

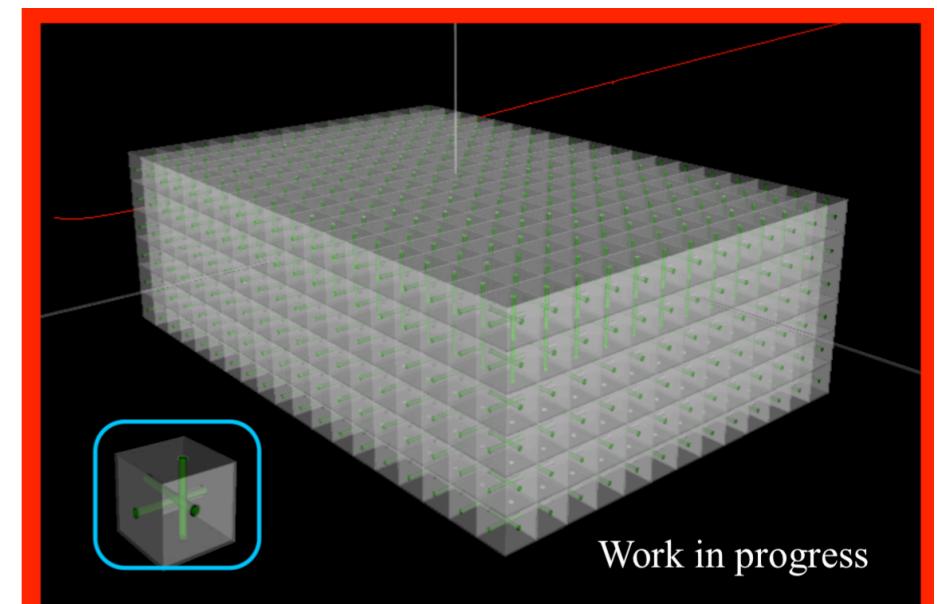
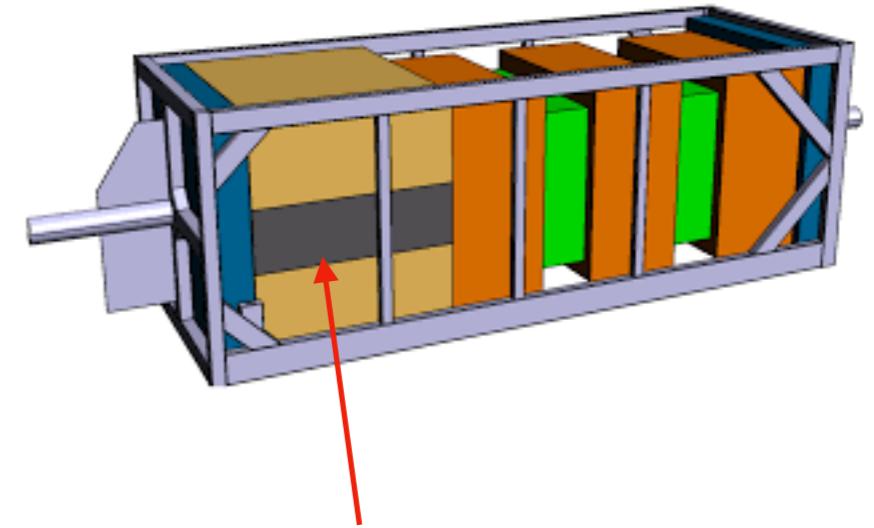
# Super-FGD に用いる シンチレータキューブの 品質チェック

2019.12.27 京都大学 谷 真央

# Super-FGD

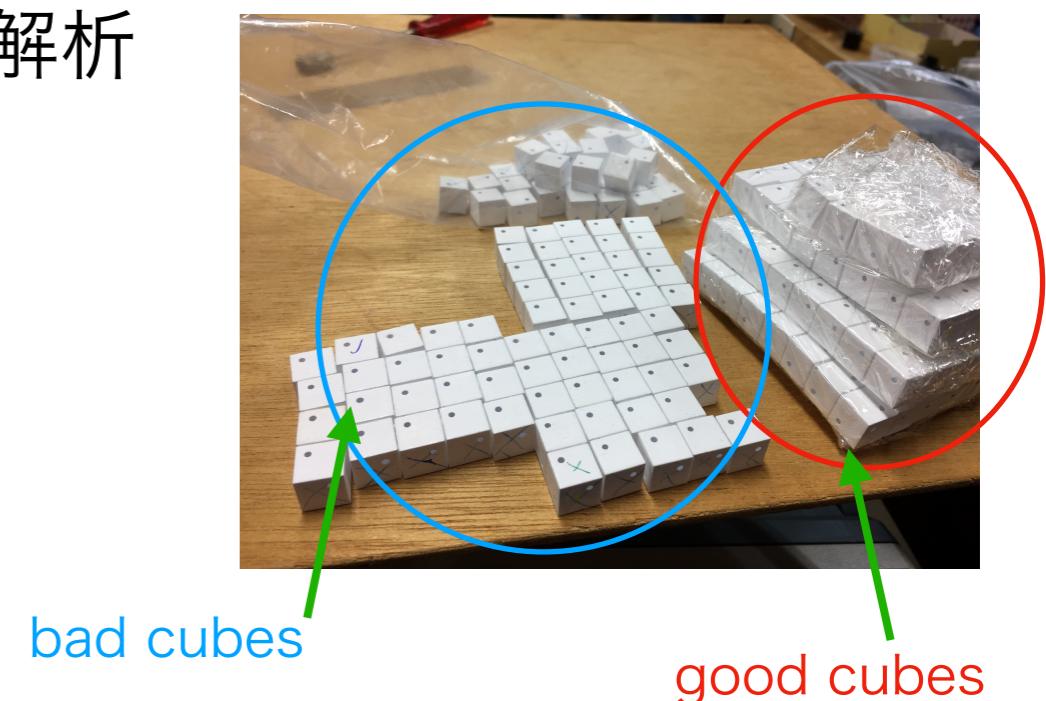
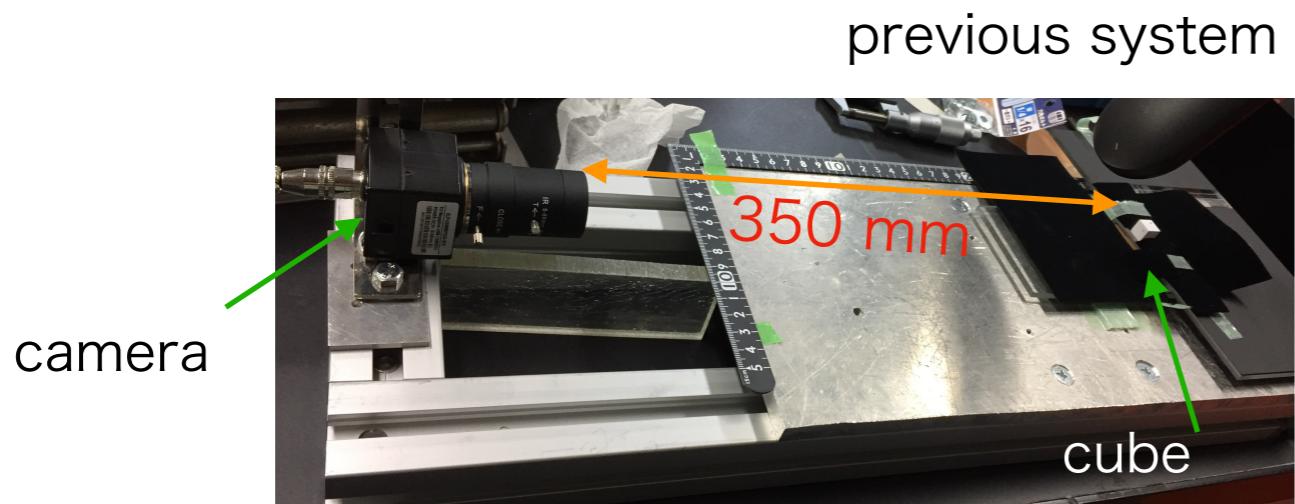
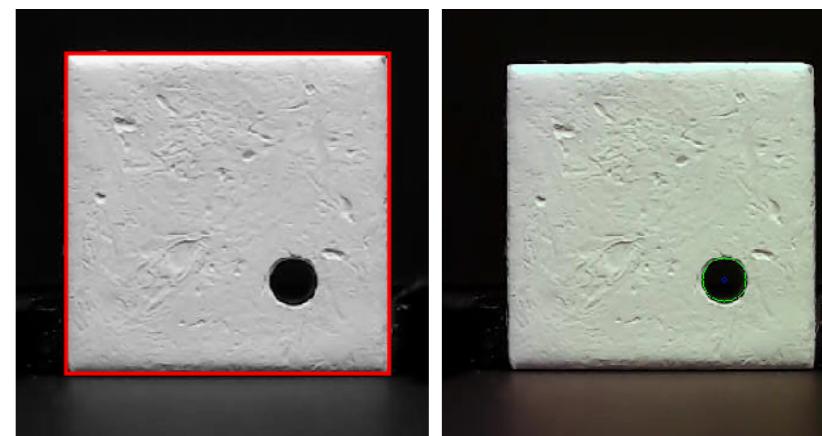
- 短い飛跡、大角度散乱粒子に対して良いアクセス
- 体積  $1.8 \times 0.6 \times 2.0 \text{ m}^3$
- $1 \times 1 \times 1 \text{ cm}^3$  のプラスチックシンチレータ・キューブを約200万個使用
- 各キューブに XYZ 方向に波長変換ファイバーを通し、MPPC で光量を測定
- 大量のキューブを 3 次元方向に並べる→事前の品質チェックが重要

$\nu$  →



# what is my work

- Super-FGD に用いるシンチレータキューブのクオリティチェック
- 大量キューブを撮影、サイズや穴の位置に関する分布をつくる
- サンプルとして、ロシアにてチェック済みの 70 good + 70 bad キューブを撮影・解析



# cube quality-check

- ロシアの方法（現行）  
[https://www.t2k.org/ndup/general/nd280-upgrade/sfgd\\_assembly/fishing-line/fishing\\_line\\_assembly\\_v0](https://www.t2k.org/ndup/general/nd280-upgrade/sfgd_assembly/fishing-line/fishing_line_assembly_v0)
- 14×14 個のキューブを正方形に並べ、金屬の棒を通す
- 金属棒が上手く通らないところがあれば、そこに bad-cube があると判断  
(例：キューブが大き過ぎて隣のキューブと干渉する、穴の位置・方向等が微妙にズれている等)
- 欠点：時間がかかる、定量的な bad-cube の判断が難しい、個人差が出る

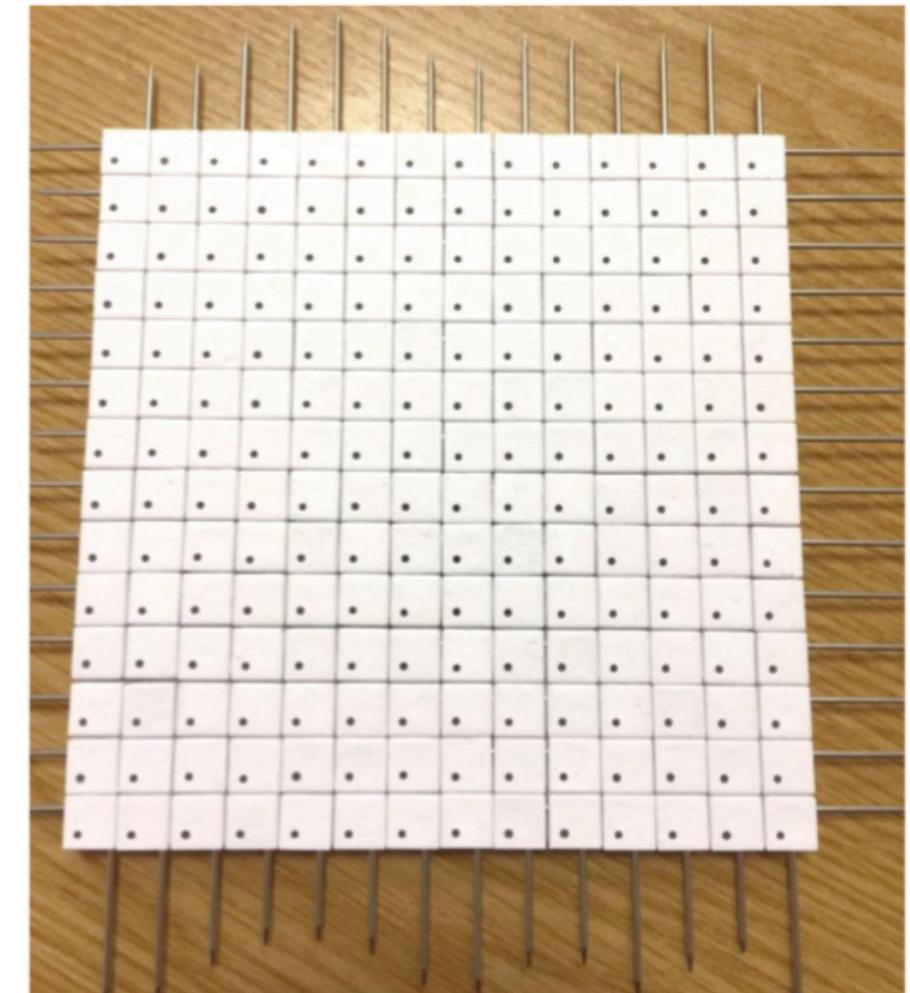


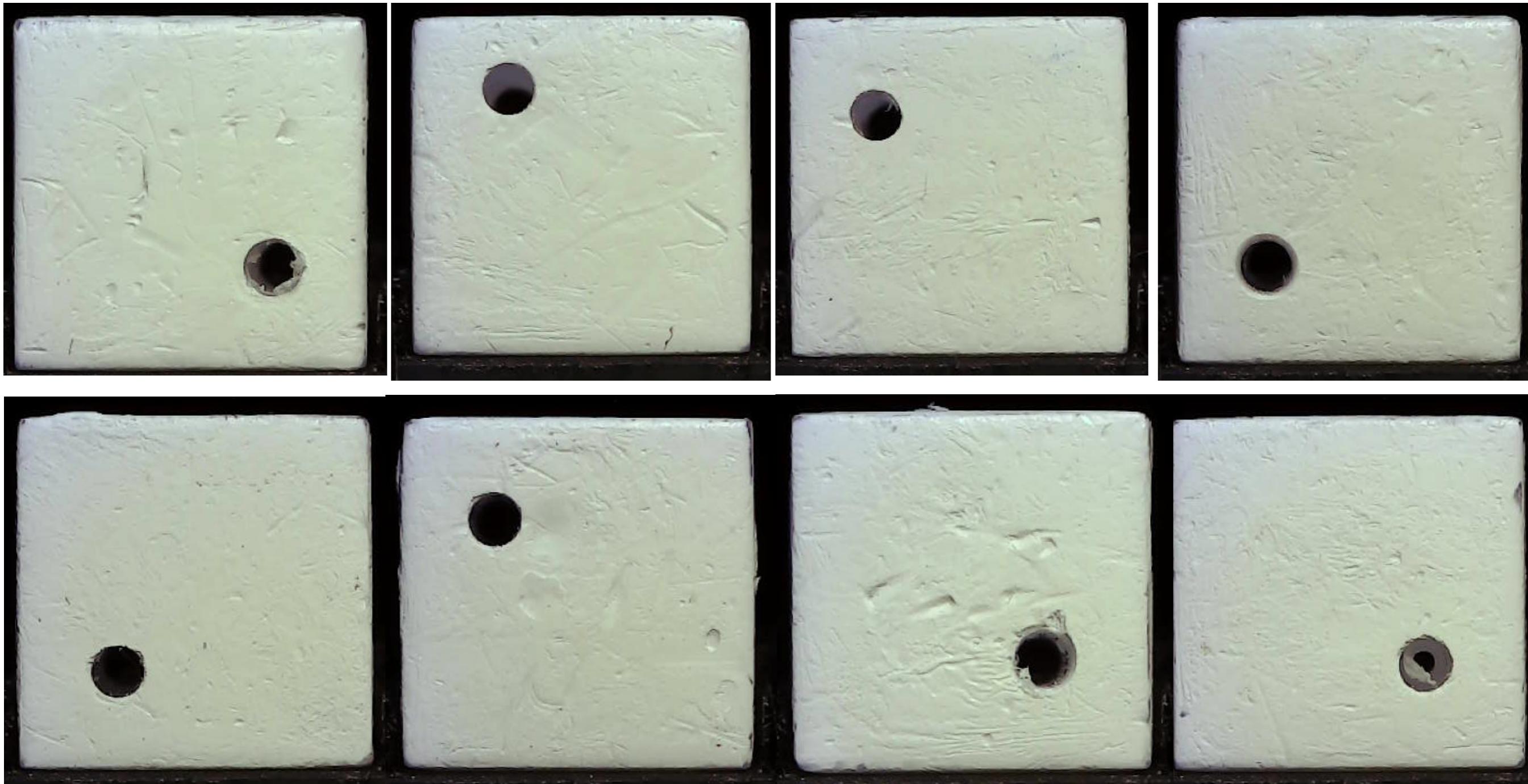
Figure 1: 196-cubes array for quality check.

この面のチェックの後、キューブを90度回転させて第3の穴についてもチェック

# good なキューブの例

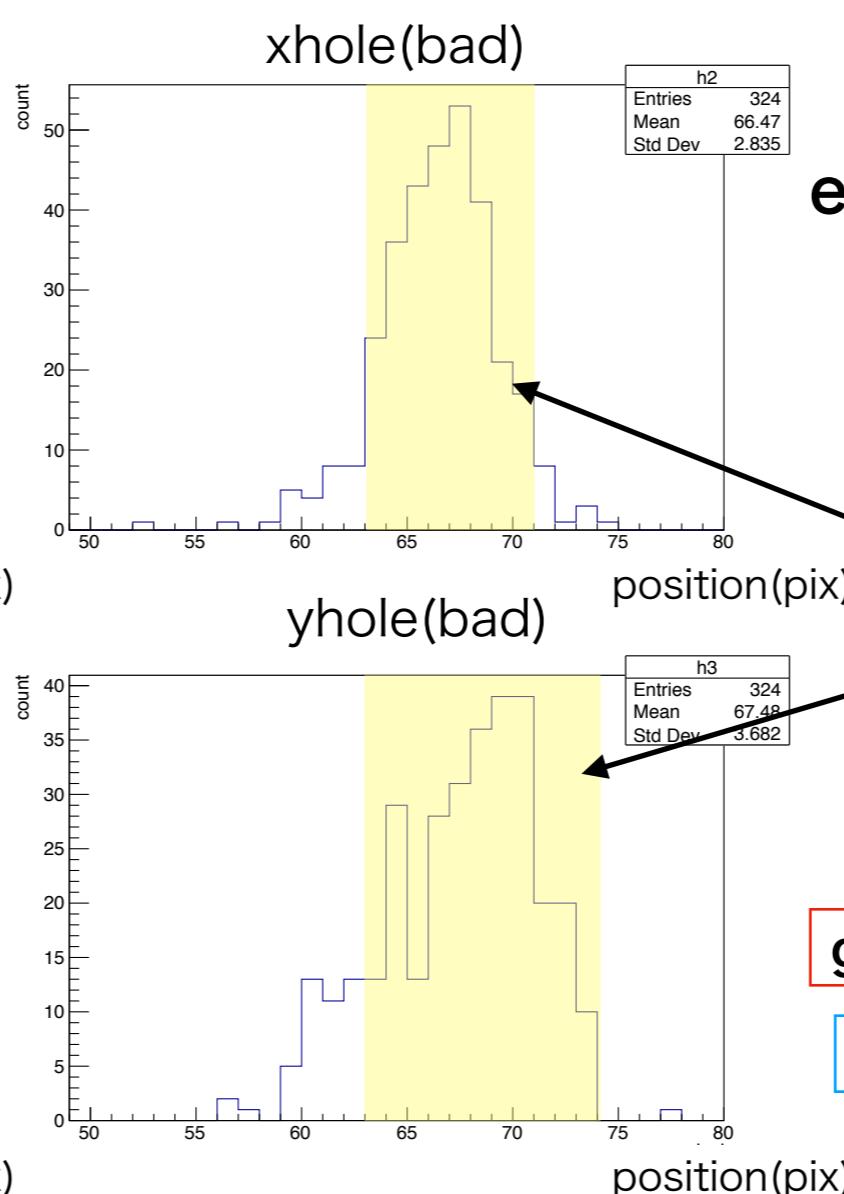
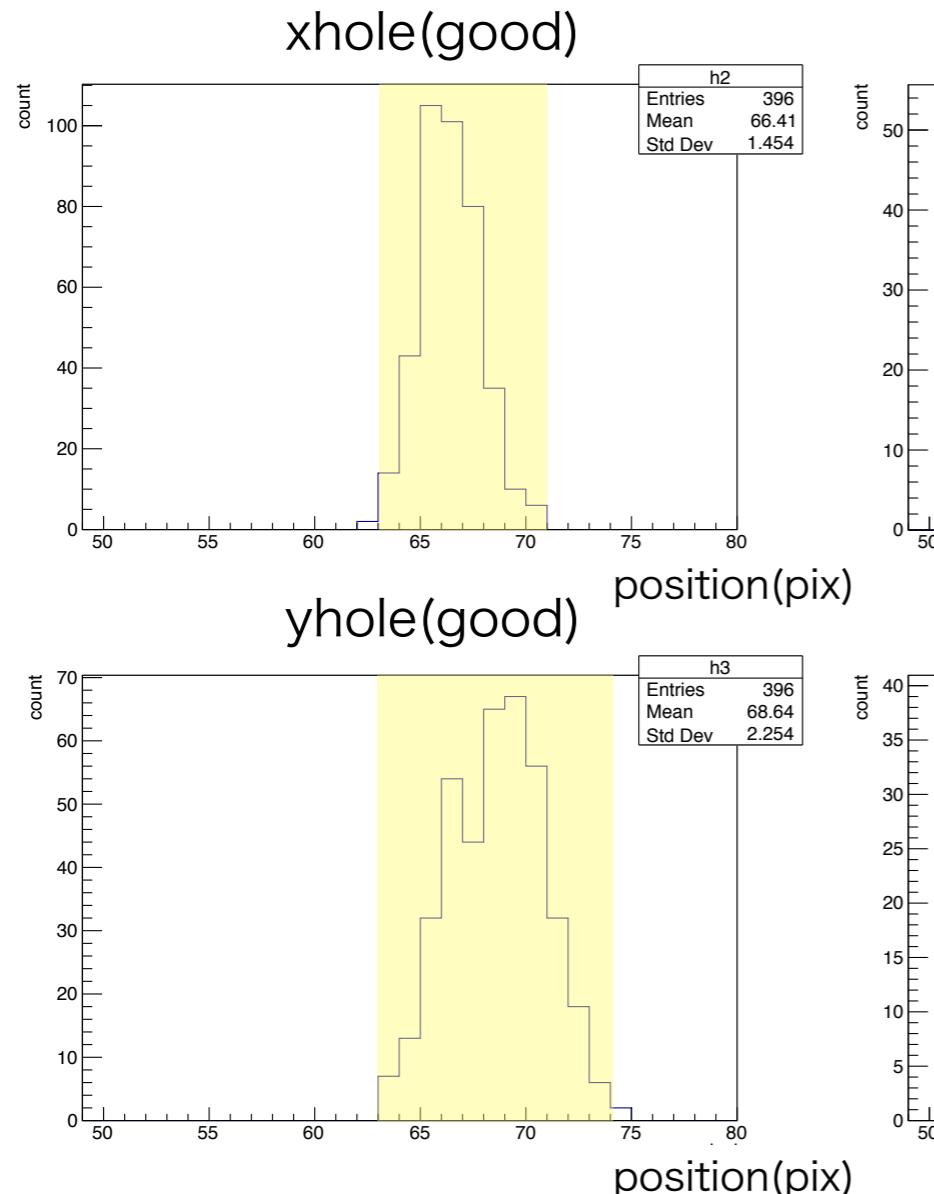


# bad なキューブの例



# analysis of good / bad cubes

- 70個ずつの good / bad キューブ解析
  - キューブサイズ、穴の位置、穴の半径についての分布作成
  - good キューブの分布を参考に bad キューブ分布にカットをかける。



ex) distributions about position of hole

decide the cut area comparing good and bad distribution.

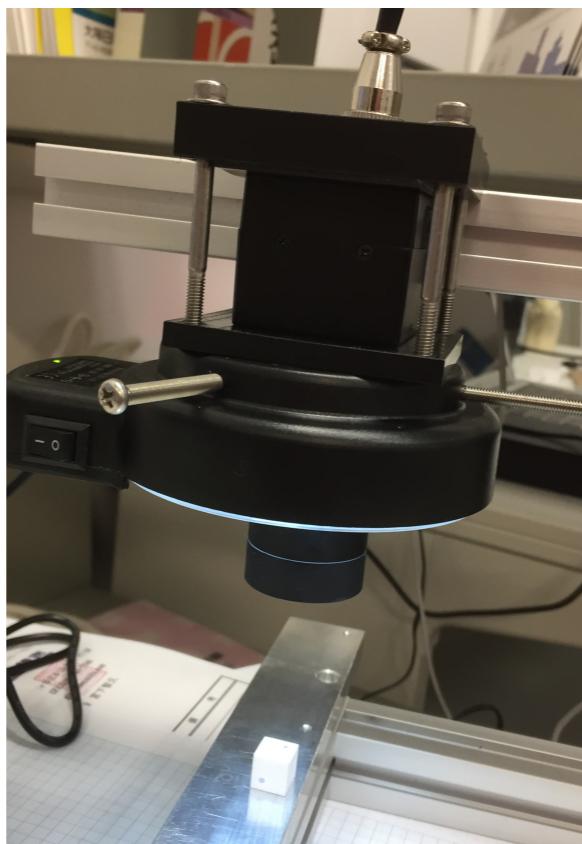
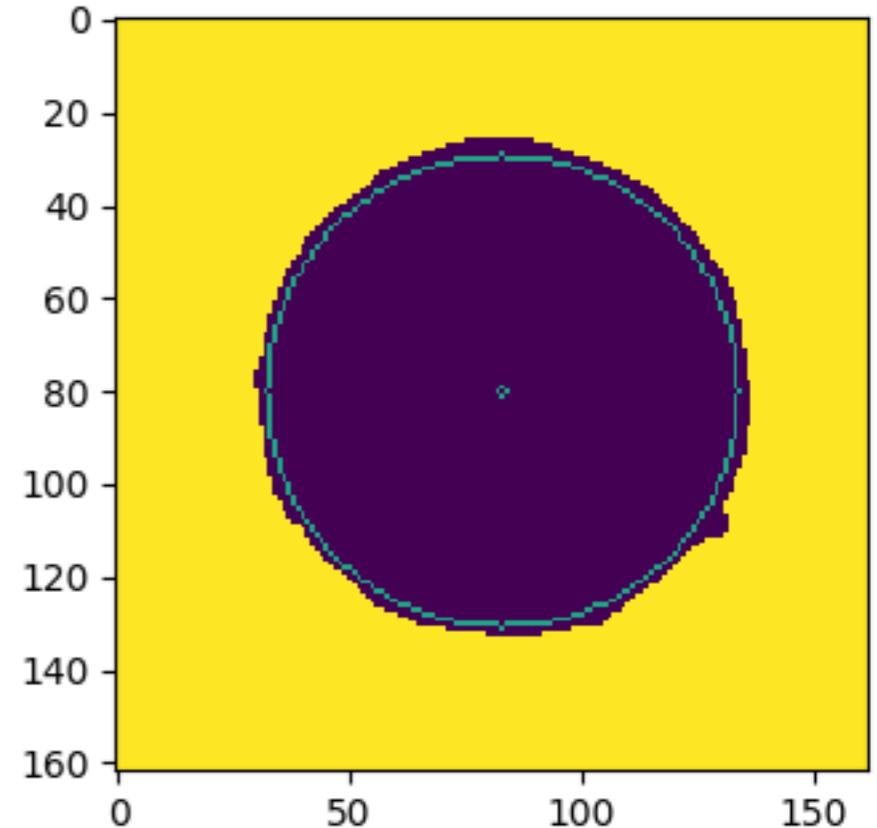
good : 70 cubes → 62 cubes

bad : 70 cubes → 15 cubes

# 直近の課題

- ・穴検出の位置精度の向上。
- ・穴周辺の画像を二値化、黒ピクセルの位置の中央値を穴の中心として検出。
- ・この穴位置を基準に円を描き、穴のエッジにフィットさせたい。

黄色：白（255）  
紫色：黒（0）に対応



- ・三次元方向から同時に撮影できるジグの制作。カメラはレコフレームに付けられるようになった。キューブの台座も作る。

# In fact ..

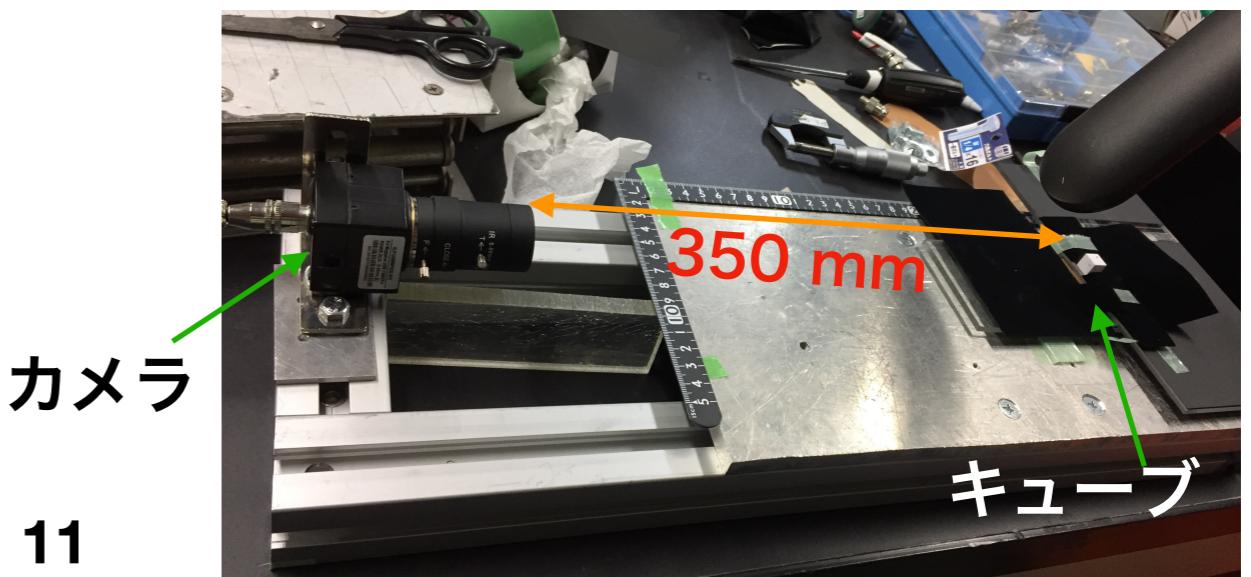
- Super-FGD の組み立てにはロシア案と日本案が存在
- 現在、ロシア案での組み立て予定
- キューブ製作・検査もロシアにて進行中
- 画像解析は第2の案として準備
- 日本でのプロトタイプ検出器制作の際に使えるか・・・？

back up

# good / bad cube の撮影

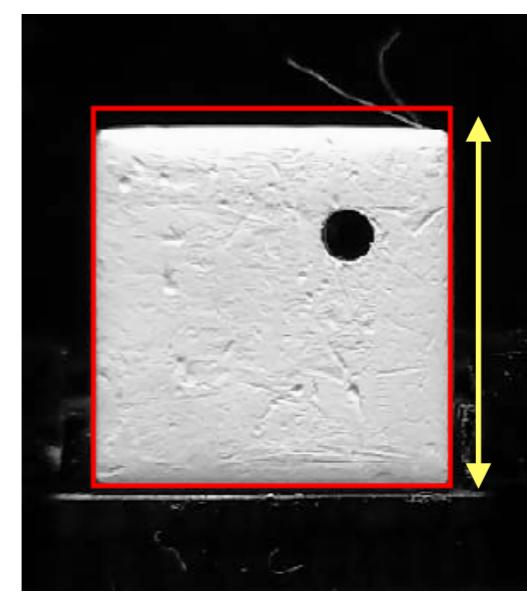
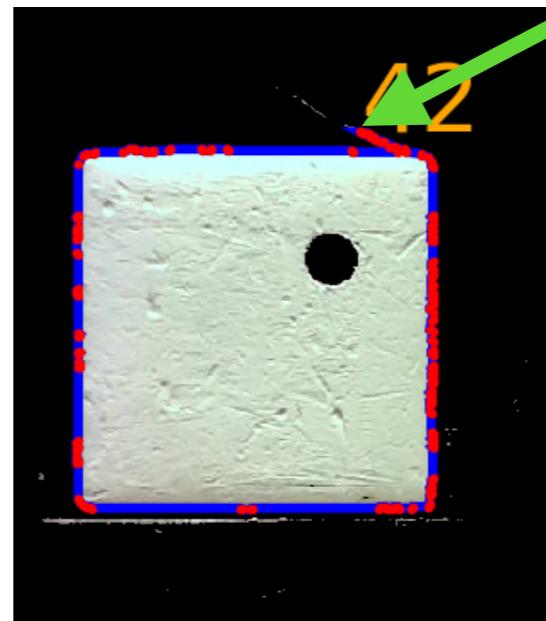
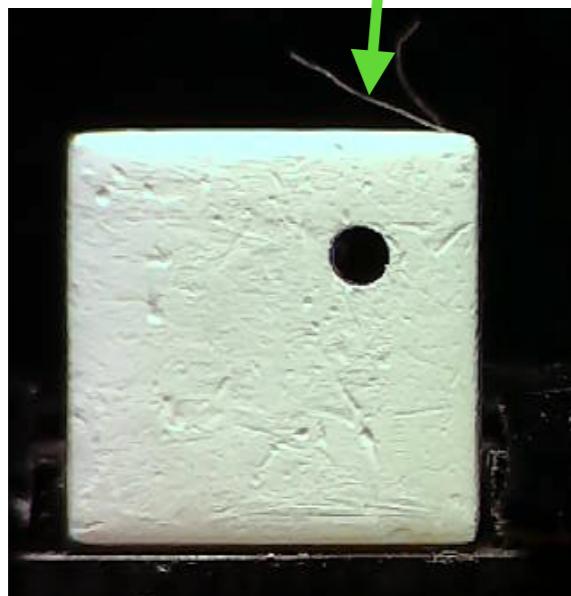
- 各 70 個の good / bad cube の撮影
  - ロシアの方法によってすでに判定されているものをサンプルとして用いる
  - 350 mm 離れた位置から焦点を穴に合わせて撮影(穴の位置は今回は右上で固定)
- ROOT による解析で good / bad の選別 (今回のデータは単位を全て pixel 数のままで扱う。)

新しい撮影台



# 注意

- 外側にゴミがついているキューブについては、今までの方法ではうまく輪郭を検出できない→コードの改善が必要。



この点をキューブの最も外側  
の点だと認識してしまう

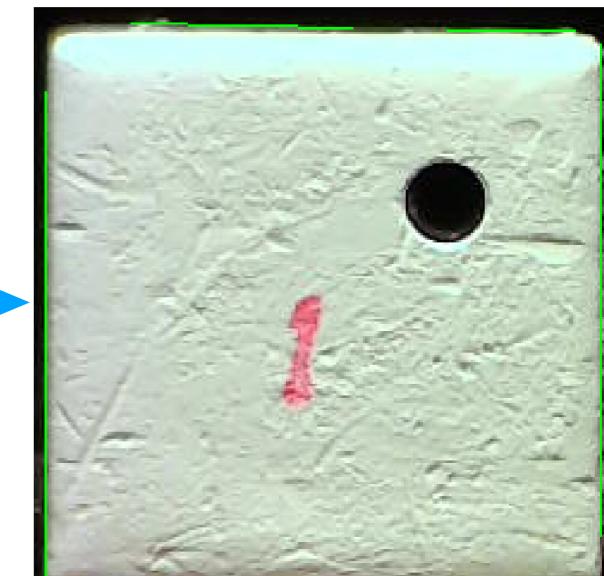
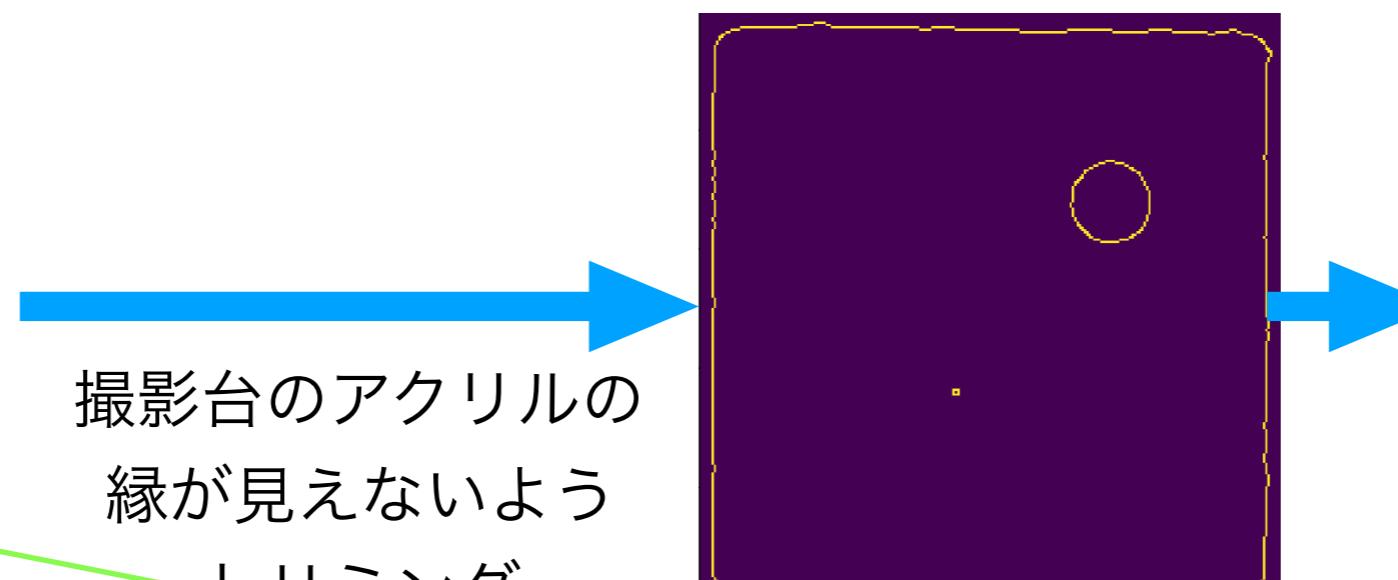
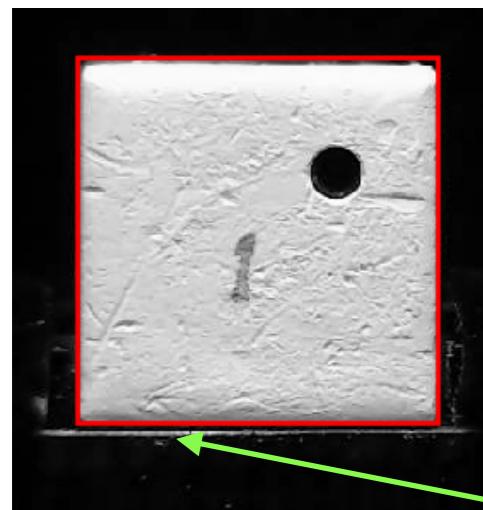
overestimate  
してしまう

今までの方法：

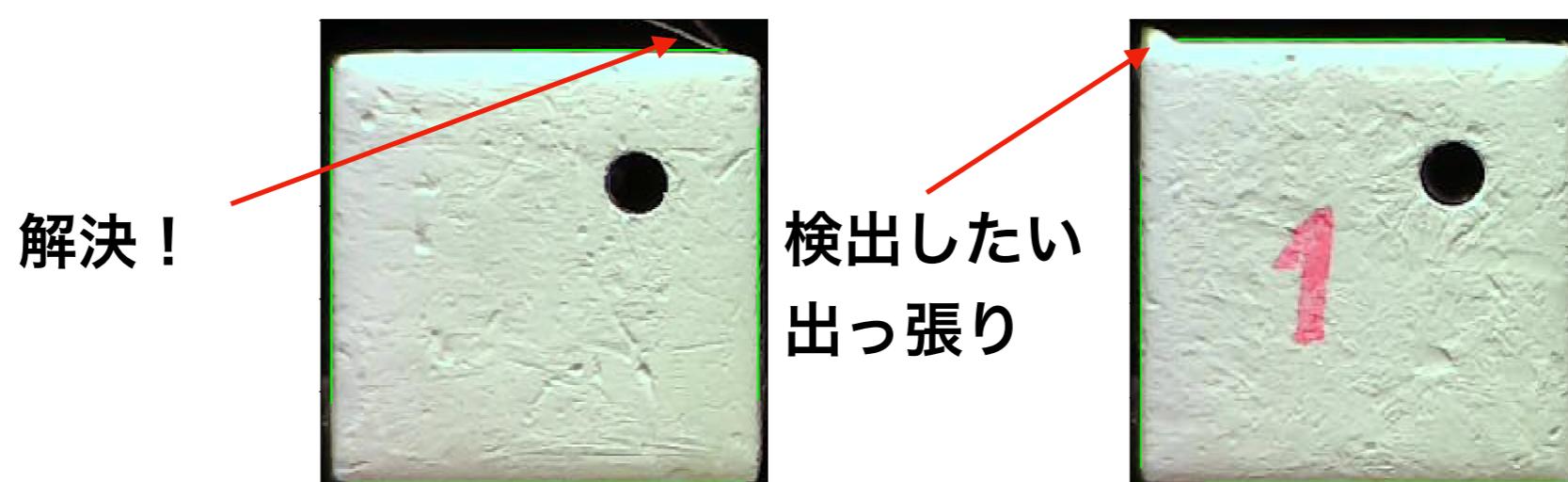
輪郭を検出、キューブの各辺のもっとも出っ  
張ったところを通るような長方形を出力

# サイズ検出の改善

- 直線を検出するメソッドを用いる。



- 前ページの問題は解決したが、今度は出っ張りが検出で  
きない。

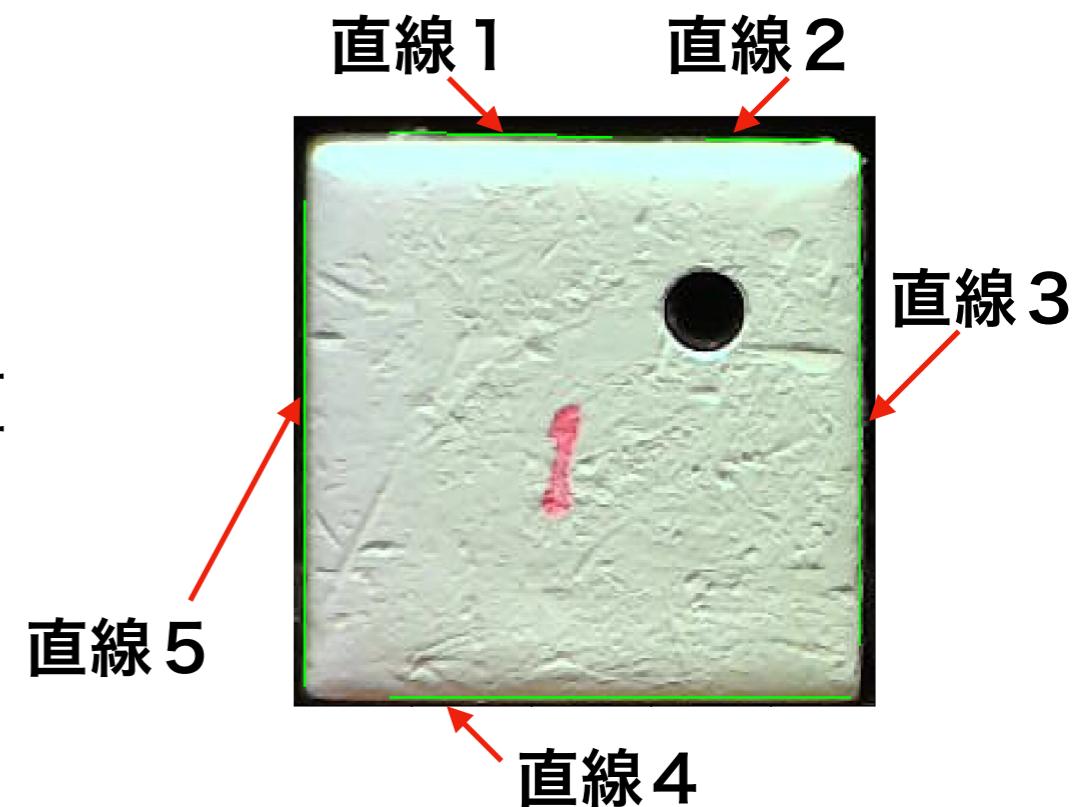


# good / bad 選別の方針

- good の分布と見比べて、badから
  - ① 検出サイズが分布から外れているものをはじく。
  - ② 穴が傾いているものをはじく。
  - ③ ①ではじけなかったもののうち、穴の位置がずれているもののをはじく。
  - ④ 穴の大きさが分布から外れているものをはじく。
- good / bad キューブを排除する割合を調べる

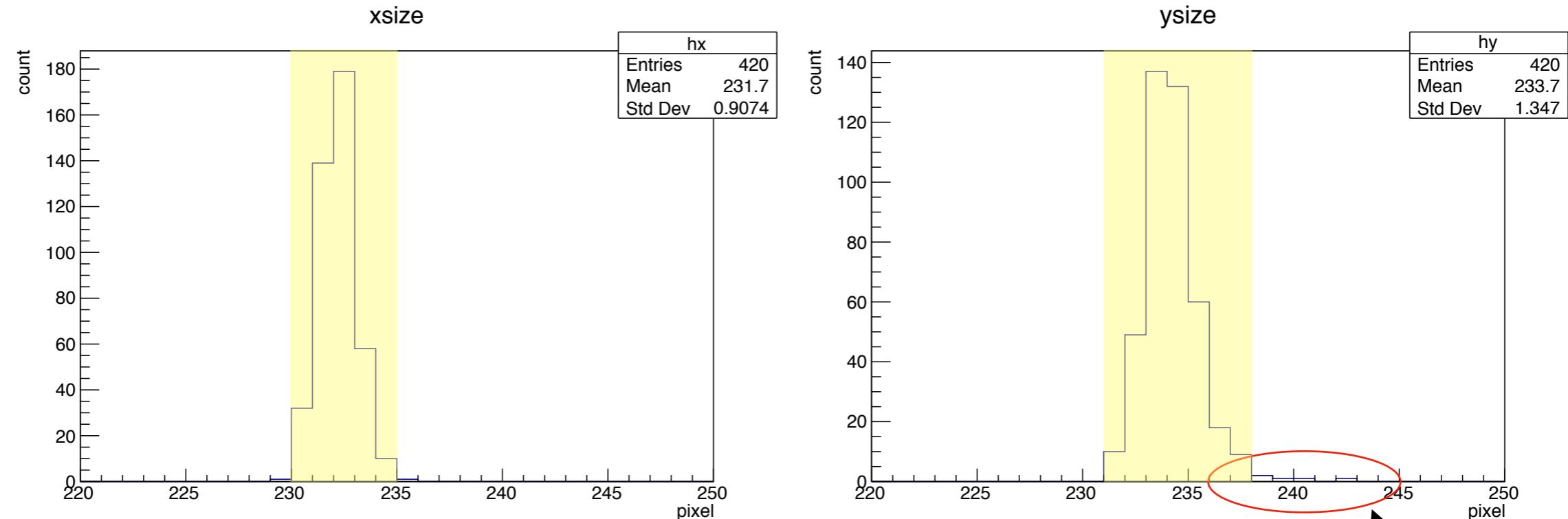
# 検出サイズの計算

- 各検出直線の始点( $x_1, y_1$ )・終点( $x_2, y_2$ )を得る
- 各直線上の中心の点  $(\bar{x}_i, \bar{y}_i)$  を計算
- $xsize, ysize$  をそれぞれ  
 $\bar{x}_{max} - \bar{x}_{min}, \bar{y}_{max} - \bar{y}_{min}$  で計算



# 検出サイズの分布 (good)

- good キューブのサイズ分布を見る。



→pixel 数

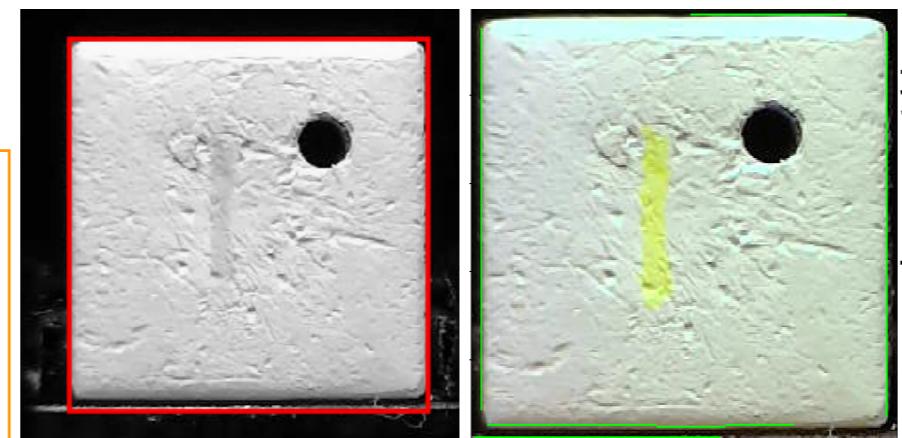
$230 \leq xsize \leq 234$

$231 \leq ysize \leq 237$

でカットをかける。

ちなみに、good キューブのサイズ~10.15 mm (前回) を用いると、  
約 0.044 mm/pixel

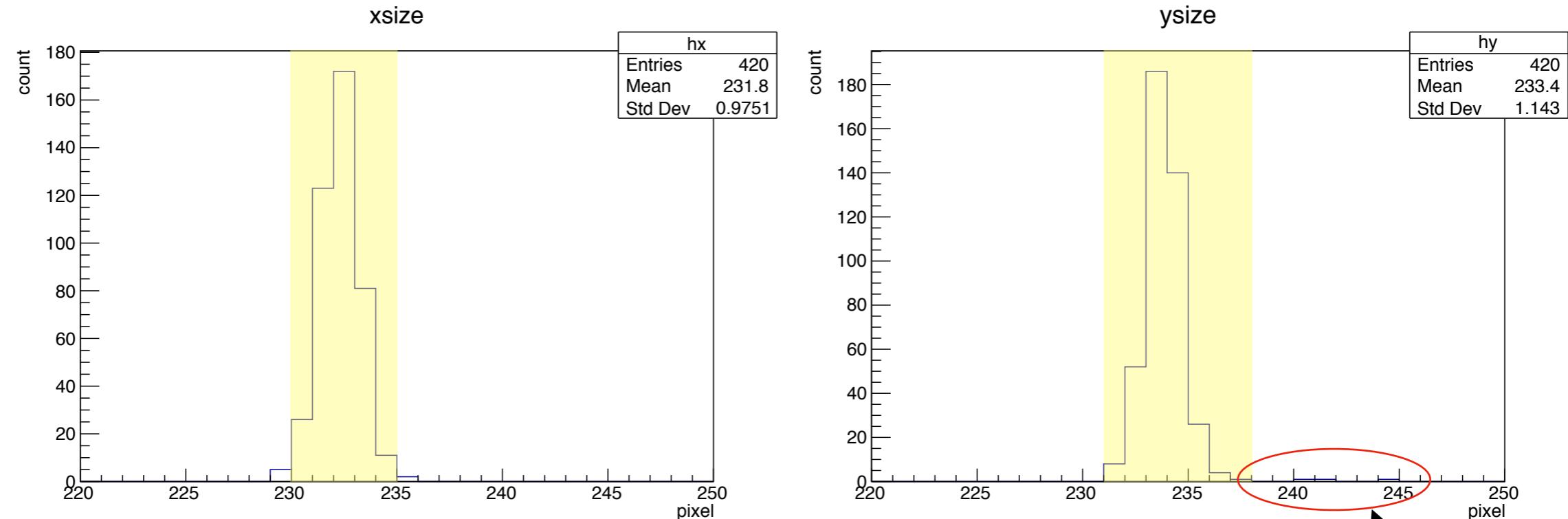
good : 70 cubes → **66 cubes**



撮影台のアクリルが見えないよう、トリミングしたが、そもそも輪郭を誤検出している場合、正しい結果が得られない→改善の余地あり

# 検出サイズの分布 (bad)

- bad キューブをサイズで選別する



good の結果：

$230 \leq xsize \leq 234$

$231 \leq ysize \leq 237$

でカットをかける。

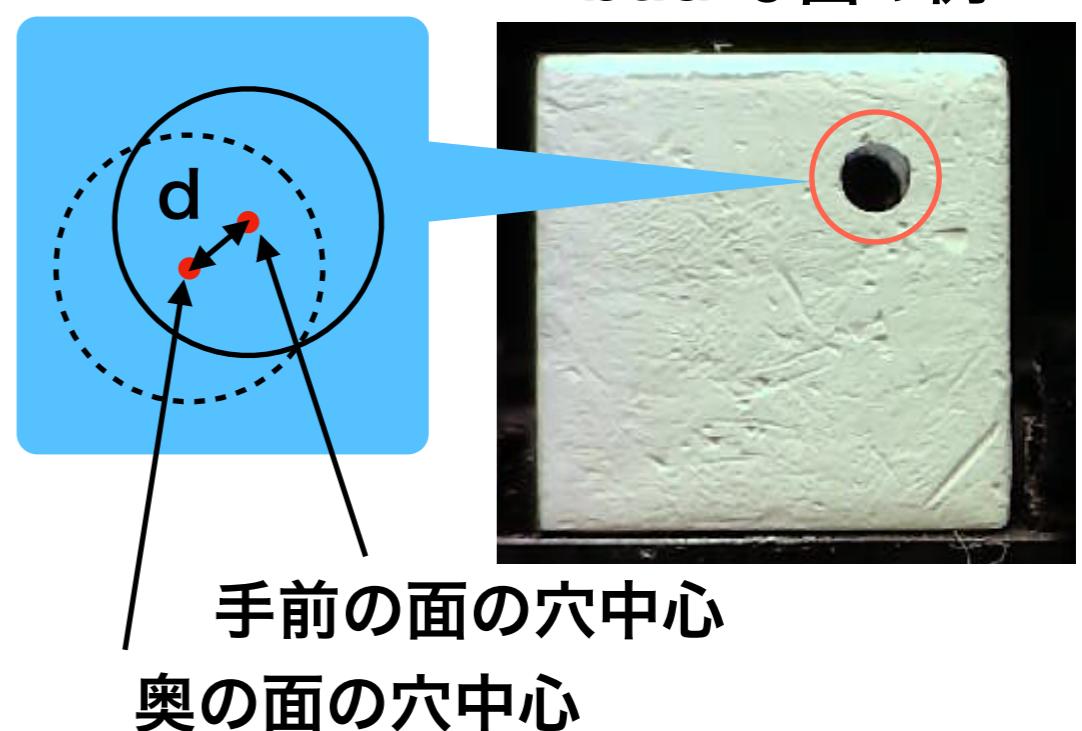
6面の各サイズについて、どれか1つでもカットがかかれればそのキューブをはじく。

bad に関しても誤検出が

**bad : 70 cubes  $\rightarrow$  63 cubes** いくつか見られる

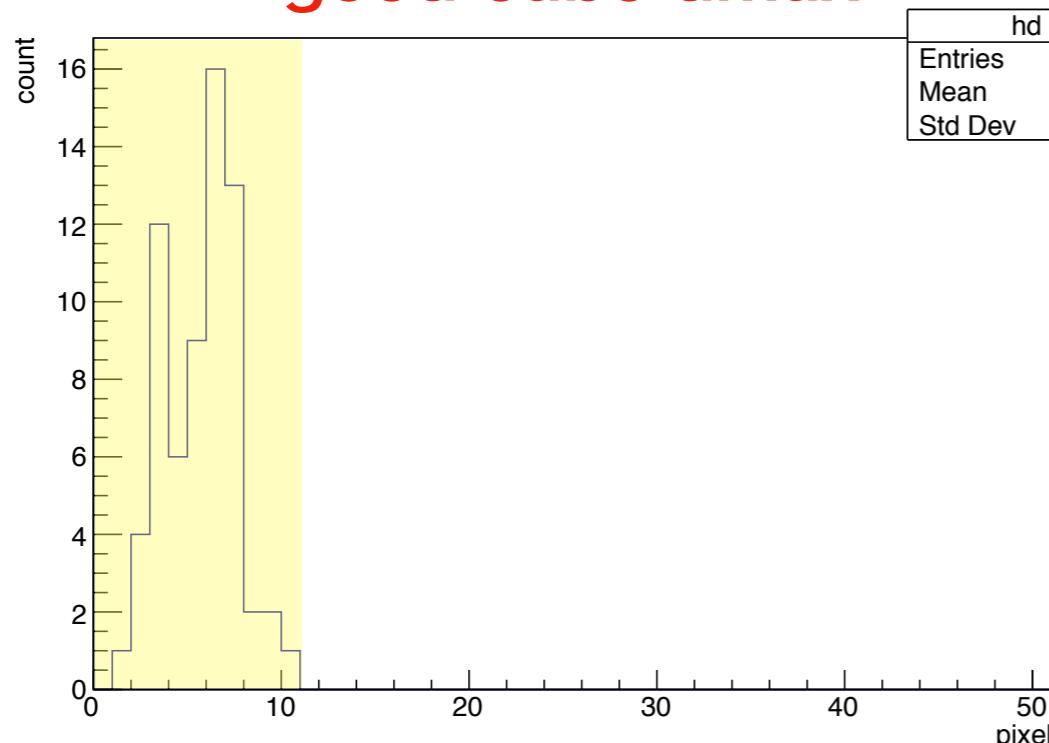
# 穴の傾きについて

- 穴が傾いている場合、ファイバーが通りにくい→ bad と判定したい
- 各穴について、 $d = \sqrt{dx^2 + dy^2}$  を定義  
( $dx$ 、 $dy$  : x、y方向の穴の位置のずれ)
- 3つの穴の  $d$  のうち、最大のものを  $d_{max}$  と定義
- good、bad について  $d_{max}$  の分布を比較する。



# 穴の傾きの分布

good cube dmax

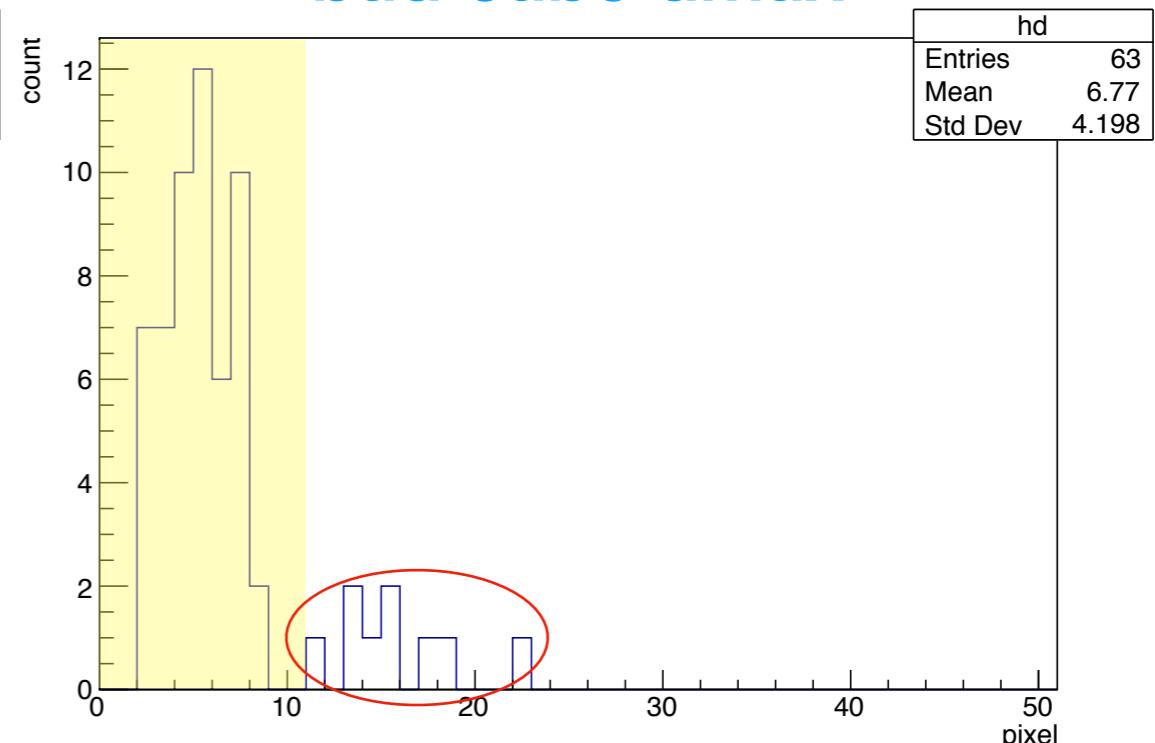


good の分布を見て、  
pixel 数

$dmax \leq 11$

でカットをかける

bad cube dmax



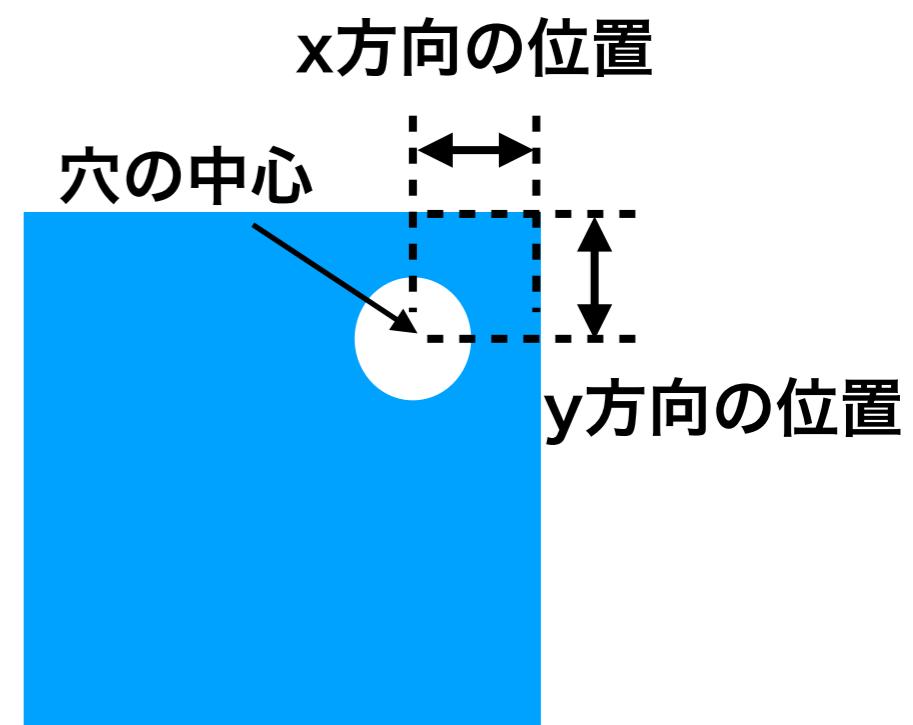
$dmax$  が大きいものをはじく

good : 66 cubes  $\rightarrow$  66 cubes

bad : 63 cubes  $\rightarrow$  54 cubes

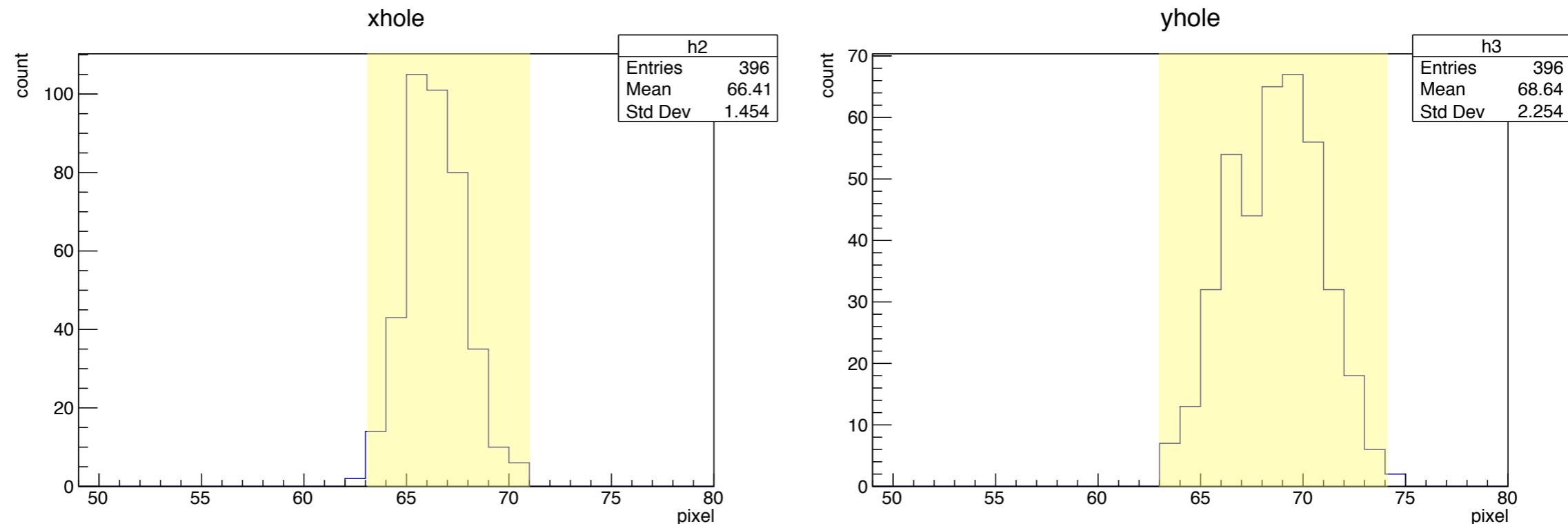
# 穴の位置について

- 穴が傾いていない場合でも、そもそも位置がずれていればファイバーは通りにくい
- good、bad について穴の位置の分布を比較する。



# 穴の位置分布 (good)

- good キューブの穴の位置分布を見る



good の結果：

$63 \leq x\text{hole} \leq 70$

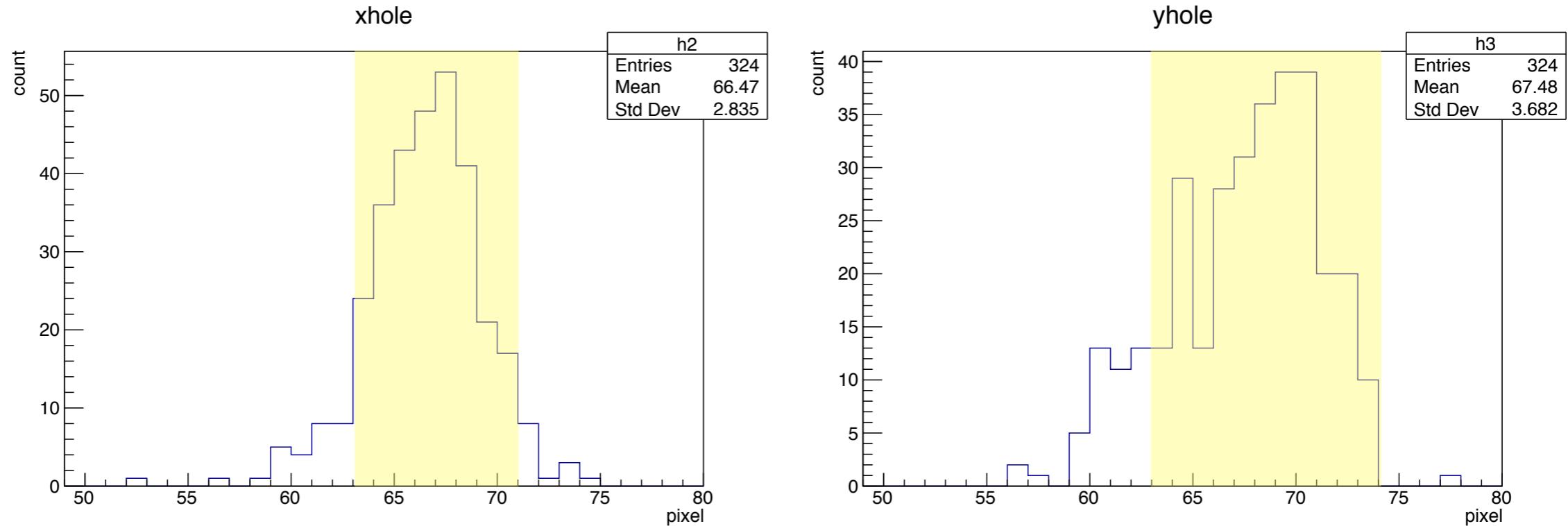
$63 \leq y\text{hole} \leq 74$

でカットをかける。

good : 66 cubes  $\rightarrow$  62 cubes

# 穴の位置分布 (bad)

- bad キューブを穴の位置分布から選別する



good の結果：

$63 \leq xhole \leq 70$

$63 \leq yhole \leq 74$

でカットをかける。

6面の xhole, yhole について、どれか1つでもカットがかかればそのキューブをはじく。

bad : 54 cubes  $\rightarrow$  19 cubes

# 穴の大きさについて

- ・ 穴はドリルで開けているので、本来ならばドリル径（あるいはそれより大きめ）の穴が空いているはず
- ・ 検出した穴の半径が小さ過ぎる、大き過ぎる場合、穴の状態が悪く、正しく検出できていないと仮定

よい穴の例：若干小さく検出している  
ように見えるが、許容範囲とみなす

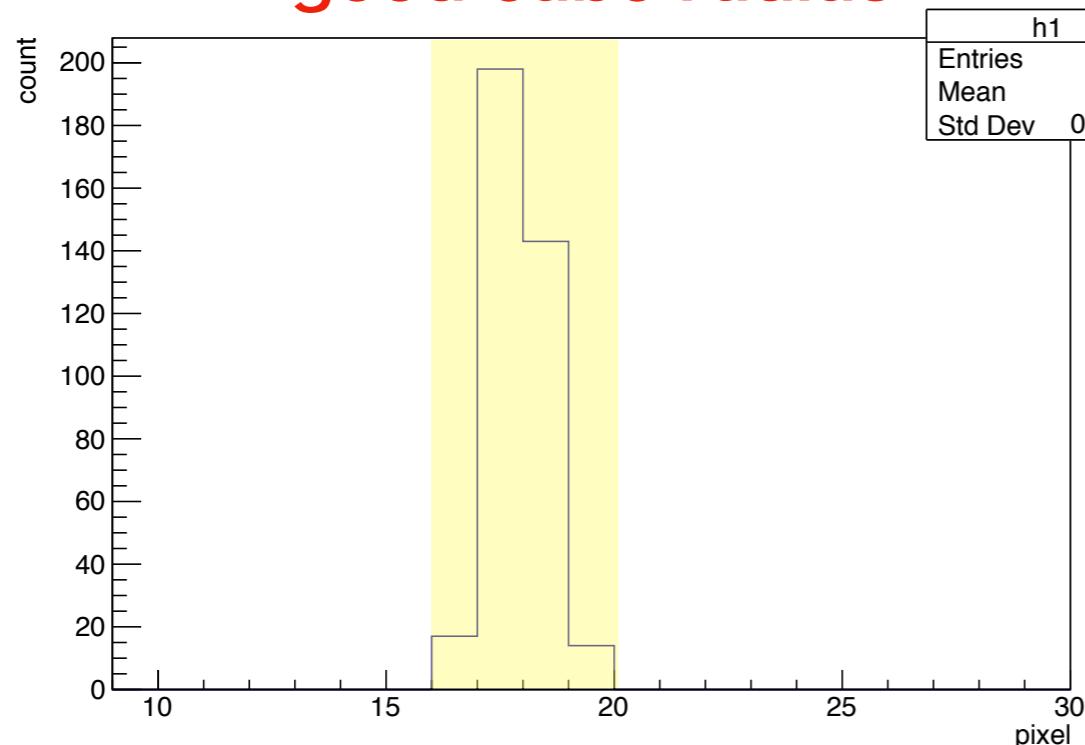


検出半径の大き過ぎる例

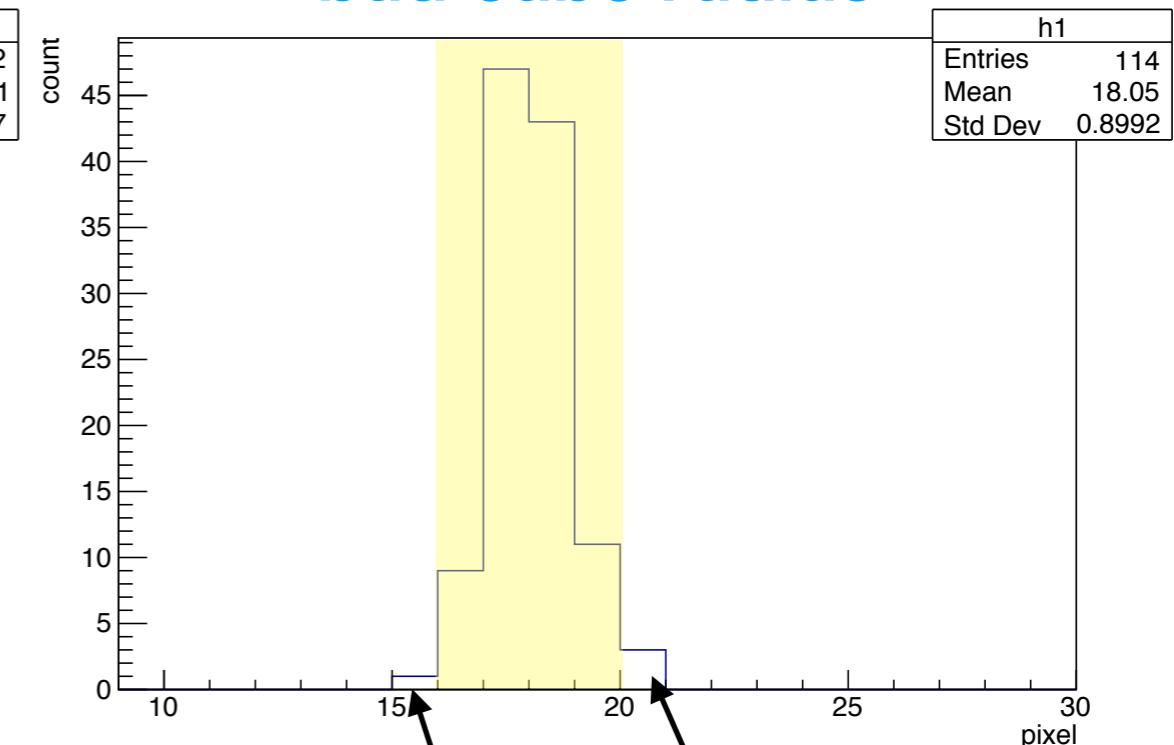


# 穴の大きさの分布

good cube radius



bad cube radius



good の分布を見て、

pixel 数

$16 \leq \text{radius} \leq 11$

でカットをかける

good : 62 cubes  $\rightarrow$  62 cubes

bad : 19 cubes  $\rightarrow$  15 cubes

# ここまでの中のまとめ

- 以上の選別で、

good : 70 cubes → 62 cubes

bad : 70 cubes → 15 cubes

となった。

- bad を排除できた割合は  $\frac{70 - 15}{70} = 78.6\%$

- good を誤って排除してしまった割合は  $\frac{70 - 62}{70} = 11.4\%$

- もう少し条件を追加して、精度を高めたい。

# 今後の方針

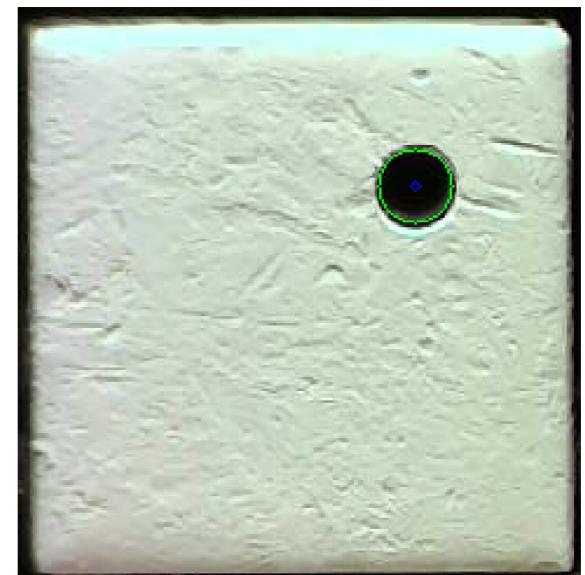
- 穴の内部のピクセルの明るさのデータを用いたキューブの選別ができるかもしれない。

(図参照)

- 今回は焦点距離 350 mm 、レンズ軸を穴に合わせた撮影

→今後レンズ軸をキューブ中心に合わせた撮影・解析を試す。

良い例：理想的な穴では、内部には黒の背景のみが映る



悪い例：穴の状態や傾きによって、穴の内部に白い部分が見られる。



# scintillator cubes

# image analysis

2019.12.10 Mao Tani

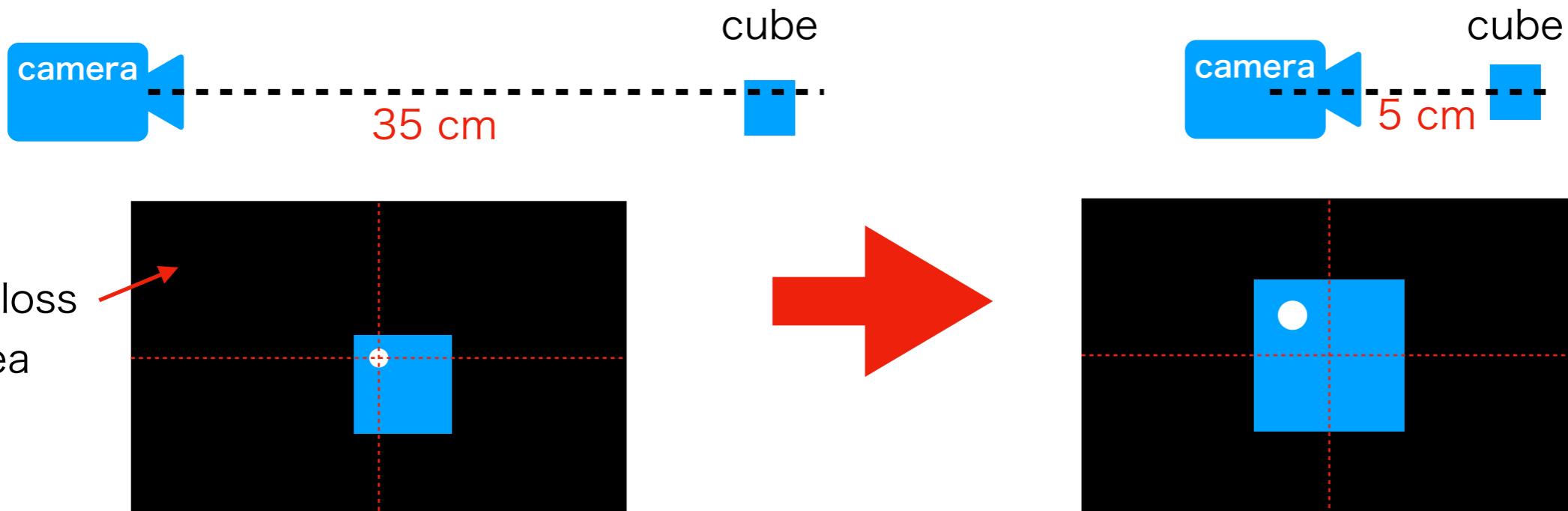
# Today's topic

- Improvement of photographic system
  - change the geometry, focal distance
  - Introduce extender (rear converter)
  - Introduce LED ring light
- Improvement of analysis codes
- Plan of three-dimension photographic system

# Improvement of photographic system

# focal distance

- To reduce the influence of parallax (視差) inside of the hole, images have been taken at long range (~35 cm).
- However, in this manner there are much pixel loss.
- Changed the focal range ( $35\text{ cm} \rightarrow 5\text{ cm}$ ).
- We were focused on the center of hole (left) before, now the lens axis is set to the center of cube surface (right)

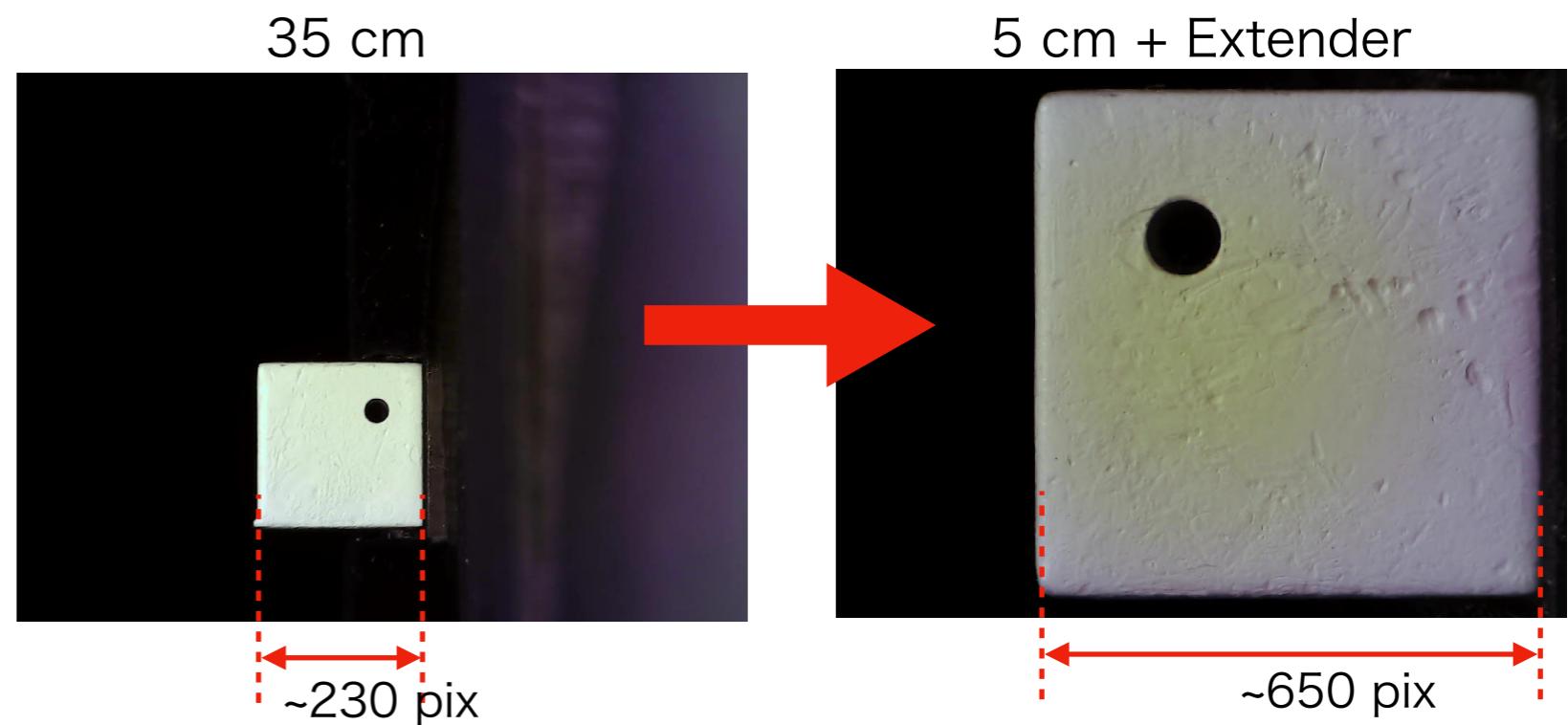
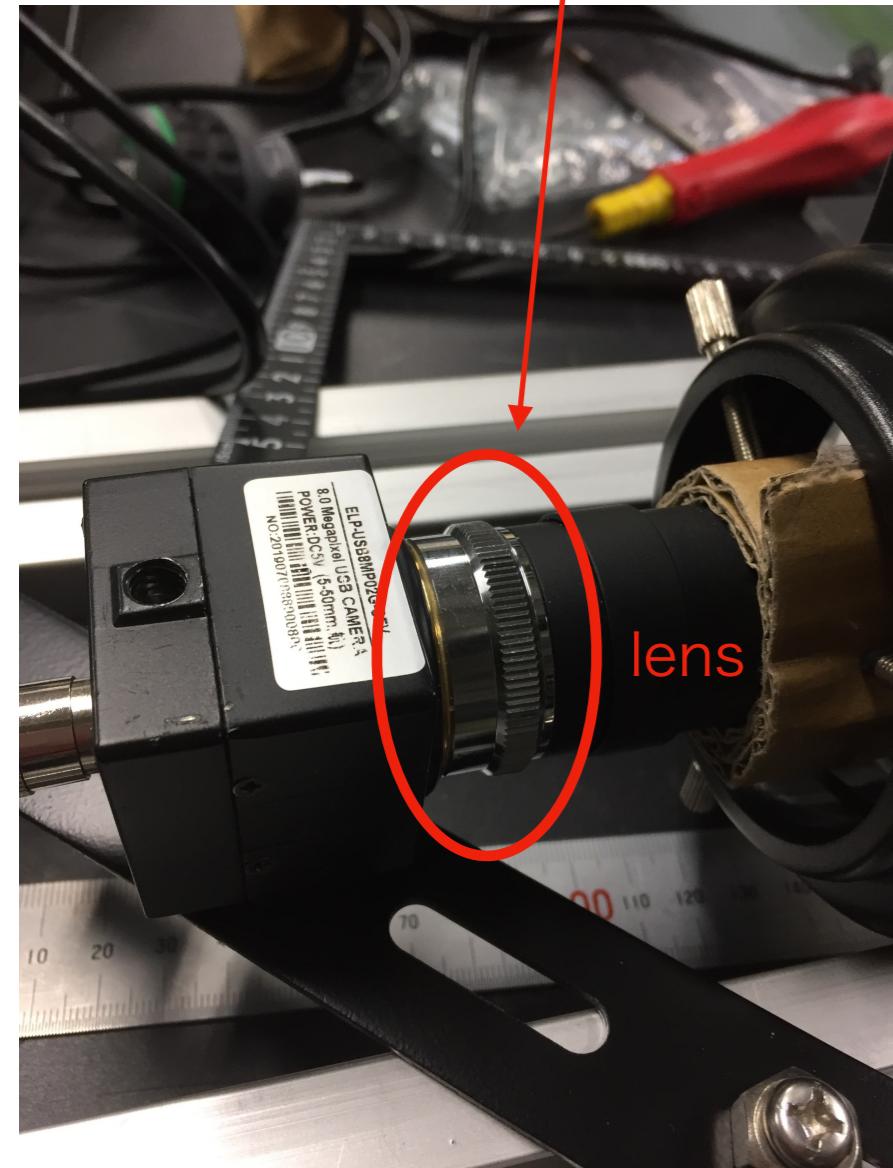


# Extender (rear converter)

- By using extender, the image of cube got more enlarged.

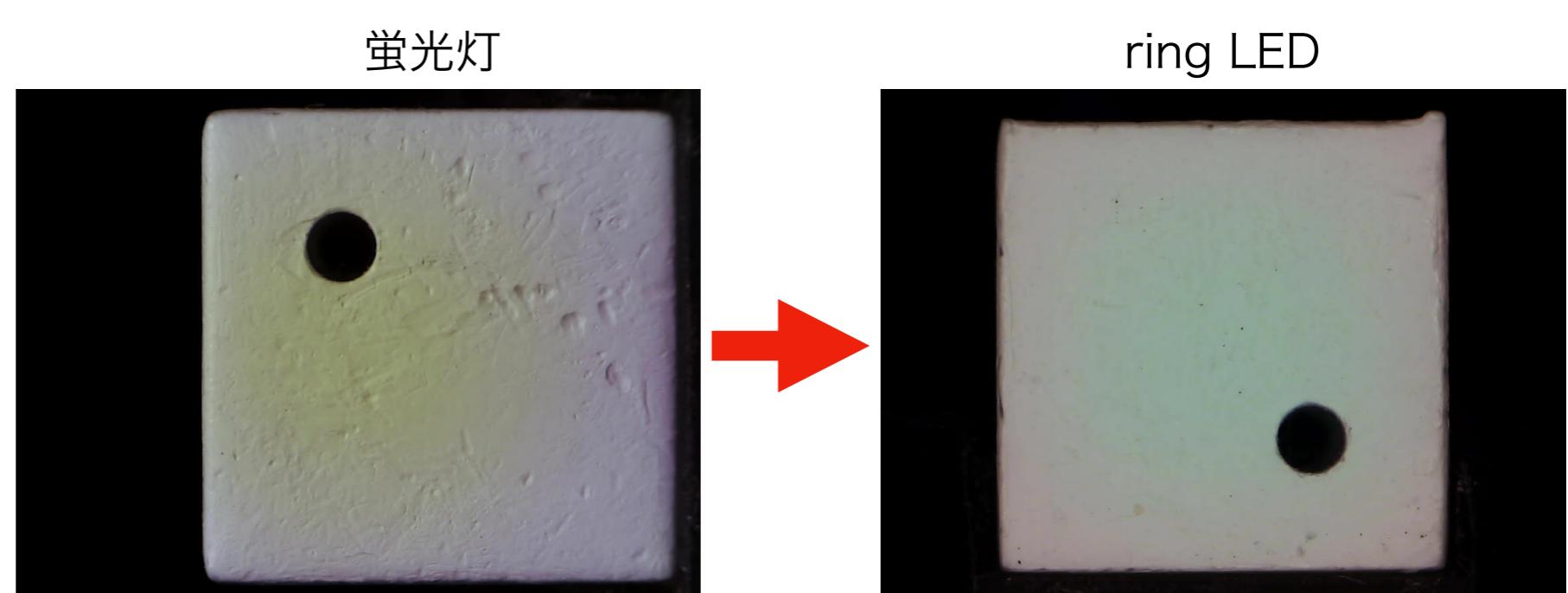
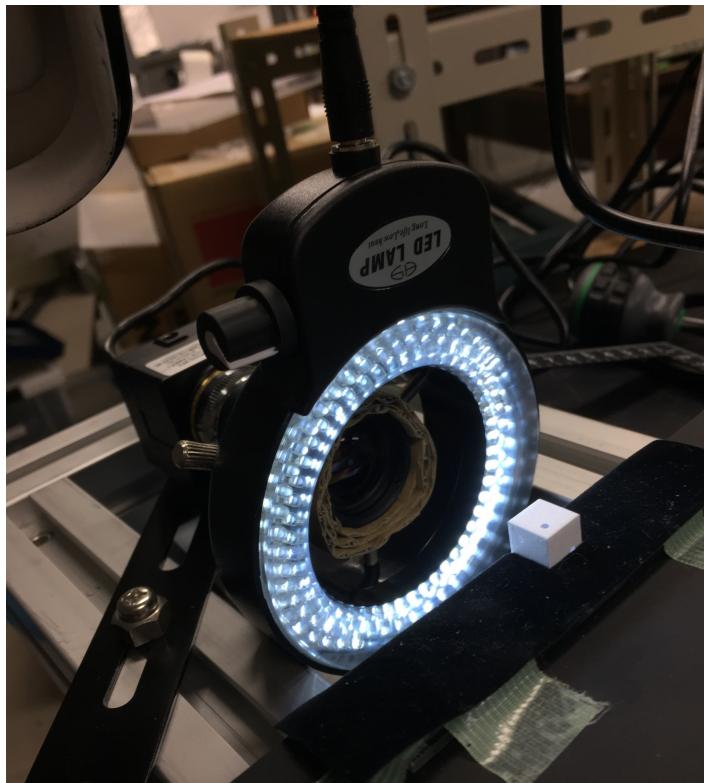


X2 TV EXTENDER JAPAN



# ring light

- As a light source, a desk lamp was used before, but now we started to use LED ring light to shine the whole surface equally.
- By using ring light, bumps on the surface got unnoticeable, which stood out with the desk lamp. Similarly, contours got independent of the direction of light.



# Improvement of analysis codes

# Improvement of accuracy of hole detection

- Previously we have detected holes automatically with the function ‘Hough Circle’ in openCV that is a library of python : (0 or 255)

1. binarization (0 or 255)

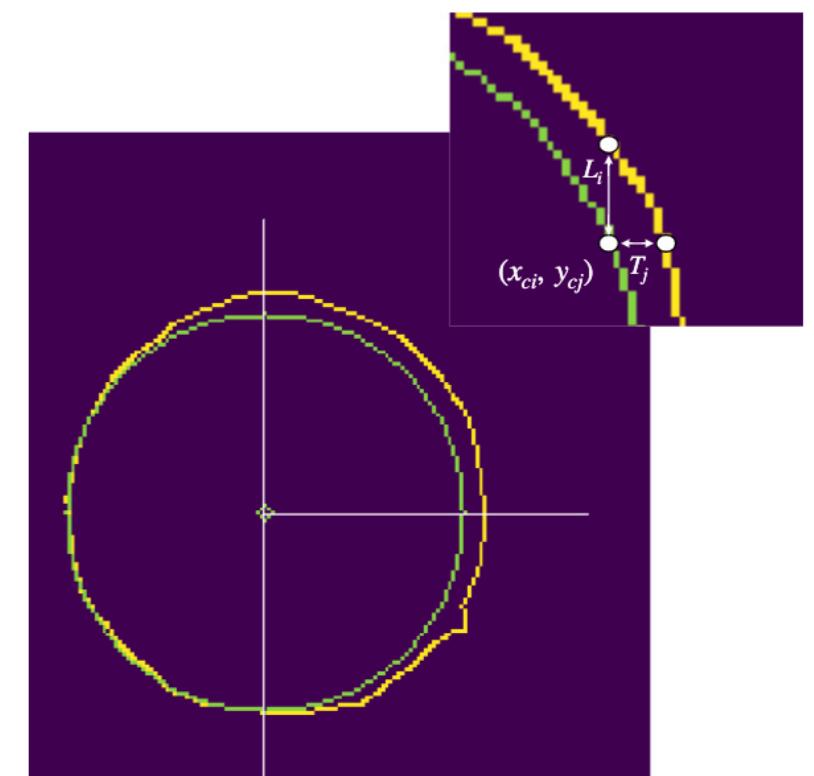
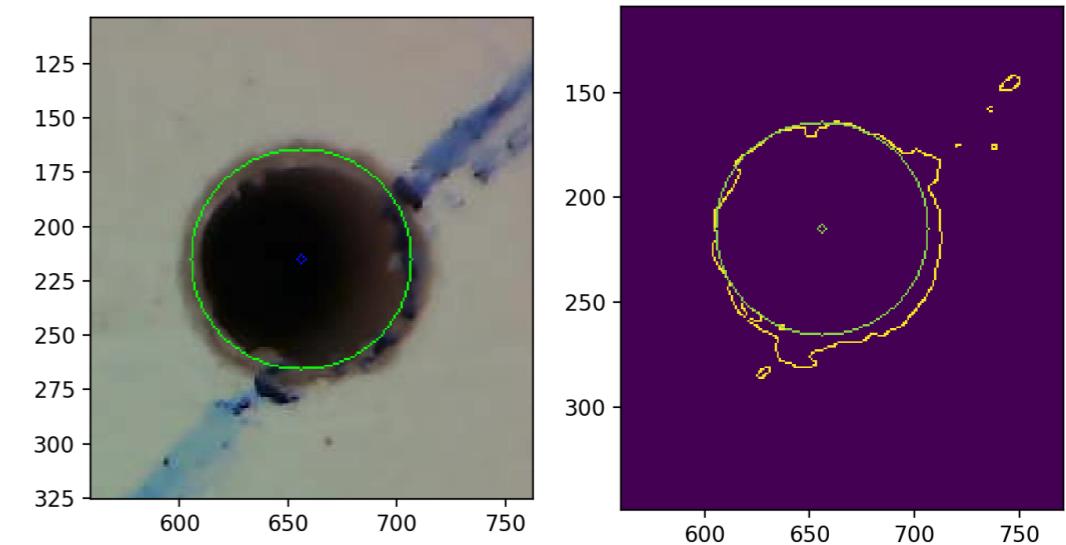
need to decide threshold value (~100)

2. edge detection

3. automatic circle detection

- Improve the hole-detection accuracy with certain correction.

Hough Circles



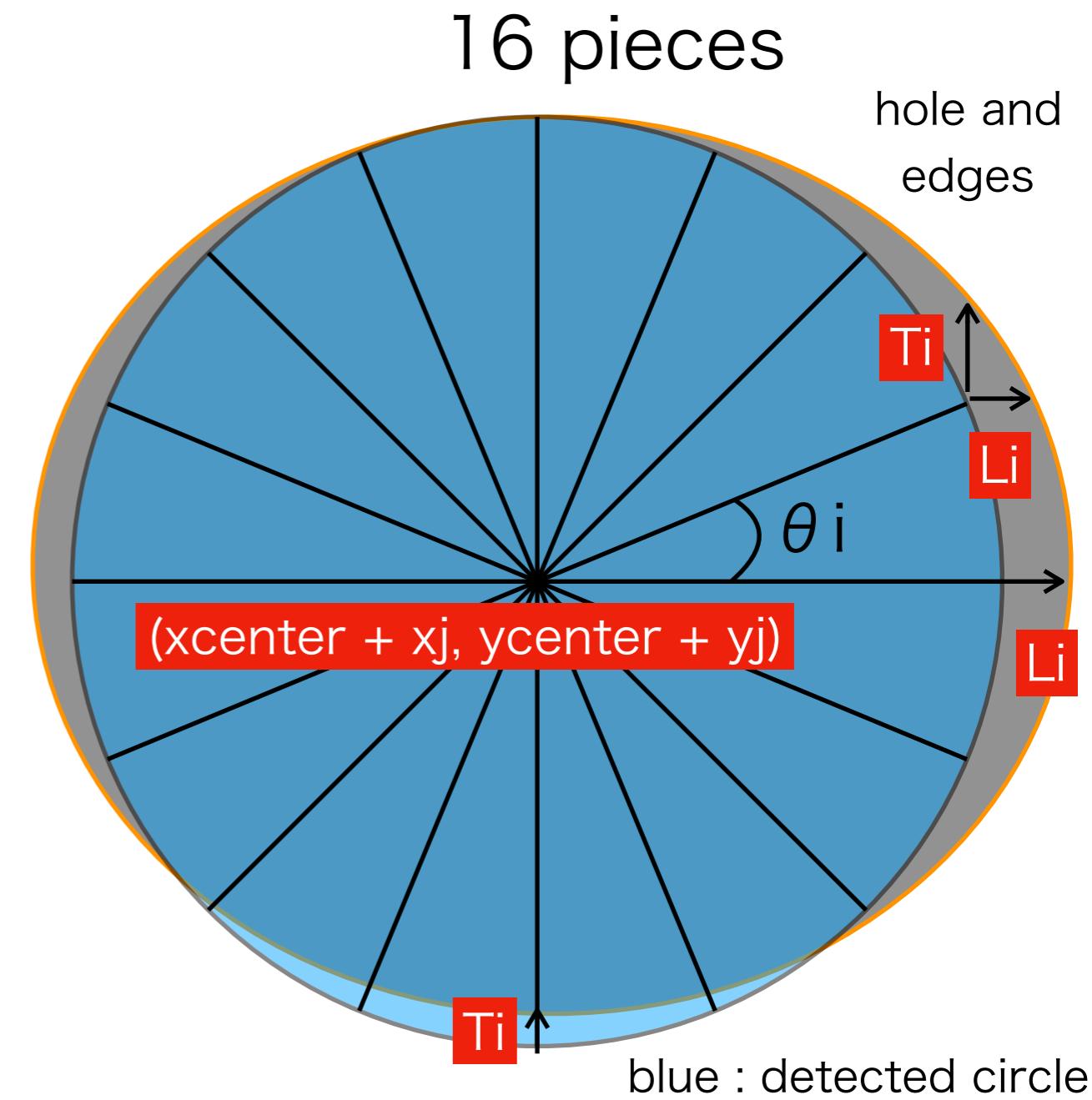
green : detected circle  
yellow : edges of hole

# minimization of distance between circle and edge

- equally devide the circumference (into 12 ~ 16 pieces)
- sum the longitudinal and lateral distance to edge :  $E = \sum (Ti + Li)$ 
  - use only  $Li$ ,  $Ti$  for  $\theta_i = 0$  or  $\pi$ ,  $\theta_i = \pi/2$  or  $3\pi/2$  respectively.
  - if there are some edges in the same direction, use the farther one.
- move the center of hole one by one ( $x_j, y_j$ ), and increase the radius by 1, minimize  $E = \sum (Ti + Li)$ .

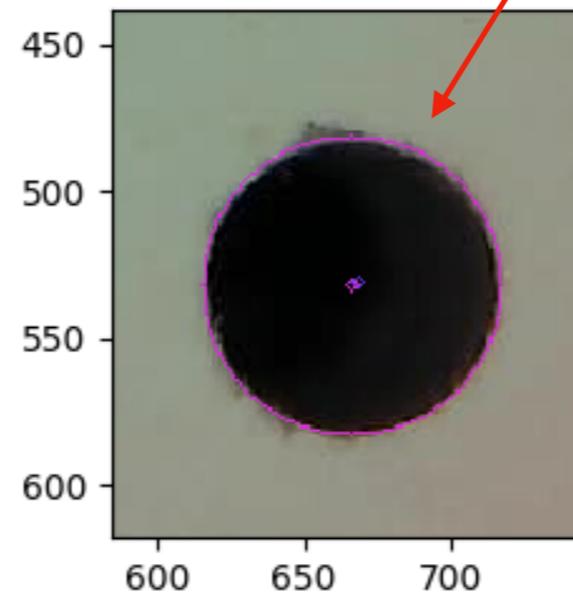
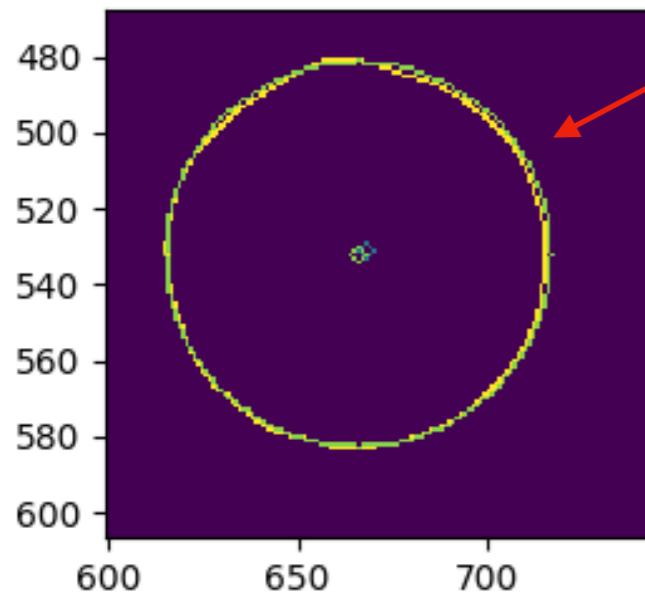
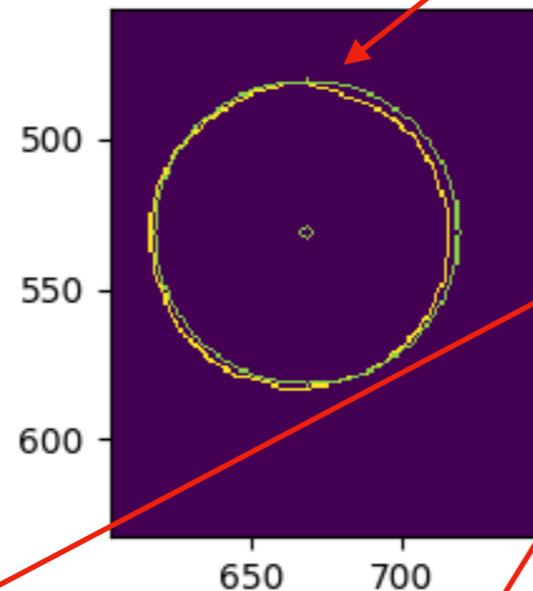
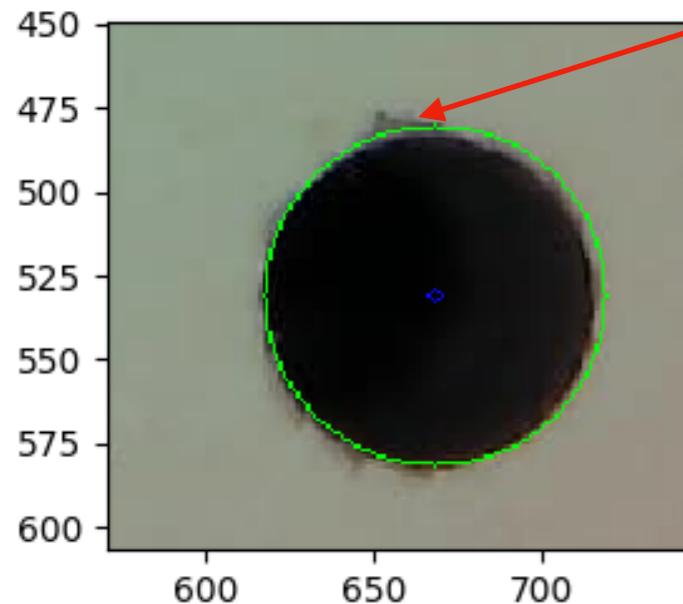
search area :  $-5 \leq x_j \leq +5, -5 \leq y_i \leq +5$

search radius :  $r = r_0 + \delta r, 0 \leq \delta r \leq 5$



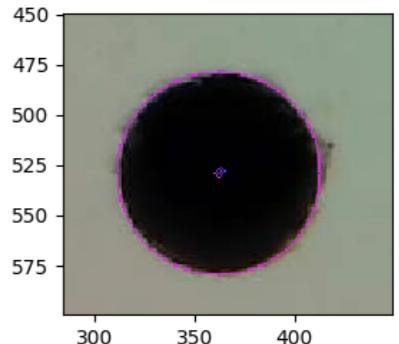
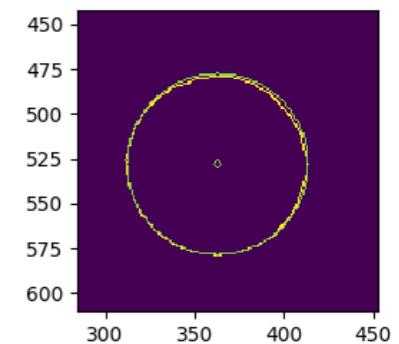
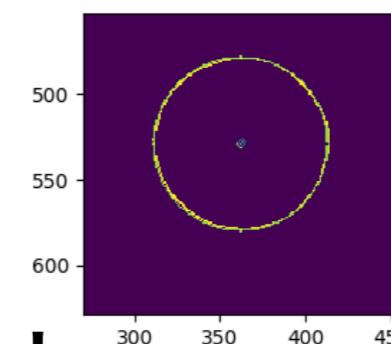
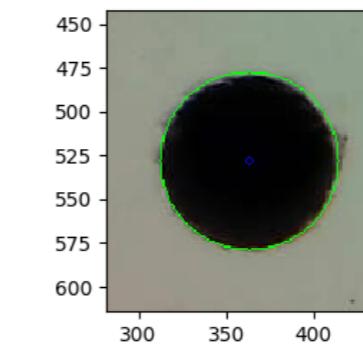
# Example of ‘good’ hole

Hough Circles



- upper row : detect hole by Hough circle transformation (green) (yellow : edge)
- bottom : edge (yellow) , circle after E minimization (green and pink)

Hough Circles

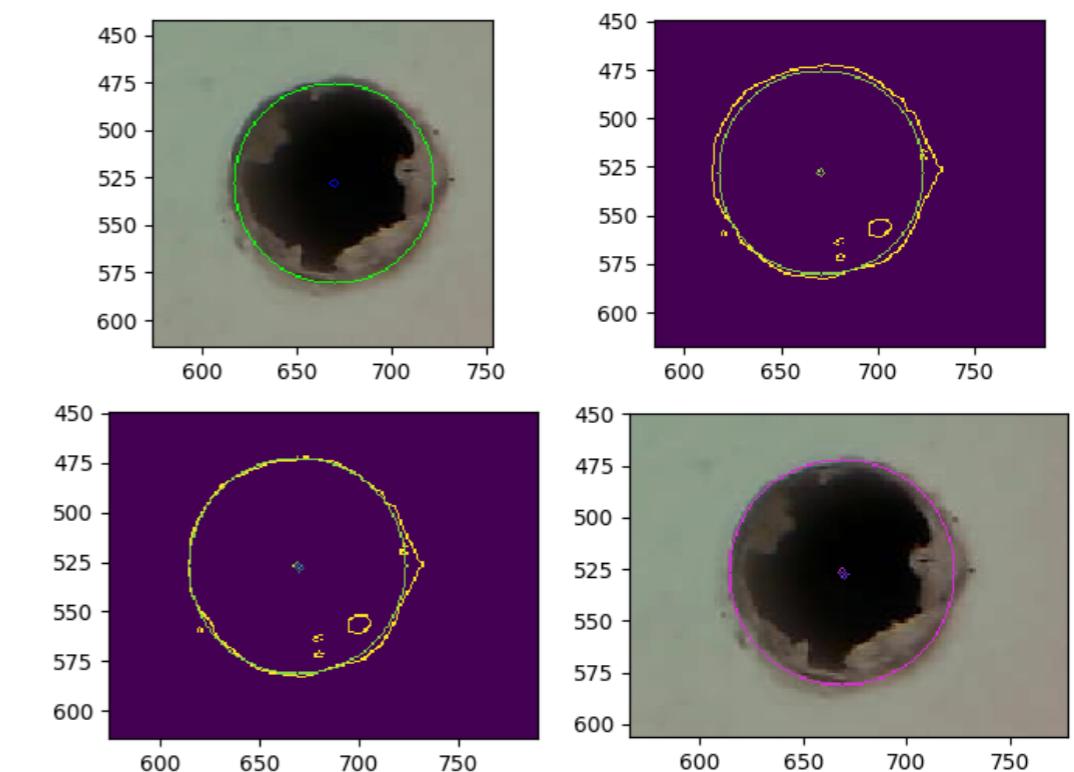
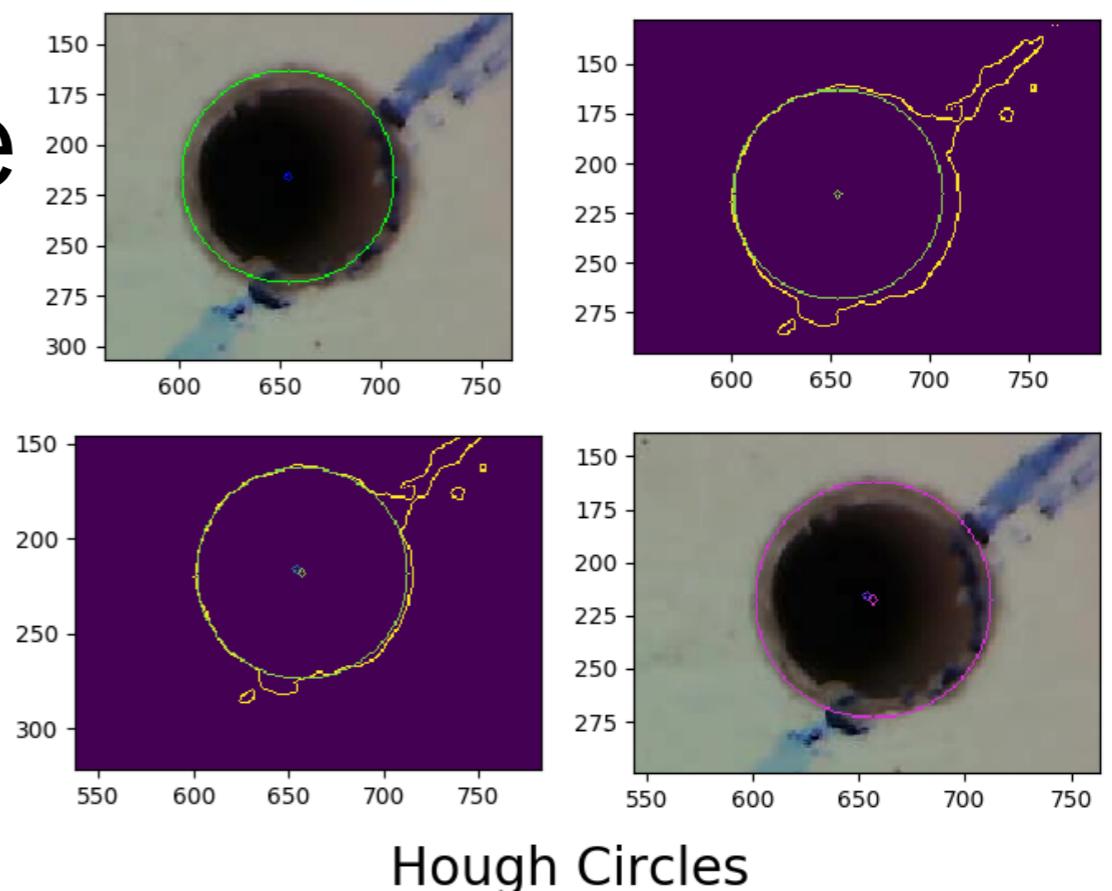
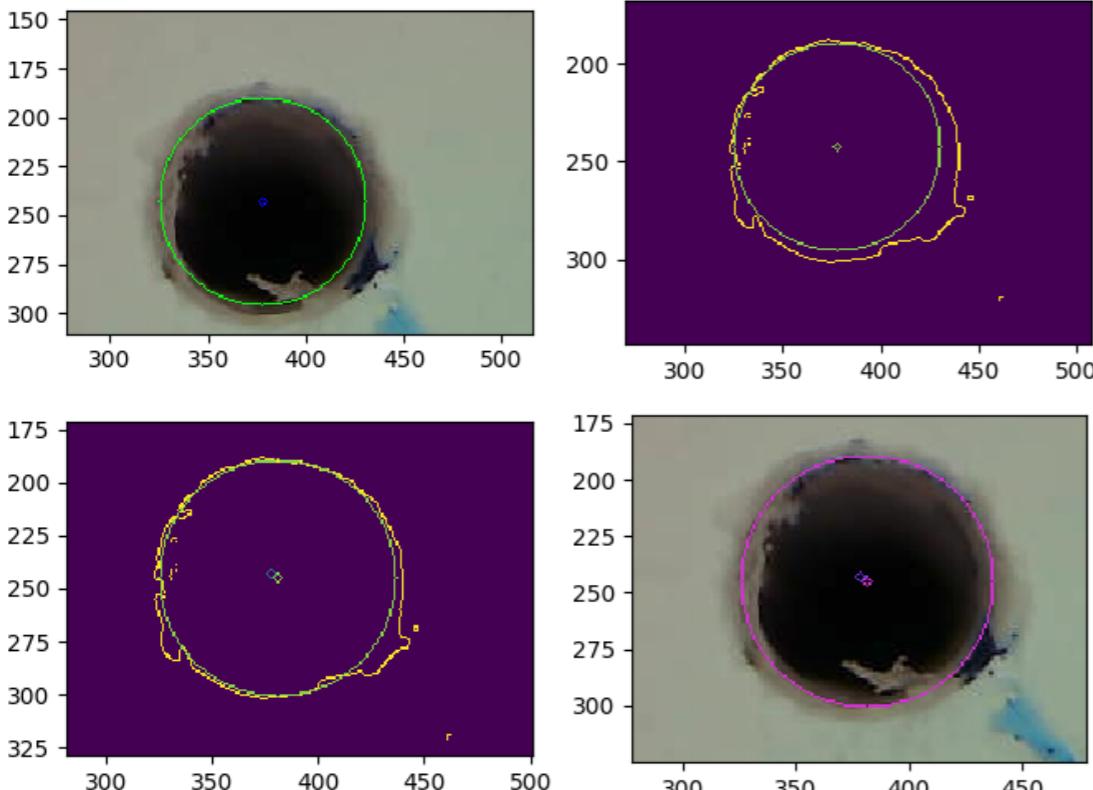


for ‘good’ hole, this corrections succeed

# Example of ‘bad’ hole

- If there is a ‘good’ edge, we get accurate center of hole by correction.
- Since we have to scan for many times, it takes time (~ 5 sec)

Hough Circles

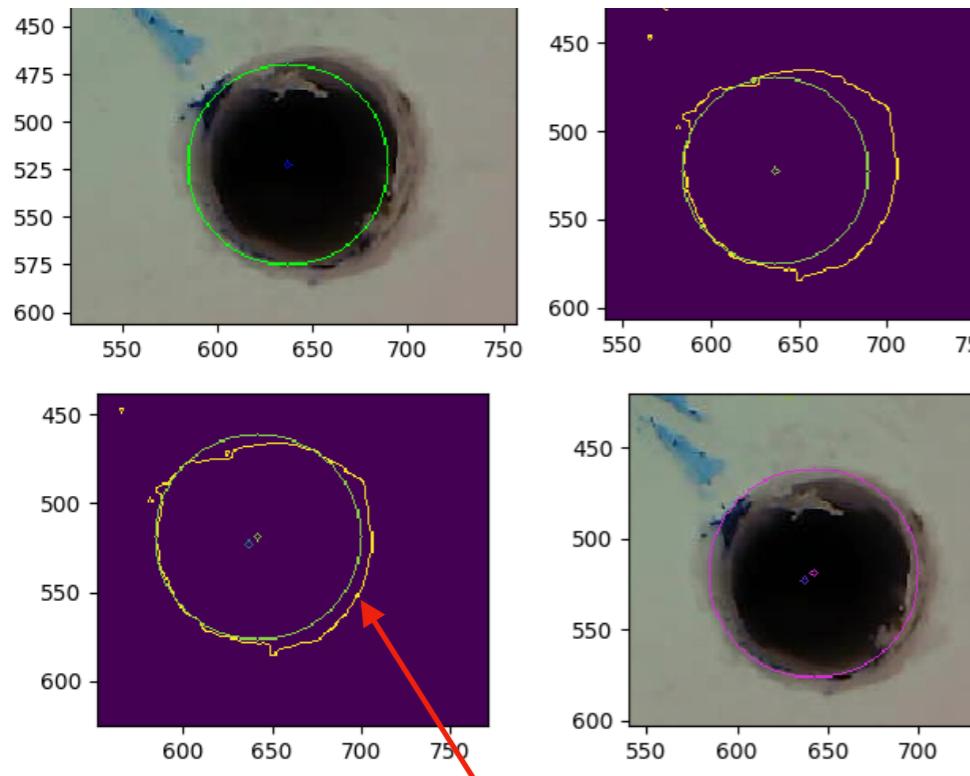


upper row : before correction  
bottom row : after correction  
(for each image matrix)

# Example that could not get ideal center

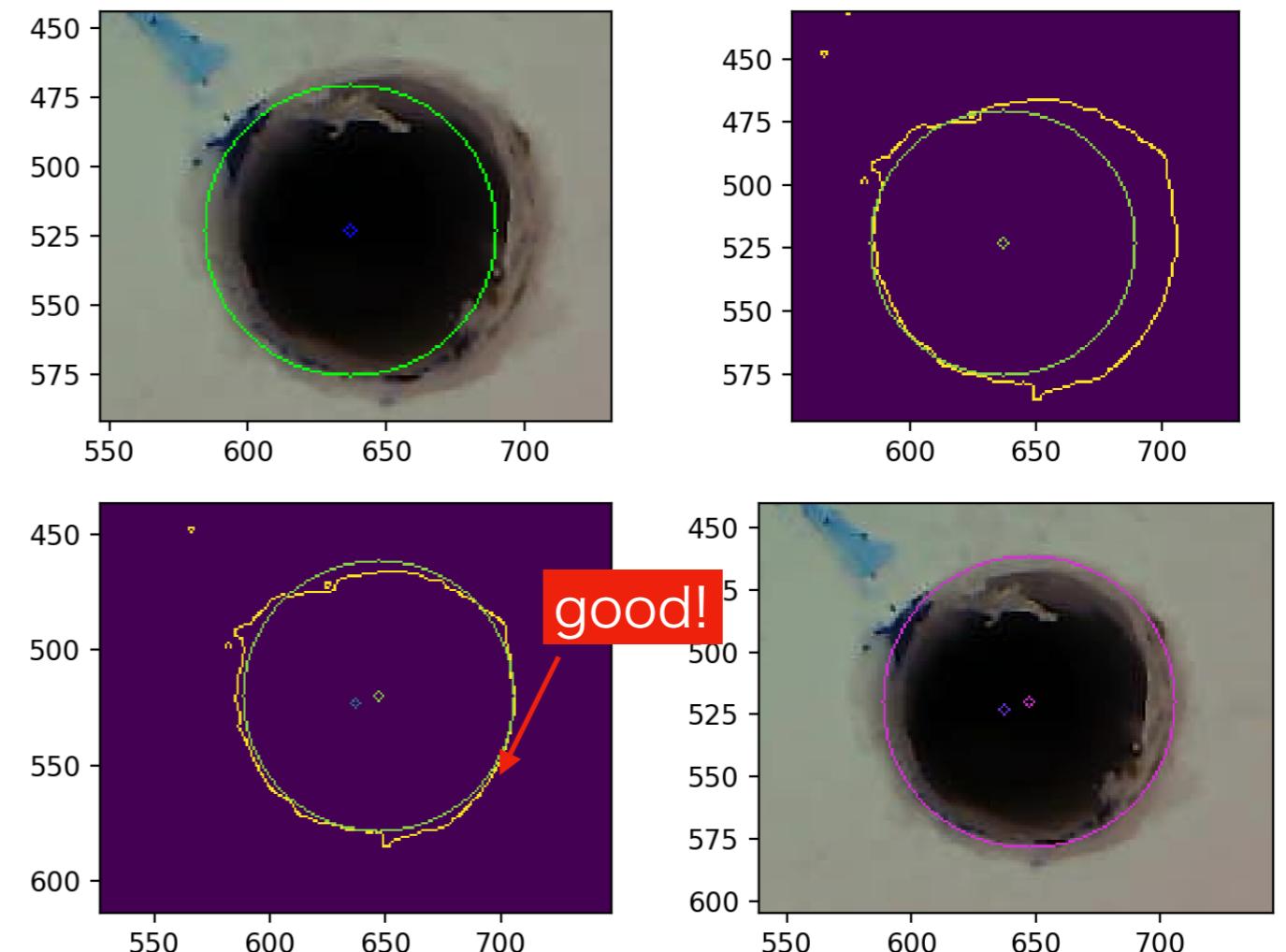
for big edge, it is difficult to correct the circle.

- enlarged the search area, then succeeded (bottom image)



upper row : before correction  
bottom row : after correction  
(for each image matrix)

- center ( $x_{\text{center}}+9$ ,  $y_{\text{center}}-4$ ),  
radius  $r = r_0+6$



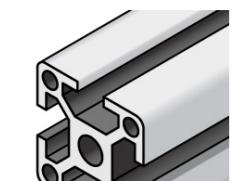
because of increment of scan area it takes more time (~25 sec)

plan of 3 dimensional  
photography jig

# photography jig

## photography platform for cubes

アルミフレーム 5シリーズ 正方形 20×20mm 1列溝 4面溝  
アルミフレーム 5シリーズ 正方形 20×20mm 1列溝 4面溝

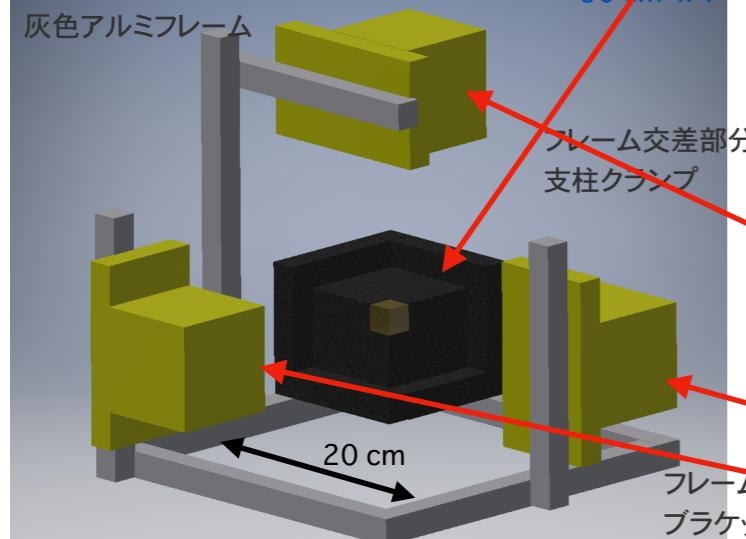


[https://jp.misumi-ec.com/vona2/detail/110302683830/?rid=c14\\_detail\\_6\\_110302683830](https://jp.misumi-ec.com/vona2/detail/110302683830/?rid=c14_detail_6_110302683830)

HFSB5-2020-[50]4000/0.5]

300 円: 20 cm x 7

30 cm x 1



5シリーズ(溝幅6mm) -1列溝用- 突起付反転ブラケット  
5シリーズ(溝幅6mm) -1列溝用- 突起付反転フ



<https://jp.misumi-ec.com/vona2/detail/110300437260/?HissuCode=HBLFSNB5-C&PNSearch=HBLFSNB5-C&KWSearc=HBLFSNB5-C&searchFlow=results2products>

HBLFSNB5-C (M5)  
200円 x 6

L字接続、クロス接続どちらの場合でも接続が可能です。

5シリーズ(溝幅6mm)20・25・40角アルミフレーム用

5シリーズ(溝幅6mm)20・25・40角アル HNTT\40 x 2



C-30-RK-3220 (ネジM5)  
160 円 x 4

ゴム足  
支柱タランプー角・角直交ー ALQOD20  
支柱タランプー角・角直交ー 3300円 x 4



- we will make a 3 dimensional photography jig (left image)

- order photography platform and camera box

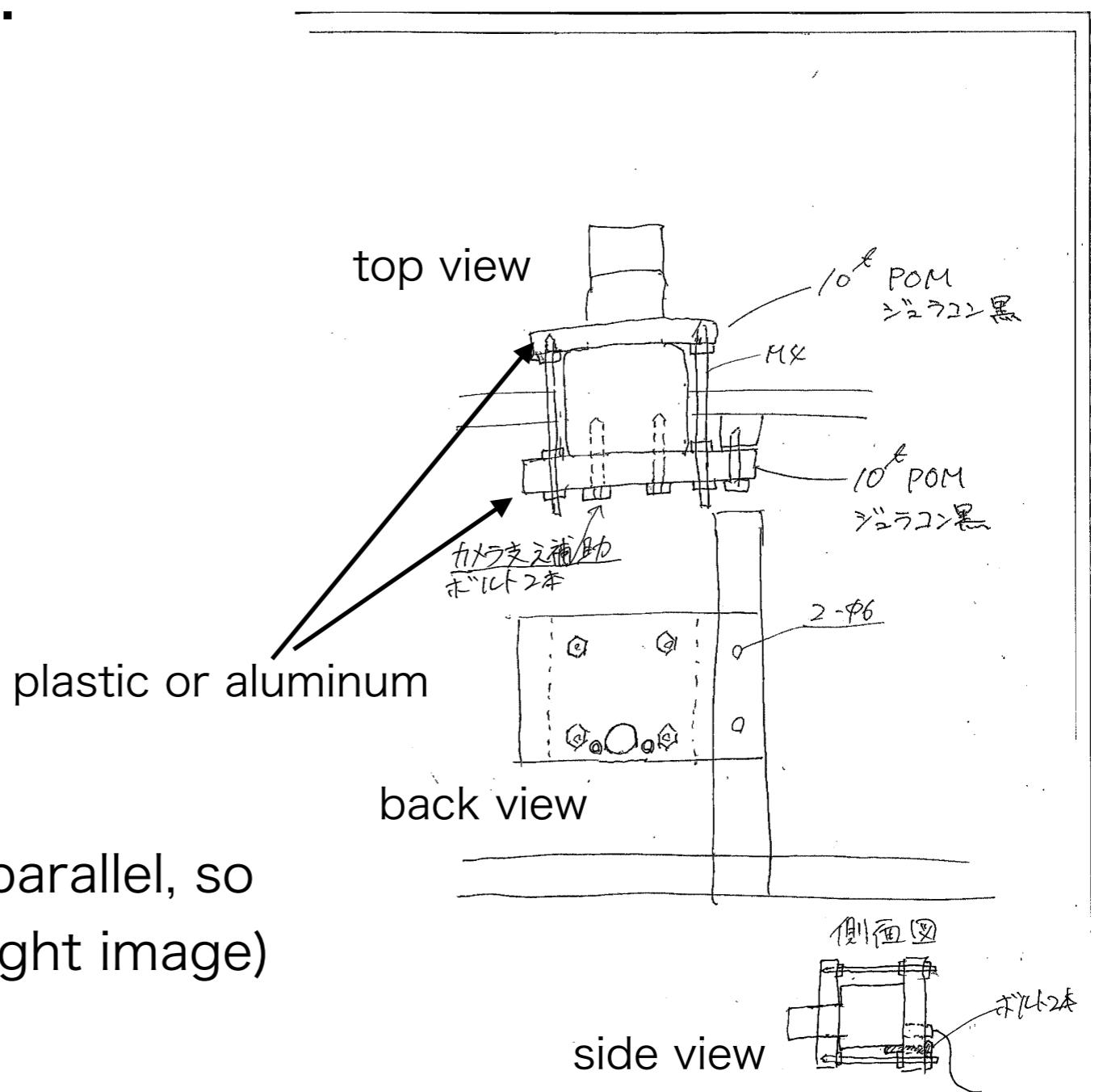
camera box

# camera box

body of this camera is not a rectangular, but  
this area protrudes.



backside and frontside look parallel, so  
consider such a camera jig (right image)



# conclusion

- improved photographic system and equipments
- succeed to correct the detected hole of cube
- going to make a new photography jig