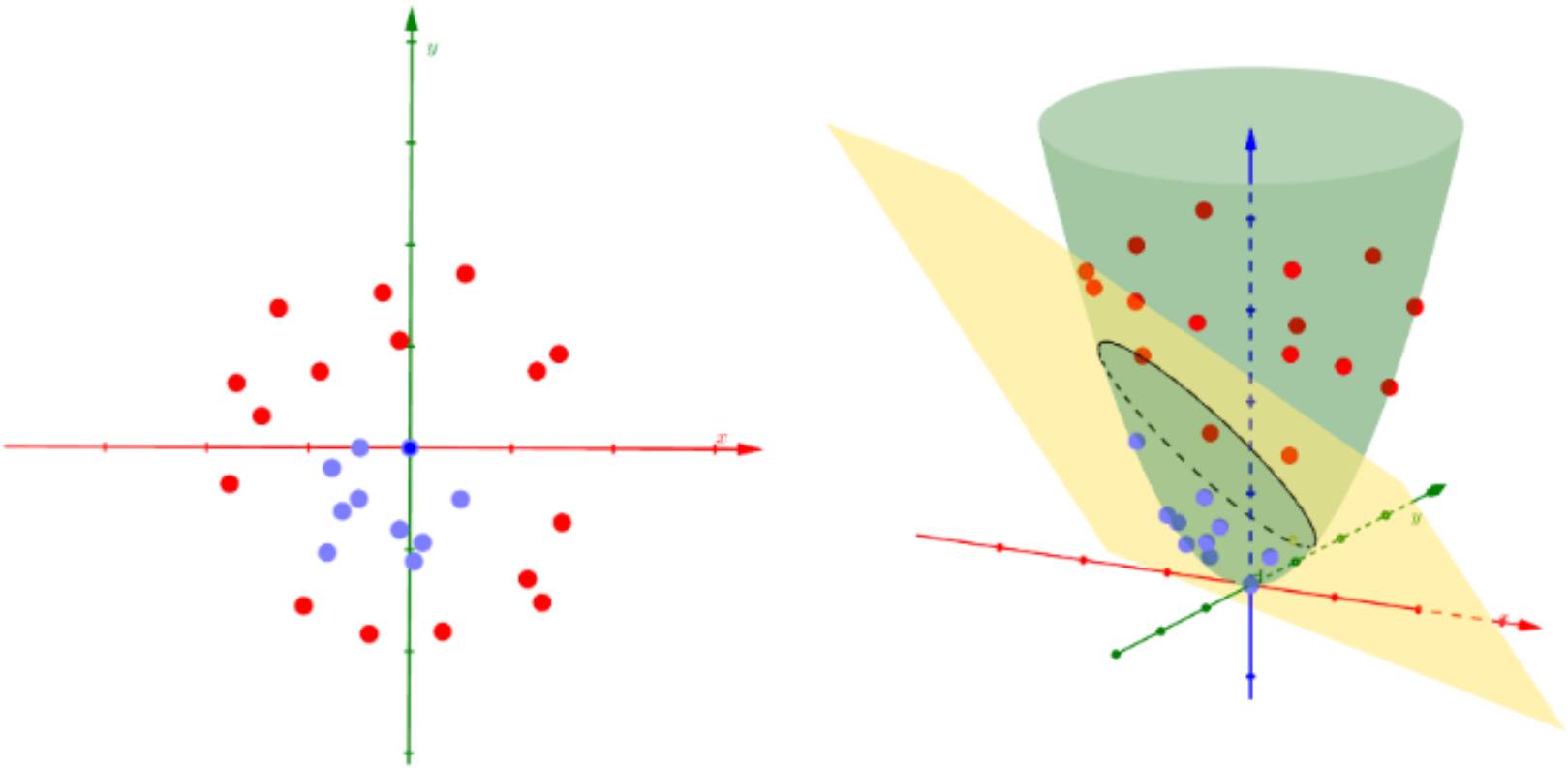
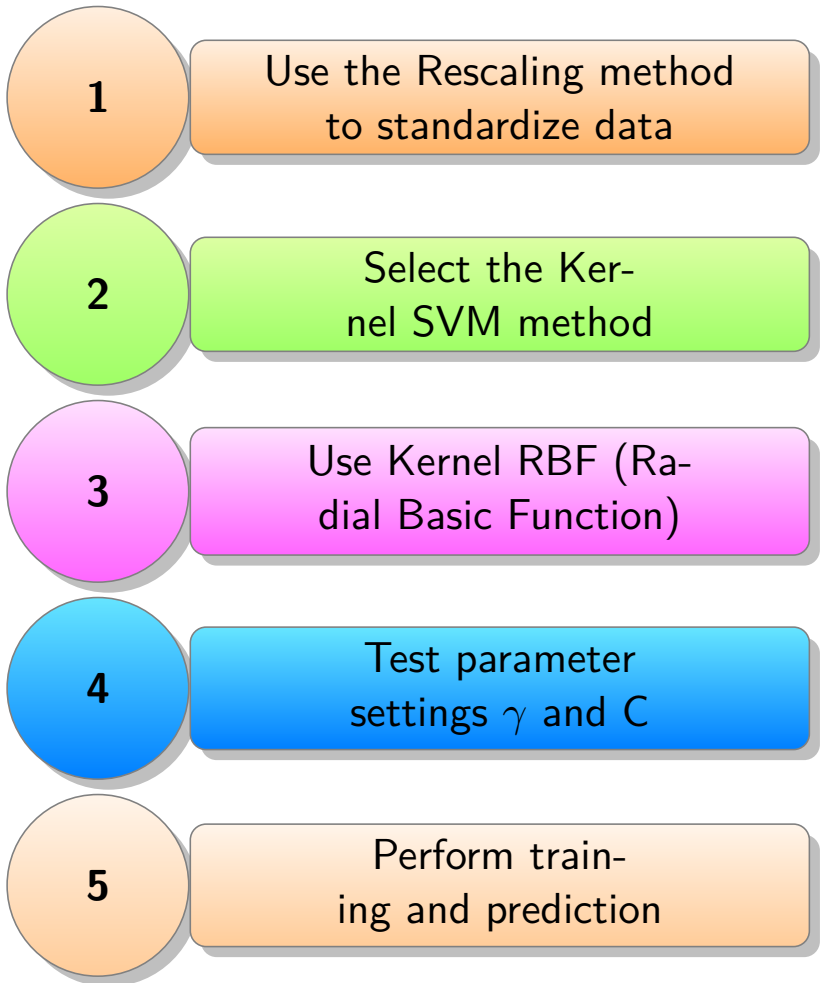


Figure 24: Expand not with linear non-discriminatory data in SVM model.

# Perform classification

- SVM Model:



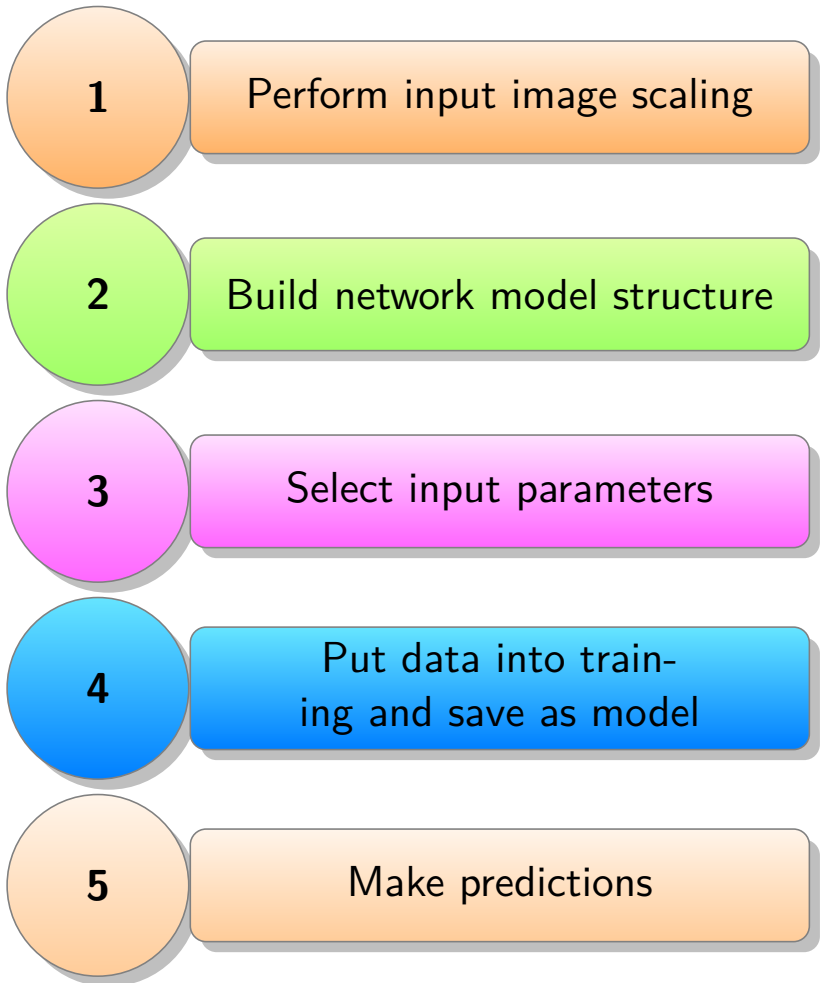
# Perform classification

| Case | Accuracy | True rate: ( NG ) | True rate: ( OK ) | Time running |
|---|---|---|---|---|
| Blur | 97.00% | 48/50(96.00%) | 49/50(98.00%) | 40ms |
| Flare | 95.00% | 45/50(90.00%) | 50/50(100.00%) | 116ms |
| Flash | 97.12% | 50/50(100.00%) | 51/54(94.44%) | 34ms |
| White | 97.30% | 58/61(95.30%) | 50/50(100.00%) | 803ms |

Table 2: SVM algorithm results

# Perform classification

- CNN Model:



# Perform classification

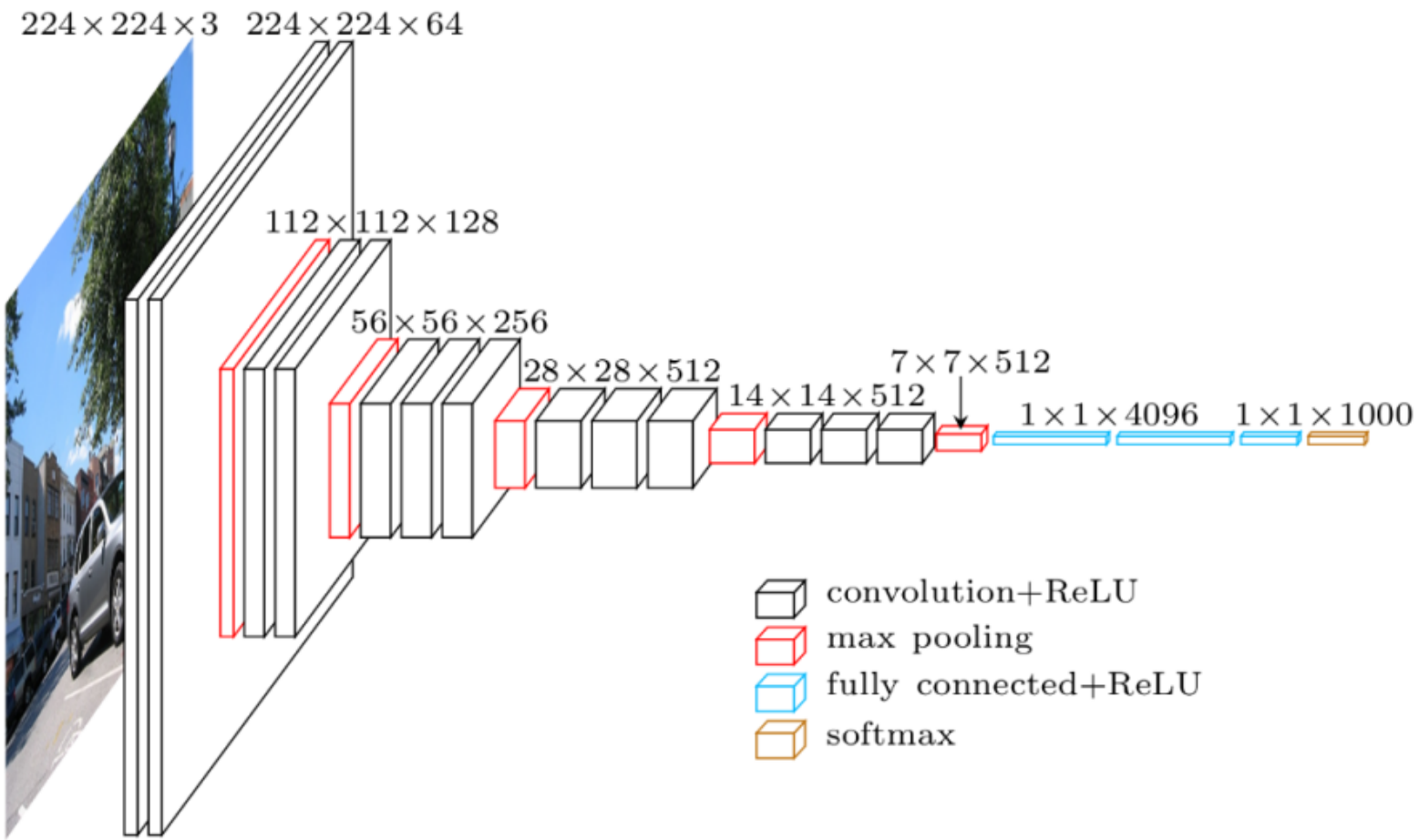



Figure 25: CNN network model architecture.

# Perform classification

| Case | Accuracy | True rate: ( NG ) | True rate: ( OK ) |
|---|---|---|---|
| Blur | 99.00% | 49/50(98.00%) | 50/50(100.00%) |
| Flare | 95.00% | 48/50(96.00%) | 47/50(94.00%) |
| Flash | 71.15% | 41/50(82.00%) | 33/54(61.11%) |
| White | 90.99% | 51/61(83.61%) | 50/50(100.00%) |

Table 3: CNN algorithm results

# Compare results

| | Linear Regression | Logistic Regression | SVM | CNN |
|---|---|---|---|---|
| Blur | 92.00% | 96.00% | 97.00% | 99.00% |
| Flare | 95.00% | 95.00% | 95.00% | 95.00% |
| Flash | 97.12% | 97.12% | 97.12% | 71.15% |
| White | 86.49% | 93.69% | 97.30% | 90.99% |

Table 4: Compare accuracy between algorithms

# Compare results

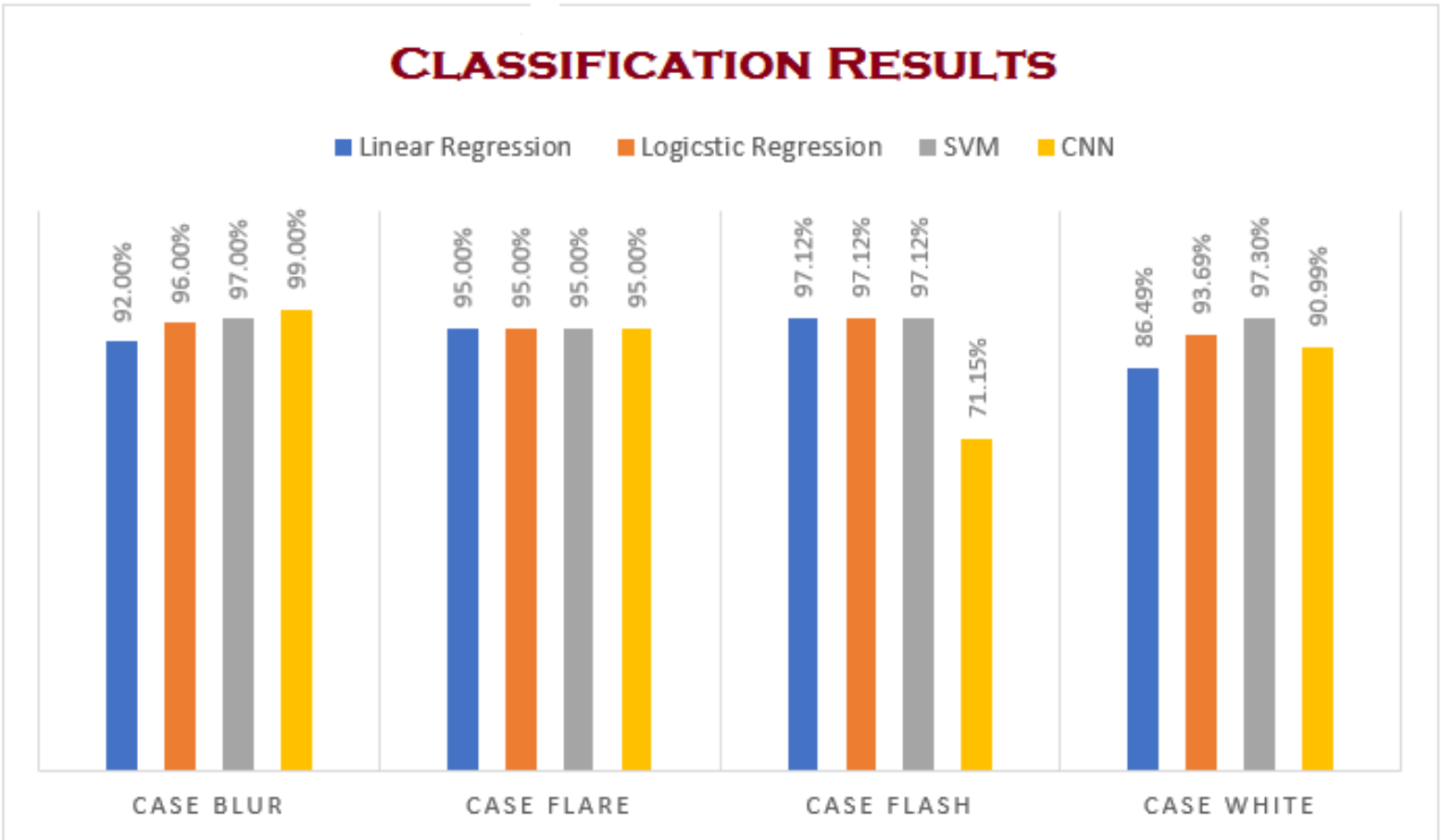



Figure 26: Classification results Chart.

# Result evaluation

- Misclassification Case:
  - Blur: Photo has a very dark background
  - Flare: Photos with slight flare rays are not well identified
  - Flash: NG image test set has images that are not properly recognized
  - White: Misclassified color images are very light

# Conclusion

## Achieved Result

- List some features of the image and perform cutting of photo areas containing the featured feature
- Pre-Processing of image input data and the application of image processing into characteristic separation
- Implement problem solving with SVM model
- Perform image stretching, training CNN model, save as model for prediction
- Build some other sorting algorithms and make comparisons

# Conclusion

## Limit

- Feature extraction is not really optimal
- Training and testing set is still quite small
- The classification of two labels is simple
- Machines for training are not strong enough for CNN networks

# Q & A

# References

1. Luong Manh Ba, Nguyen Thanh Thuy. Nhập môn Xử lý ảnh số. *Hanoi University of Science and Technology*, 2007.
2. PhD. Do Nang Toan, PhD. Pham Viet Binh. Giáo trình Xử lý ảnh. *Thai Nguyen University, University Of Information and Communication Technology*, 11/2007.
3. M. K. Hu. Visual Pattern Recognition by Moment Invariants. *IRE Trans. Info. Theory, vol. IT-8*, 1962.
4. Suzuki, S. and Abe, K. Topological Structural Analysis of Digitized Binary Images by Border Following. *pp. CVGIP 30 1, pp 32-46*, 1985.
5. C.-W. Hsu and C.J.Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks, 13(2):415-425*, 2002.
6. Boser, B, E, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. *In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pages. 144 – 152. ACM Press, 1992*, 1992.
7. Vu Huu Tiep. http://machinelearningcoban.com, *Online*.
8. .v.v.v