

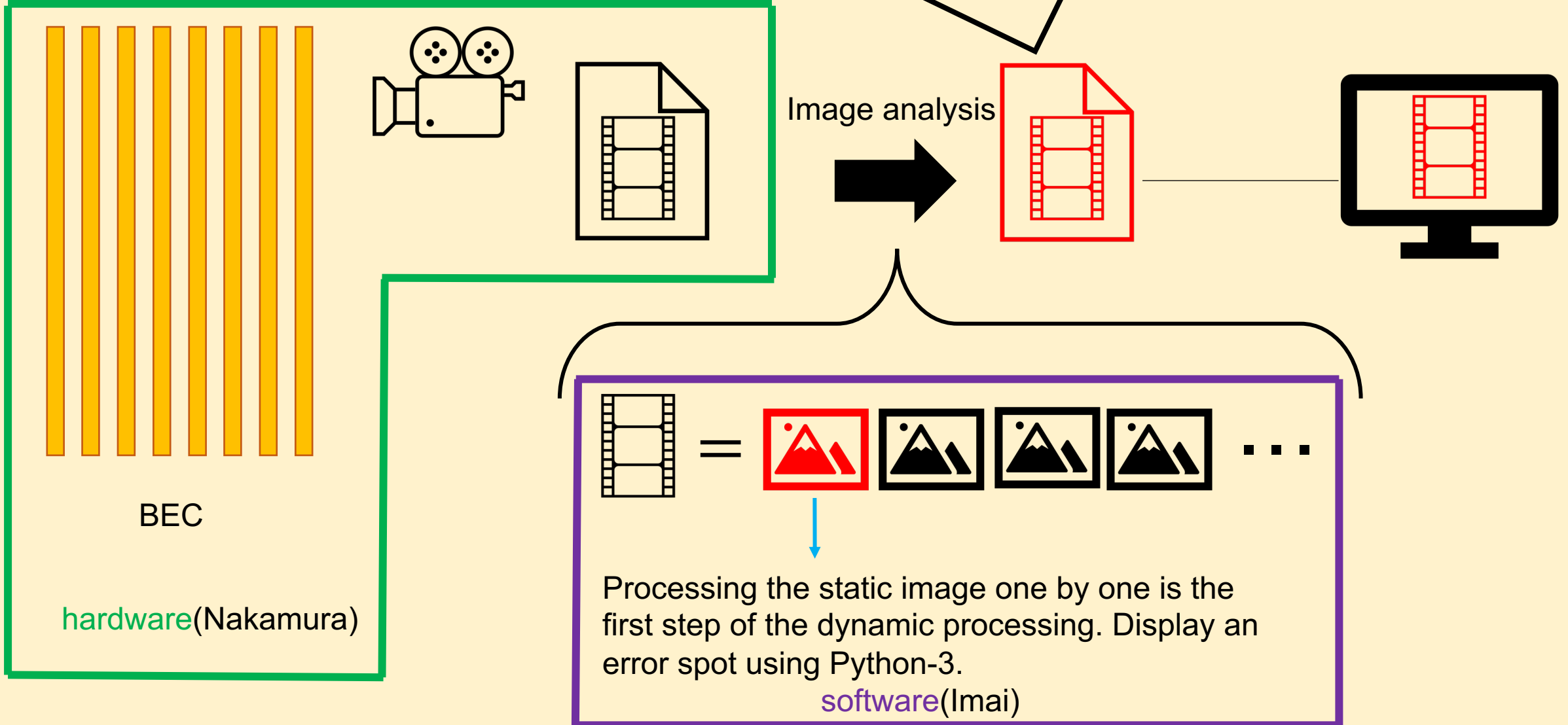
Development of the inspection fixture of the line & space pattern for the INTT busextender inner signal layer ---software---

Hikaru Imai (Rikkyo University:M1)

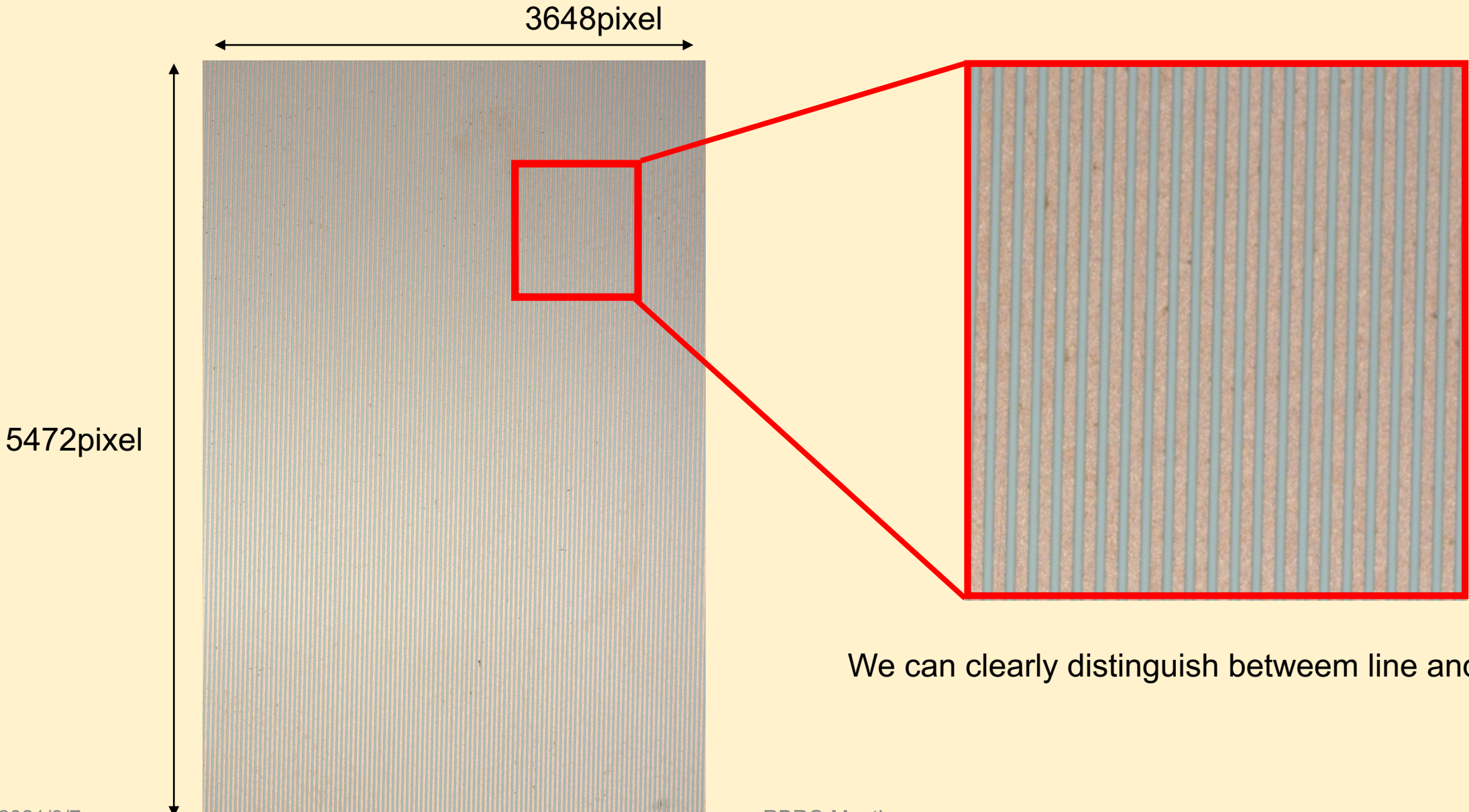
2021/9/07

Hardware and software

A real time (dynamic) scanning image is preferable rather than a static image processing

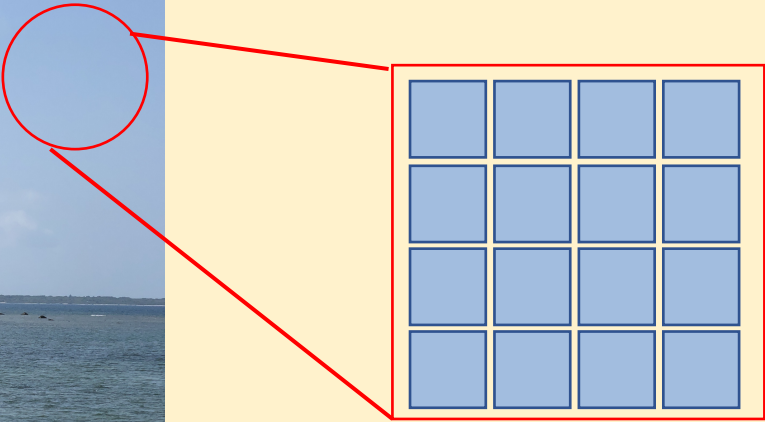


Sample image



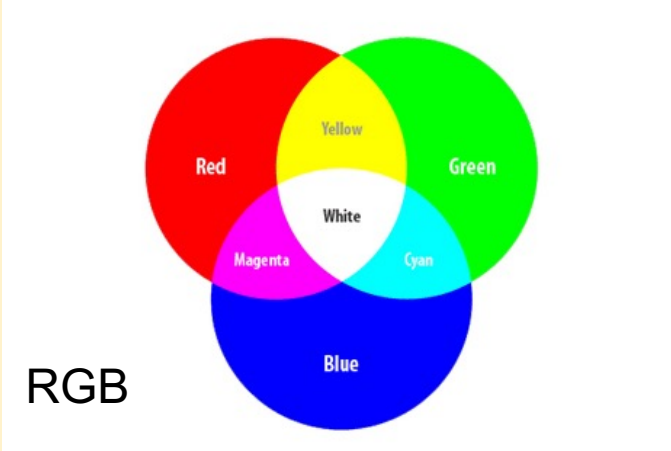
We can clearly distinguish between line and space.


Pixel and RGB



 ...pixel

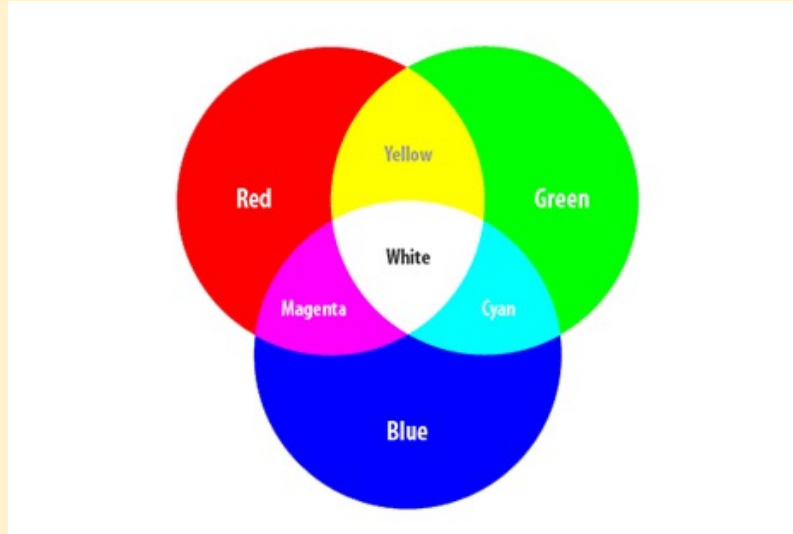
Image consist of pixel.



R G B
 = [70, 20, 100]

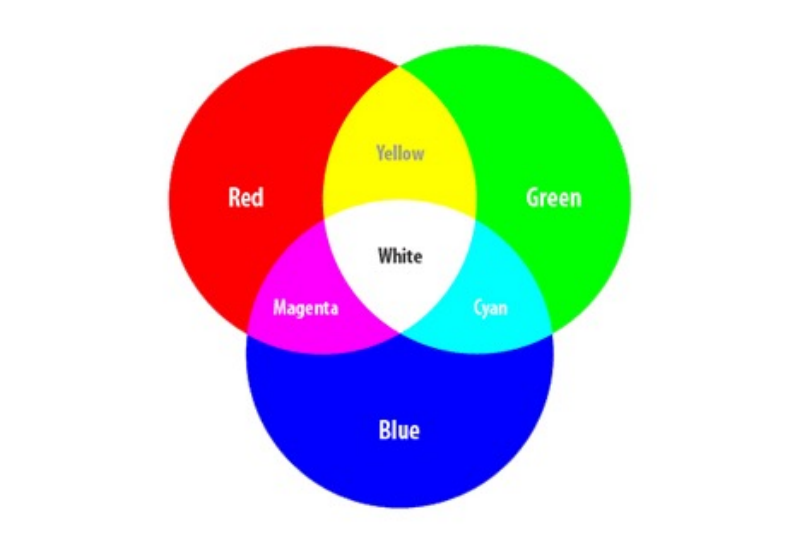
Pixel have three value which is RGB for expressing color in digital.

RGB and HSV

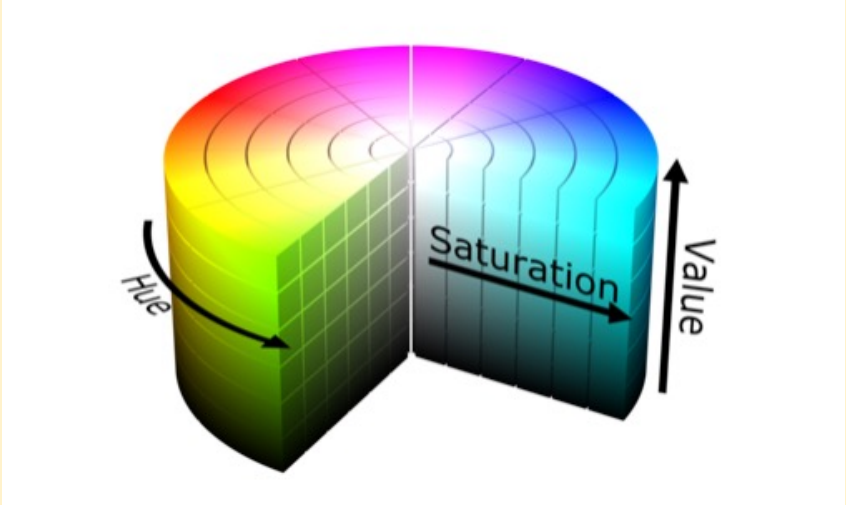


$$\begin{array}{c} \text{R} \quad \text{G} \quad \text{B} \\ \square = [70, 20, 100] \end{array}$$

RGB and HSV



VS

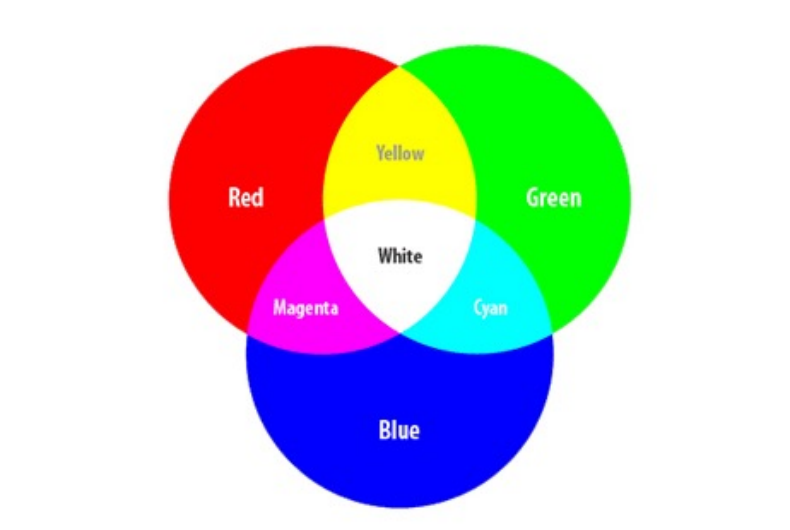


Hue ··· a colour or type of colour
Saturation ··· chroma
Value ··· Brightness

R G B
[70,20,100]

H S V
[10,150,220]

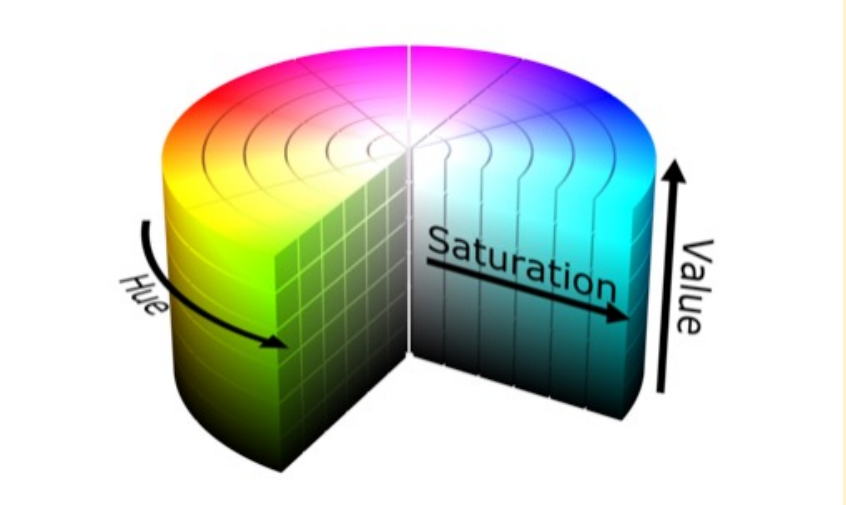
RGB and HSV



VS

R G B
[70,20,100]

I choice this method.



- Hue ··· a colour or type of colour
- Saturation ··· chroma
- Value ··· Brightness

H S V
[10,150,220]

HSV expression easily understandable for human.

Hue distribution



Python library can read this matrix.

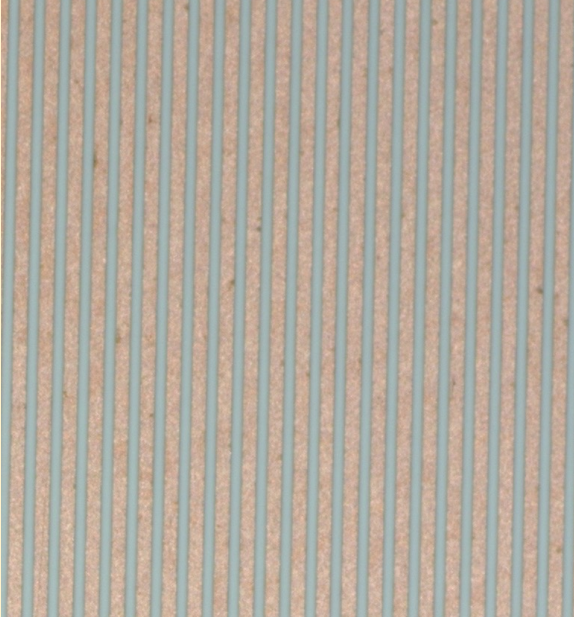
Value T_{ij2}

= T_{ijl}

$0 \leq i < 5472$

$0 \leq j < 3648$

$l = 0,1,2$



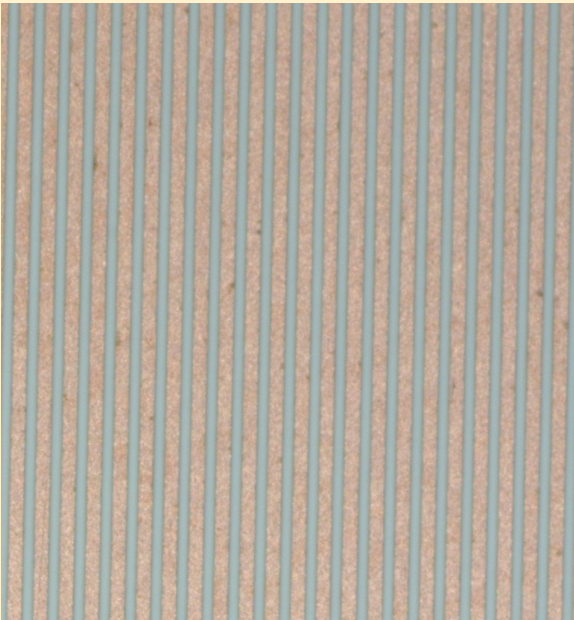
=

				10	100	22	30
				11	111	122	21
				9	101	29	111
			10	100	111	22	
			11	200	201	109	
			9	101	29	28	
					99	30	
		20	87	19	22		
		11	111	122	21		
		103	101	29	28		
		133	22	99	30		

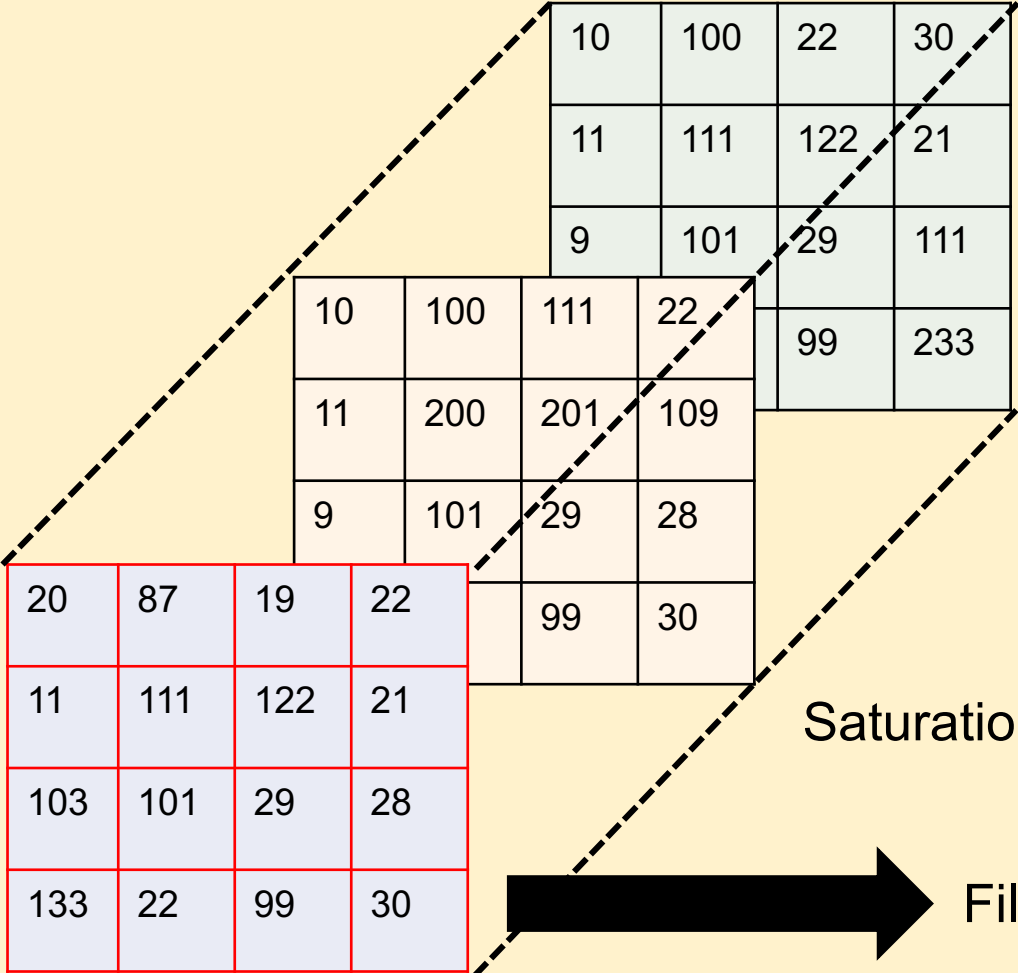
Saturation T_{ij1}

Hue T_{ij0}

Hue distribution



=



Value T_{ij2}

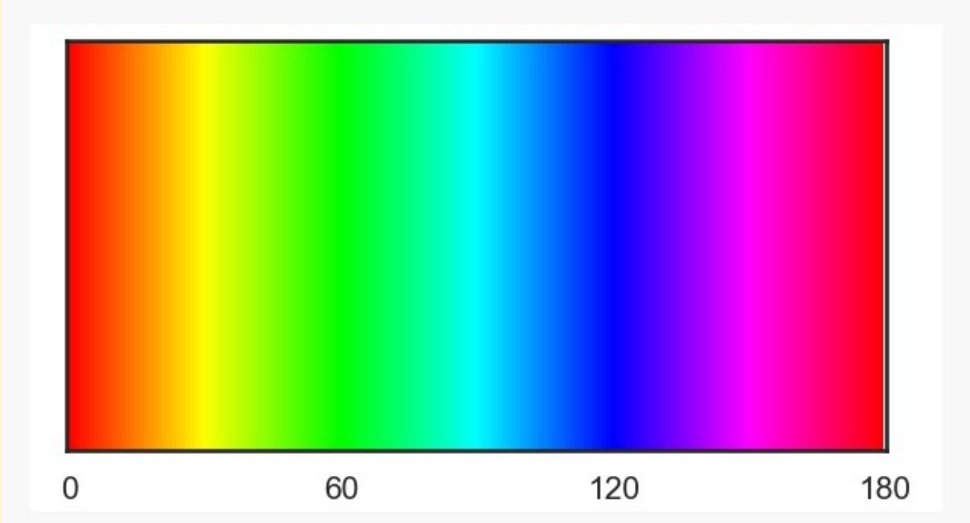
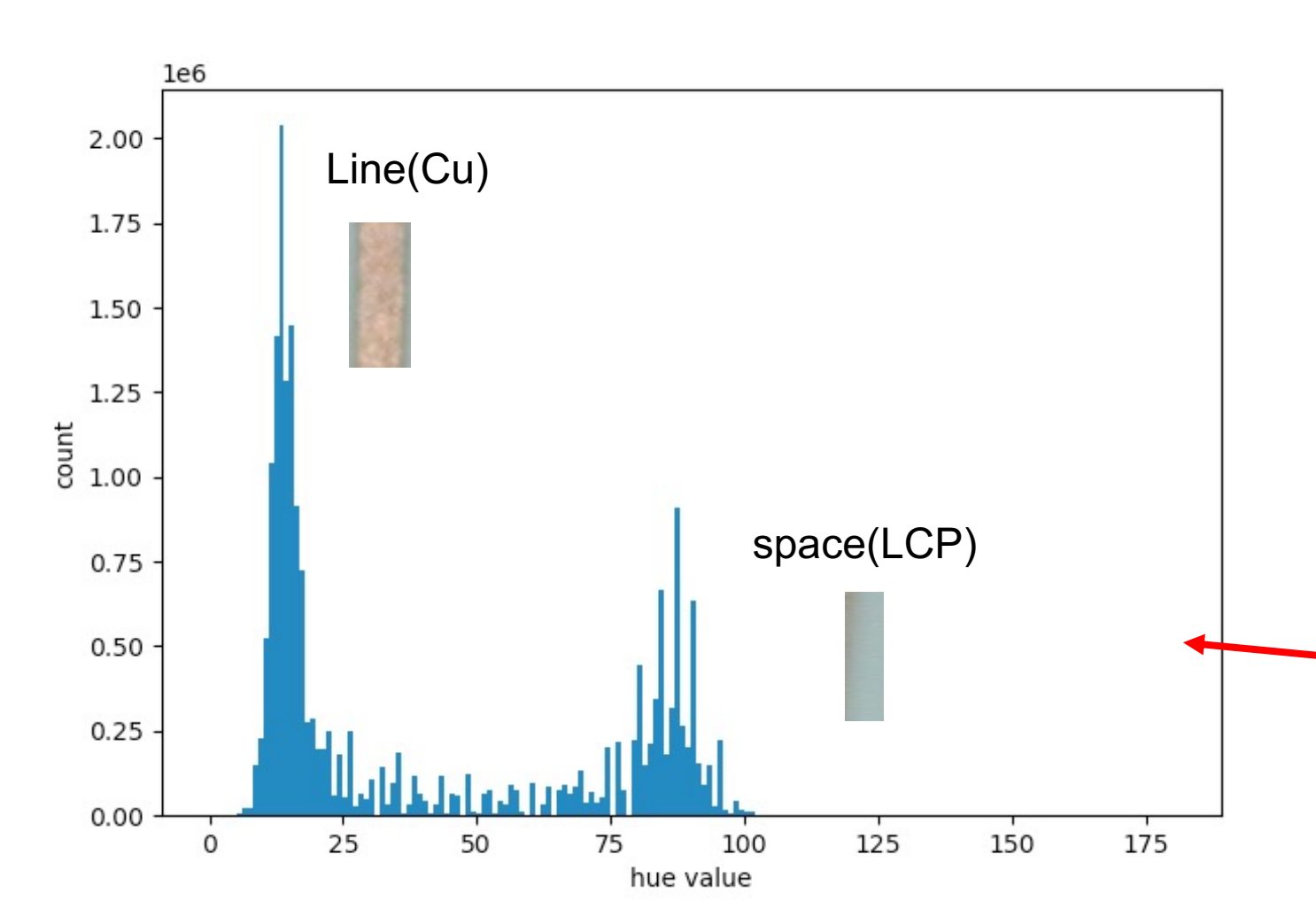
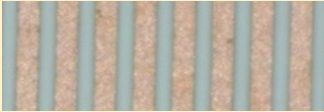
Saturation T_{ij1}

Hue T_{ij0}



Fill hue to histogram

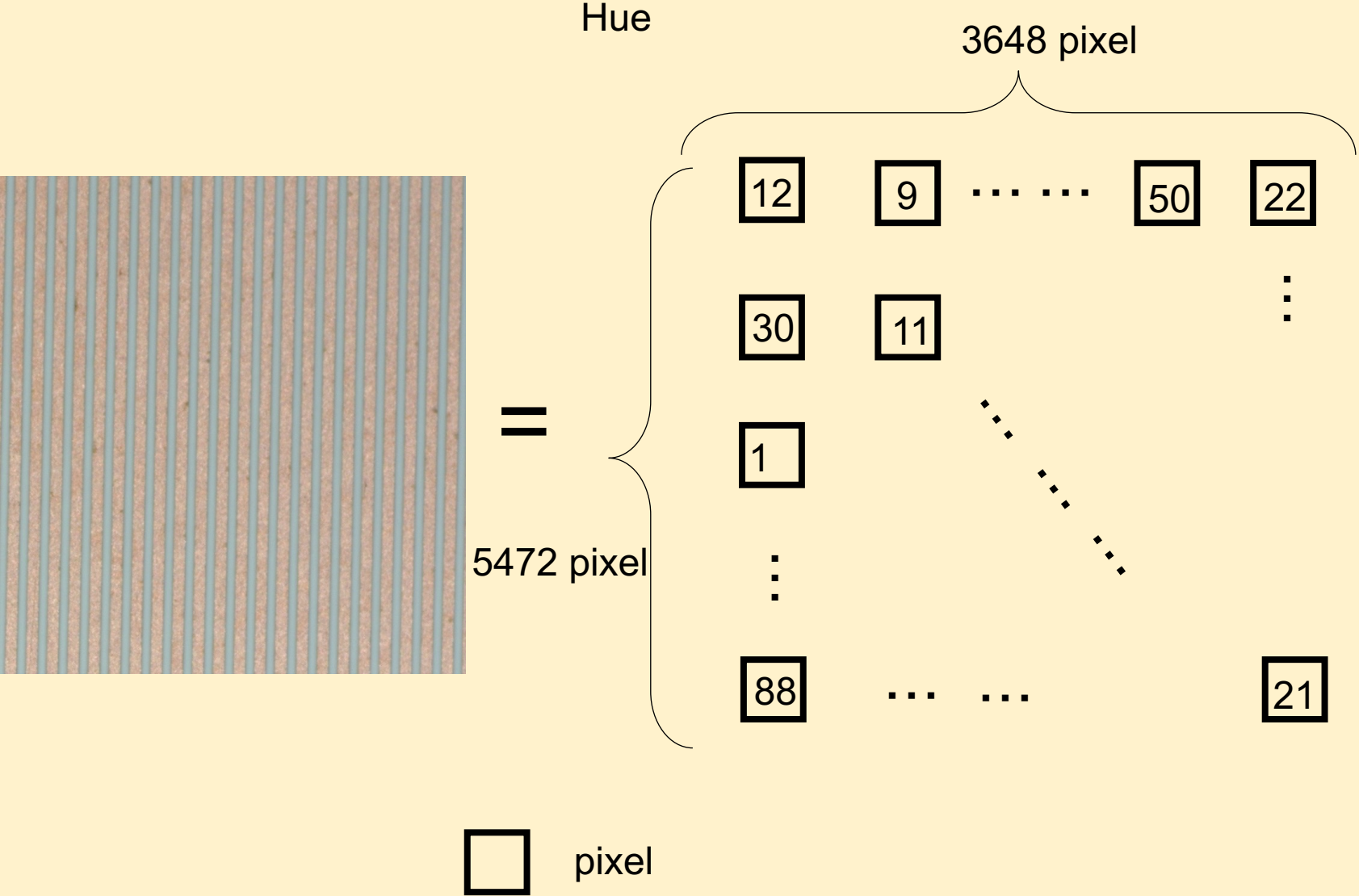
Hue distribution



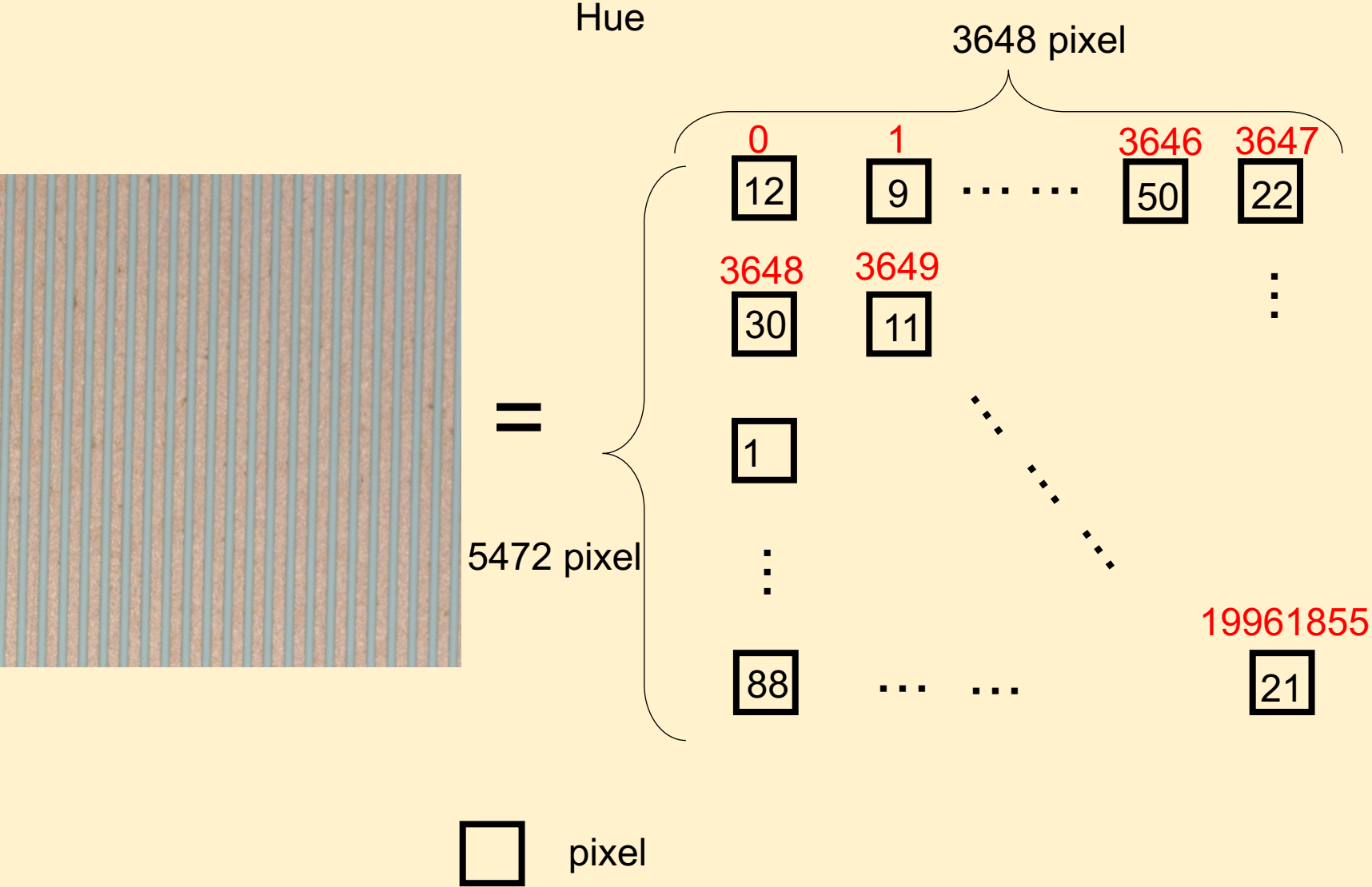
This shows distribution of hue at S=255,V=255.

We can separate two peak because using good camera(CS2000) and thinking way of taking photo.

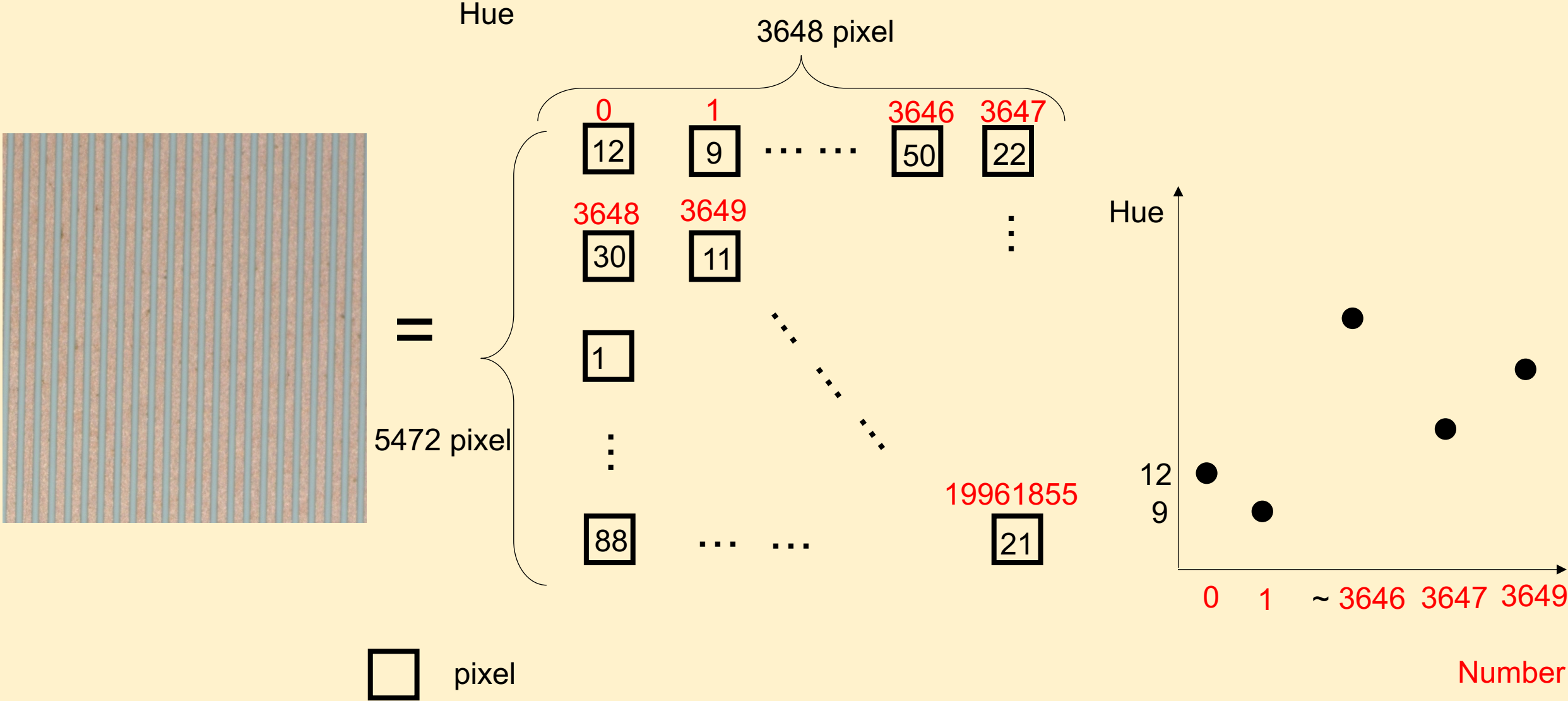
Hue graph



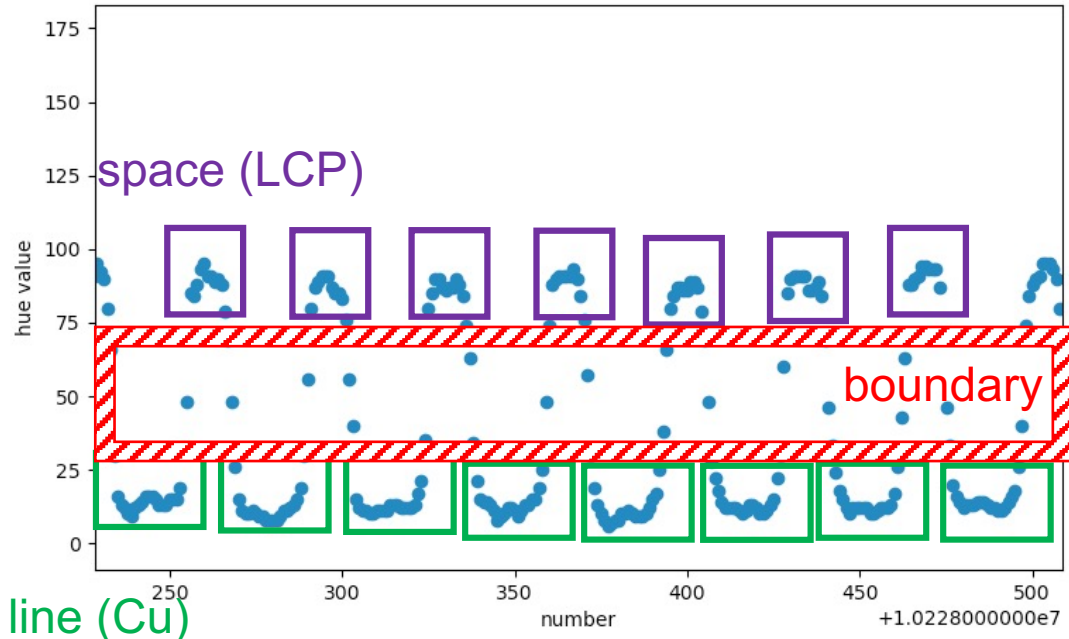
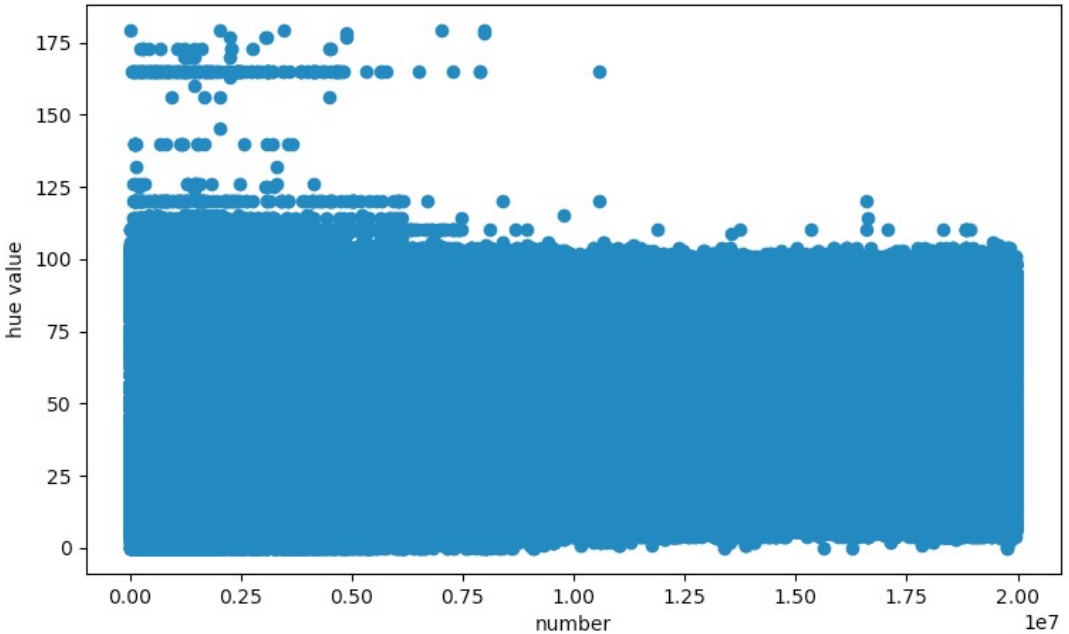
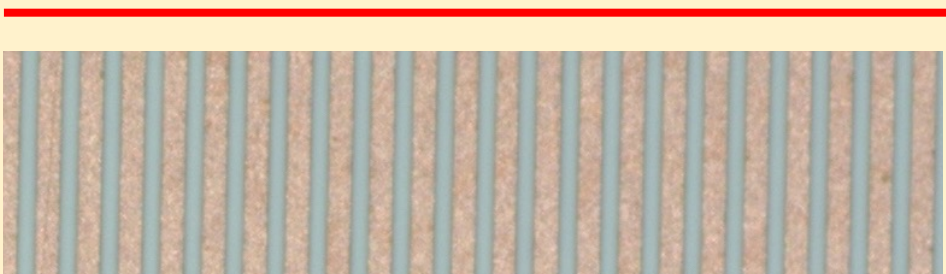
Hue graph



Hue graph

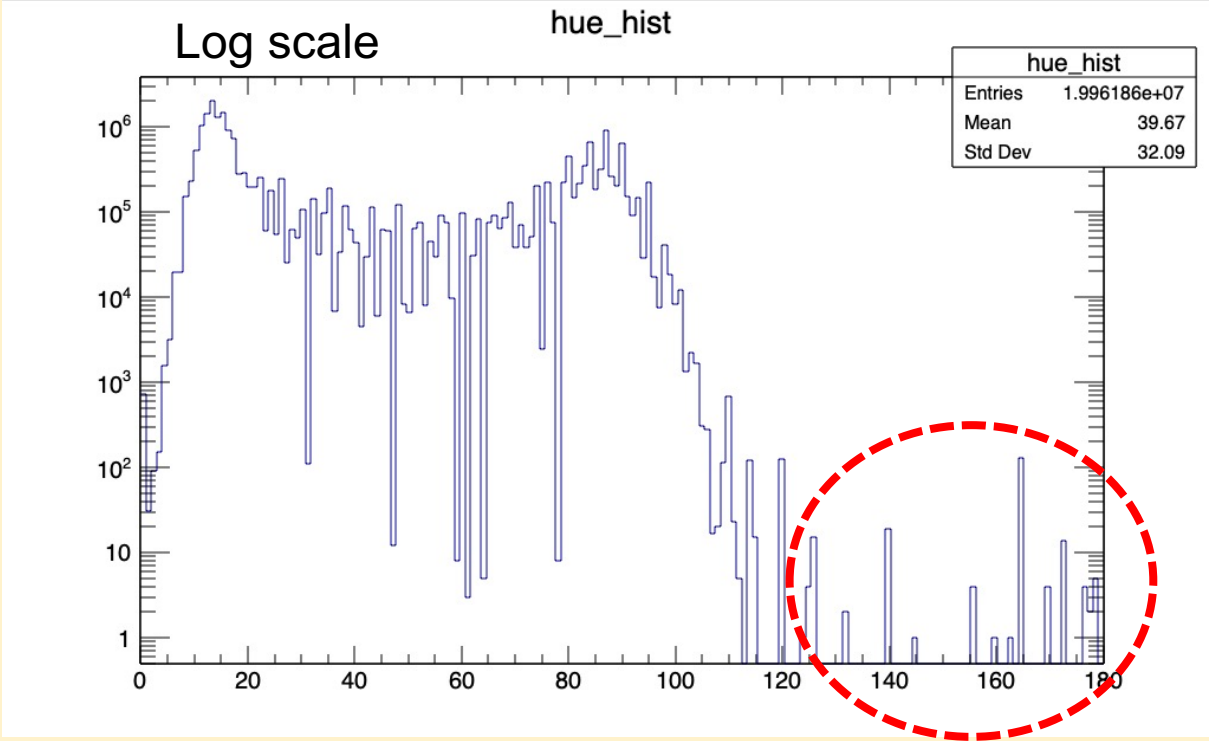
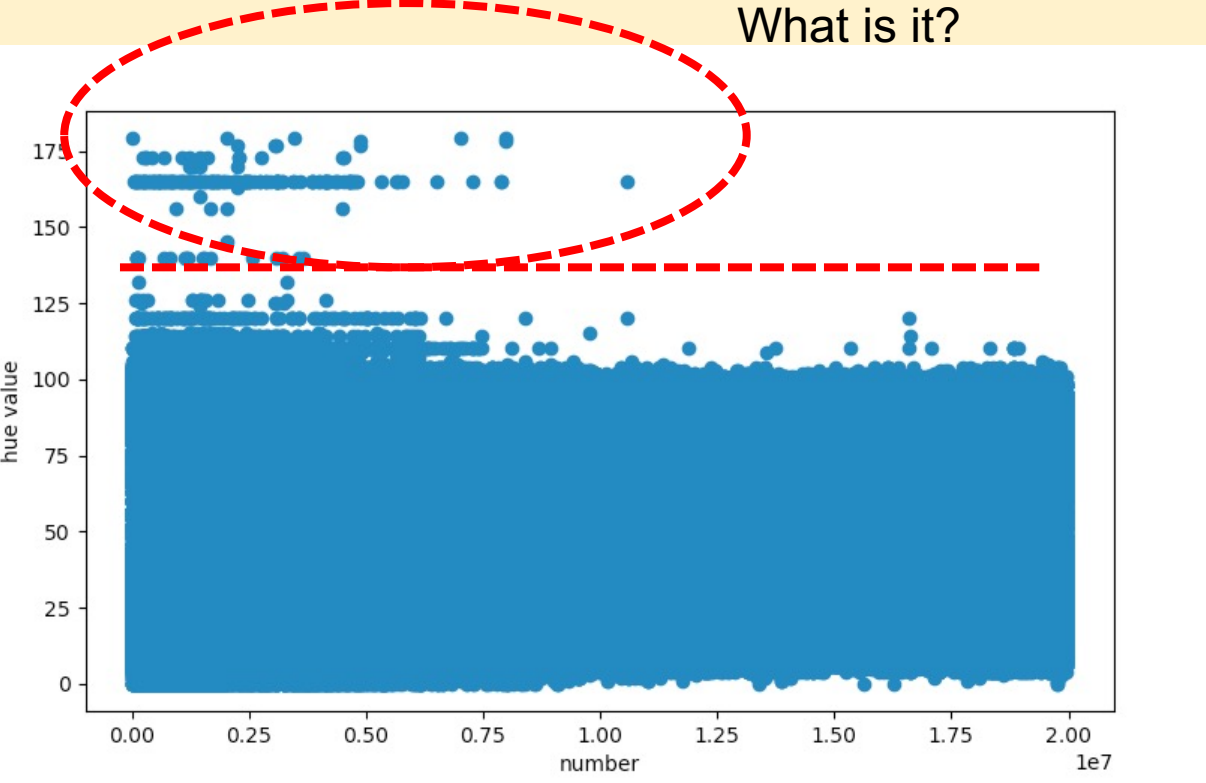


Hue graph

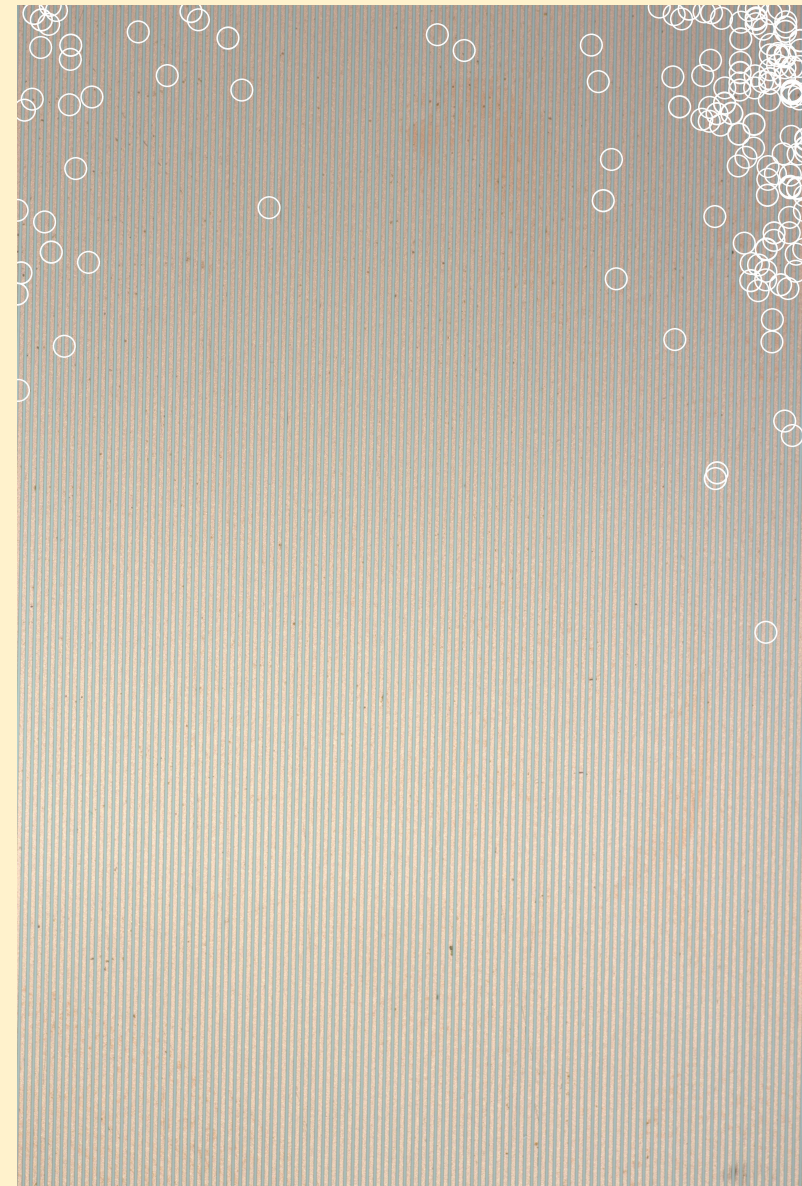
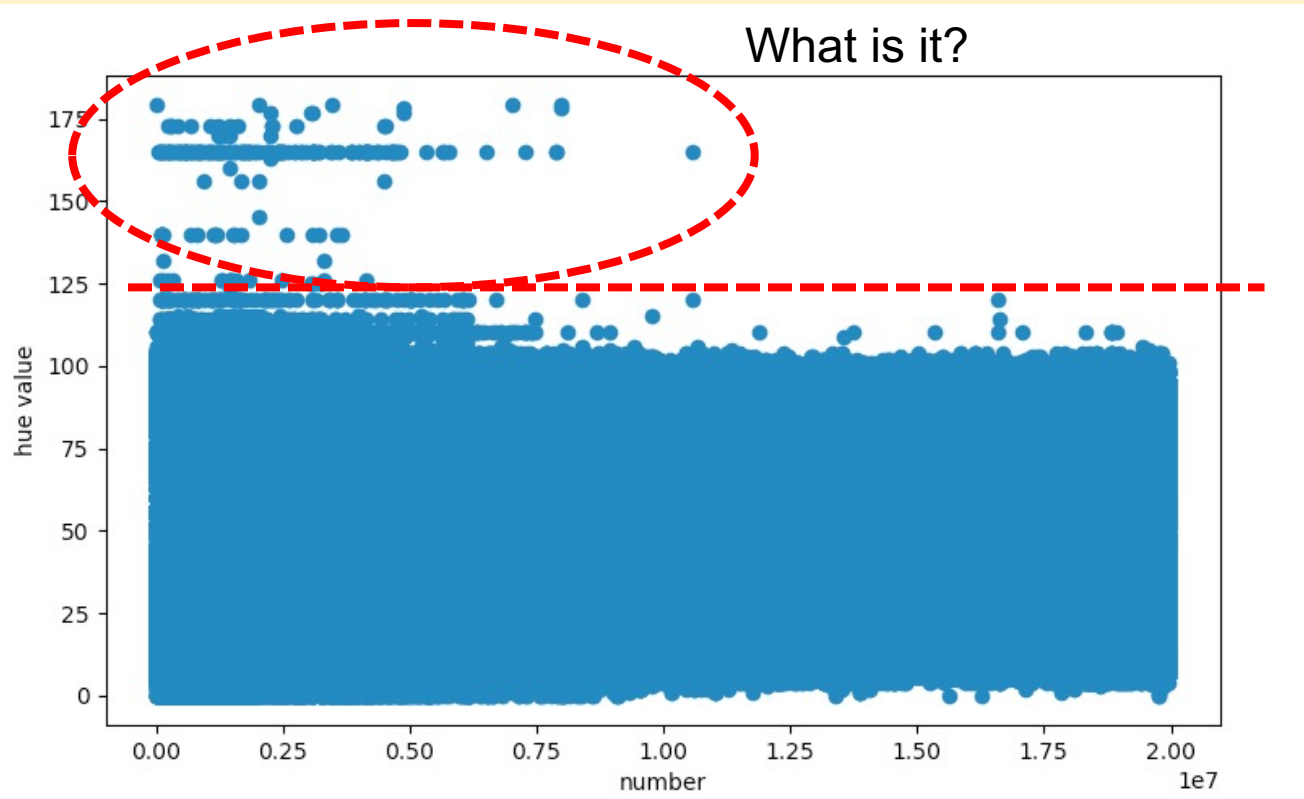


This graph be seem to be separate line and space.

Over 125 hue value

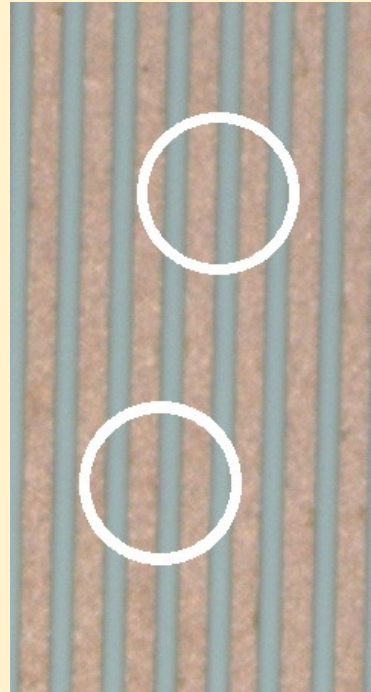
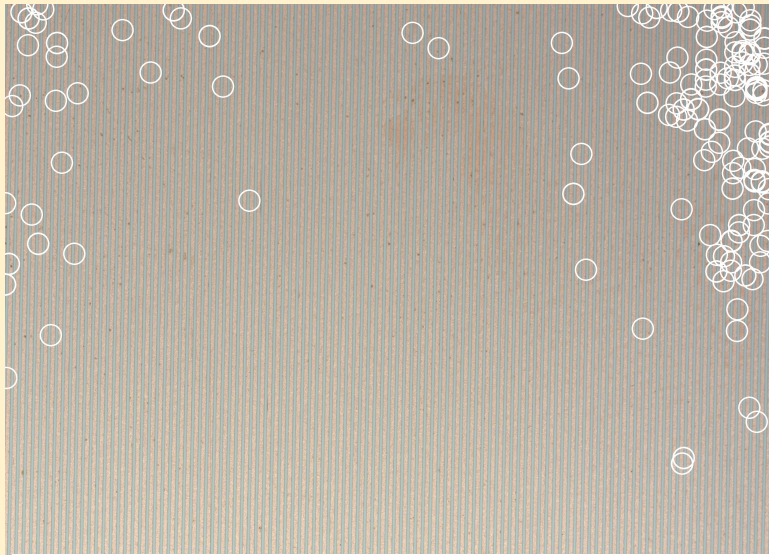


Over 125 hue value

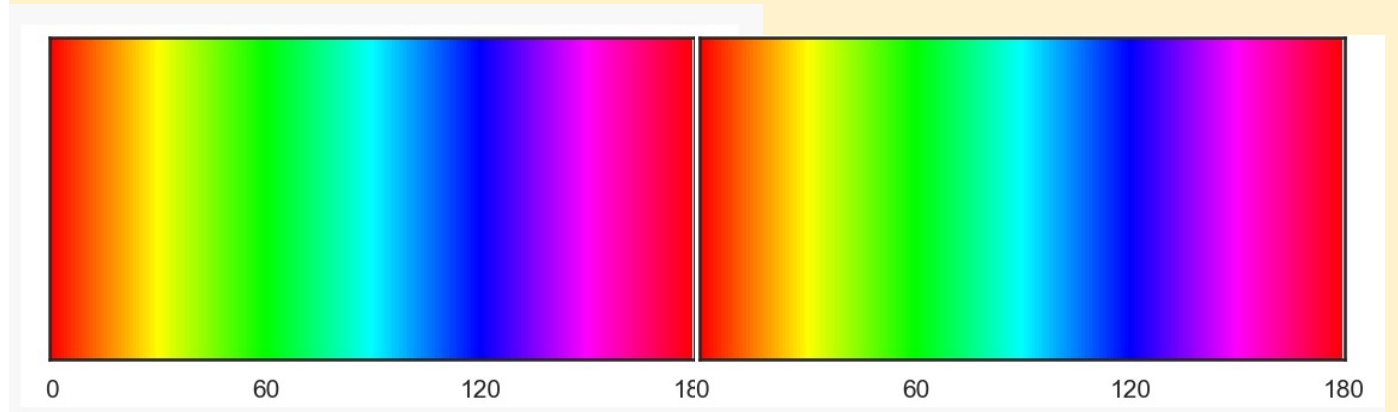
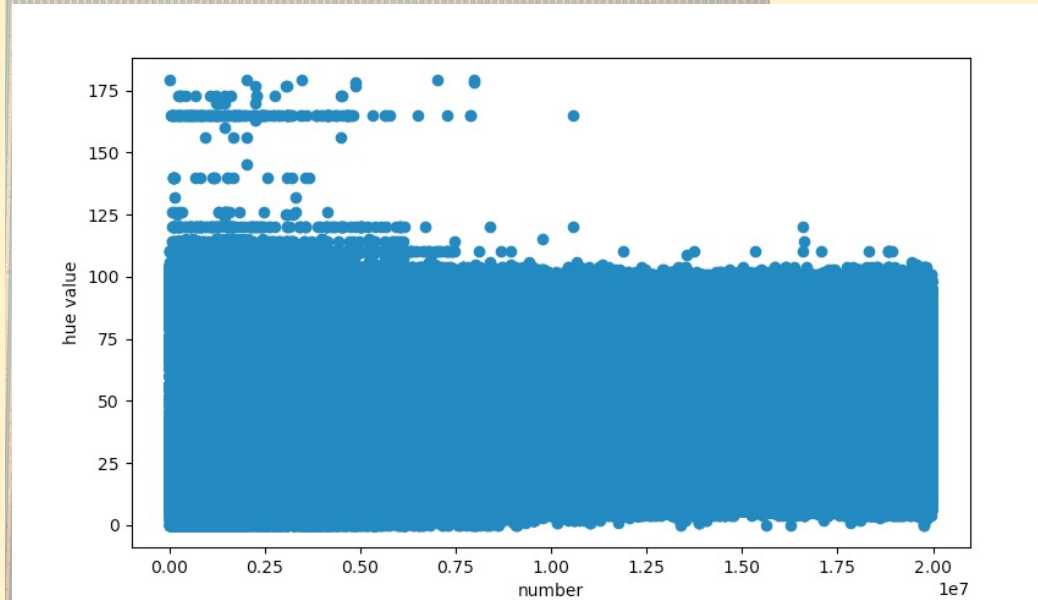
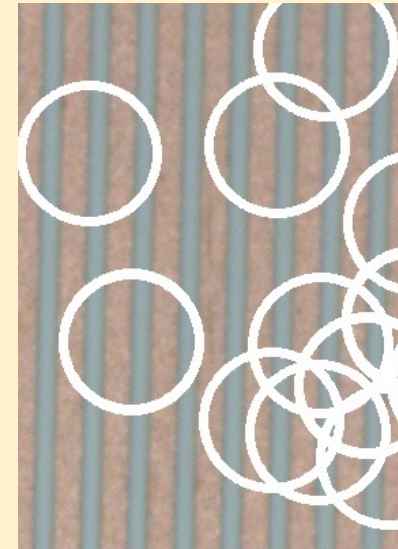


White circles means pixels with a hue value greater than 125.

Over 125 hue value

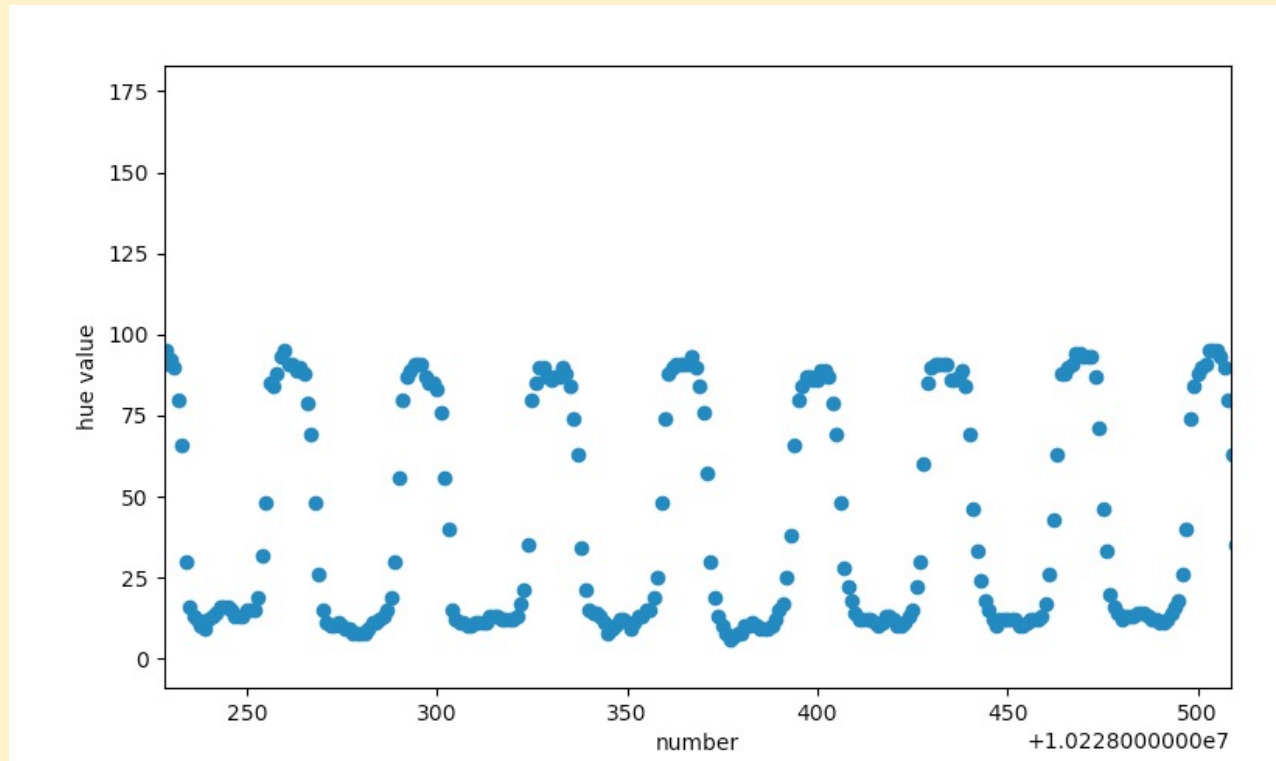


I can not understand difference.



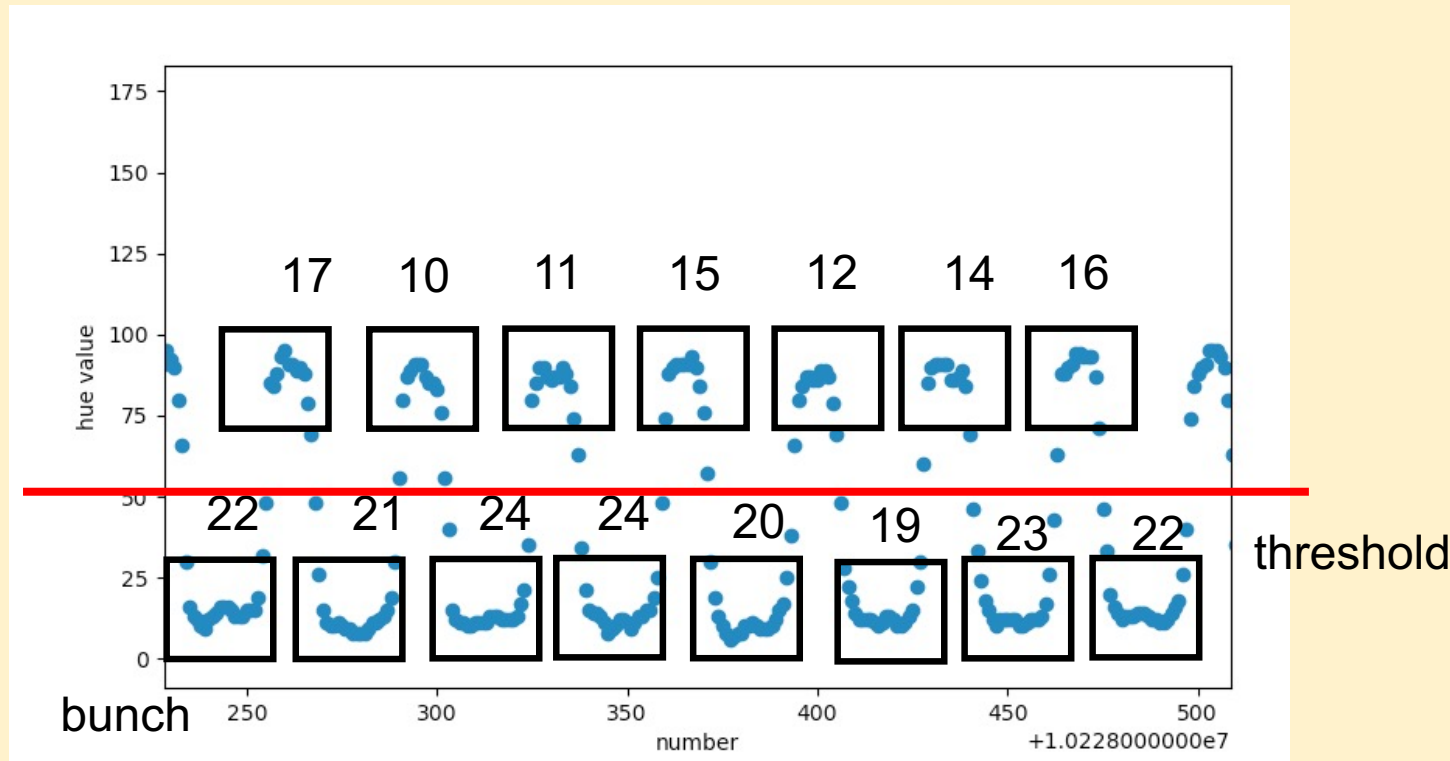
Hue have periodicity.

Algorithm



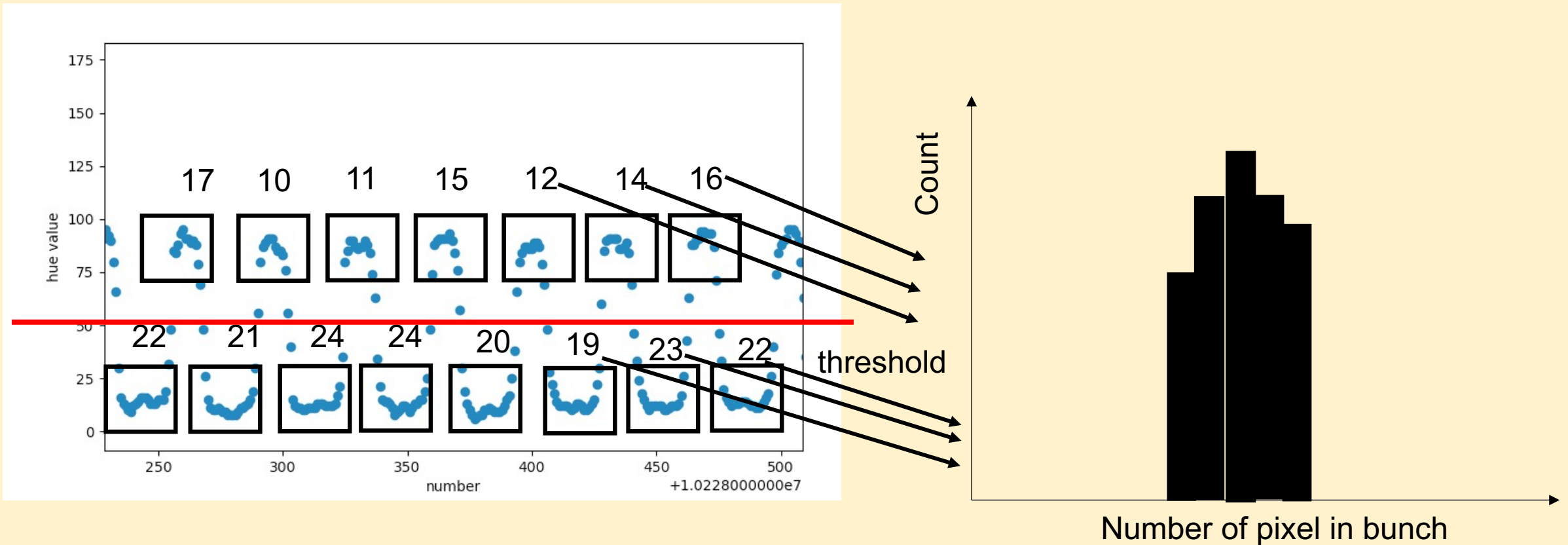
Algorithm

1.Counting dot(pixel) in bunch.



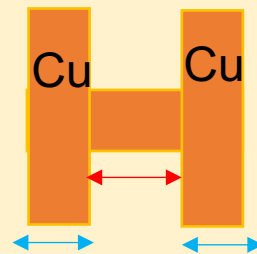
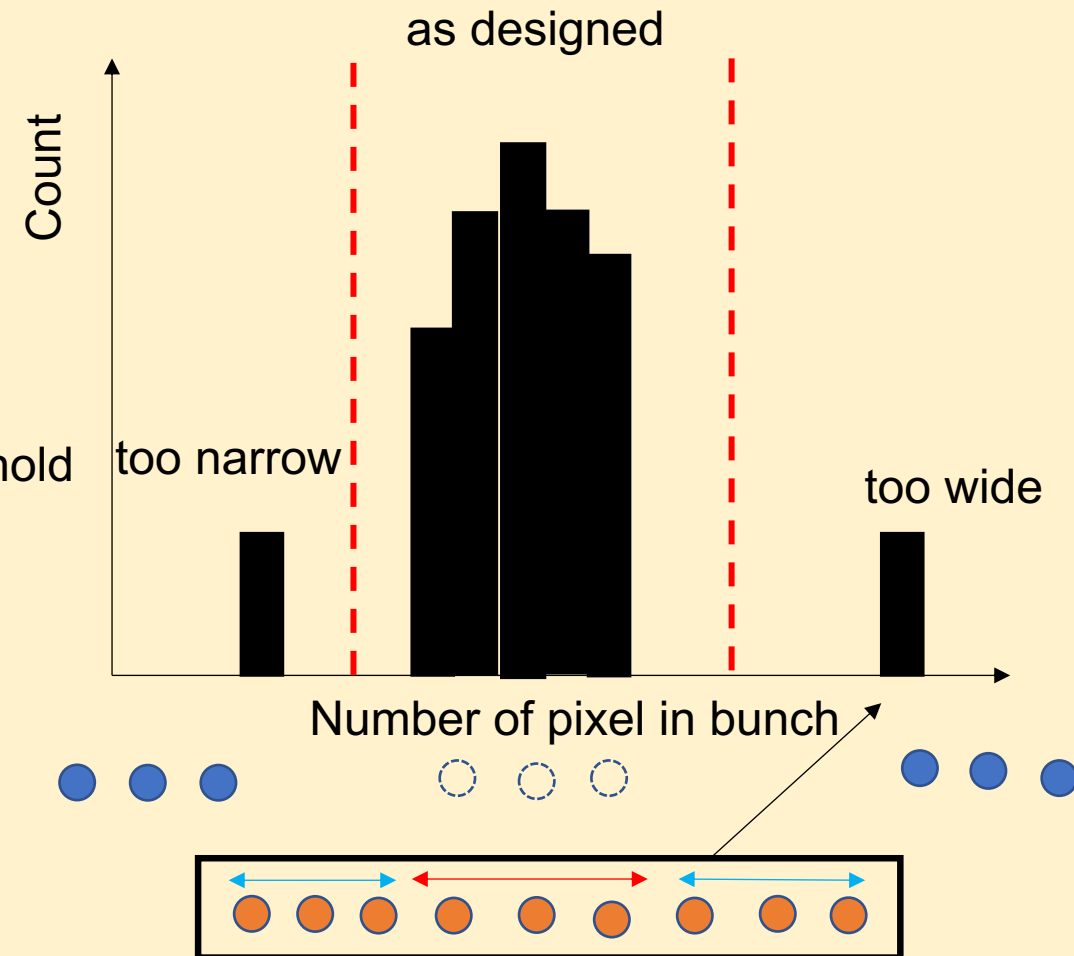
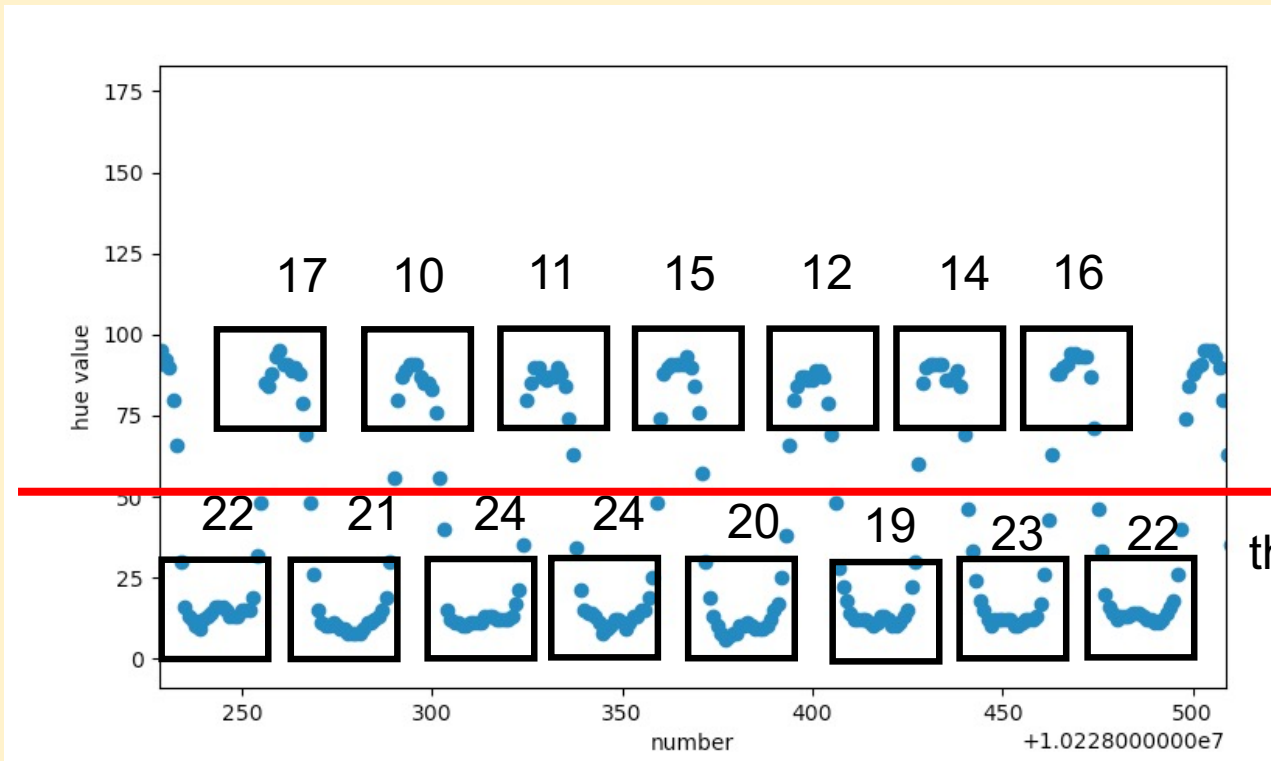
Algorithm

- 1.Counting dot(pixel) in bunch.
- 2.Fill number of dot in bunch to histogram.



Algorithm

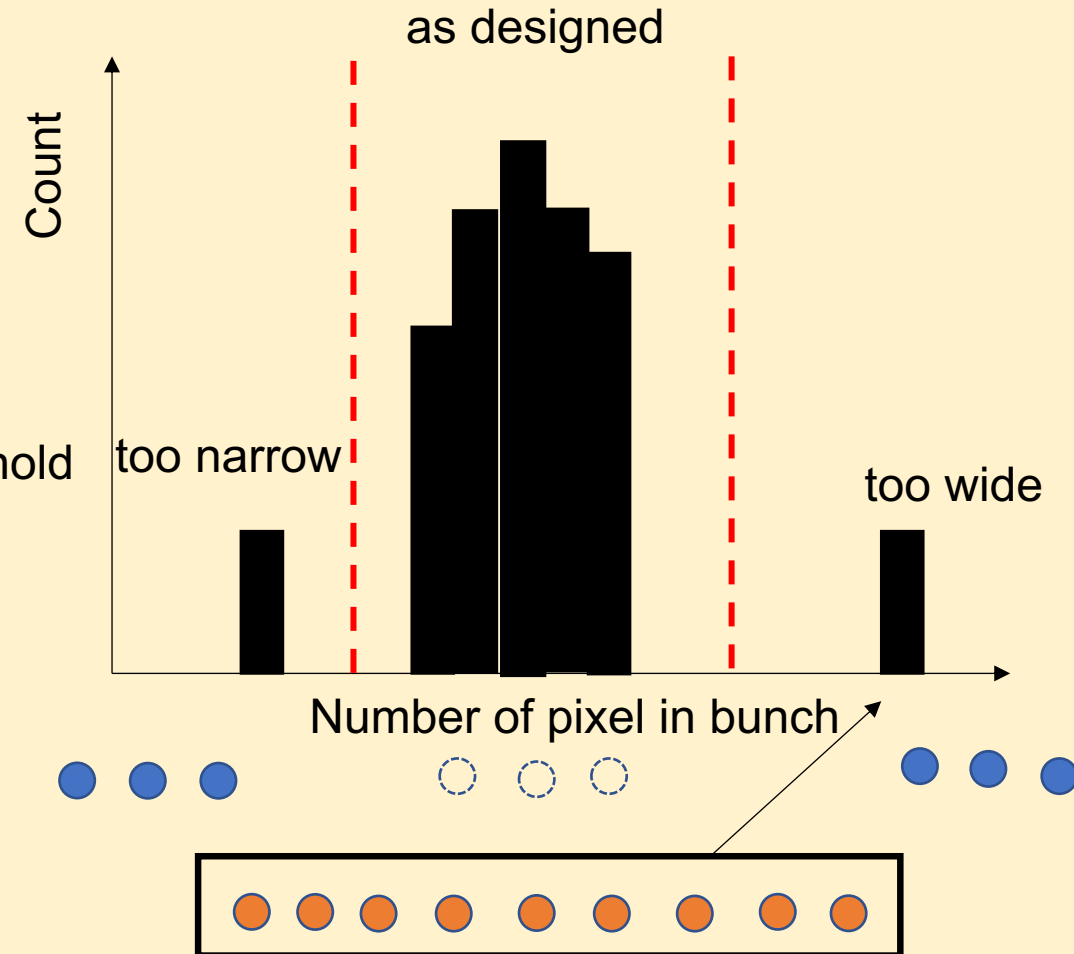
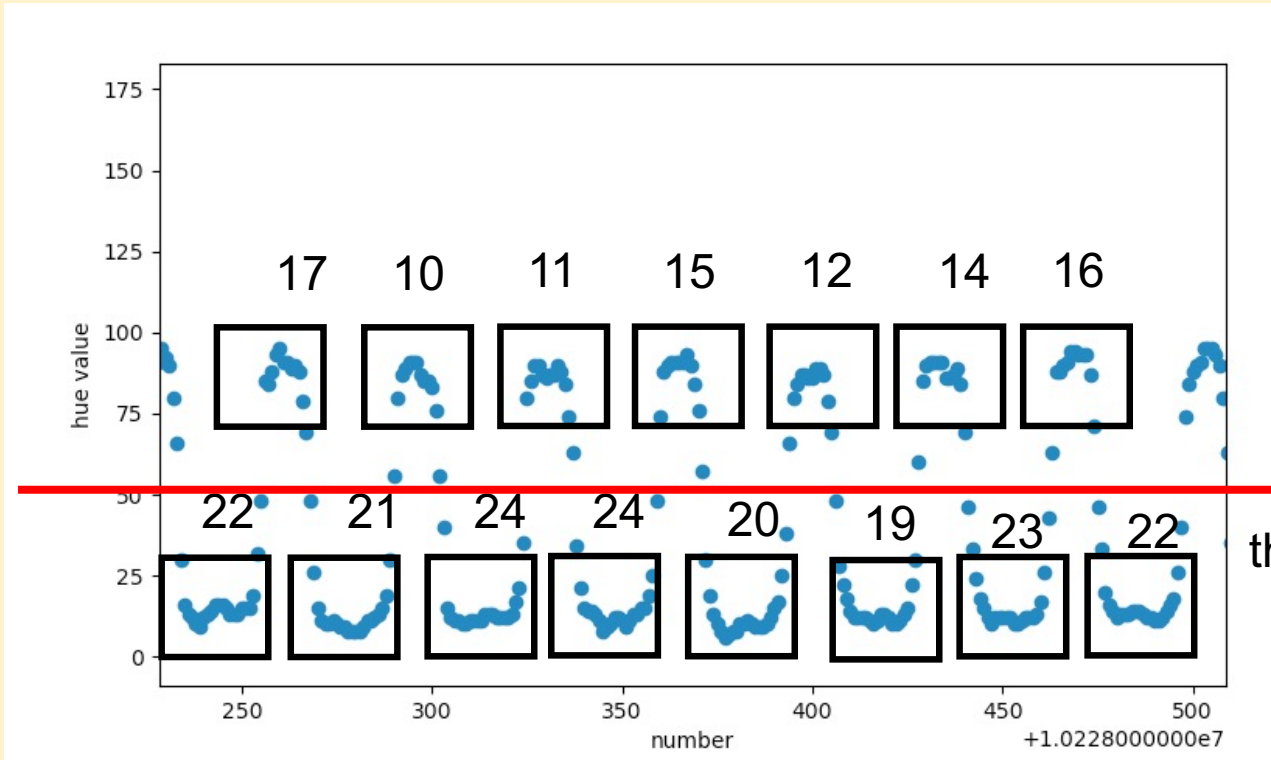
- 1.Counting dot(pixel) in bunch.
- 2.Fill number of dot in bunch to histogram.
- 3.Determine threshold.



Short situation. RBRC-Meeting

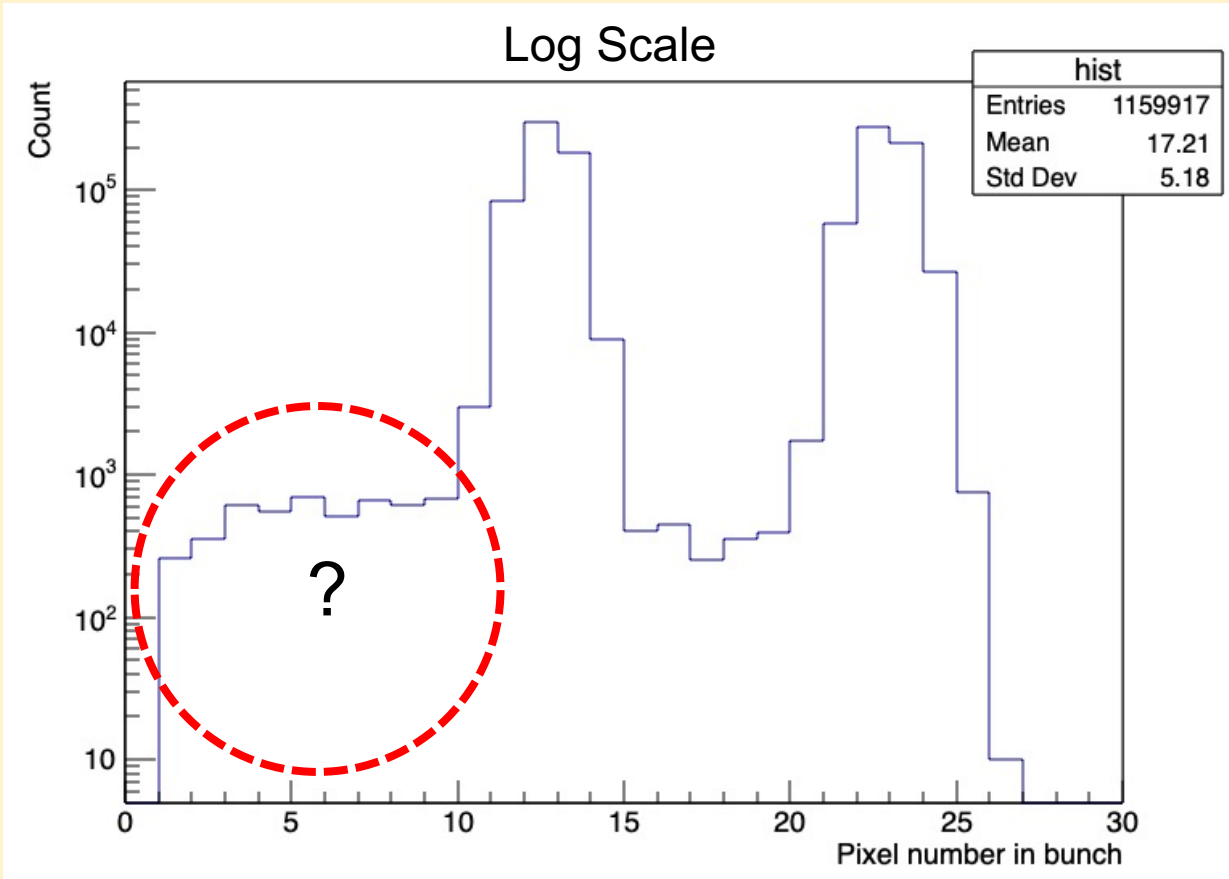
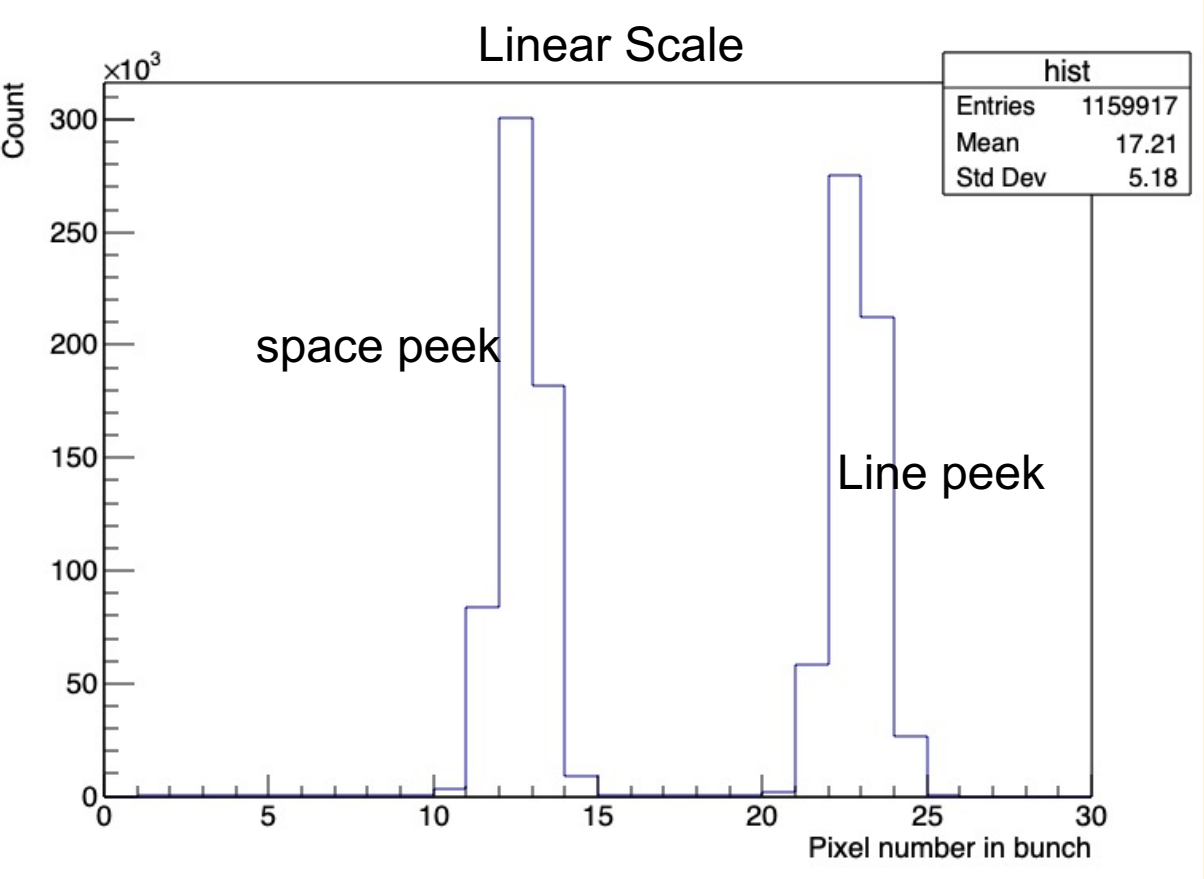
Algorithm

- 1.Counting dot(pixel) in bunch.
- 2.Fill number of dot in bunch to histogram.
- 3.Determine threshold.
- 4.Marking



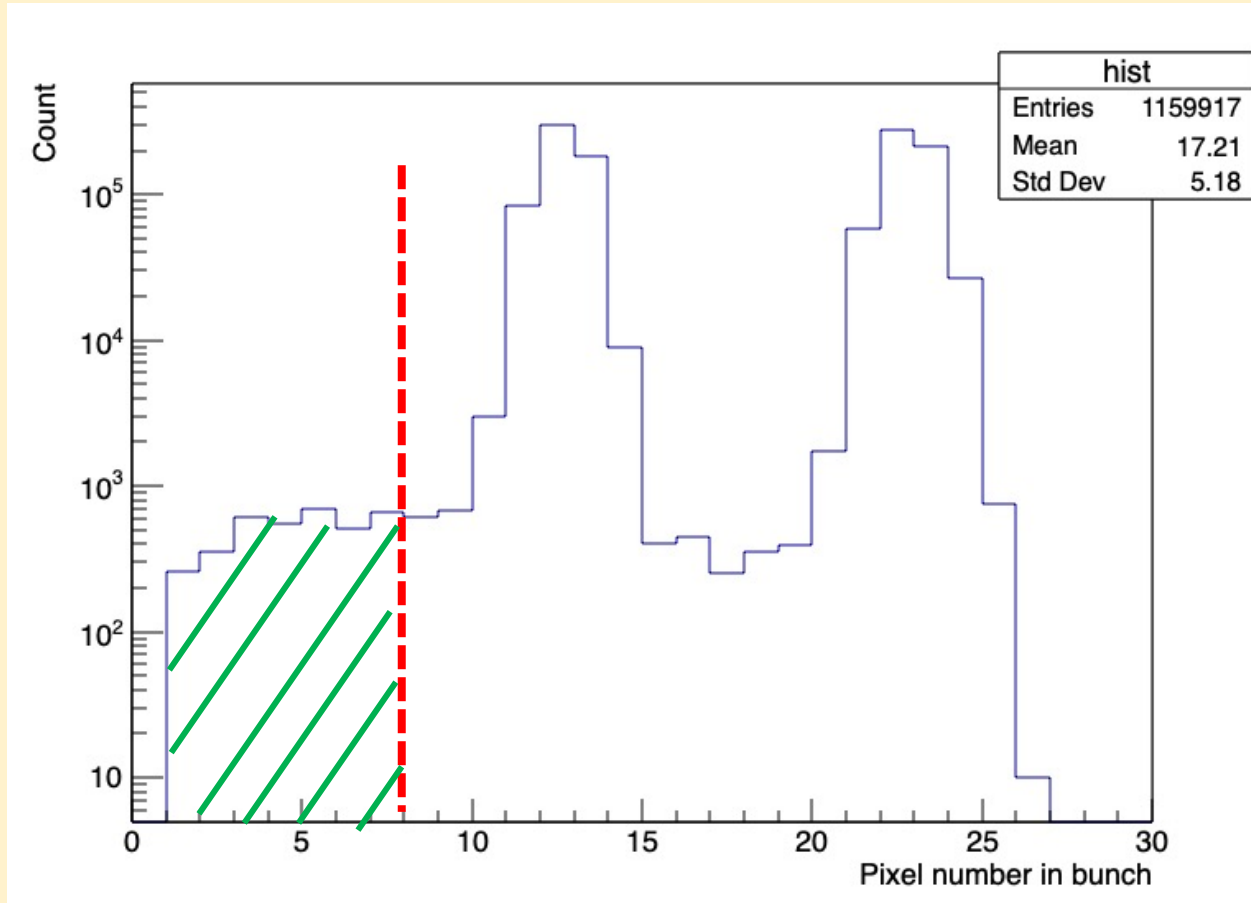
To apply this algorithm, the LCP and Copper lines have to be separated in two clear peeks.

Number of pixel in bunch

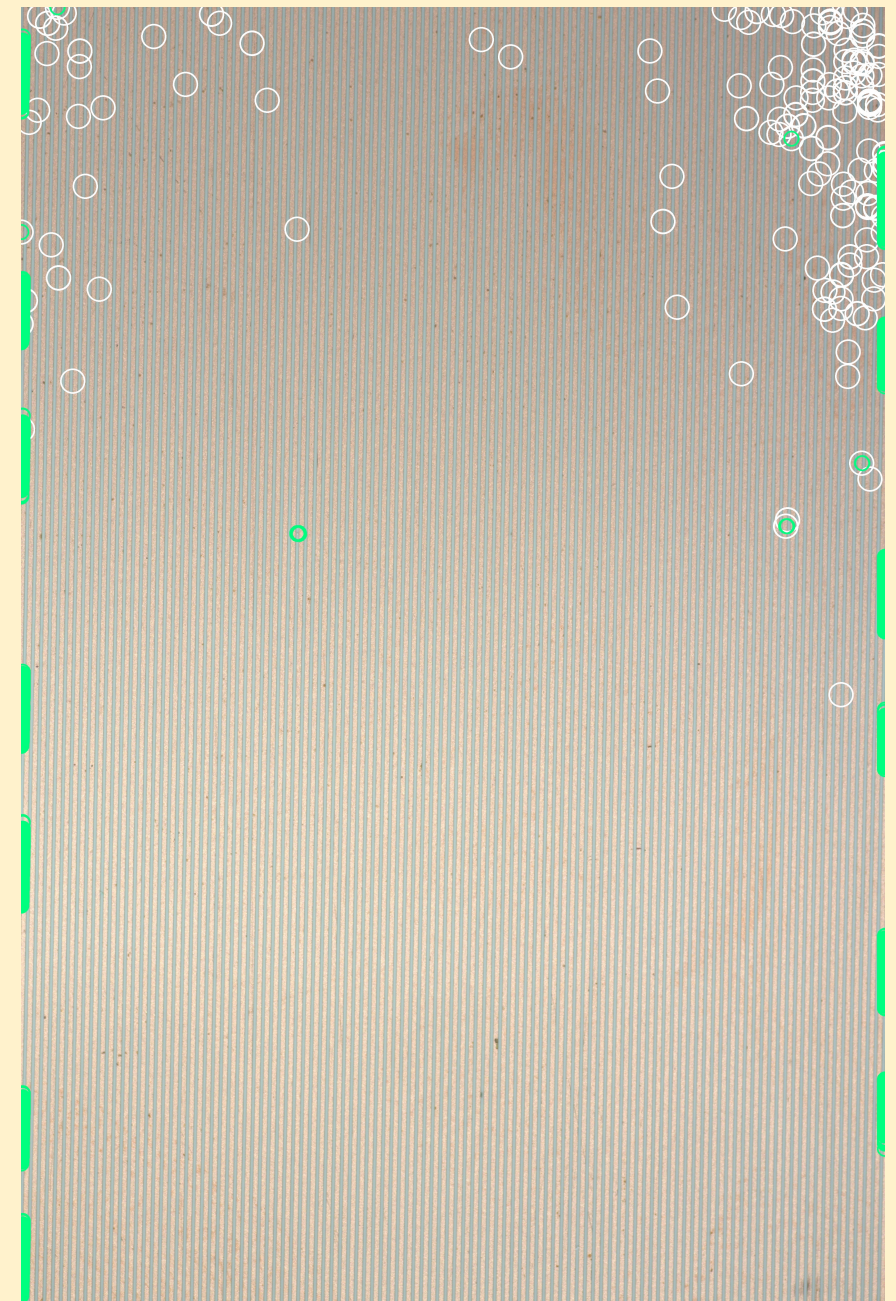


short in pixel number counting. (see next page)

marking

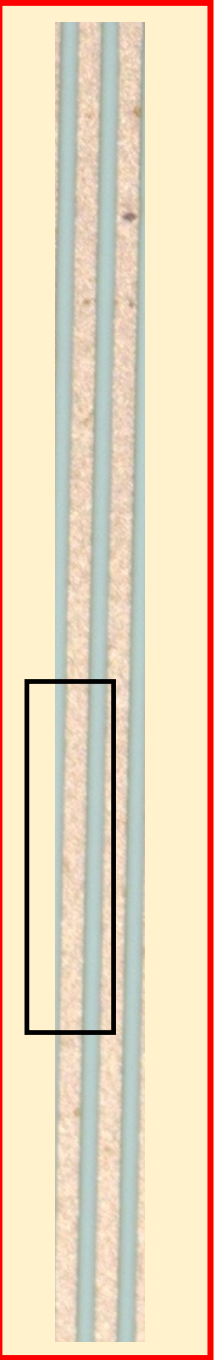


"Pixel number in bunch < 8"



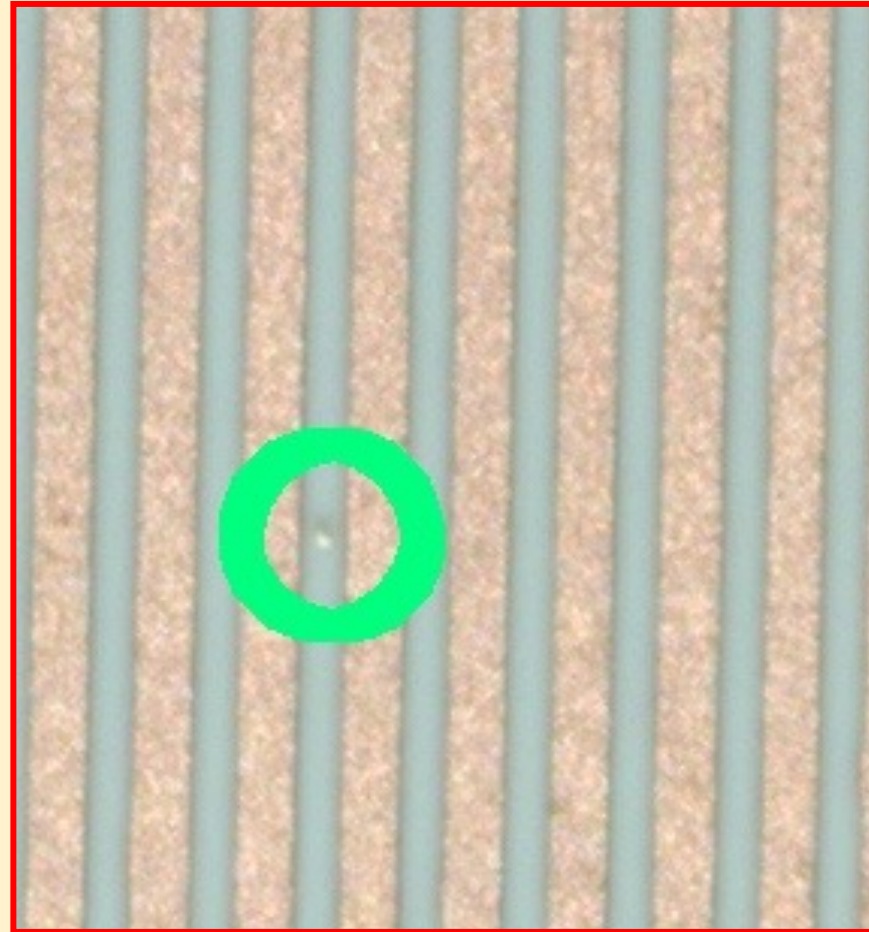
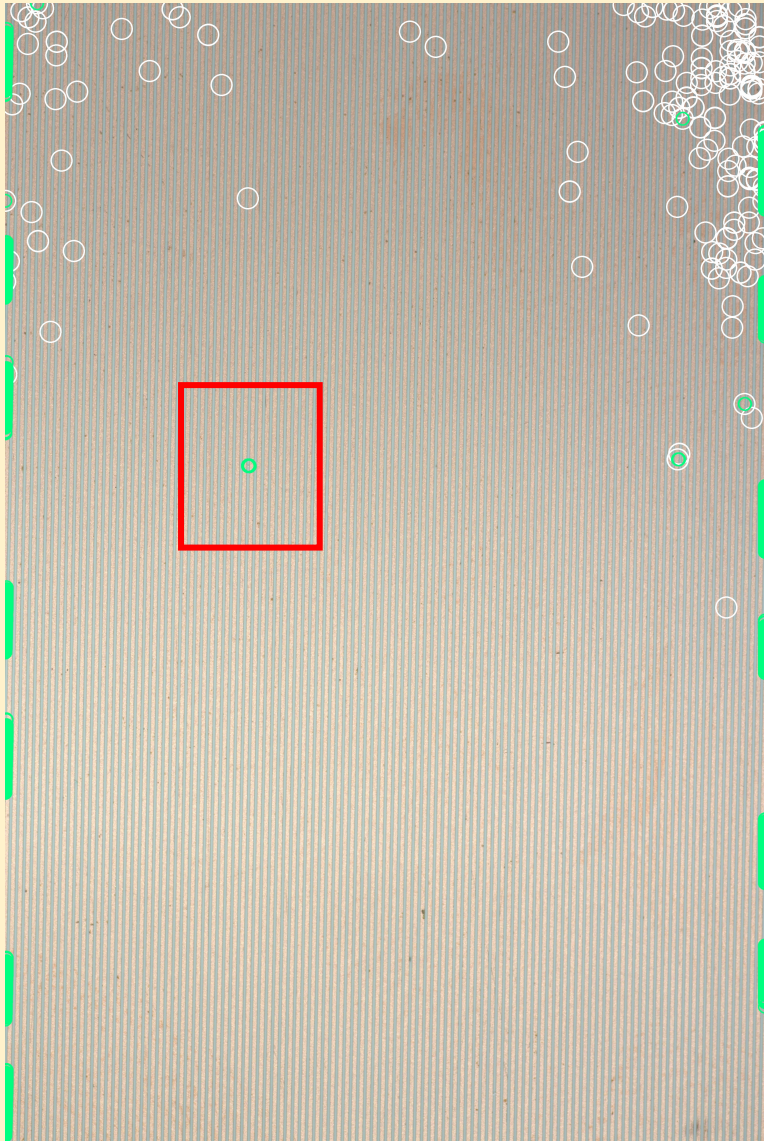
"Pixel number in bunch < 8" spot is indicated by a green circle.

Marking



Edge

Marking



A small dot in the LCP zone was detected as the error.

Remaining Issue: Processing speed

Reading sample image.
Extracting RGB data from image.
Converting RGB to HSV.



Remaining Issue: Processing speed

Reading sample image.
Extracting RGB data from image.
Converting RGB to HSV.



Count number of pixels in a given bunch.

Fill hue histogram for the threshold determination.

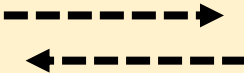
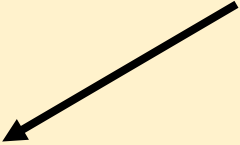
Remaining Issue: Processing speed

Reading sample image.
Extracting RGB data from image.
Converting RGB to HSV.



Count number of pixels in a given bunch.

Fill hue histogram for the threshold determination.



Display the error mark on the pixel which didn't match with the expected pattern.

Remaining Issue: Processing speed

Reading sample image.
Extracting RGB data from image.
Converting RGB to HSV.



Count number of pixels in a given bunch.



Fill hue histogram for the threshold determination.



Display the error mark on the pixel which didn't match with the expected pattern.

The 1st attempt was to process the pixel number counting process within Python, but it took longer than 30 minutes and never finished.

The data volume (5472 x 3649 pixels) was too large for Python.

To achieve the real time image processing, the goal of image processing time is <1sec.

Remaining Issue: Processing speed

Reading sample image.
Extracting RGB data from image.
Converting RGB to HSV.



Count number of pixels in a given bunch.

Fill hue histogram for the threshold determination.



Display the error mark on the pixel which didn't match with the expected pattern.

By replacing the pixel counting routine by c++ code, the total process time is reduced by 2orders of magnitude. Now it is ~5sec.

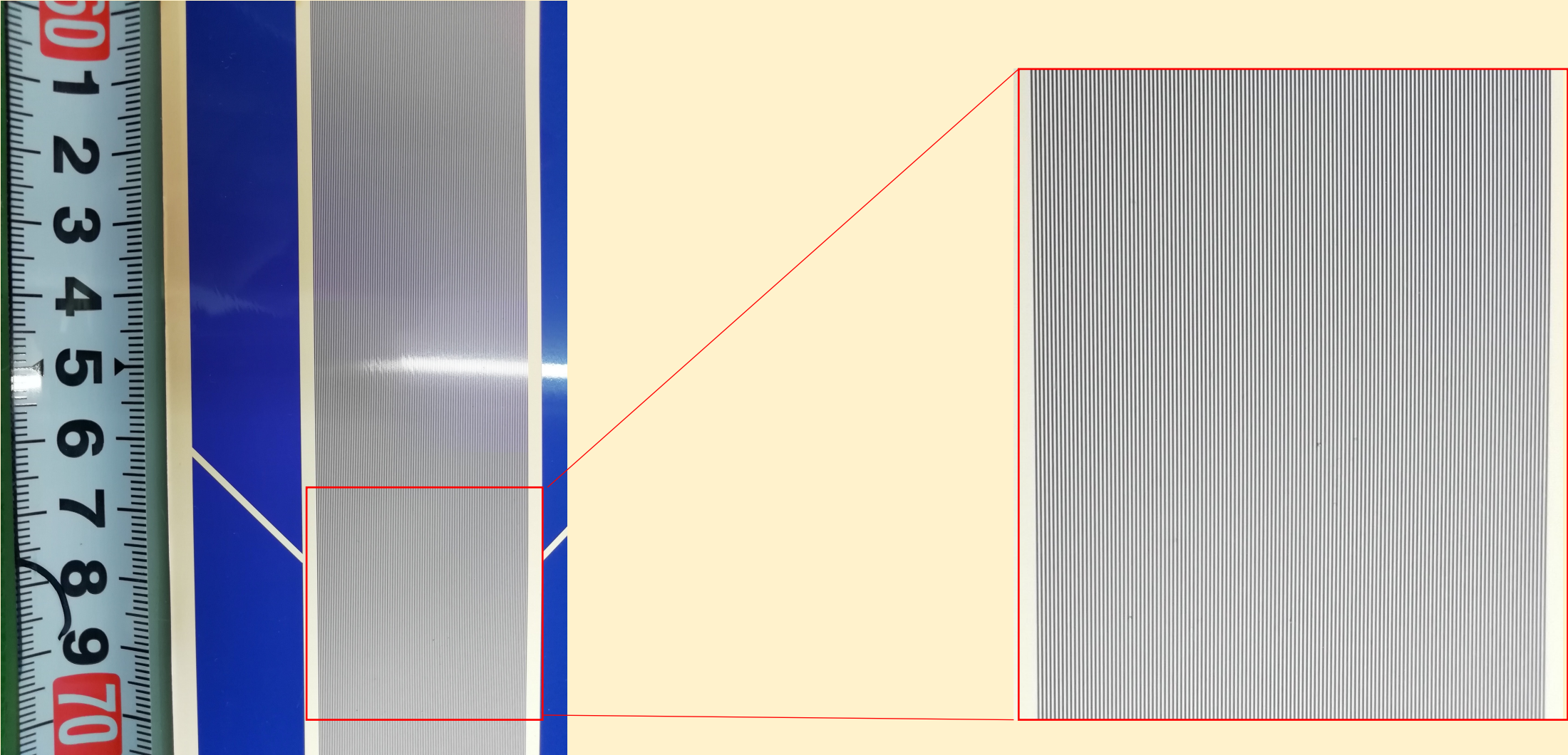
The algorism has to be kept simple and avoid any sophisticated analysis to avoid increasing process time.

Summary and next step

- I develop software which can detect abnormal patterns in the signal line layer of the BEC.
- CS2000 camera is under PO now. Once it is delivered, the algorithm will be tested under various conditions.

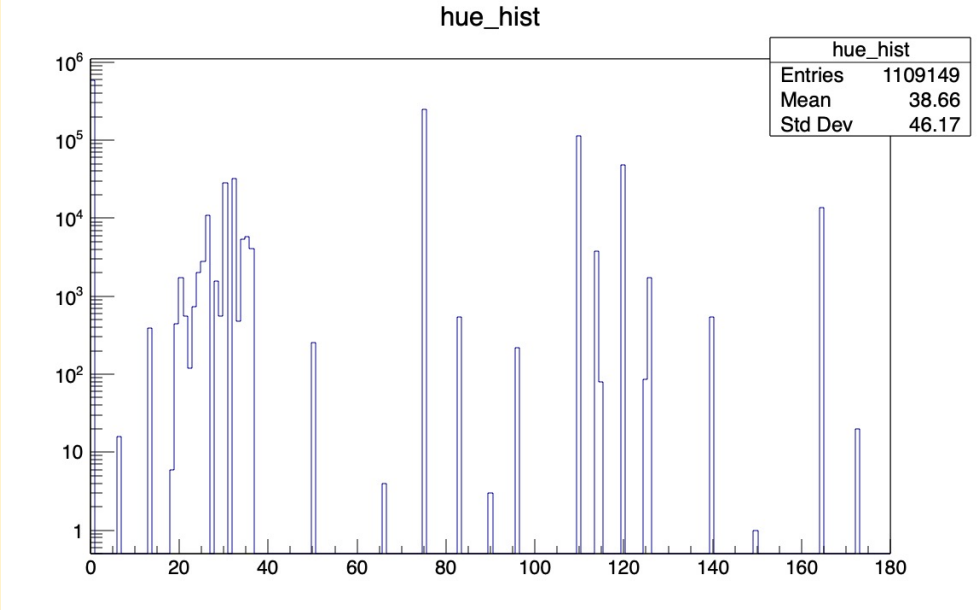
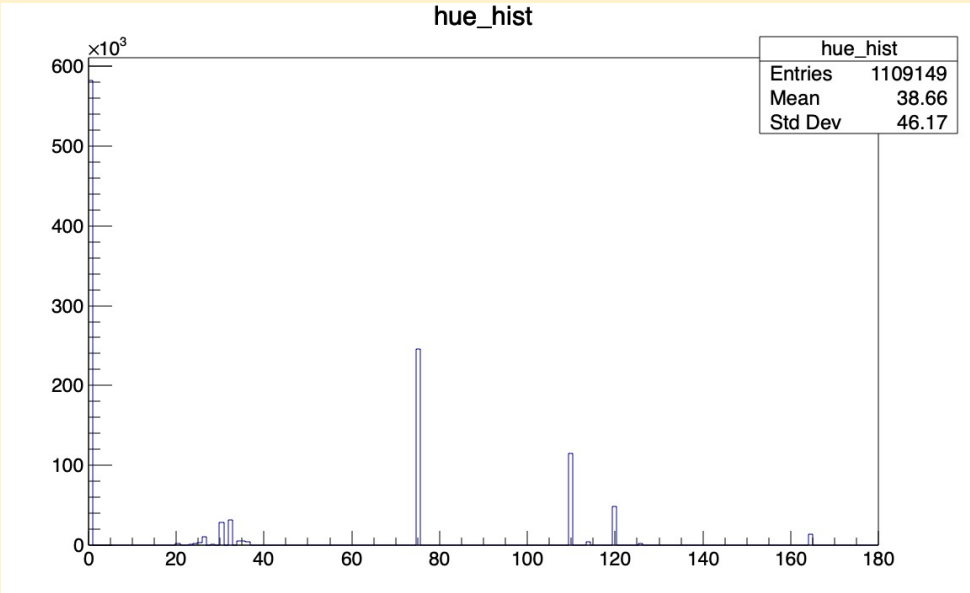
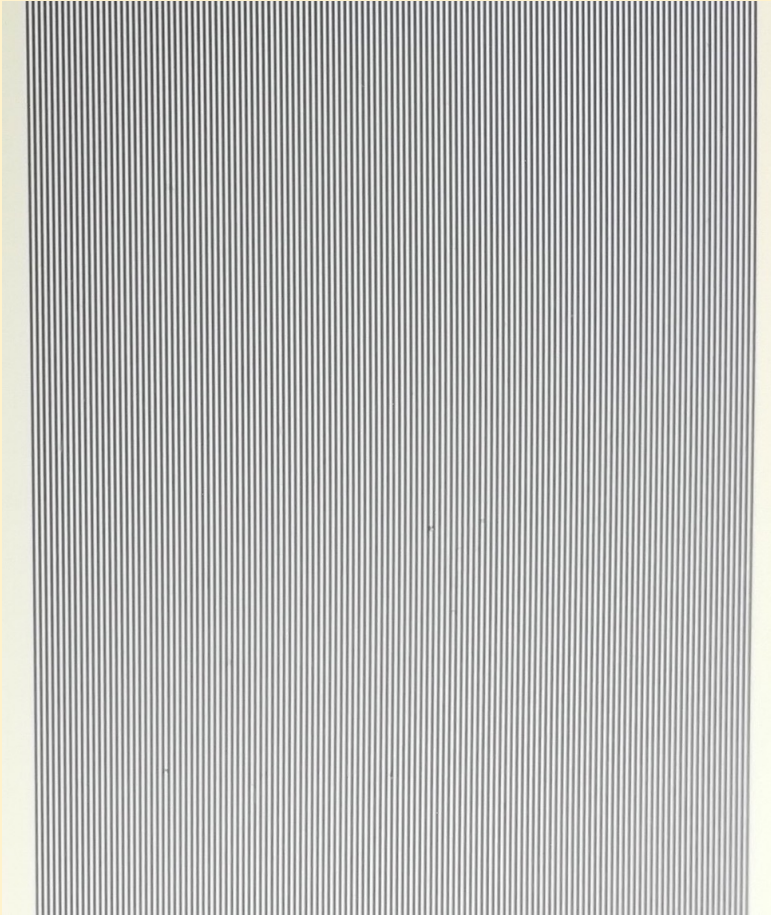
BACK UP

Dry film



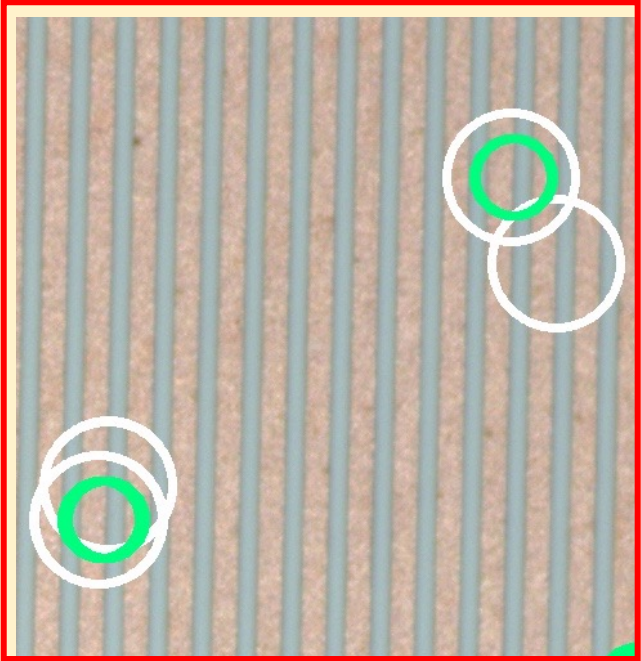
This photo was taken by Nakagawa Print Electronics Laboratory co.

Dry film



I can not know feature.

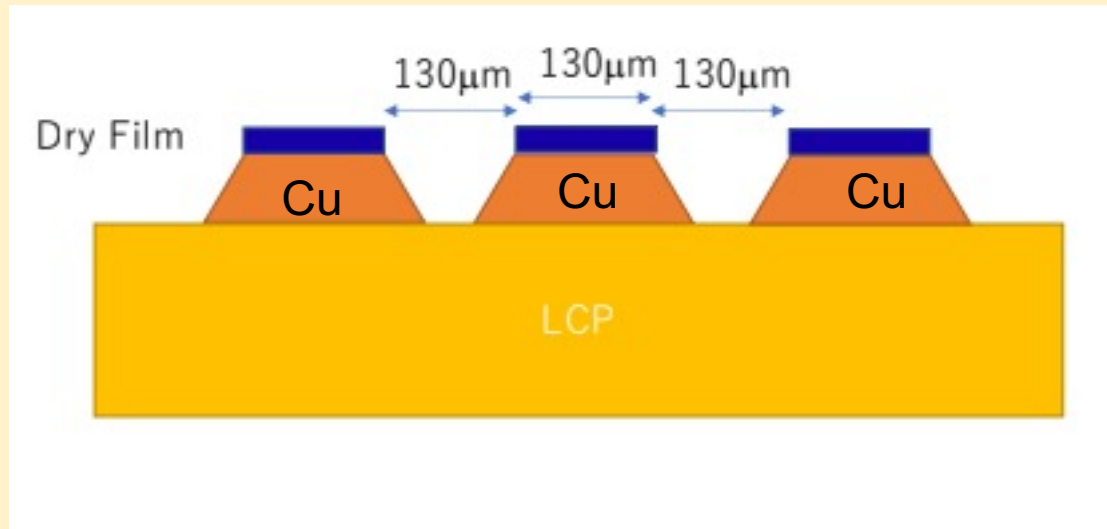
Marking



I can not find error.



I guess bunch are separated by dot which is greater than 125.



RGB

