

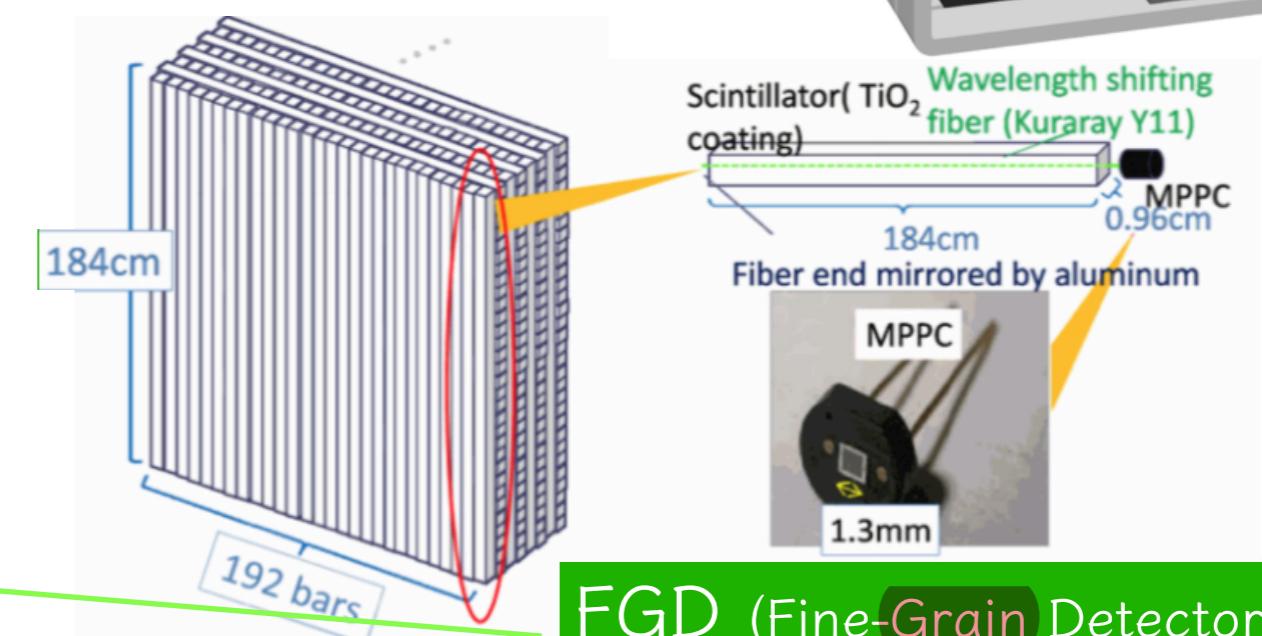
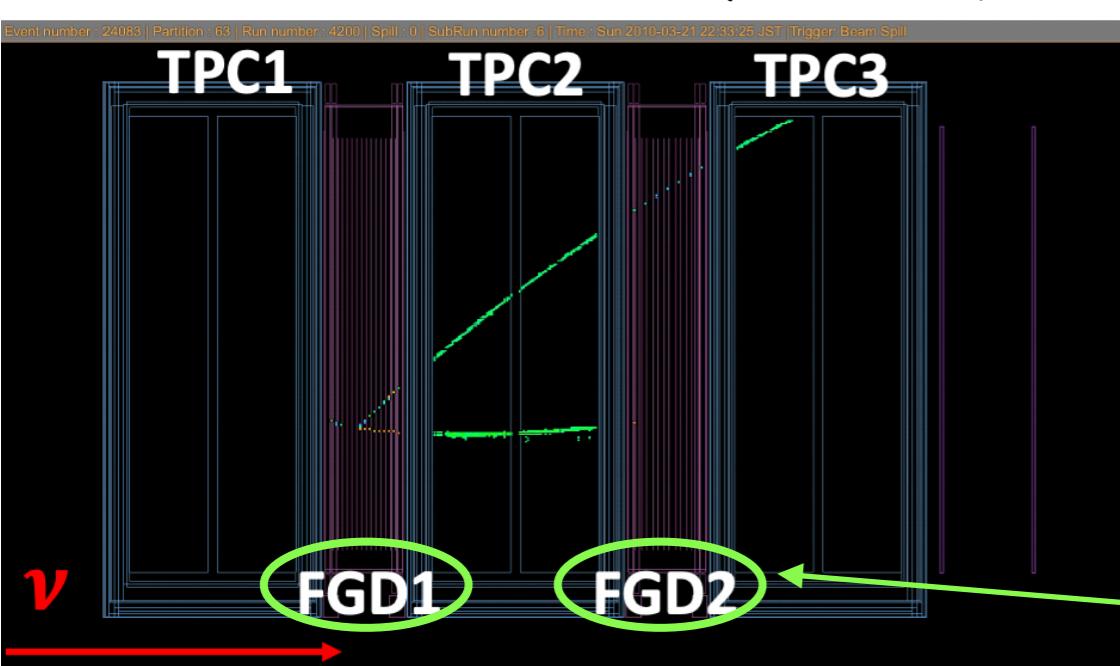
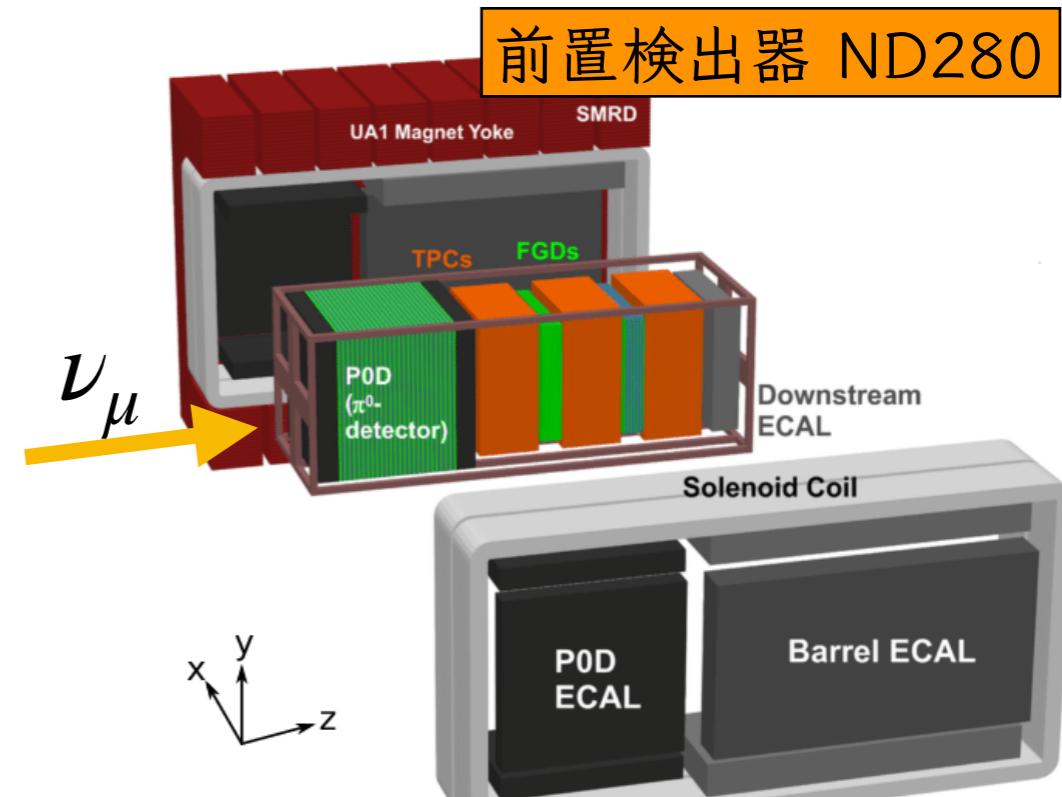
T2K 前置検出器アップグレード用
シンチレータキューブの画像認識による
自動検査システム開発

京大理, 東大理A, 首都大理B, KEK素核研C, 総研大D
谷真央, 栗林宗一郎, 木河達也, 市川温子, 中家剛, 江口碧A, 鞠谷温士A,
岩本康之介A, 横山将志A, 在原拓司B, 粟田口唯人B, 角野秀一B, 小川智久C,
松原綱之C, 中平武C, 藤井芳昭C, 小林隆C, Jakkapu MaheshD,
他T2K Collaboration

T2K 実験概要

- T2K: 長基線ニュートリノ振動実験

- J-PARC で生成した大強度ニュートリノビームを、前置検出器・後置検出器（スーパーカミオカンデ）で測定
- ニュートリノ振動におけるCP対称性の破れの証拠をとらえる
- 系統誤差の要因：ニュートリノの反応断面積の不確定性が大きい
 - 前置検出器 ND280 等によってニュートリノ反応の正確な理解を目指す

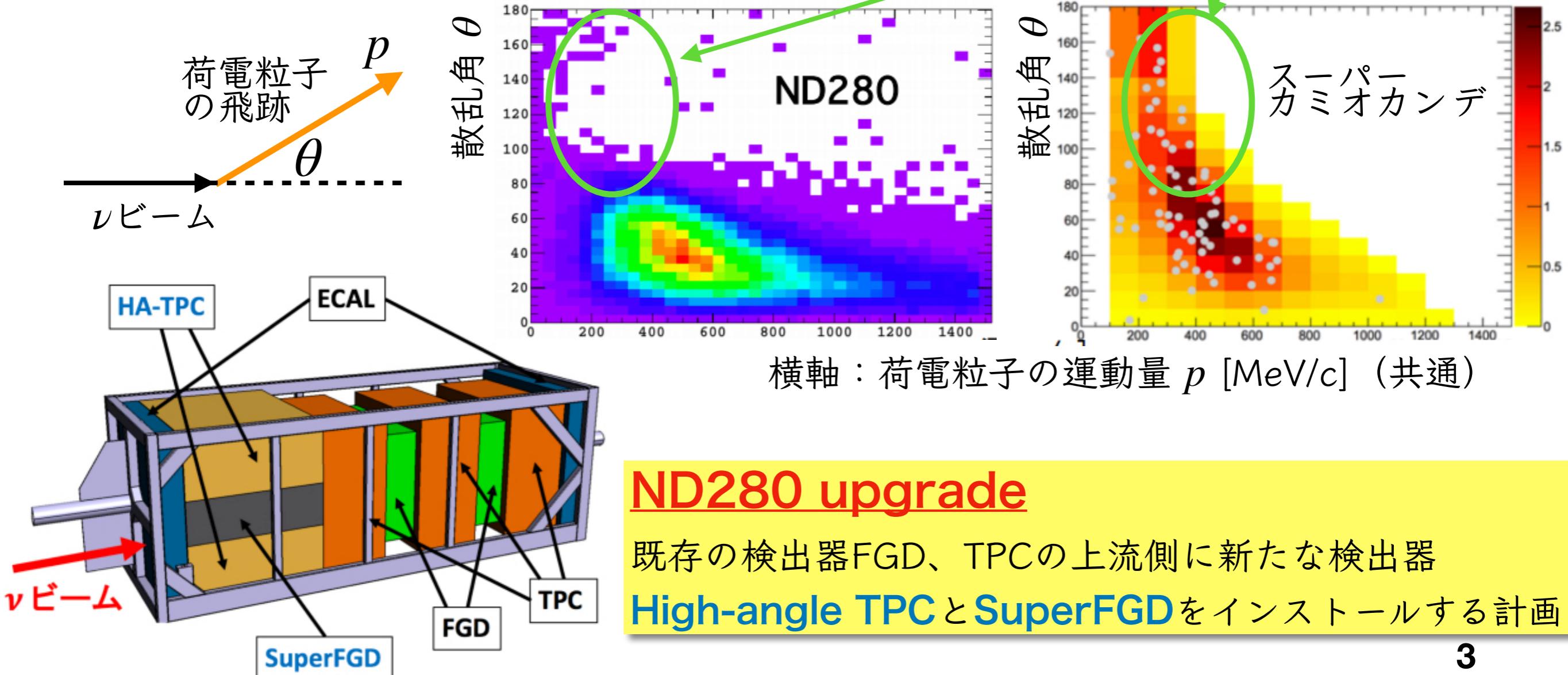


前置検出器 ND280 upgrade

現在のND280 における問題点：

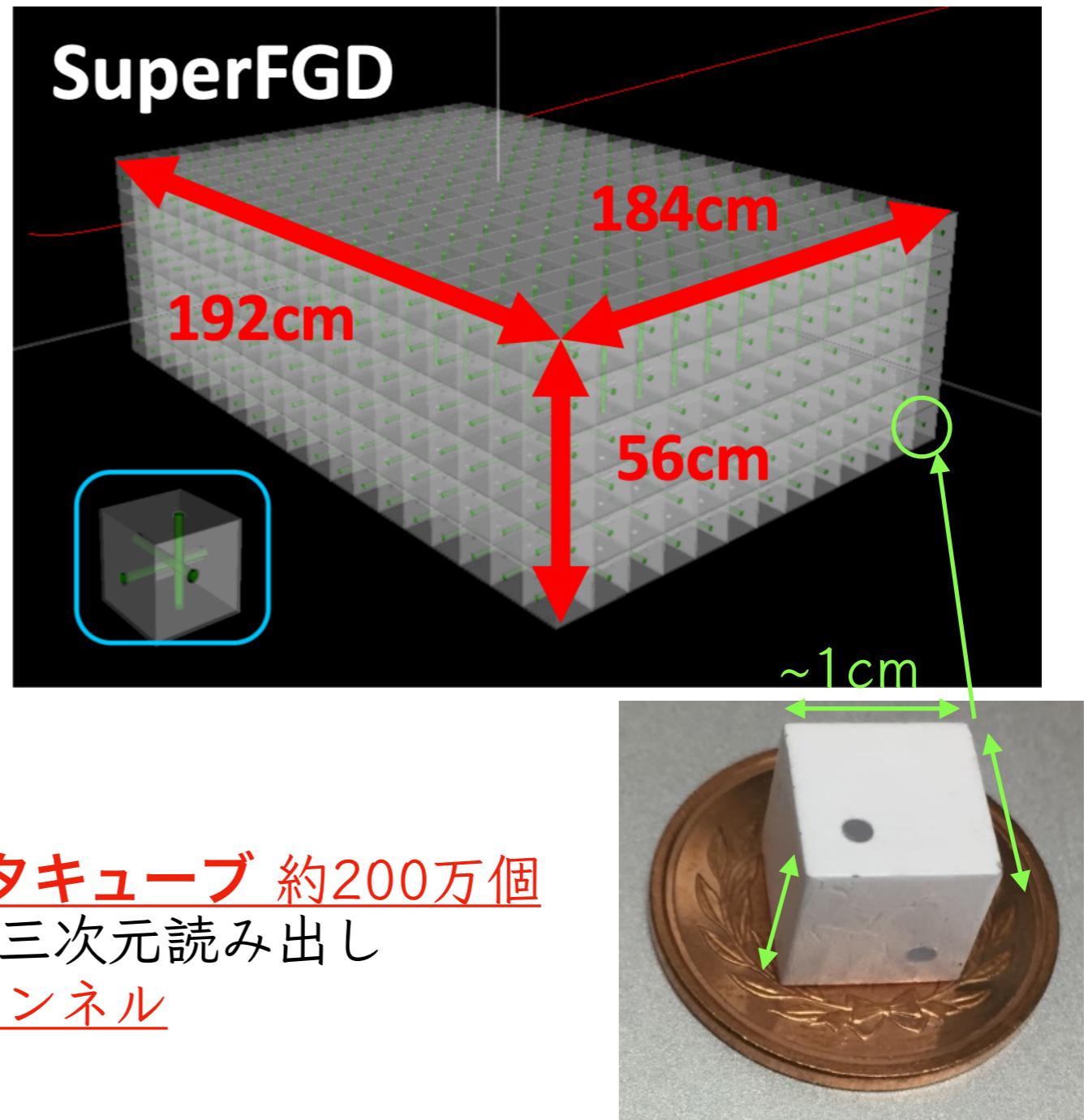
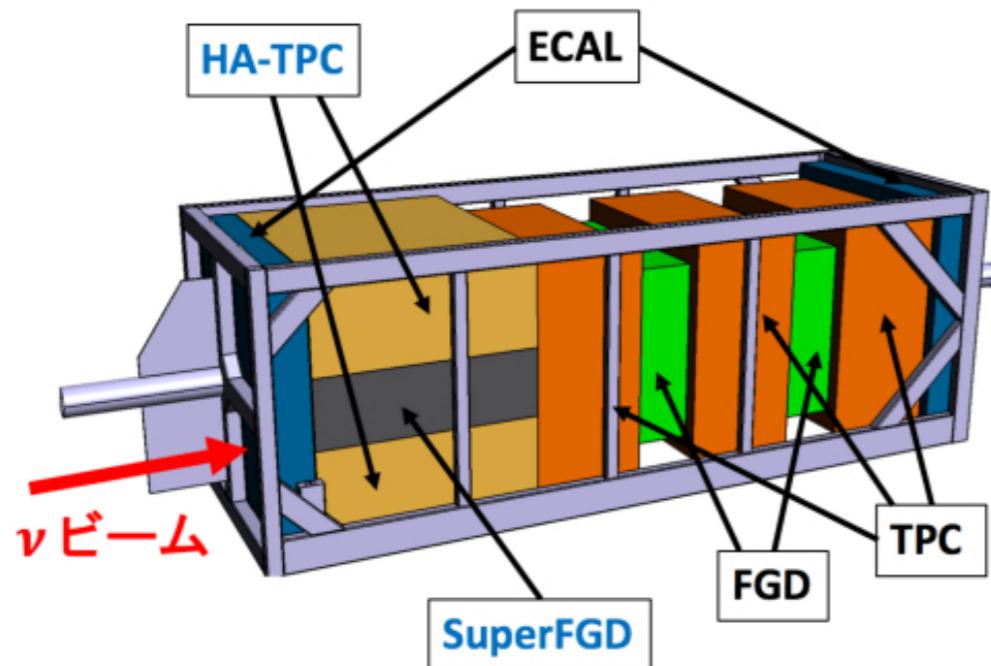
- 大角度散乱に対する検出効率が低い
(スーパー・カミオカンデでは 4π に対し acceptance をもつ)
- 低運動量の荷電粒子の軌跡検出が難しい

低運動量・大角度散乱の
粒子に対する振る舞いが
大きく異なる



Super-FGD

Super Fine-Grain Detector



- 1立方センチメートルのシンチレータキューブ 約200万個
- $\phi 1\text{ mm}$ 波長変換ファイバーにより三次元読み出し
- ピクセル光検出器 MPPC 約6万チャンネル
- キューブはINR(ロシア)にて製造
 - ポリスチレンベースのキューブを成形
 - 薬液に浸けて表面を発泡させ、反射層化(ケミカルエッティング)
 - $\phi 1.5\text{ mm}$ のドリルで穴あけ

精度 σ : $\sim 30\mu\text{m}$ (外形)、 $\sim 50\mu\text{m}$ (穴位置)

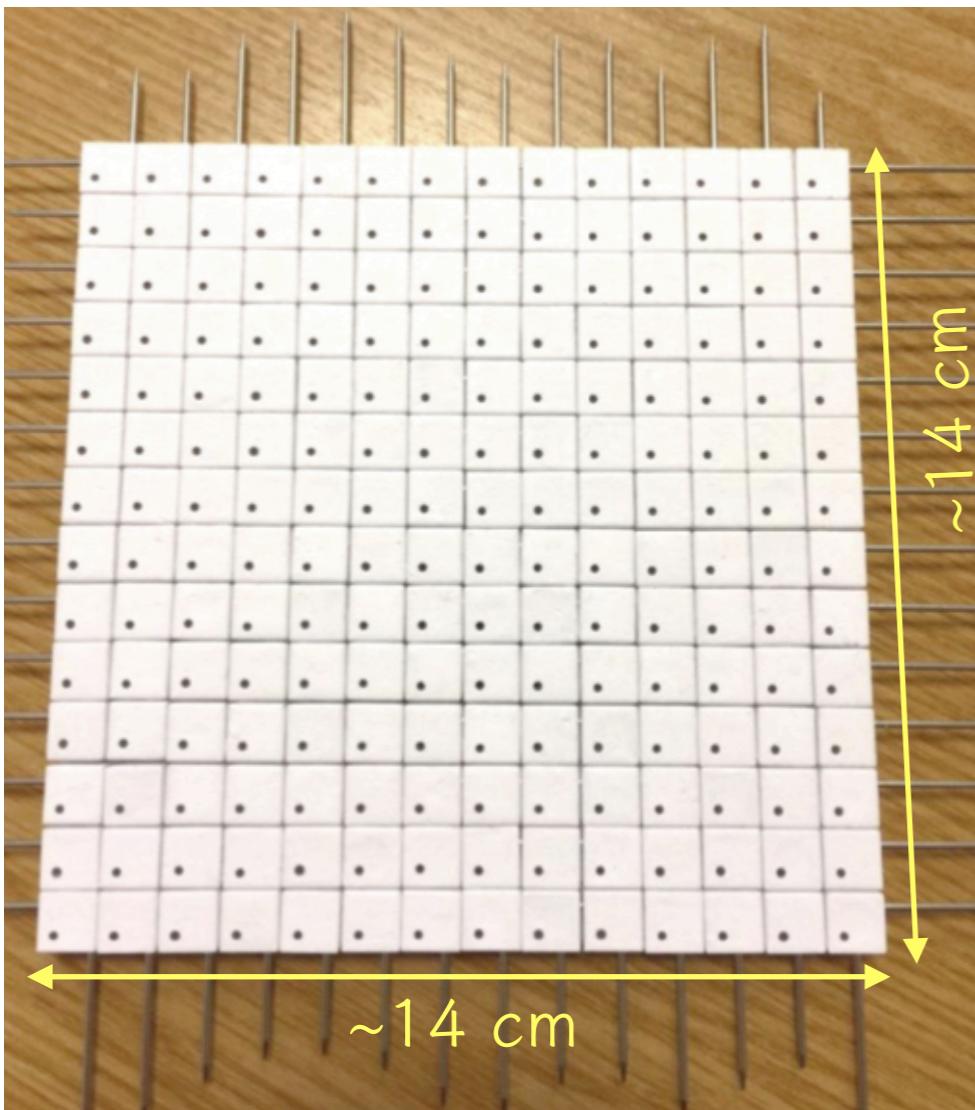
表面の発泡化の度合い・穴位置のズレ・穴の傾き等によって、最終的な組み上げに影響を与える可能性がある → キューブの事前チェックが必要

現行のロシアでの Quality Check

- $14 \times 14 (=196)$ 個のキューブを正方形に並べ、
2方向から金属の棒 ($\phi 1.4$ mm) を通す
- 金属棒が上手く通らなければ、該当部のキューブ
の穴の周囲にドリルの際に生じたバリ等を取り除
いて再度試験を行う
- それでも金属棒がなめらかに動かなければ、それ
は bad cube であると判断
 - 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・方向等が微妙にずれている

金属棒試験の欠点

- 時間かかる
- 定量的な bad cube の判断が難しい、
- 個人差が出る (複数人での並行作業)



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



カメラによりキューブ面を撮影し、
画像解析により必要な情報を抽出、
定量的にキューブの良し悪しを判断
する自動システムの開発

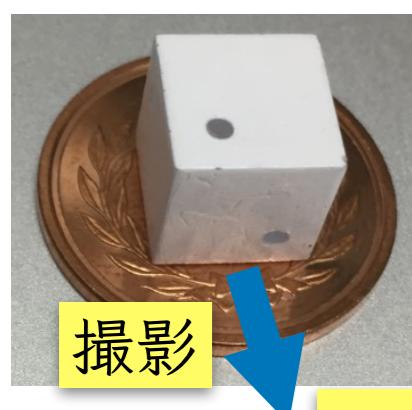
カメラを用いたキューブ撮影

大まかな手順

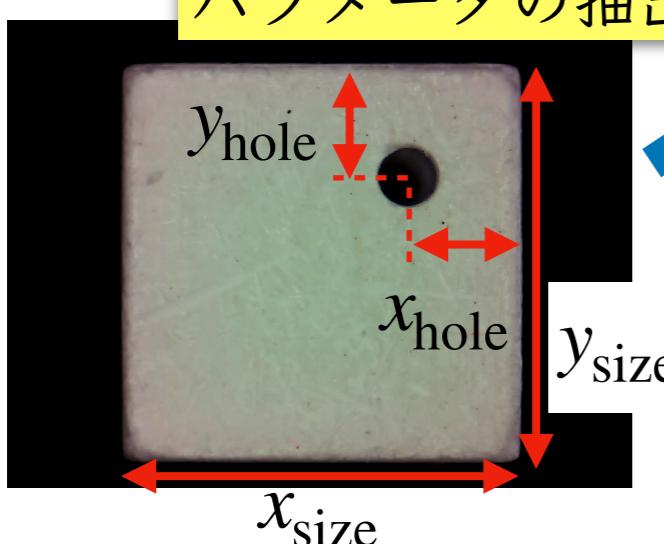
- キューブを固定・6面を撮影
- 6枚の画像から選別のためのパラメータを抽出
- 予め用意した条件を参照、各パラメータが許容範囲内かどうかチェック
- 選別結果により、キューブを分ける



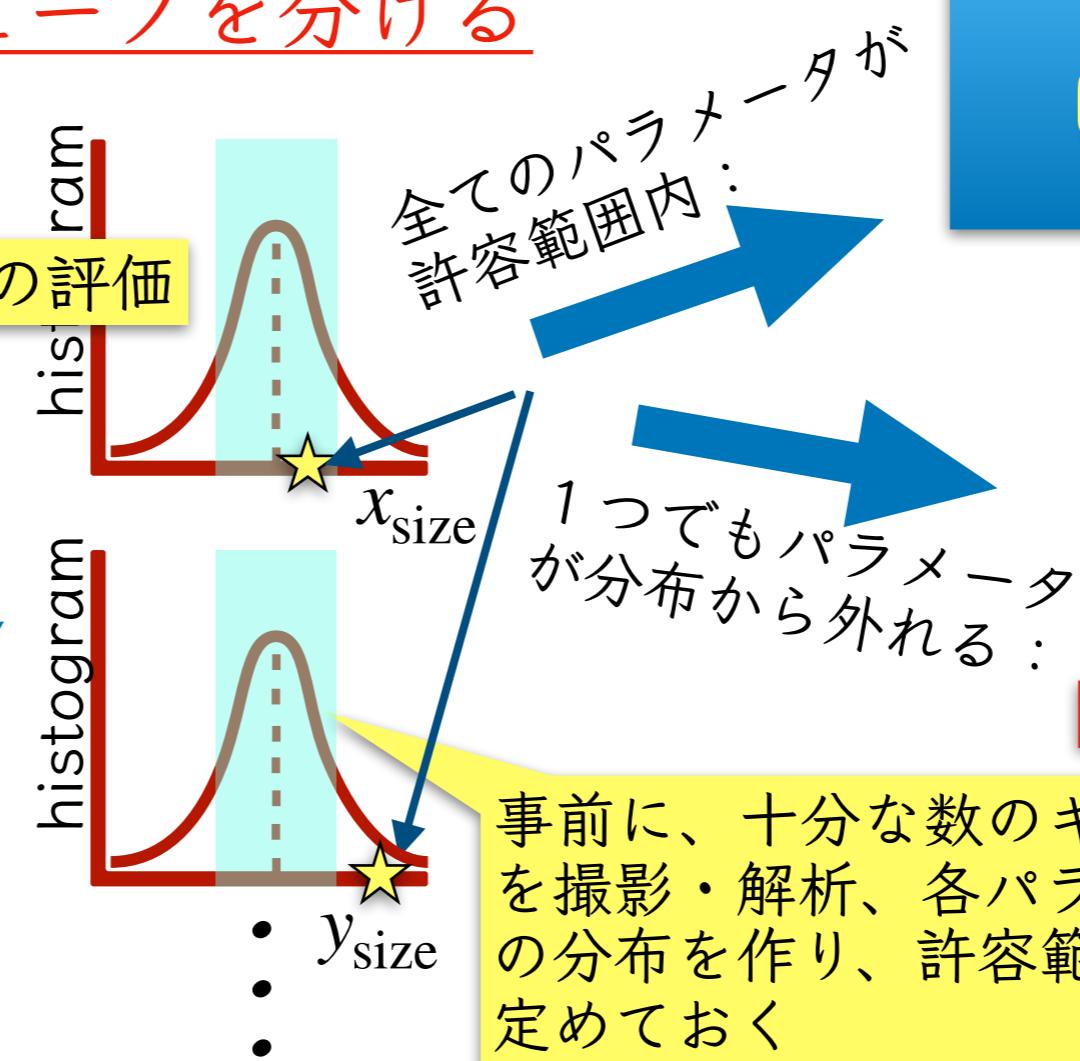
ELP 社 800万画素 Webcam



撮影



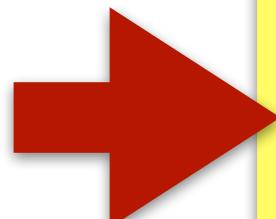
パラメータの抽出



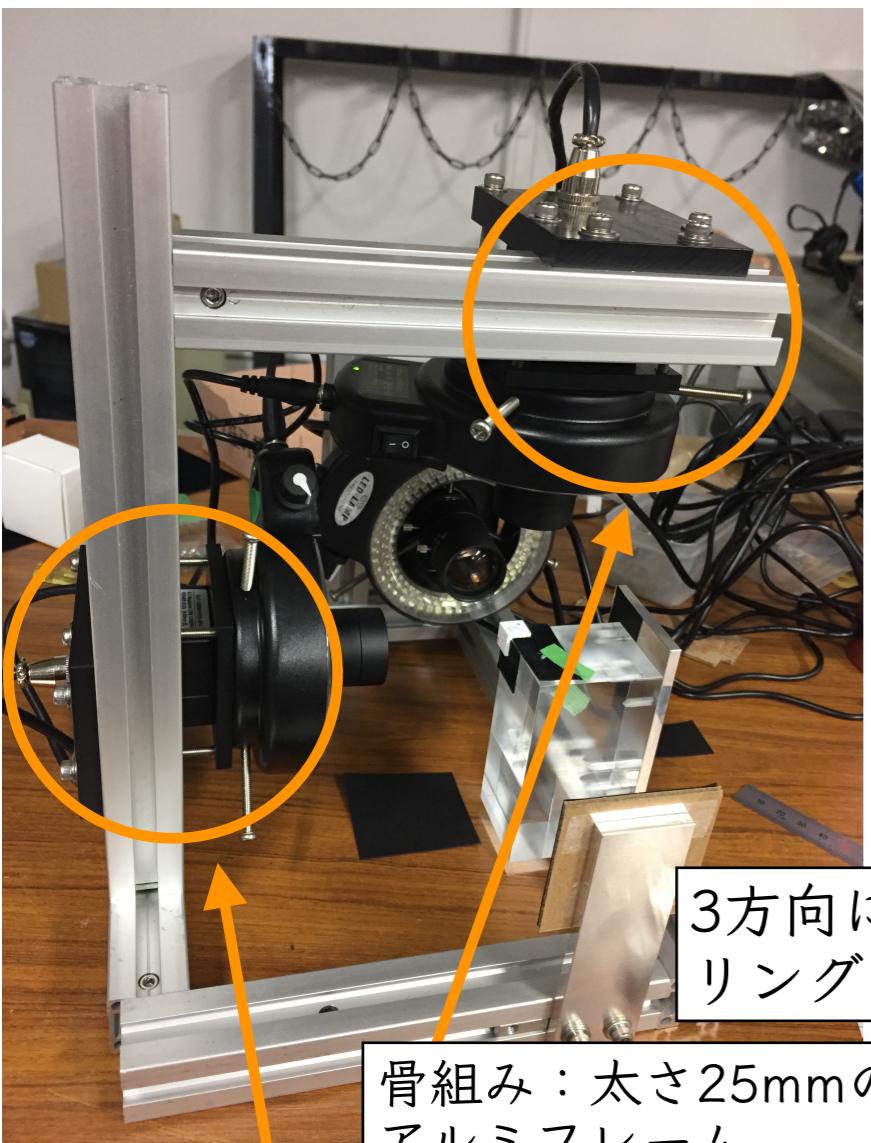
Bad Cubes

本研究の目的

- キューブ画像の解析
 - 画像解析による必要なパラメータの抽出法開発
 - 十分な統計量(数百~?)のパラメータを用いた分布生成、許容範囲の決定
- ジグの開発
 - カメラの画素数を最大限に活用する撮影ジオメトリ
 - 大量(数万個~)の撮影に耐える再現性
 - 短時間(~数秒/キューブ)でのスムーズな撮影
 - キューブの6面全てを同条件で撮影
- **暫定目標：プロトタイプ検出器の組み上げ**
 - 約12000個 のキューブを用いた Super-FGD の プロトタイプ検出器の製作を計画
 - 組み上げ前に本システムを用いた Quality Check を予定



現行の撮影ジグ



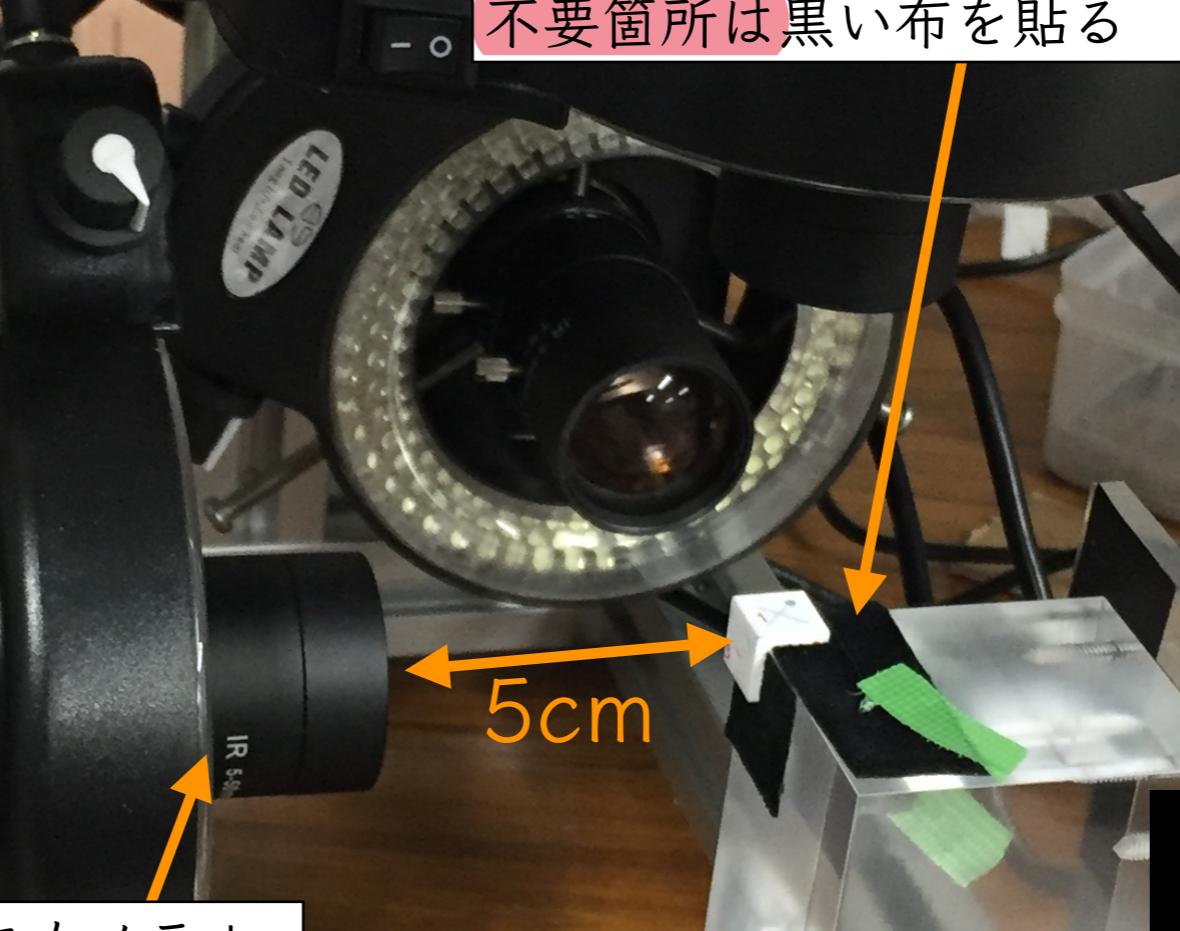
カメラを水平・垂直に固定するため、専用のカメラジグを制作(スズノ技研)



3方向にカメラ+リングライト設置

骨組み: 太さ25mmのアルミフレーム

- 3方向からの三面同時撮影
- 焦点距離5cmでの撮影
- 各カメラはリング状のLED、リアコンバータを装備
- 手でキューブを回転し、残りの三面の撮影



キューブ台座（アクリル、スズノ技研）
不要箇所は黒い布を貼る

リアコンバータ



X2 TV EXTENDER JAPAN



使用前

約2倍の分解能



使用後

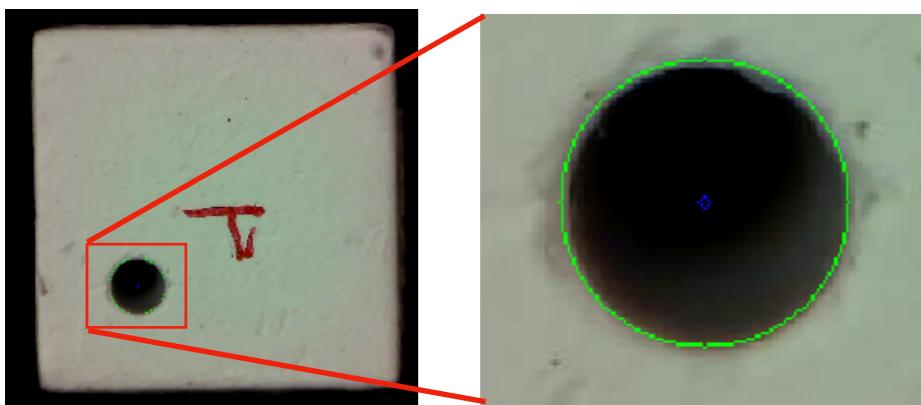
1 pixel ~ 15 μm

パラメータの取得

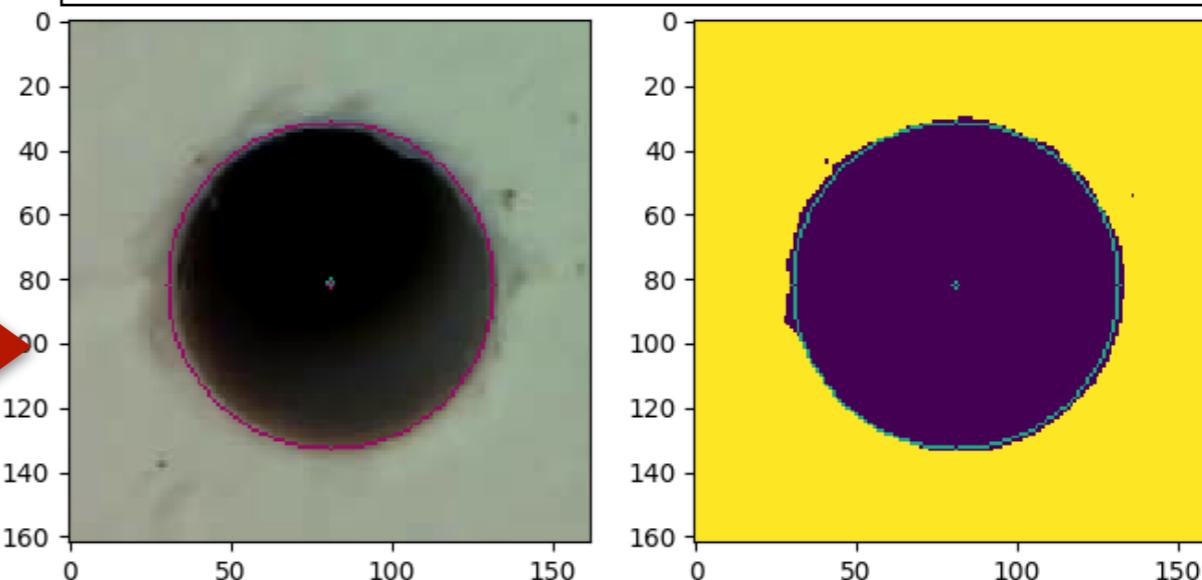
解析コード: 主に python + openCV(画像処理モジュール)

- 穴検出

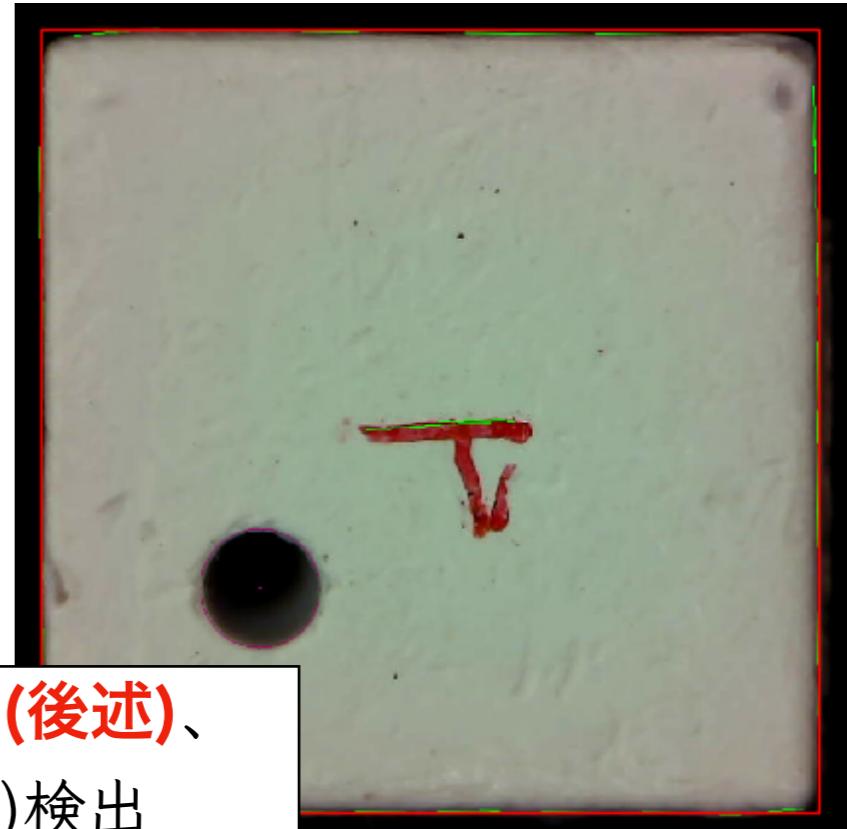
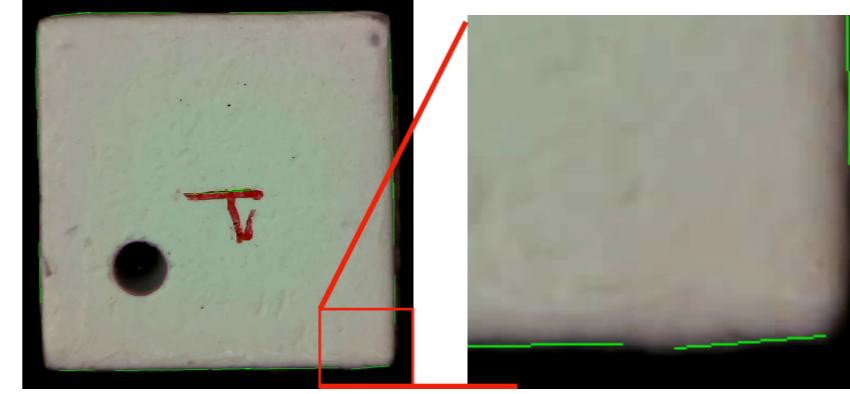
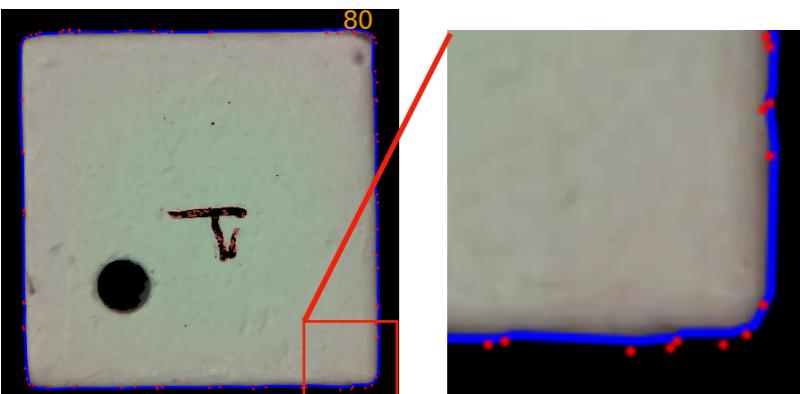
openCV 上の円検出関数による
大まかな穴検出



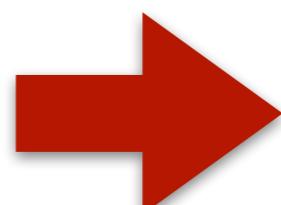
χ^2 最小化による詳細な穴位置検出 (後述)



- 辺検出・キューブのサイズ算出



openCV 上の関数を用いた
輪郭検出、直線検出



撮影時の傾き補正 (後述)、
表面のバンプ(凹凸)検出

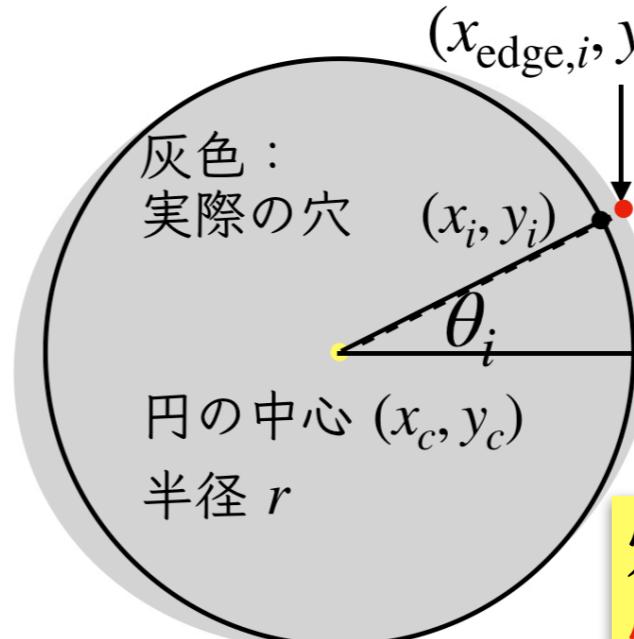
キューブ穴位置の検出 - カイニ乗値の最小化 -

$$E_{\text{sum}}(x_c, y_c, r) = \sum \left(|x_i - x_{\text{edge},i}|^2 + |y_i - y_{\text{edge},i}|^2 \right)$$

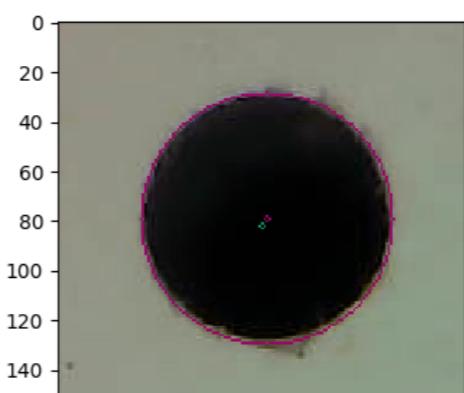
$$x_i = x_c + r \cos \theta_i, y_i = y_c - r \sin \theta_i$$

E_{sum} が最小となる (x_c, y_c, r) を求める。

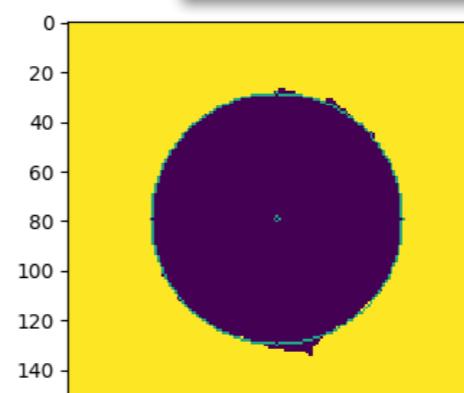
- $x_{\text{edge}}, y_{\text{edge}}$ は穴のエッジ上の点
- (x_c, y_c, r) の初期値は二値化画像から得た中央値を使う。
- 今回は $\theta_i = 0, \frac{\pi}{8}, \frac{\pi}{4}, \dots, \frac{15}{8}\pi$ で E_{sum} を計算。



穴の状態に関わらず、
穴位置の最適化に成功!

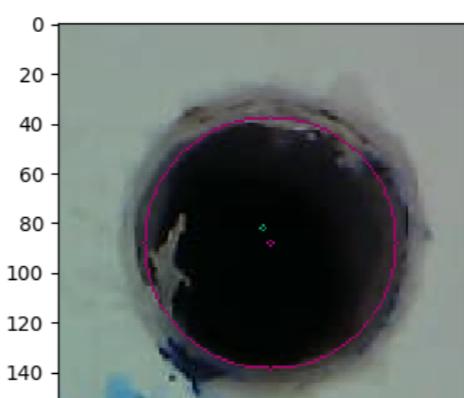


$(x, y, r):(364.1, 230.1, 52.4)$
 $\rightarrow(363.8, 229.4, 51.1)$

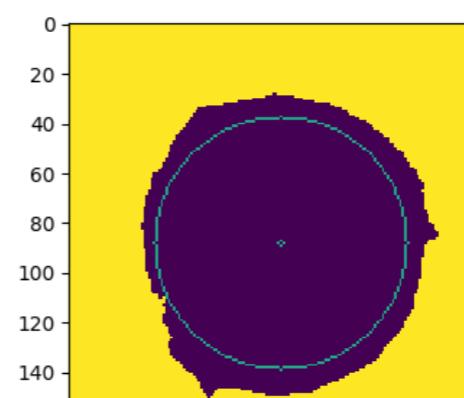


$E_{\text{sum}}:$
 $50.5 \rightarrow 13.1$

良い状態の穴



$(x, y, r):(652.1, 254.1, 52.4)$
 $\rightarrow(652.1, 253.8, 58.1)$



$E_{\text{sum}}:$
 $626.5 \rightarrow 106.1$

悪い状態の穴

* 画像上の円は
最適化前のもの

勾配法

ある $\mathbf{x} = (x, y, r)$ での E の勾配

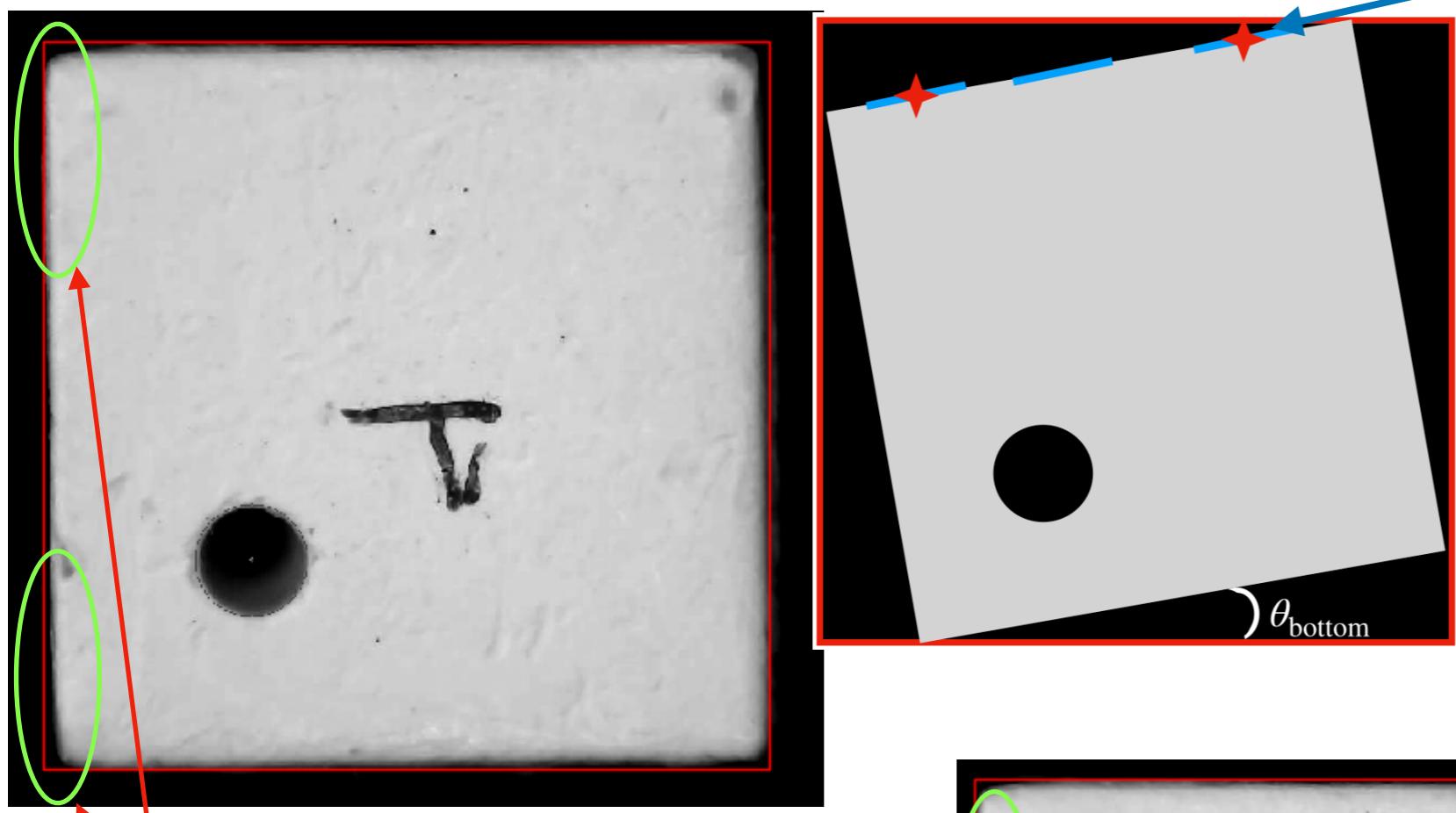
$$\nabla E = \left(\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}, \frac{\partial E}{\partial r} \right)$$

を求め、 \mathbf{x} を更新：

$$\mathbf{x} \rightarrow \mathbf{x} - \eta \nabla E \quad (\text{これを繰り返す})$$

η : 学習率

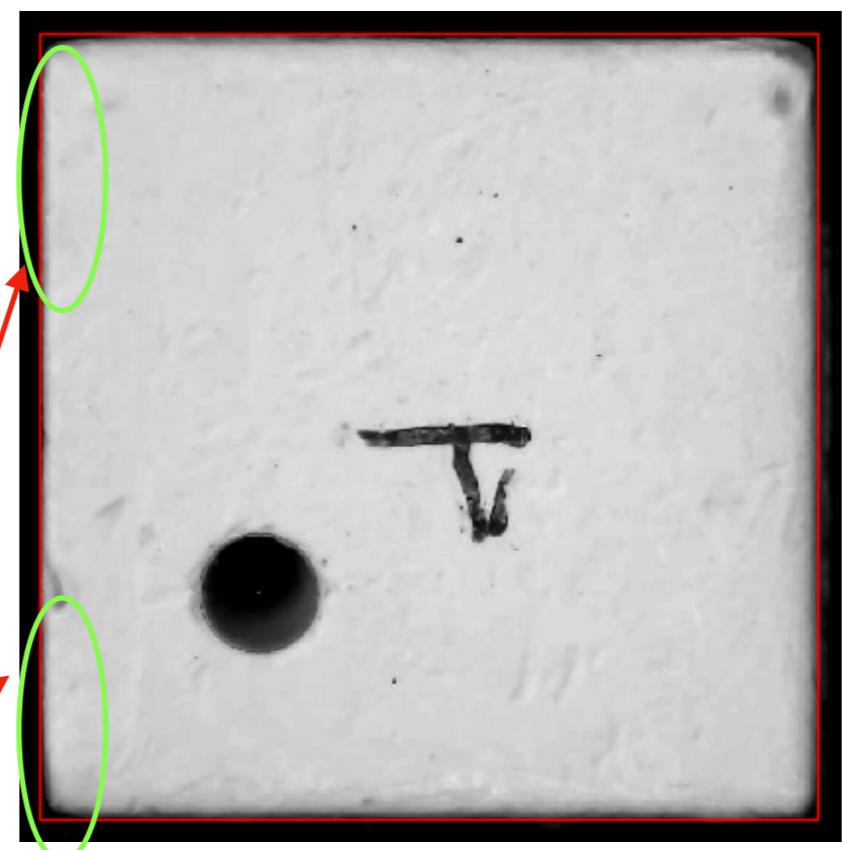
キューブの傾き補正



緑の丸：キューブが傾いているので、下の方では赤線とキューブ辺の間に隙間ができる。穴の相対位置は辺の座標を基準に決めるので、辺の正確な位置検出が重要

回転後：補正により、隙間は解消された

- 各辺ごとに、関数により複数の直線(青線)を検出
- それらの中心 \star を結ぶ直線の傾き $T = \tan \theta$ を求める
- 4つの辺について傾きを求め、平均 \bar{T} を得る
- 平均の傾き角 $\bar{\theta} = \arctan \bar{T}$ を求め、座標・画像の回転



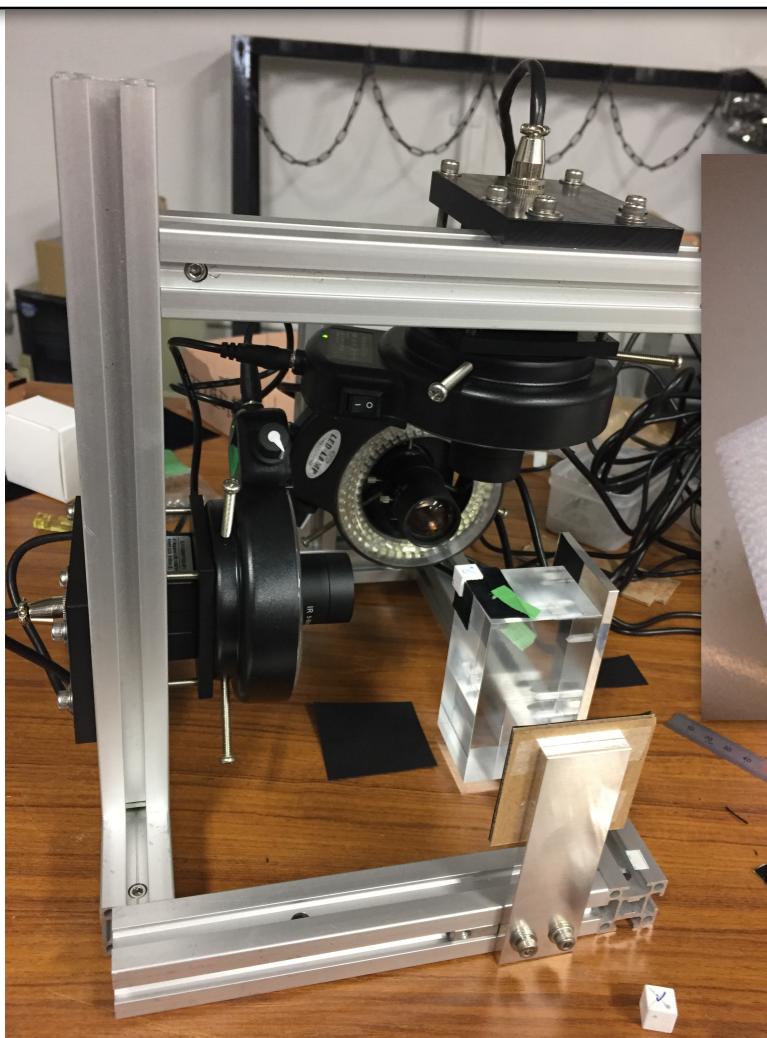
キューブの検出サイズ
(単位 pixel):
回転前 : (667, 665)
 \rightarrow 回転後 : (661.0, 667.7)

x方向のサイズは補正された
y方向は過剰に回転してしまった？統計をためて
詳しい確認が必要

この画像の場合 $\bar{T} = 0.0139$, $\bar{\theta} = 0.796^\circ$ に対応

現行の撮影ジグの問題点

- キューブを設置→撮影→回転→撮影→選別の繰り返し
- 問題点：
 - 確実に回転しないと6面撮影できない
 - 選別ミスの可能性
 - 各キューブにつき上記操作の繰り返し：時間がかかる



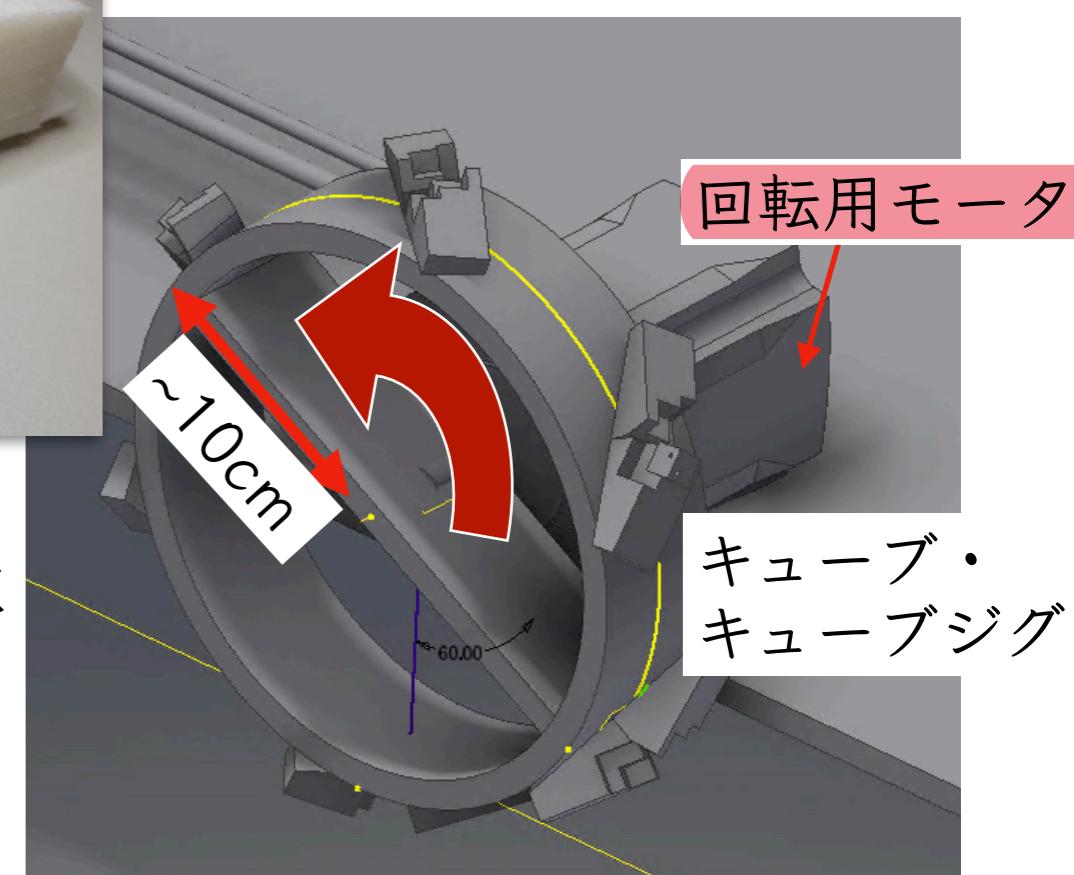
現行の撮影ジグ

新しい撮影ジグ

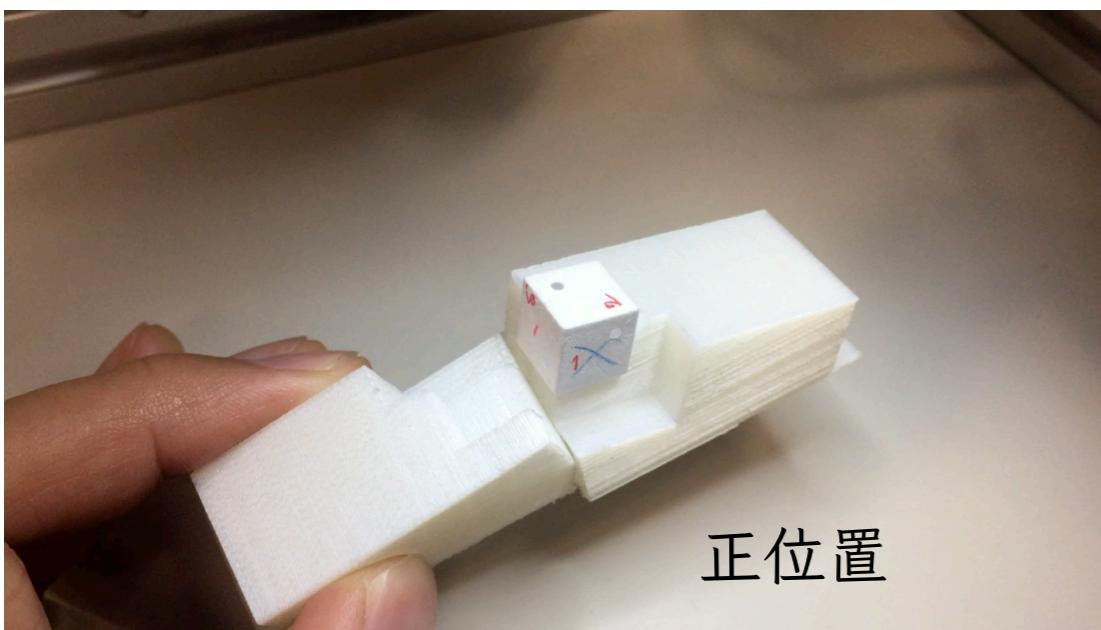
- 2つの台座が向かい合う形。
- カメラを更に3台用意 (合計6台)、残りの三面を別の場所で撮影。
- キューブを転がして向かい側に移動させれば残りの三面が現れる。



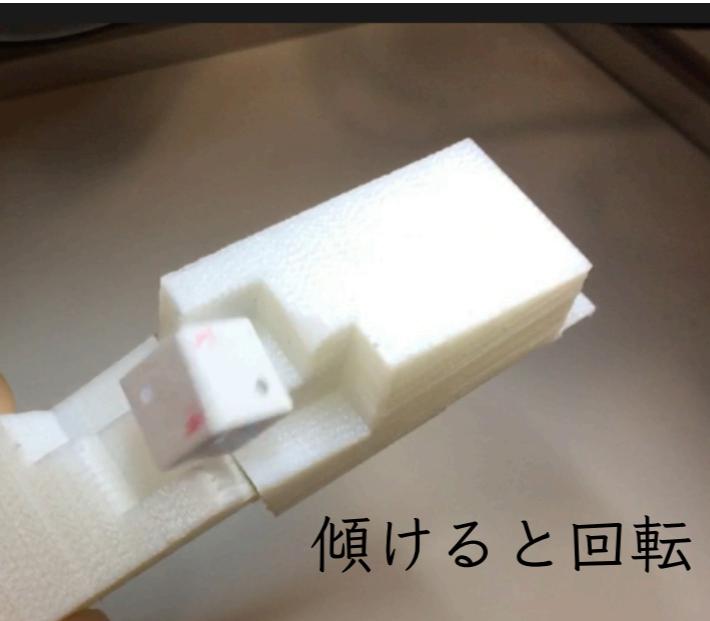
新しいキューブ台座を複数台用意して回転させる。台座を回転させることで、自然にキューブが向かいの台座に移る(次ページ)



回転のようす



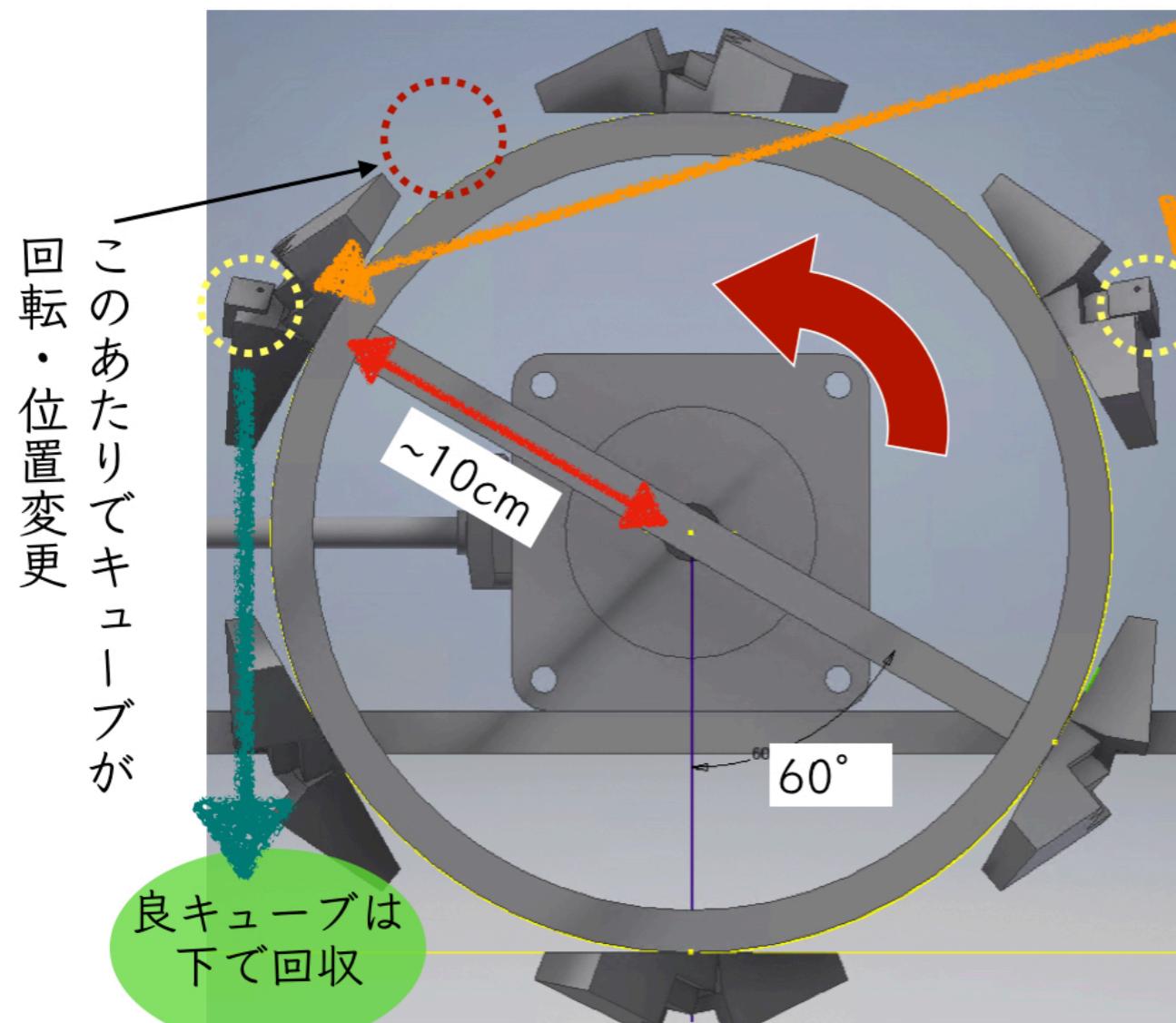
正位置



傾けると回転



回転後



良キューブは
下で回収

撮影 (2箇所)

悪キューブは2度目の撮影点ではじく

良キューブはそのまま下へ落下

→後で箱に詰める

課題：微妙な角度でのカメラの固定方法を考える必要あり

手元の PC 操作で、
回転→静止→撮影→
回転→静止→撮影…
を行いたい

まとめと今後の展望

- T2K 実験の高度化に向けた検出器のアップグレードが進行中
 - 新検出器 Super FGD の素子となる1立方センチメートルシンチレータキューブの品質検査が必要
 - キューブ検査のための撮影ジグの開発、画像解析の手法の開発を行っている
- 12000 個のキューブを用いたプロトタイプ検出器の組み上げ
 - 初めの数百程度のキューブの情報から、各パラメータの分布生成、許容範囲の決定
 - 短時間で試験可能な再現性のあるジグを用いたキューブ撮影・解析を目指す



back up

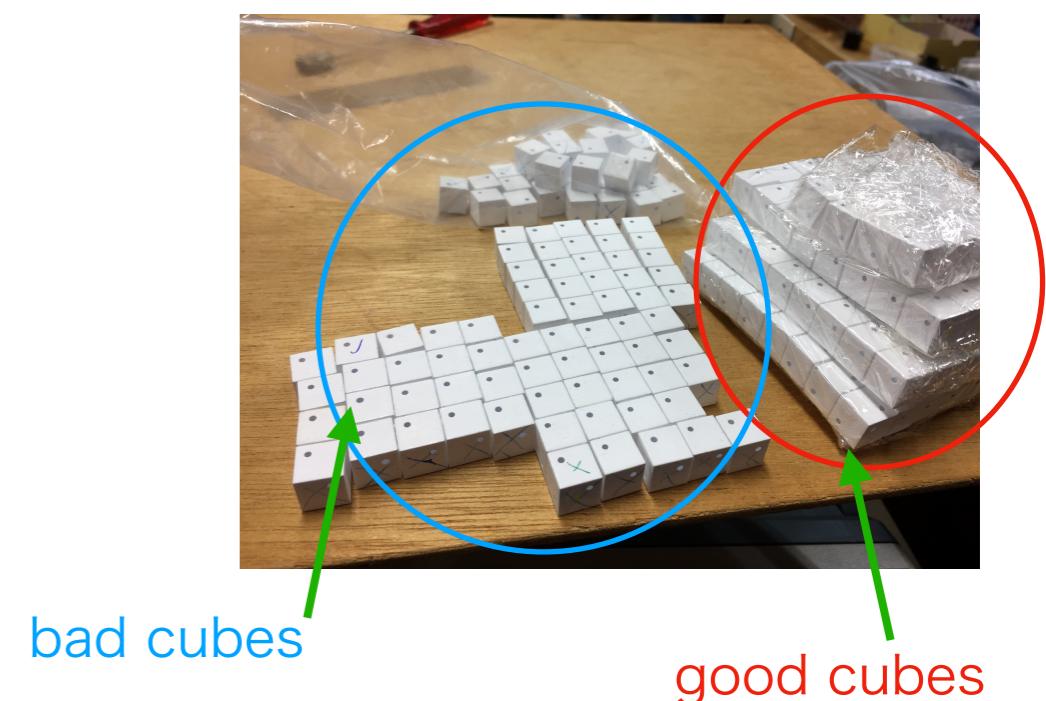
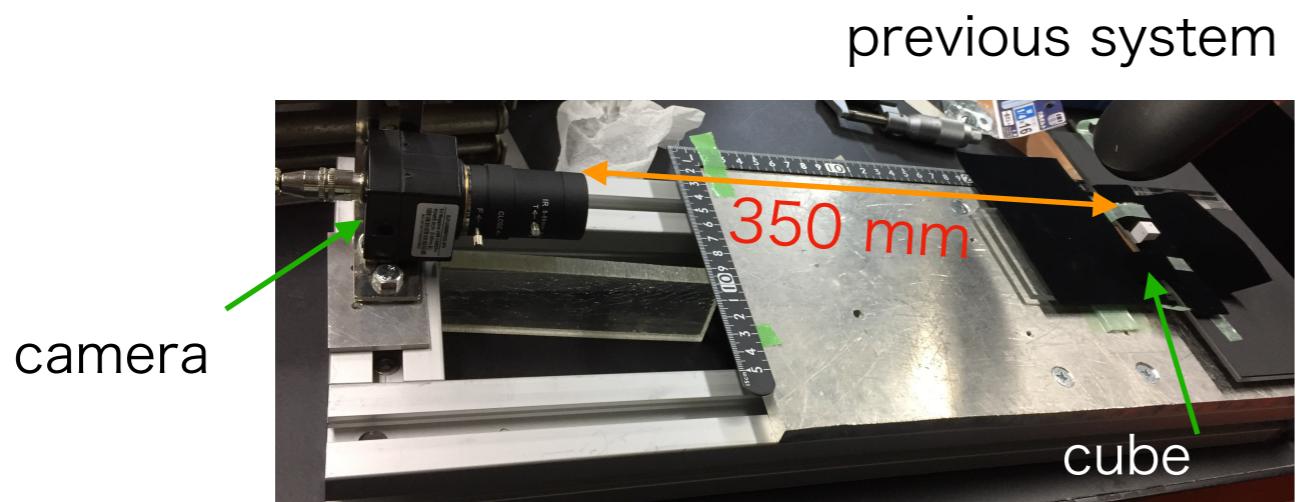
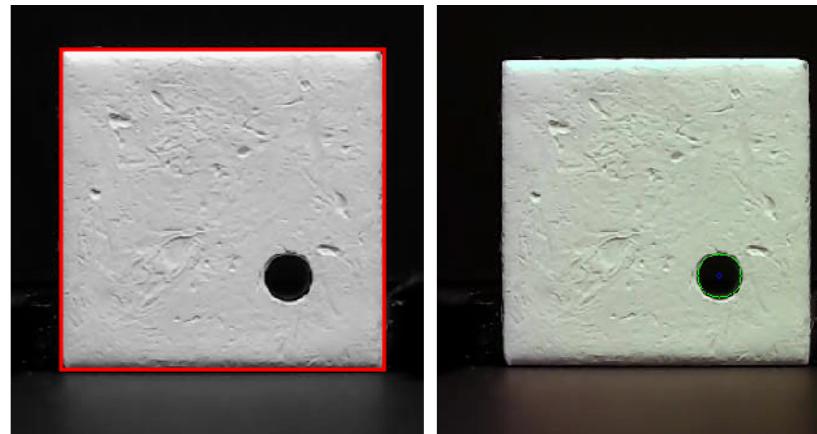
scintillator cubes

image analysis

2019.12.10 Mao Tani

what is my work

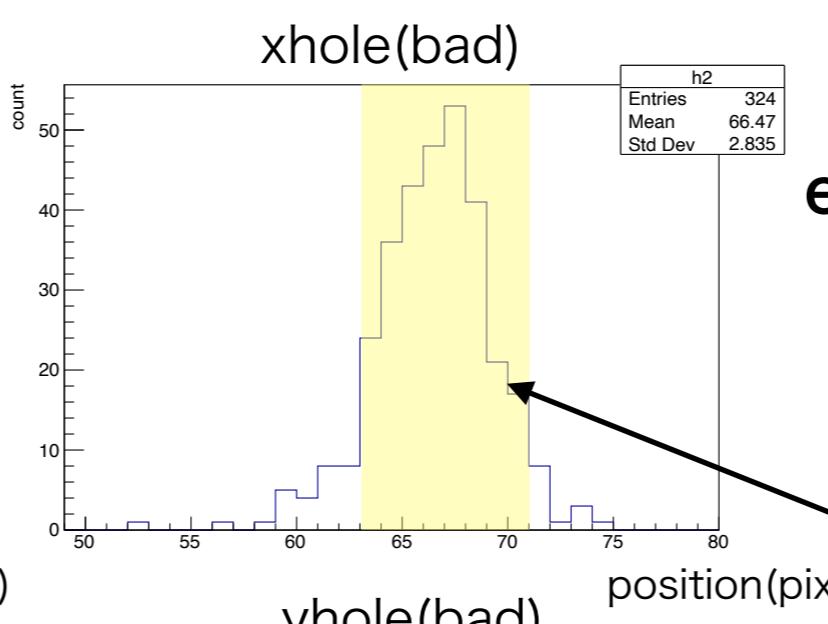
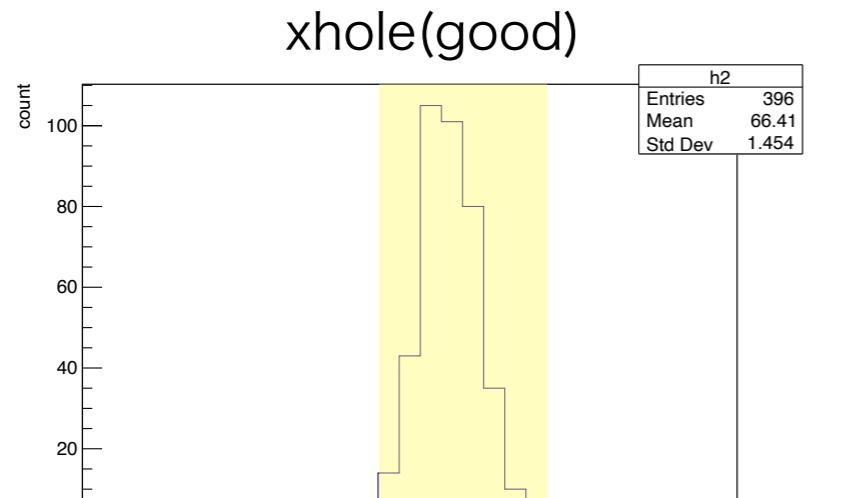
- Quality check of scintillator cubes for super-FGD.
- Take pictures of cubes, then get the information of cube size, position and radius of hole.
- I checked 70 good + 70 bad cubes before (these are already distinguished by manual quality check in Russia).



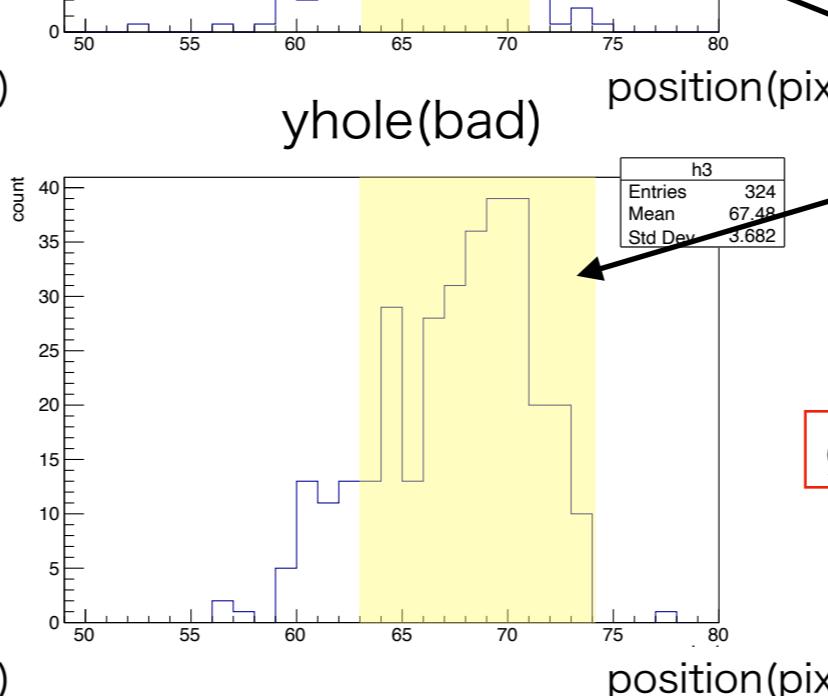
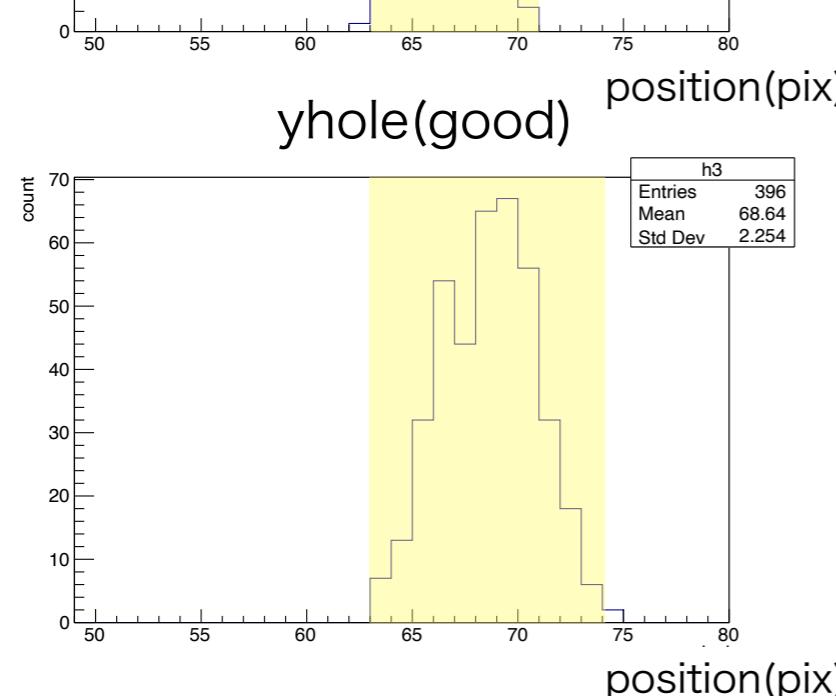
analysis of good / bad cubes

(previous theme)

- each 70 good / bad cubes analysis
 - get distribution about cube size, position and radius of hole
 - cut the bad-cube distribution referring to good cubes' one.



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

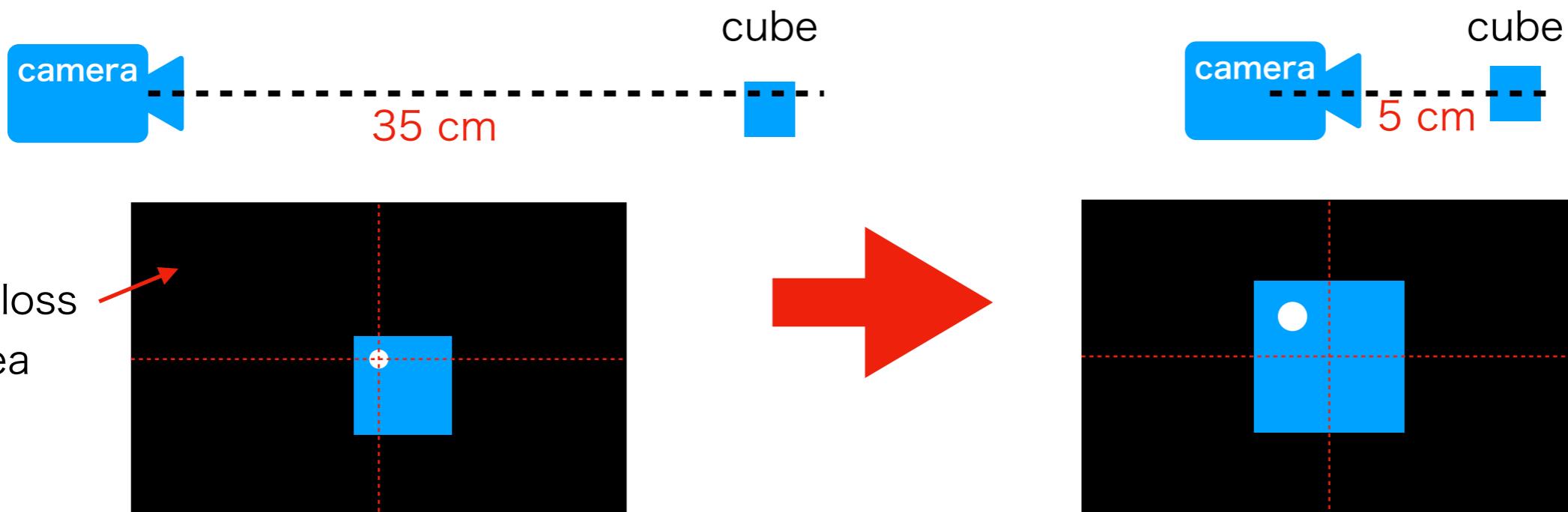
Today's topic

- 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)

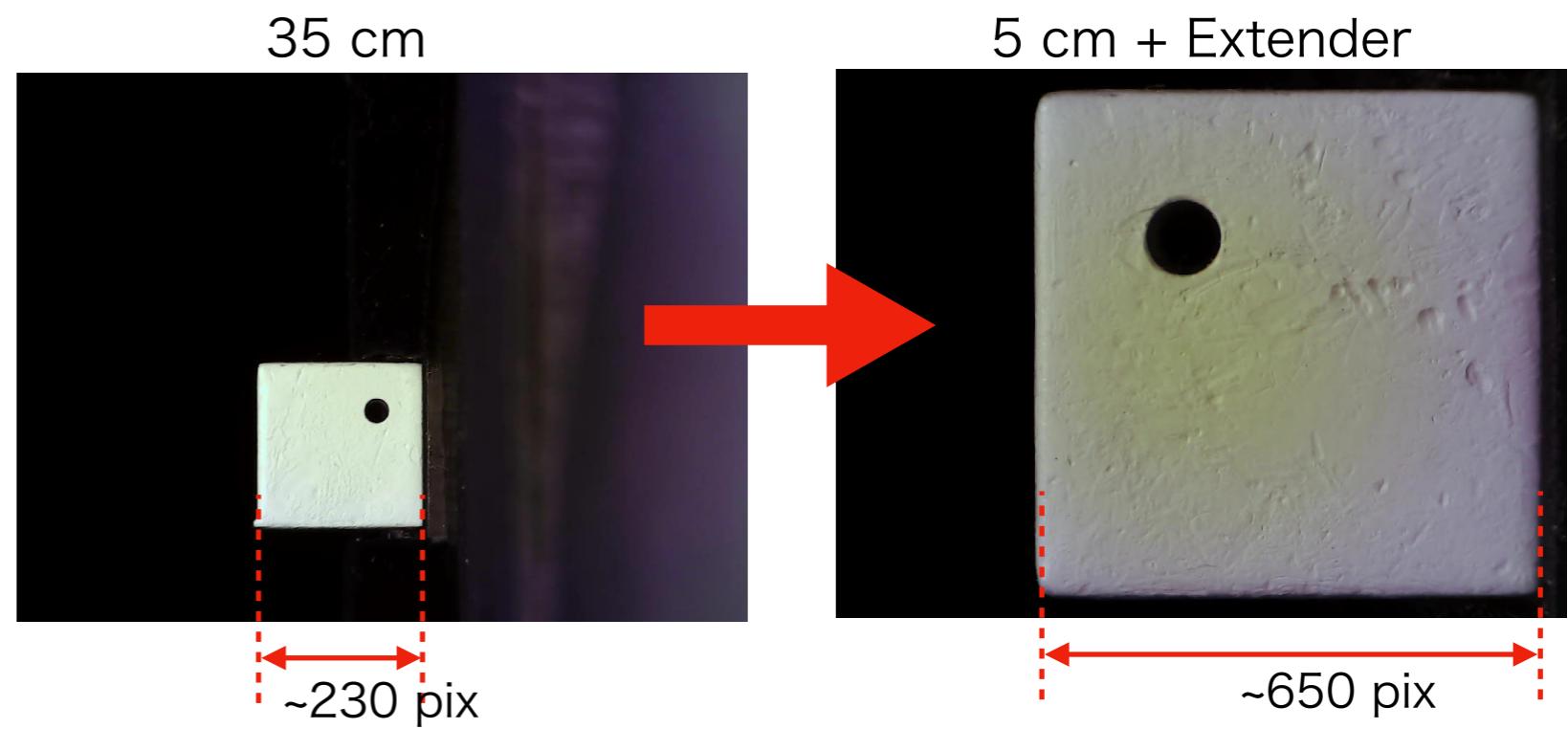
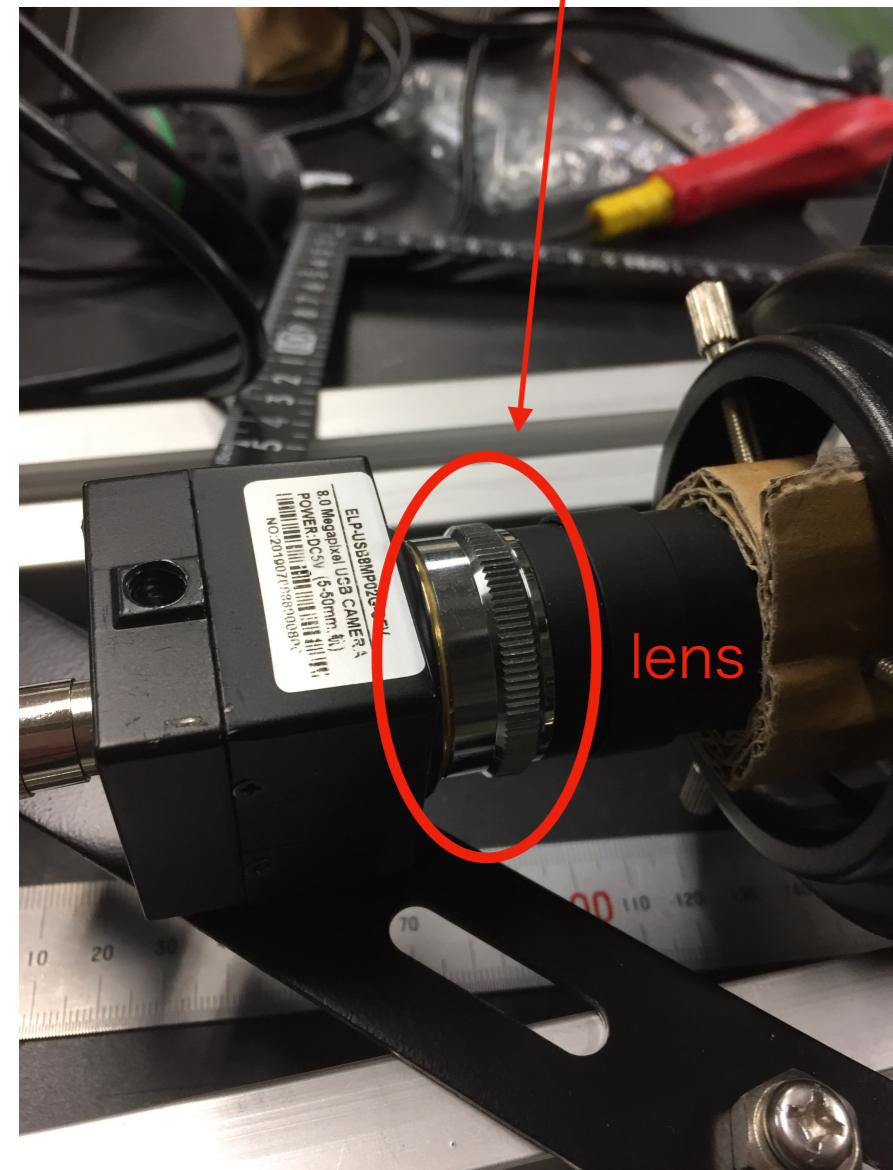


Extender (rear converter)

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

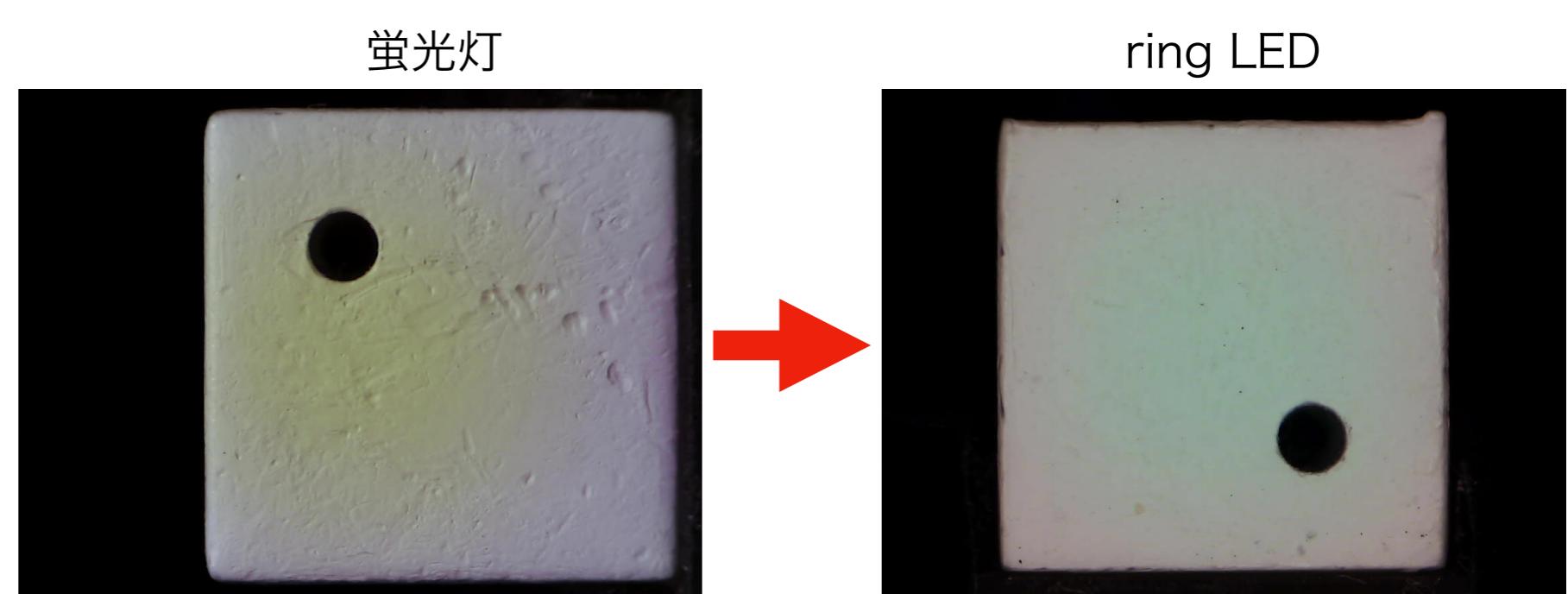
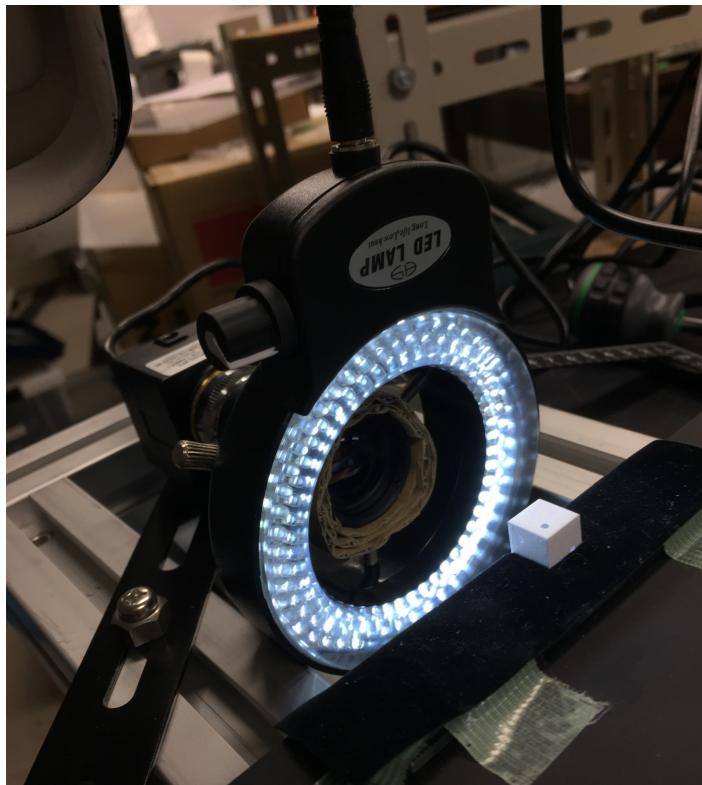


X2 TV EXTENDER JAPAN



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

Hough Circles

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

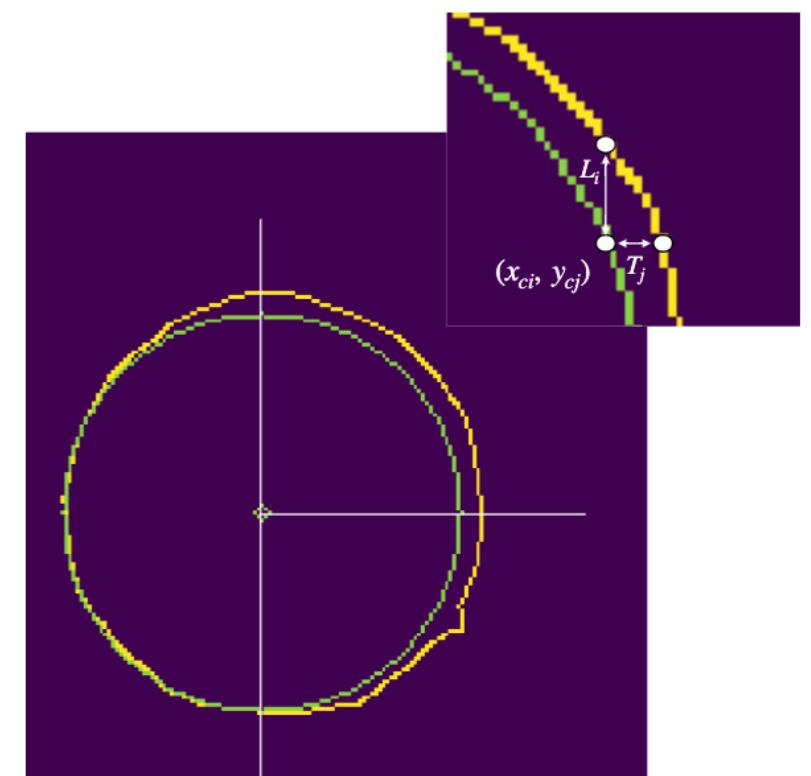
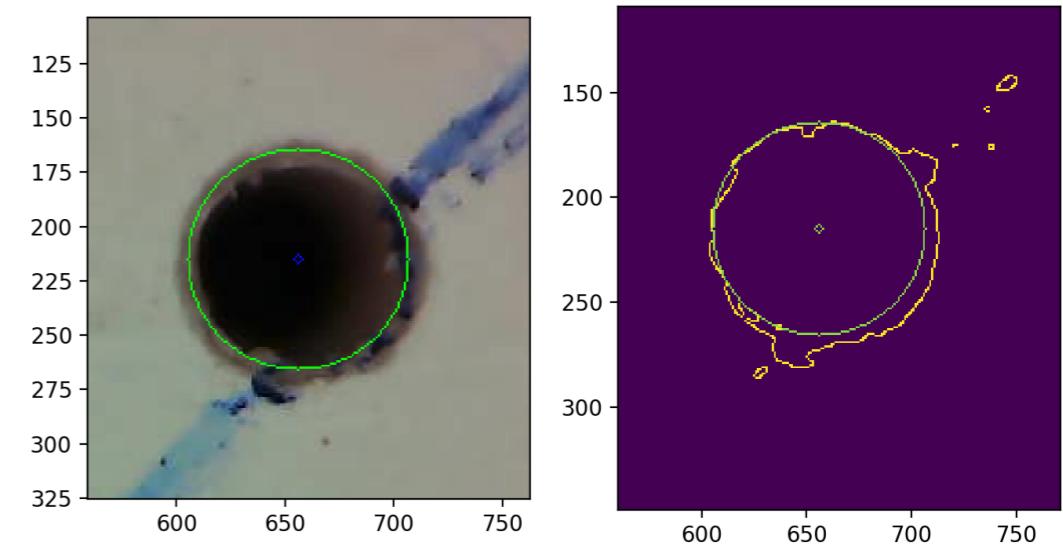
1. binarization (0 or 255)

need to decide threshold value (~100)

2. edge detection

3. automatic circle detection

- Improve the hole-detection accuracy with certain correction.



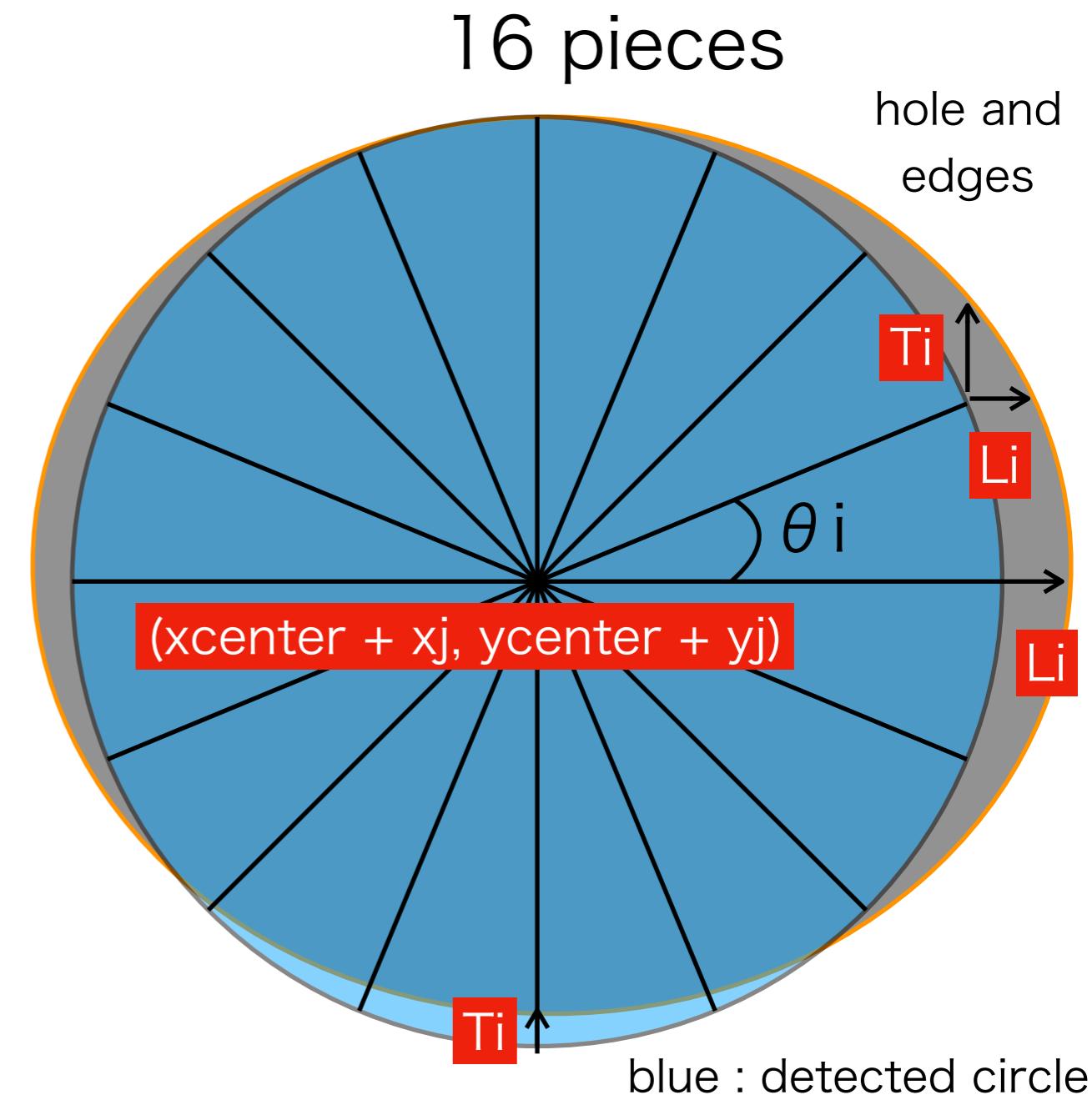
green : detected circle
yellow : edges of hole

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 π , $\theta_i = \pi/2$ 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 $E = \sum (Ti + Li)$.

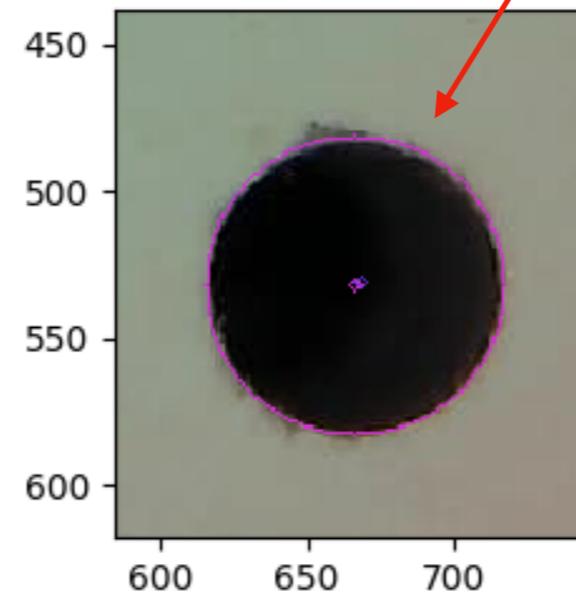
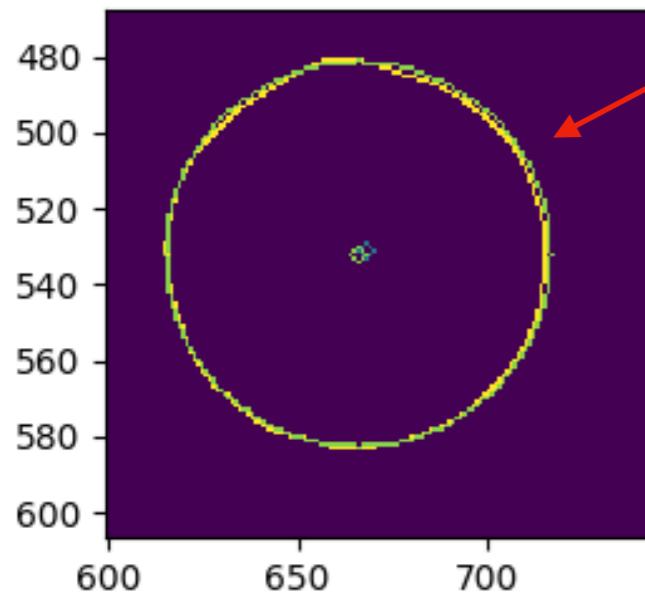
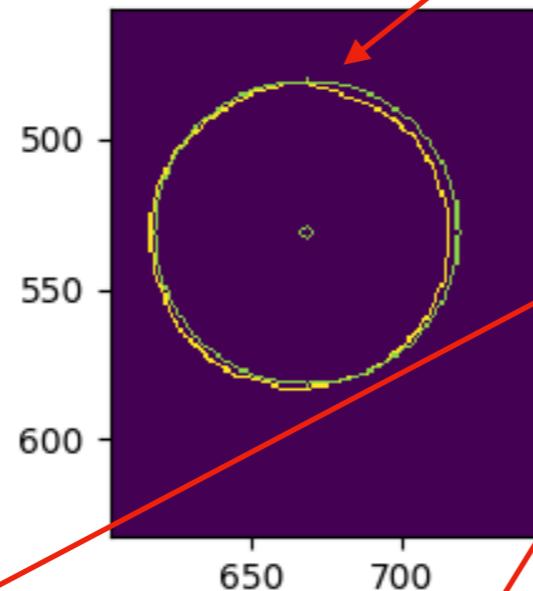
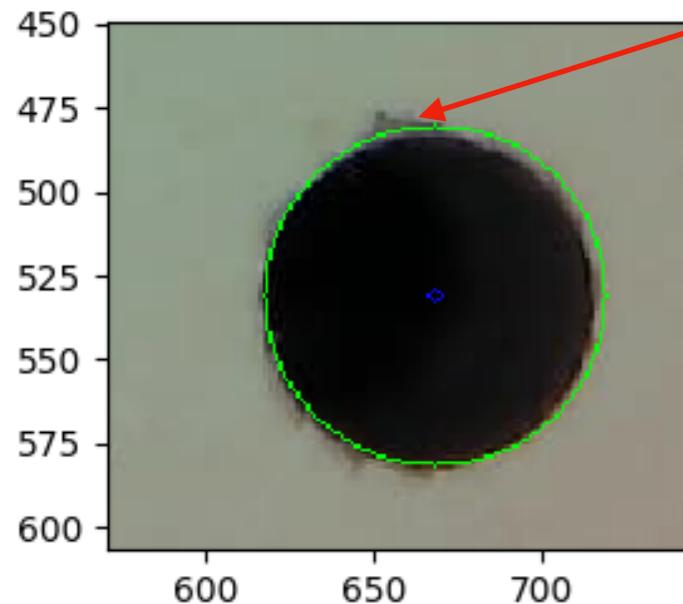
search area : $-5 \leq x_j \leq +5, -5 \leq y_i \leq +5$

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



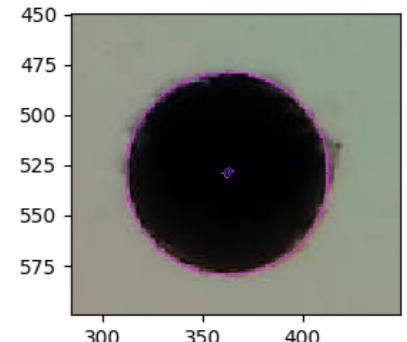
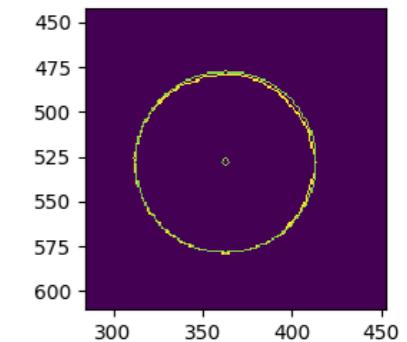
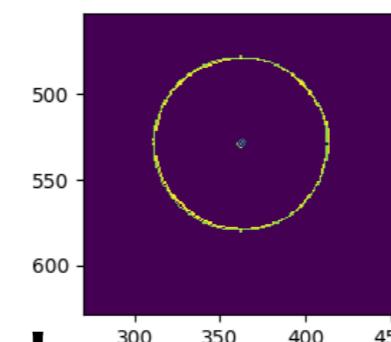
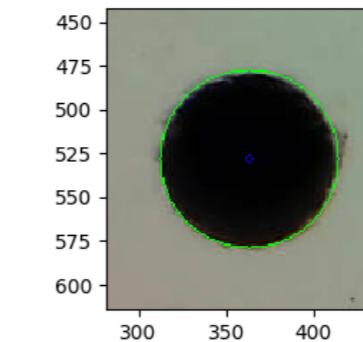
Example of ‘good’ hole

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

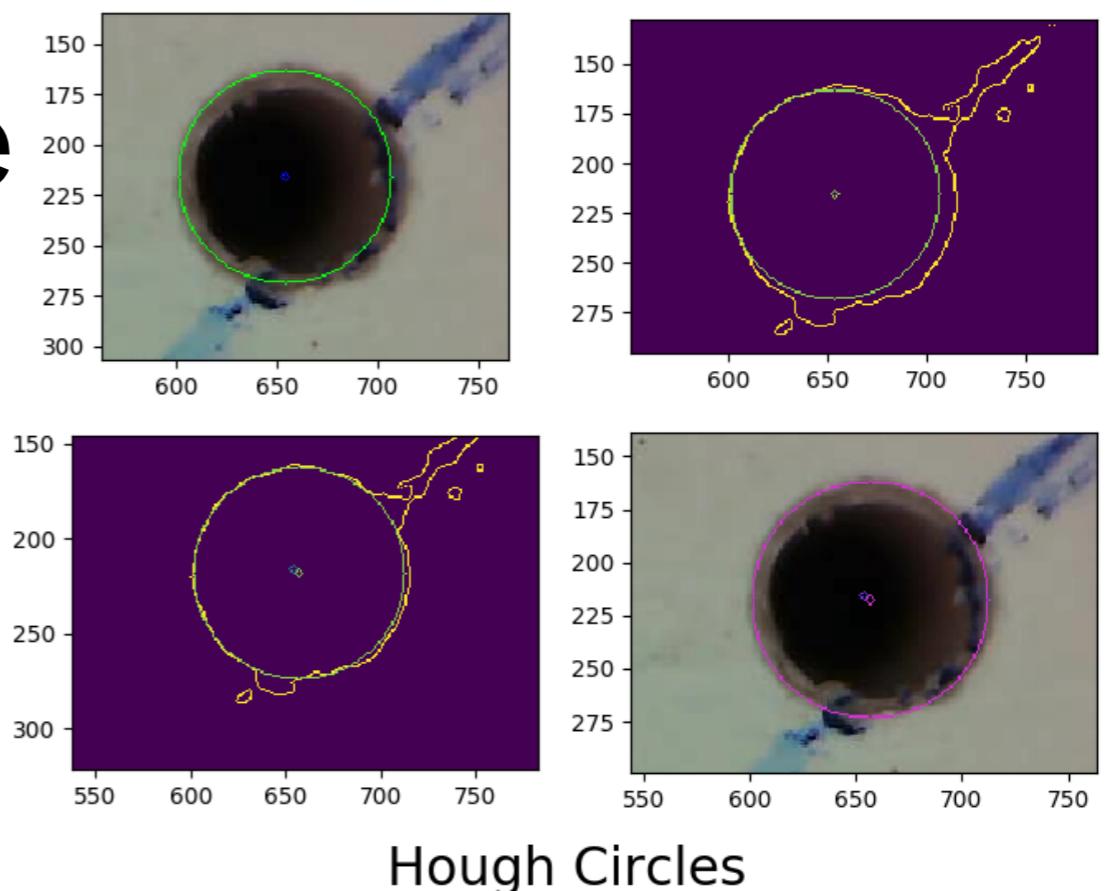
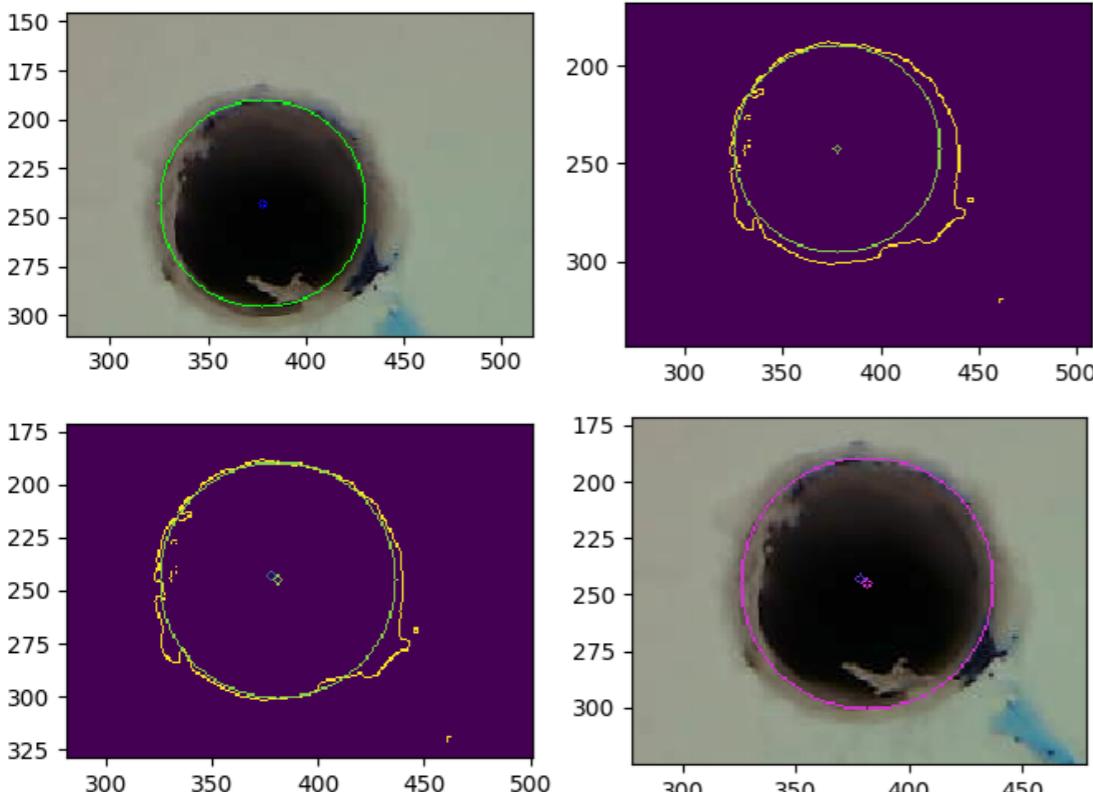


for ‘good’ hole, this corrections succeed

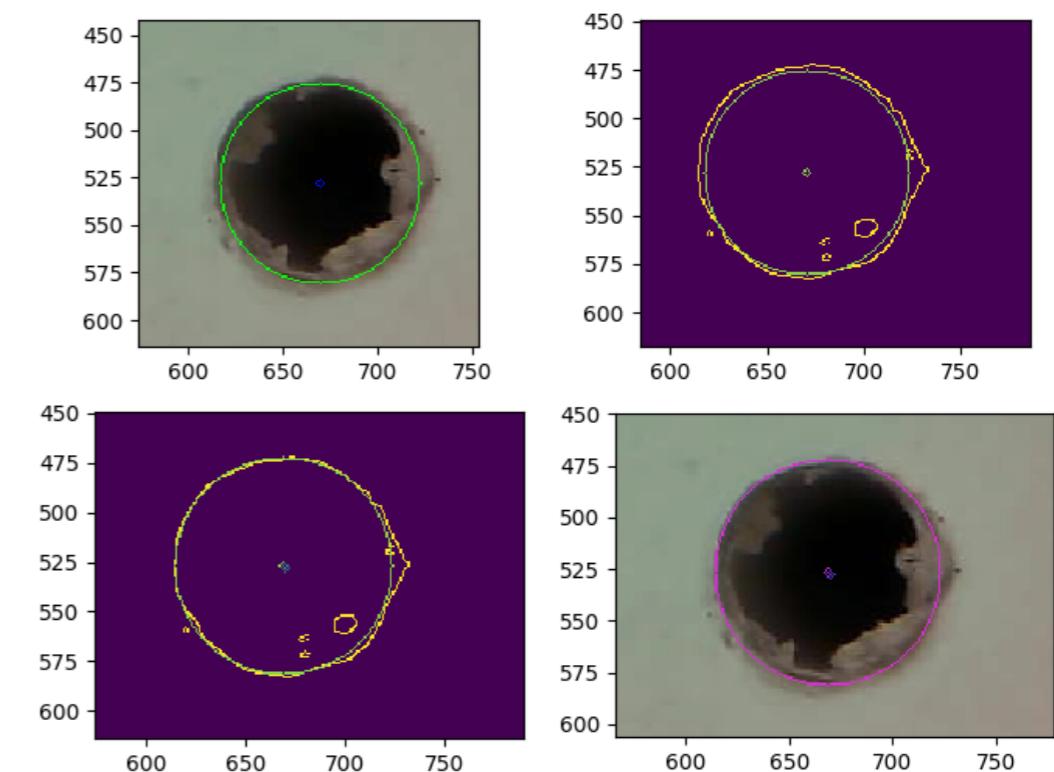
Example of ‘bad’ hole

- 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



Hough Circles

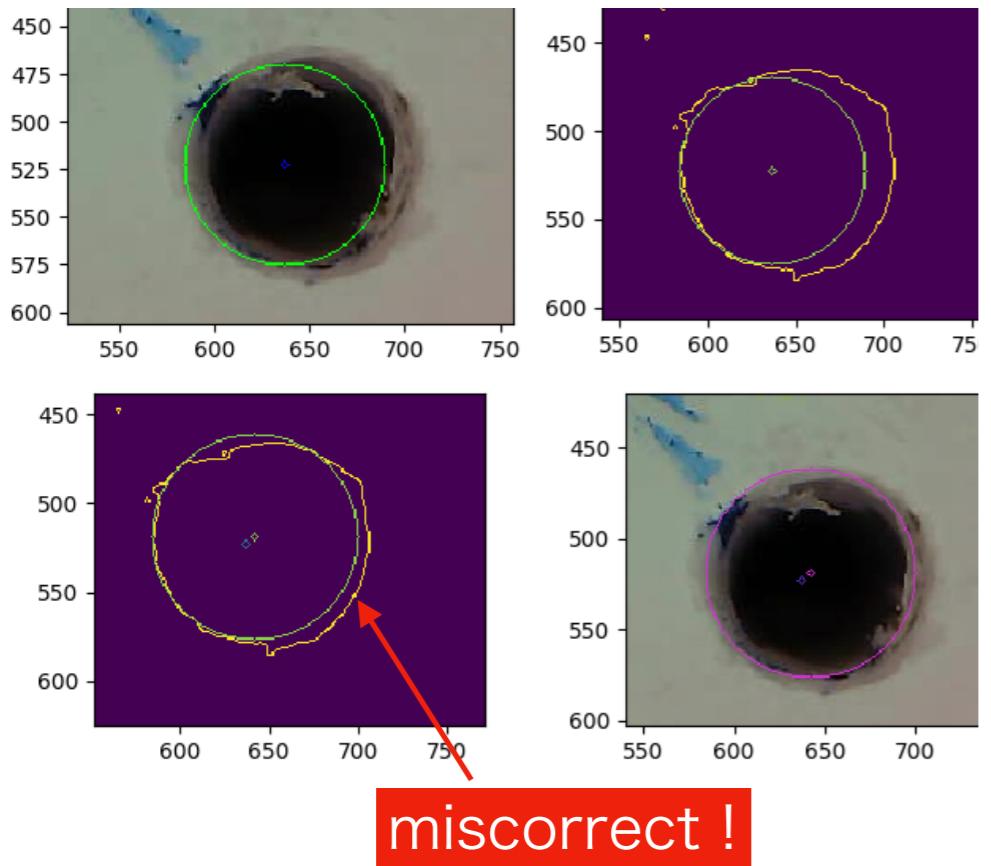


upper row : before correction
bottom row : after correction
(for each image matrix)

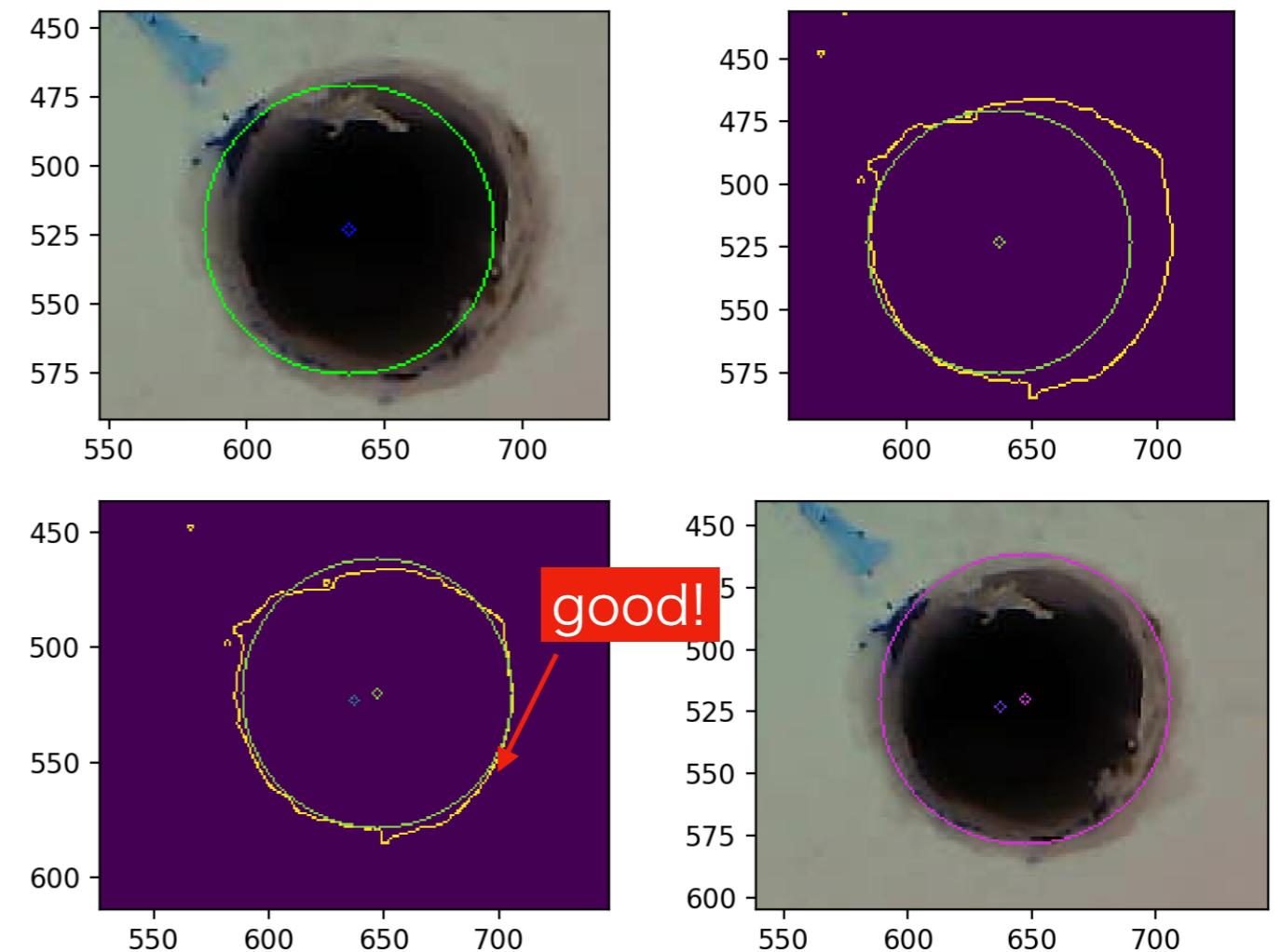
Example that could not get ideal center

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

- enlarged the search area, then succeeded (bottom image)
- center ($x_{\text{center}}+9$, $y_{\text{center}}-4$),
radius $r = r_0+6$



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



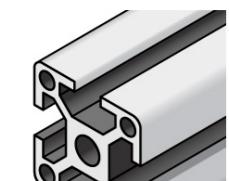
because of increment of scan area it takes more time (~25 sec)

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

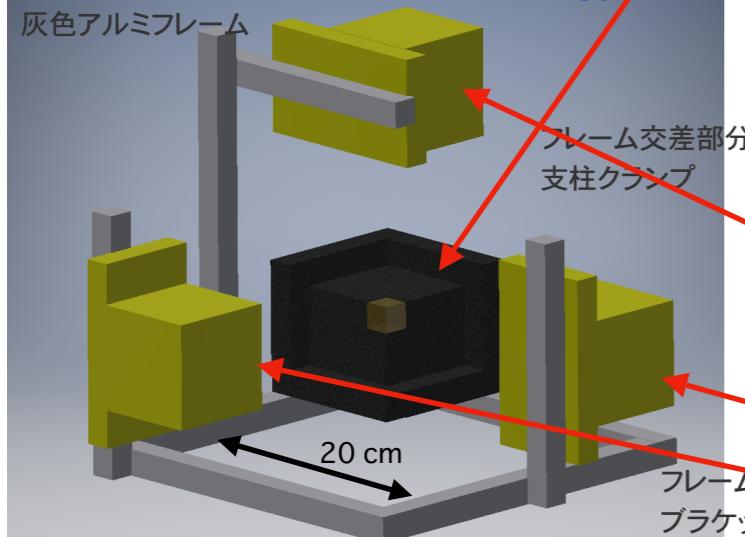


https://jp.misumi-ec.com/vona2/detail/110302683830/?rid=c14_detail_6_110302683830

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

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

https://jp.misumi-ec.com/vona2/detail/110500157910/?clkid=clkid_000025_partslist_21_partsname_11050015

サイズバリエーションが豊富です

https://jp.misumi-ec.com/vona2/detail/110500157910/?clkid=clkid_000025_partslist_21_partsname_11050015

支柱タランプー角・角直交ー ALQOD20

支柱クランプー角・角直交ー 3300円 x 4



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

- order photography platform and camera box

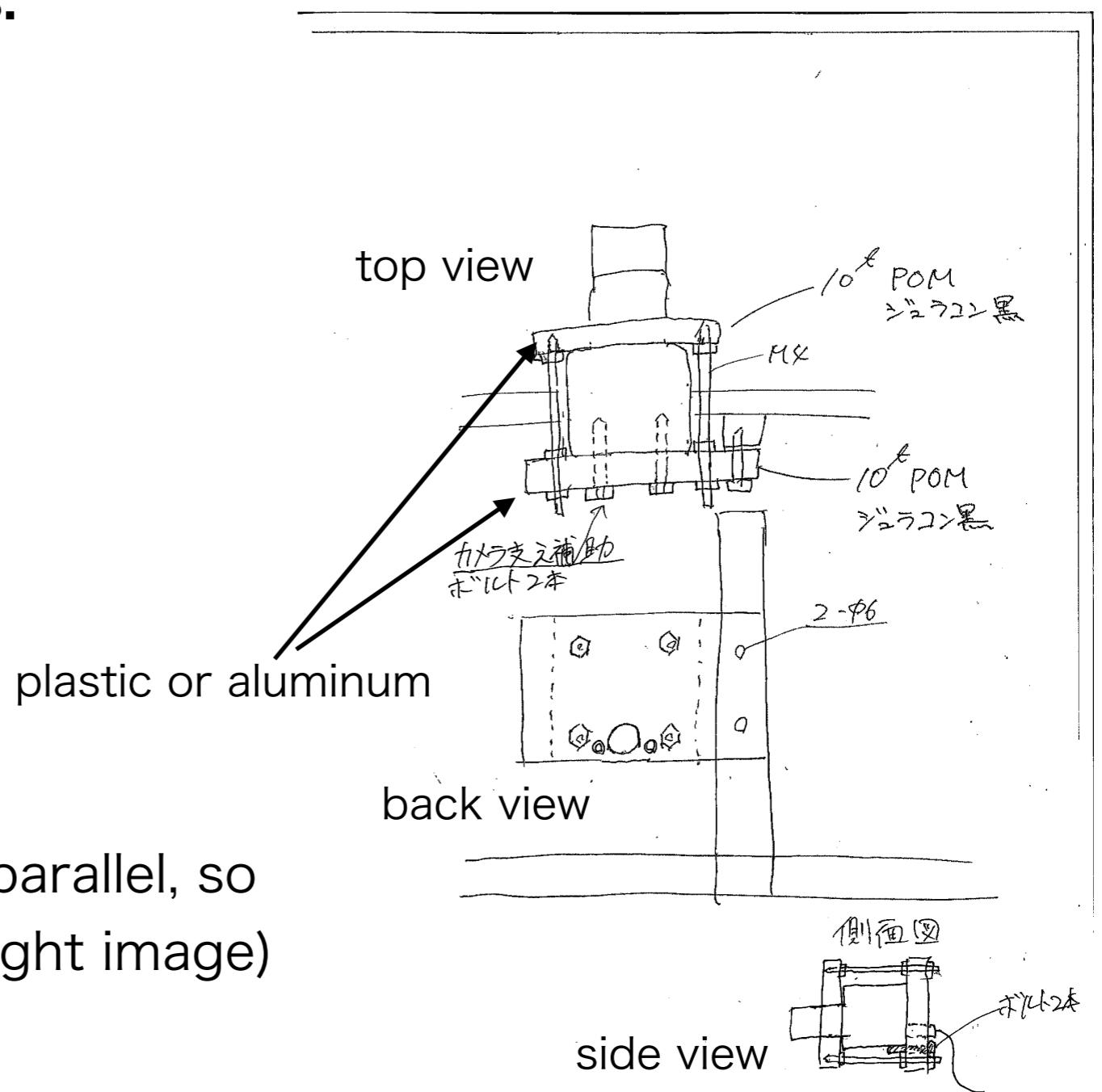
camera box

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

カイニ乗値の最小化

2020.01.15 谷

カイ二乗値の最小化

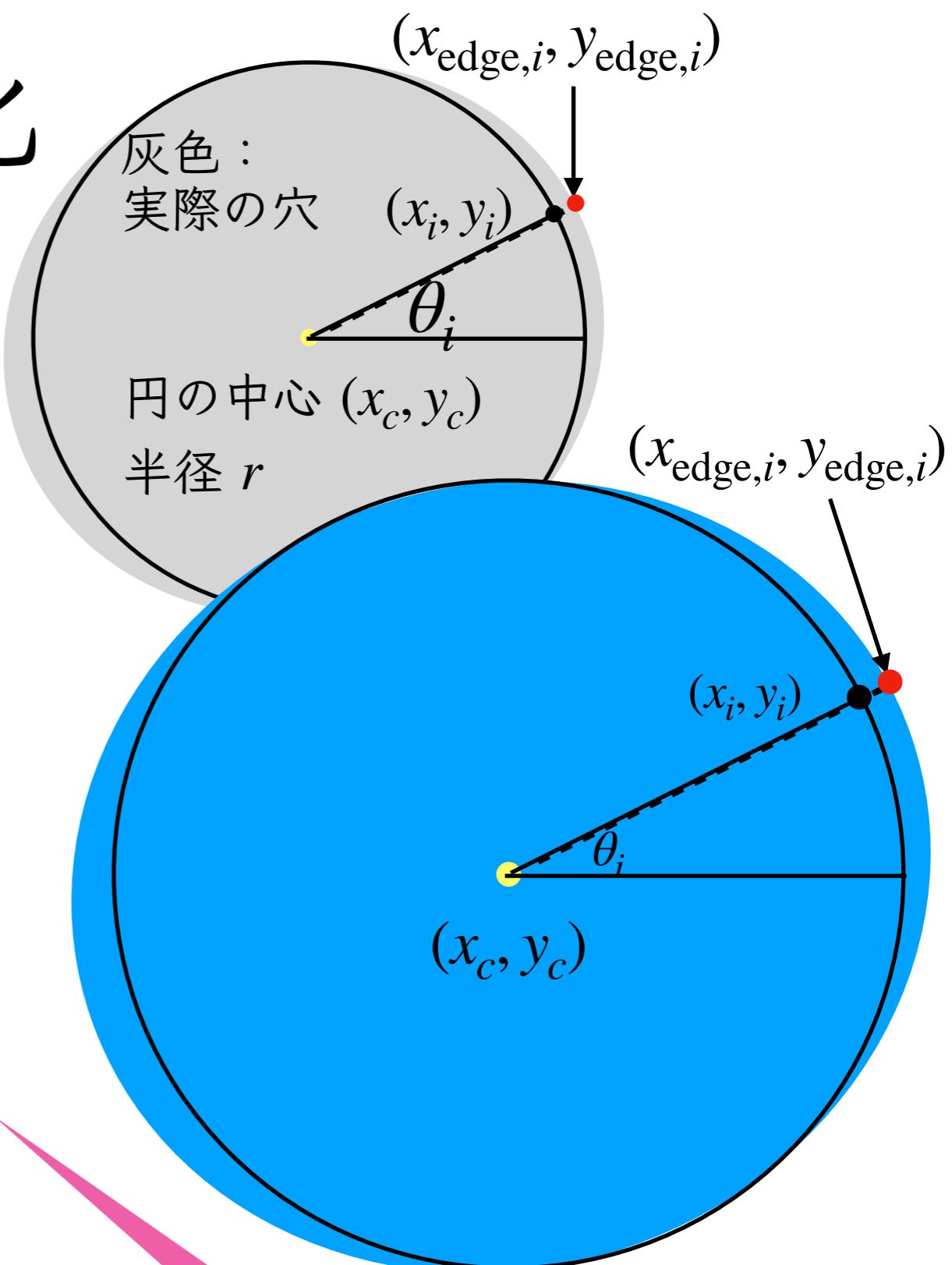
$$E_{\text{sum}}(x_c, y_c, r) = \sum_i \left(|x_i - x_{\text{edge},i}|^2 + |y_i - y_{\text{edge},i}|^2 \right)$$

$$x_i = x_c + r \cos \theta_i, y_i = y_c - r \sin \theta_i$$

E_{sum} が最小となる (x_c, y_c, r) を求める。

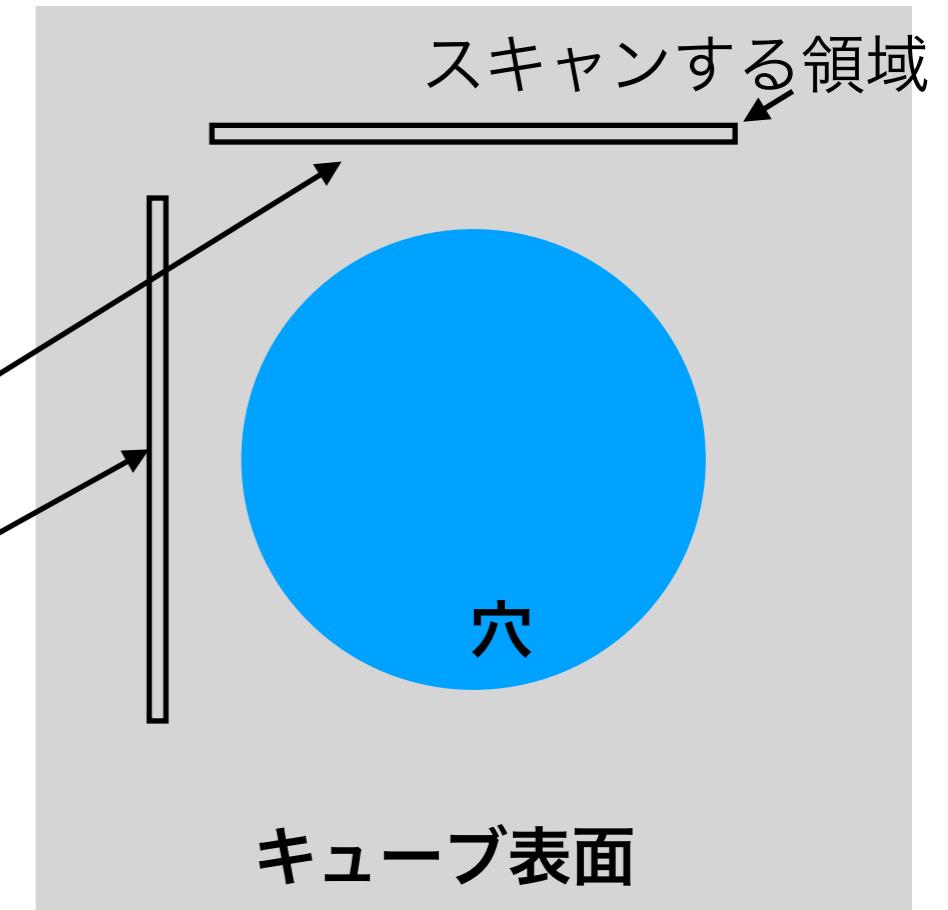
- (x_c, y_c) は円の中心、 r は円の半径
- $x_{\text{edge}}, y_{\text{edge}}$ は穴のエッジ上の点
- (x_c, y_c, r) の初期値は二値化画像から得た中央値を使う。
- 今回はエッジ上の16点について和を

とった ($\theta_i = 0, \frac{\pi}{8}, \frac{\pi}{4}, \dots, \frac{15}{8}\pi$)。



画像の二値化

- 二値化のための threshold の決め方
- 穴の周囲の色の平均を参照値とする。
- 参照値の候補をふたつ (orそれ以上) 用意する
 - x方向にスキャンした平均
 - y方向にスキャンした平均
- キューブ表面の傷・印等の影響を減らすため、候補のうち、最も白いもののを参照値として採用する。



今回は以前提案していただいたように、参照値の 30% カットの値を threshold として用いた

最小化の手順

ある $\mathbf{x} = (x, y, r)$ での Esum の勾配

$$\nabla E = \left(\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}, \frac{\partial E}{\partial r} \right)$$

を求め、設定した learning rate η をかけて (x, y, r) から引き更新する。

$$\mathbf{x}' = \mathbf{x} - \eta \nabla E$$

この操作を指定ステップ数だけ繰り返す。

徐々に最小化されている様子がわかる
(右の出力参照)。

時間もそれほどかかっていない。

```
learning_rate = 0.01, steps = 50
initial_position = [356.099998474, 244.099998474, 51.4]
はじめのEsumの値:294.19271881806037

gradient[0]=[ 3.19995117  17.19995117 -127.88263133]
Esum[0]=154.2483374368868

gradient[1]=[ 2.1759668  11.6959668 -86.9601893]
Esum[1]=89.53805548782361

gradient[2]=[ 1.47965742  7.95325742 -59.13292872]
Esum[2]=59.61602111522834

gradient[3]=[ 1.00616705  5.40821505 -40.21039153]
Esum[3]=45.78007242172495

gradient[4]=[ 0.68419359  3.67758623 -27.34306624]
Esum[4]=39.38232974596808

gradient[5]=[ 0.46525164  2.50075864 -18.59328504]
Esum[5]=36.42401353280427

.
.

.
.

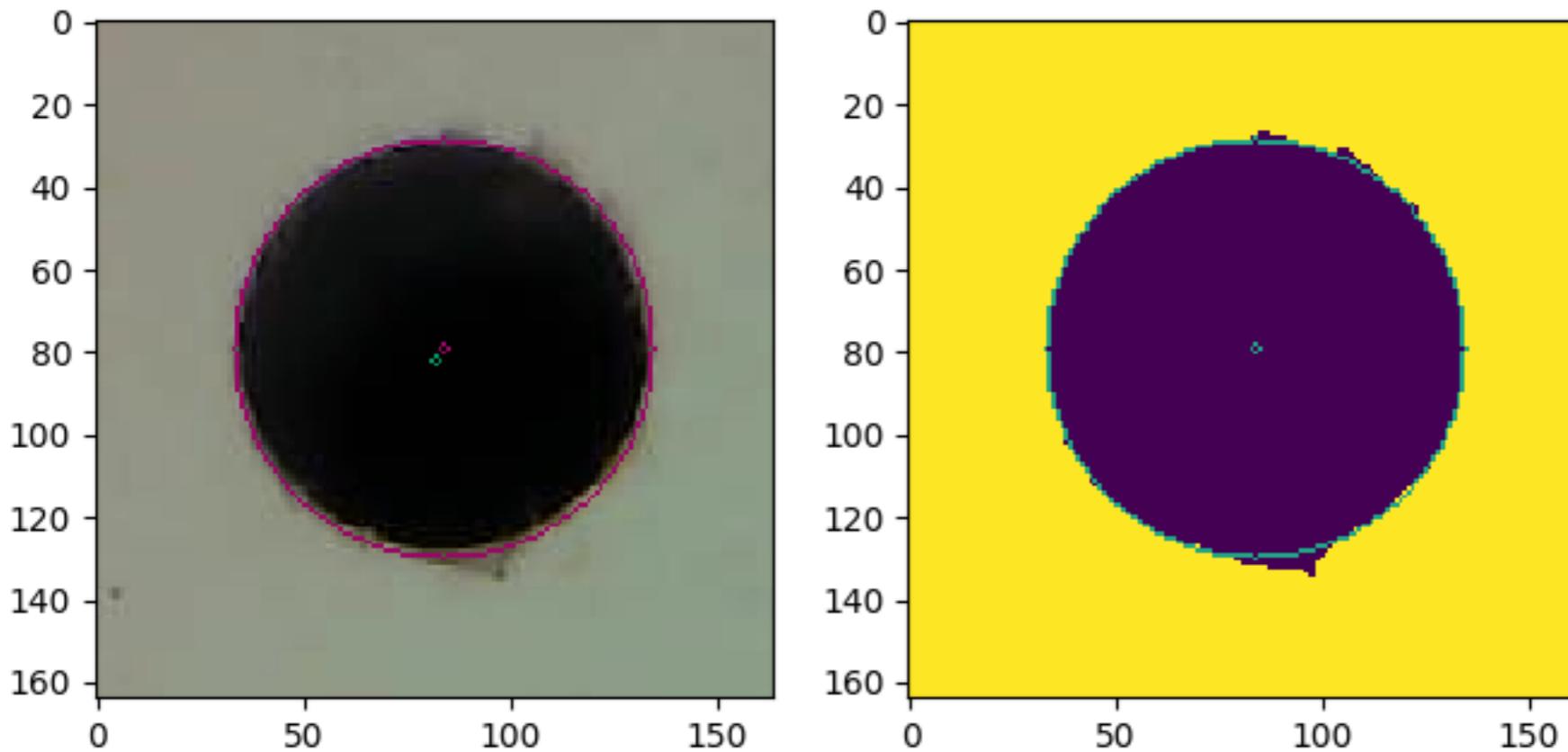
gradient[47]=[ 4.295230e-08  2.309263e-07 -1.716955e-06]
Esum[47]=33.87950940915731

gradient[48]=[ 2.923883e-08  1.570299e-07 -1.167386e-06]
Esum[48]=33.879509409157265

gradient[49]=[ 1.982414e-08  1.067590e-07 -7.938183e-07]
Esum[49]=33.87950940915729
勾配法が見つけた最小値でのx: [356.          243.5625   55.396332]
time:0.057205915451049805 sec
Esumの最小値:33.87950940915729
```

実行例

good の例：円は初期 (x, y, r)によるもの



```
learning_rate = 0.01, steps = 50
initial_position = [364.0999984741211, 230.0999984741211, 52.4]
はじめのEsumの値:50.57412827491975
```

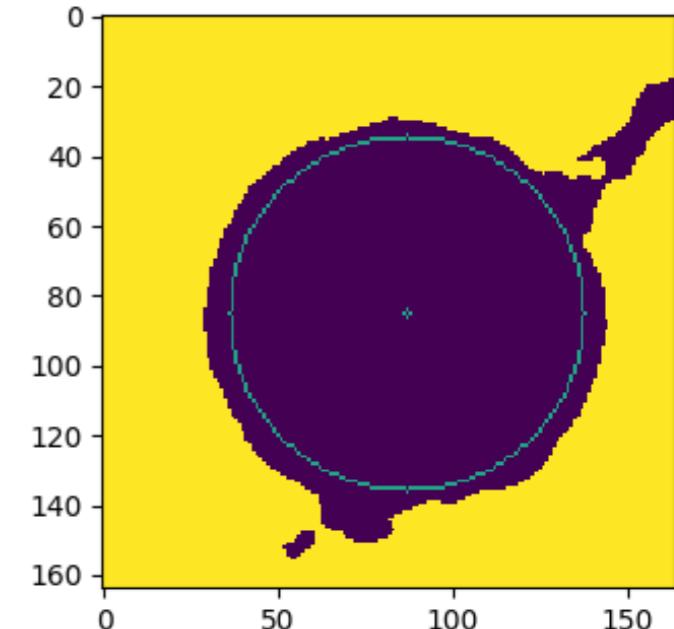
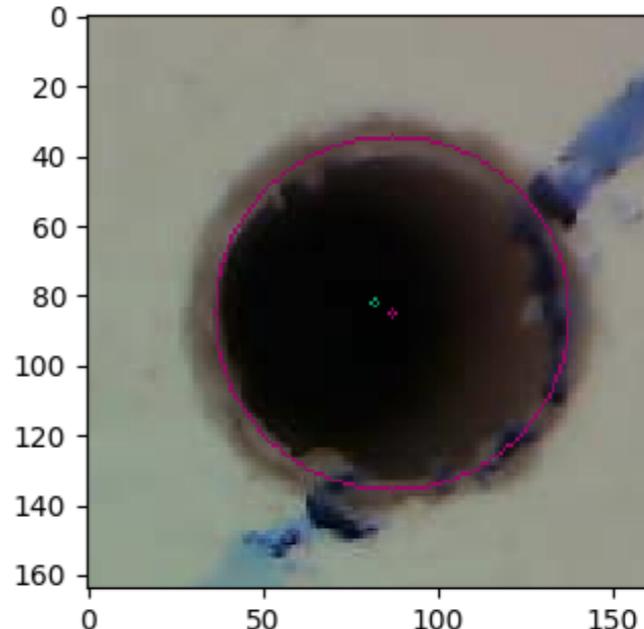
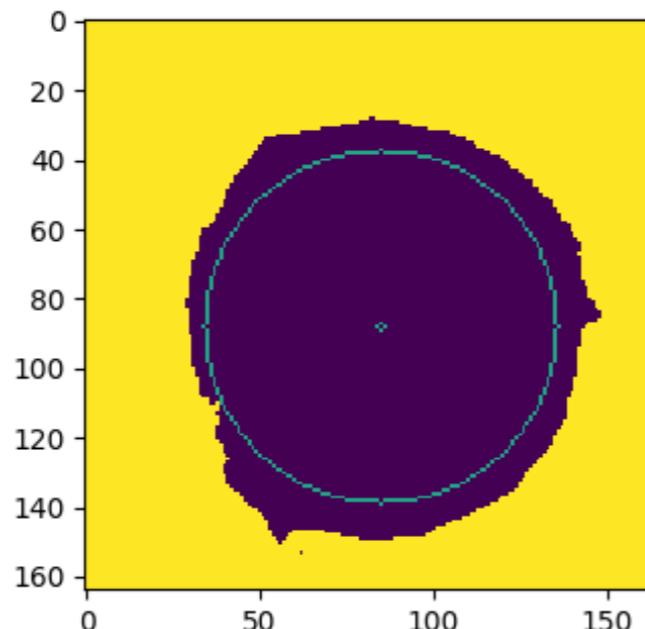
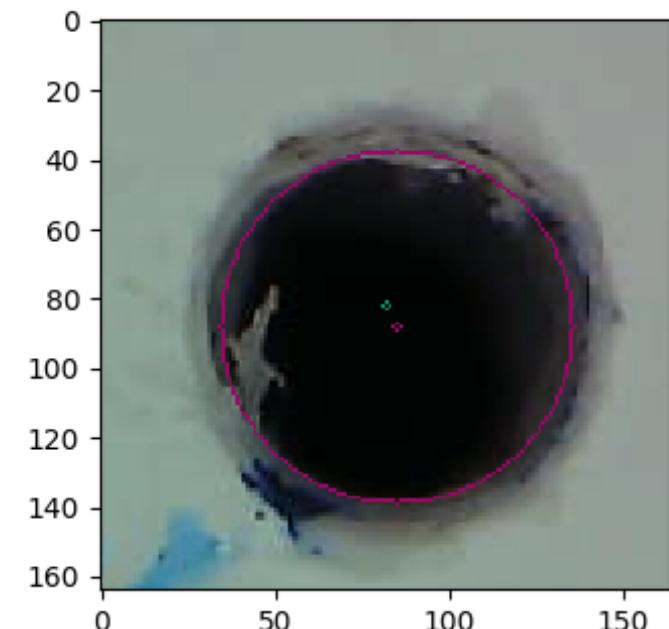
```
勾配法が見つけた最小値でのx: [363.8125      229.375      51.08272056]
```

```
time:0.04395008087158203 sec
```

```
Esumの最小値:13.078075424774163
```

(x, y, r):(364.1, 230.1, 52.4)→(363.8, 229.4, 51.1)
Esum : 50.5 → 13.1

bad の例：円は初期 (x , y , r)によるもの



```
learning_rate = 0.01, steps = 50  
initial_position = [652.099998474, 254.099998474, 52.4]  
はじめのEsumの値:626.501403830376  
勾配法が見つけた最小値でのx: [652.0625 253.8125 58.09540] 勾配法が見つけた最小値: [658.1875 217.625 57.347693]  
time:0.04877614974975586 sec  
Esumの最小値:106.15383203261482
```

$(x, y, r):(652.1, 254.1, 52.4)$
 $\rightarrow(652.1, 253.8, 58.1)$

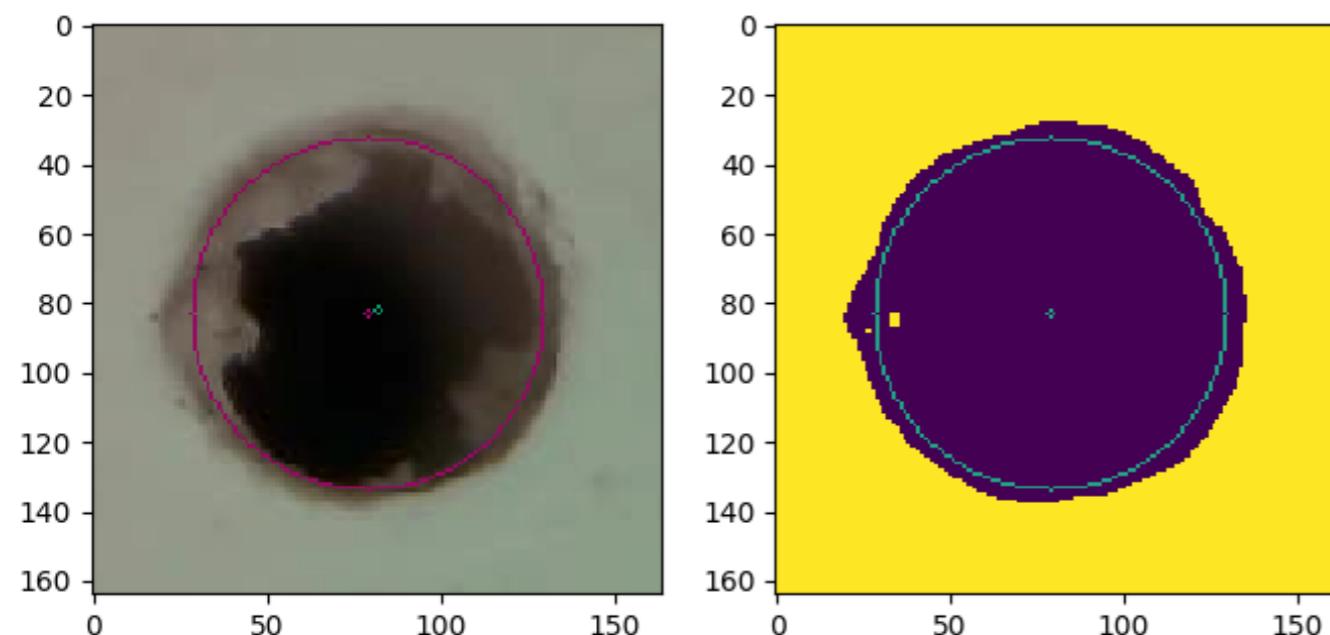
Esum : 626.5 → 106.1

```
learning_rate = 0.01, steps = 50  
initial_position = [659.09999847, 217.09999847, 52.4]  
はじめのEsumの値:651.4675248444642  
勾配法が見つけた最小値: [658.1875 217.625 57.347693]  
time:0.044788360595703125 sec  
最小値でのEsumの値:242.06030867727364
```

$(x, y, r):(659.1, 217.1, 52.4)$
 $\rightarrow(658.2, 217.6, 57.3)$

Esum : 651.5 → 242.1
(右上の印に引っ張られている?)

bad の例：円は初期 (x , y , r)によるもの



```
learning_rate = 0.01, steps = 50
initial_position = [356.09999847, 244.09999847, 52.4]
はじめのEsumの値:182.3100874940509
勾配法が見つけた最小値: [356.      243.5625  55.396332]
time:0.04555487632751465 sec
最小値でのEsumの値:33.87950940915727
```

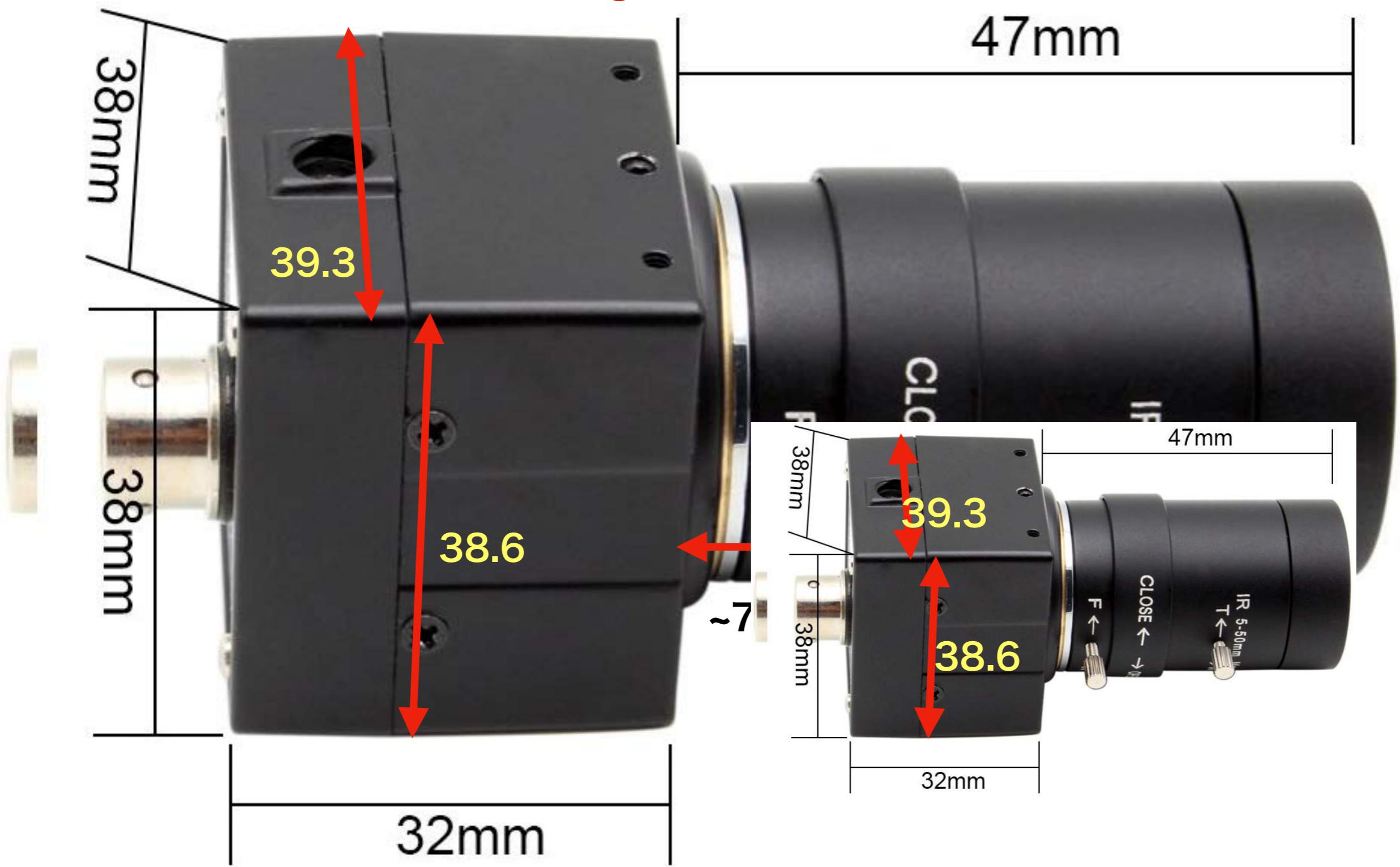
(x, y, r) : $(356.1, 244.1, 52.4)$

$\rightarrow(356.0, 243.6, 55.4)$

Esum : 182.3 \rightarrow 33.9

- どれも中心(x, y)については最小化してもほぼ変わらず→中央値作戦はうまくいっている
- r の変化は大きい
 - 二値化の threshold によって期待される r の値は変わる。
- 異なる threshold でこの操作を行い、Esum の最小値が最も小さくなるものをその画像での threshold とする？（未実行）

カメラの側面、上面、底面がそれぞれ平らではなく、
この線の辺りが出っ張っている



superFGD シンチレータキューブ
撮影時の傾き補正、
撮影ジグ改善案

2020/3/10 京都大学 谷
collaborate with 小川さん (KEK)

Introduction

- 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

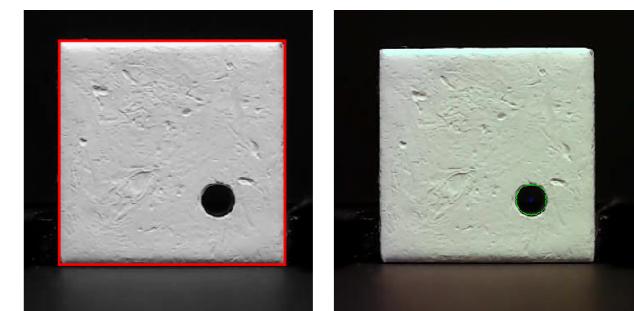
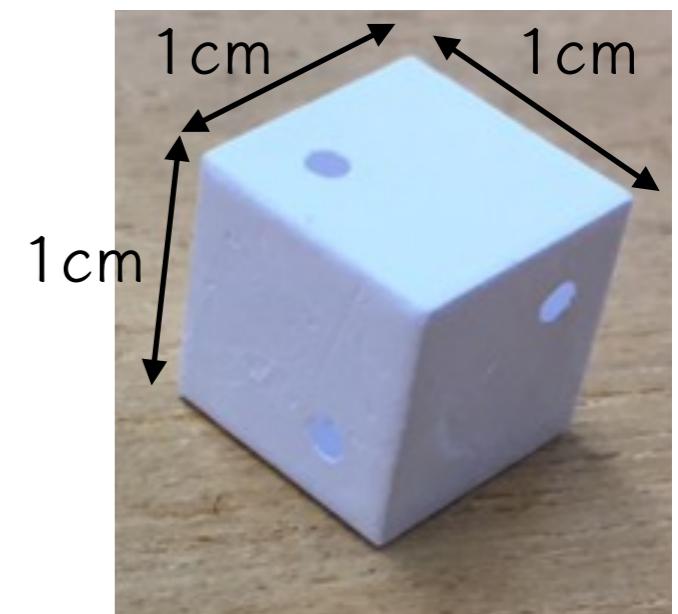
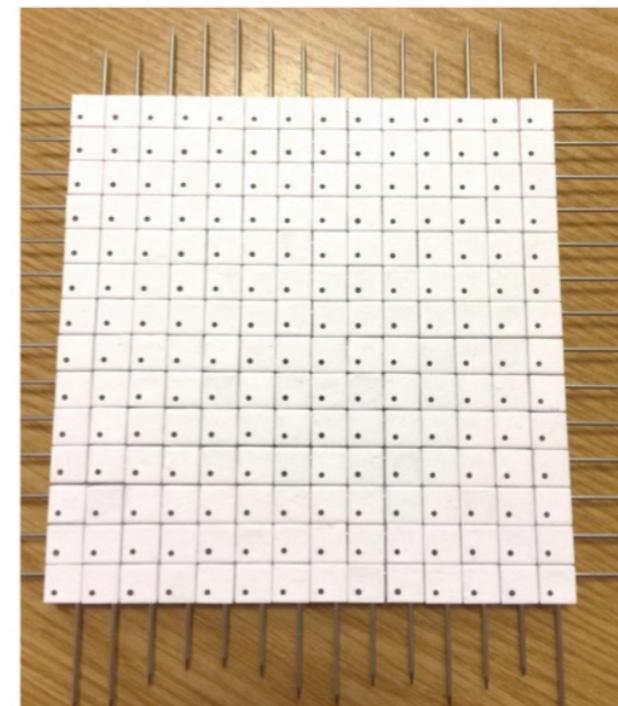
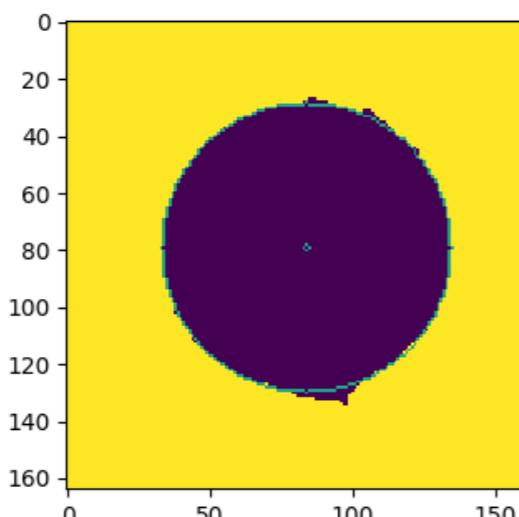
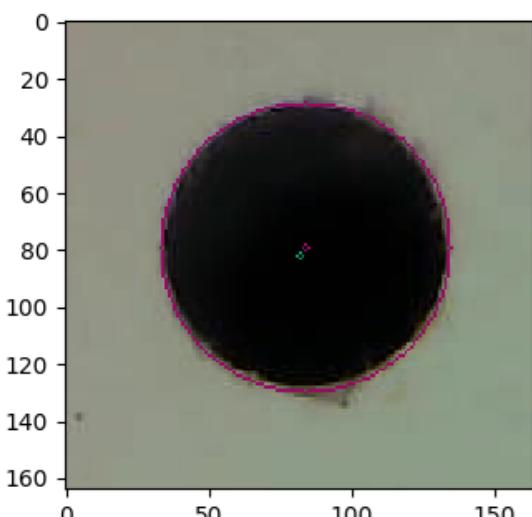
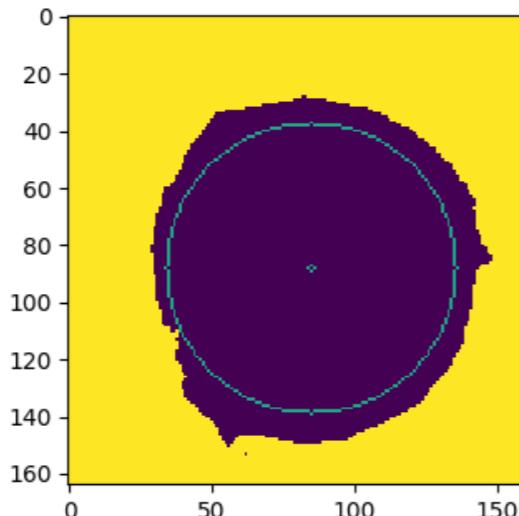
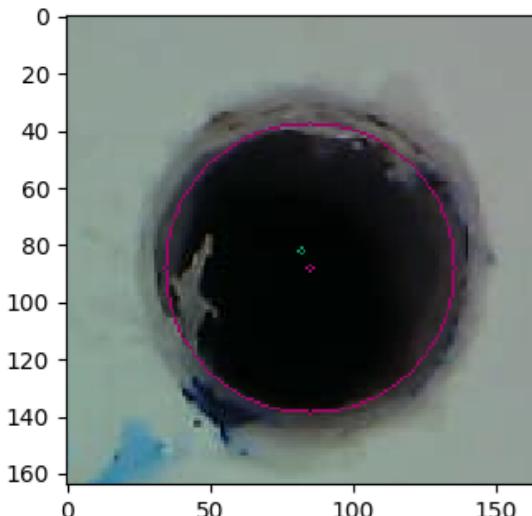


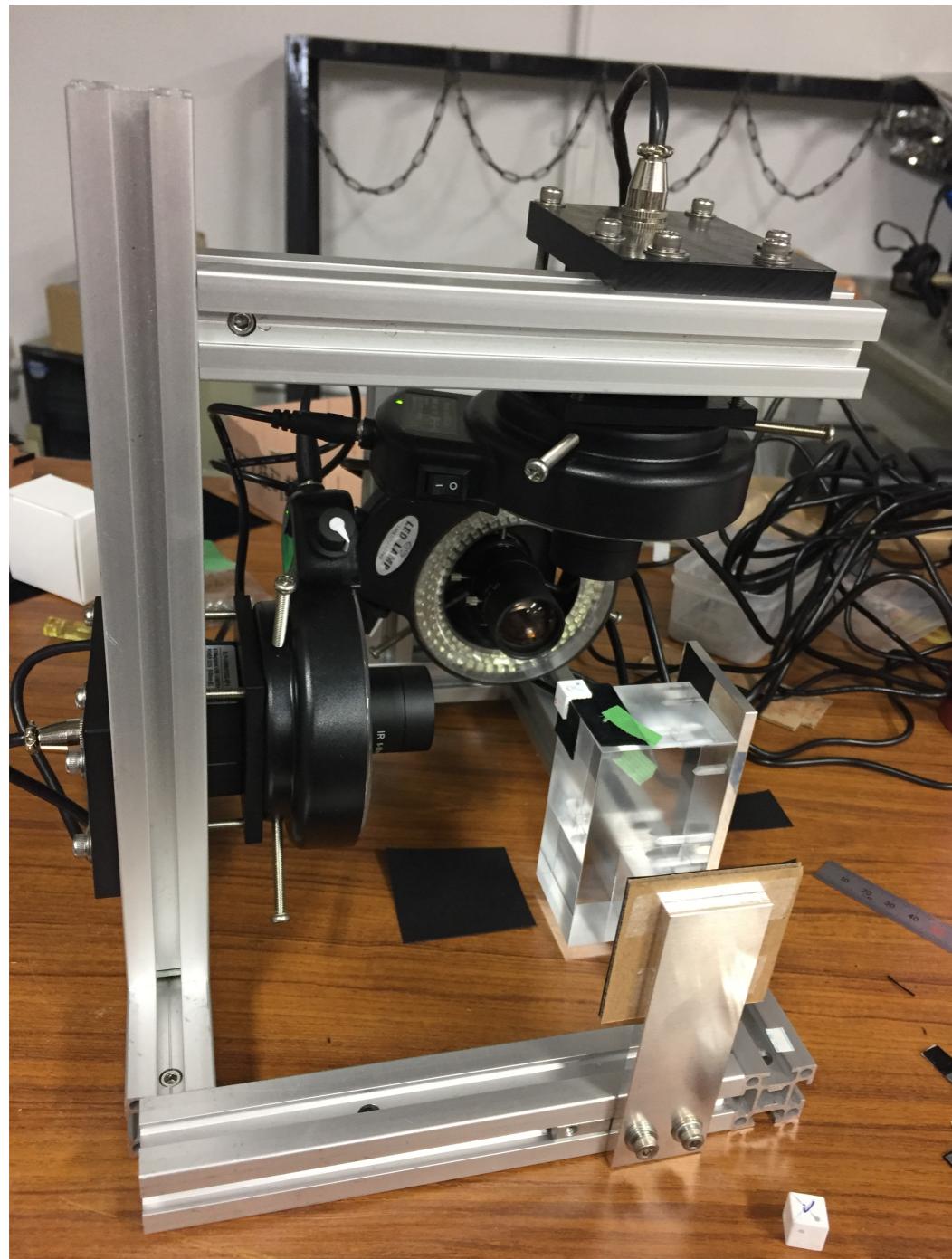
Figure 1: 196-cubes array for quality check.
ロシアの手動QC

目次

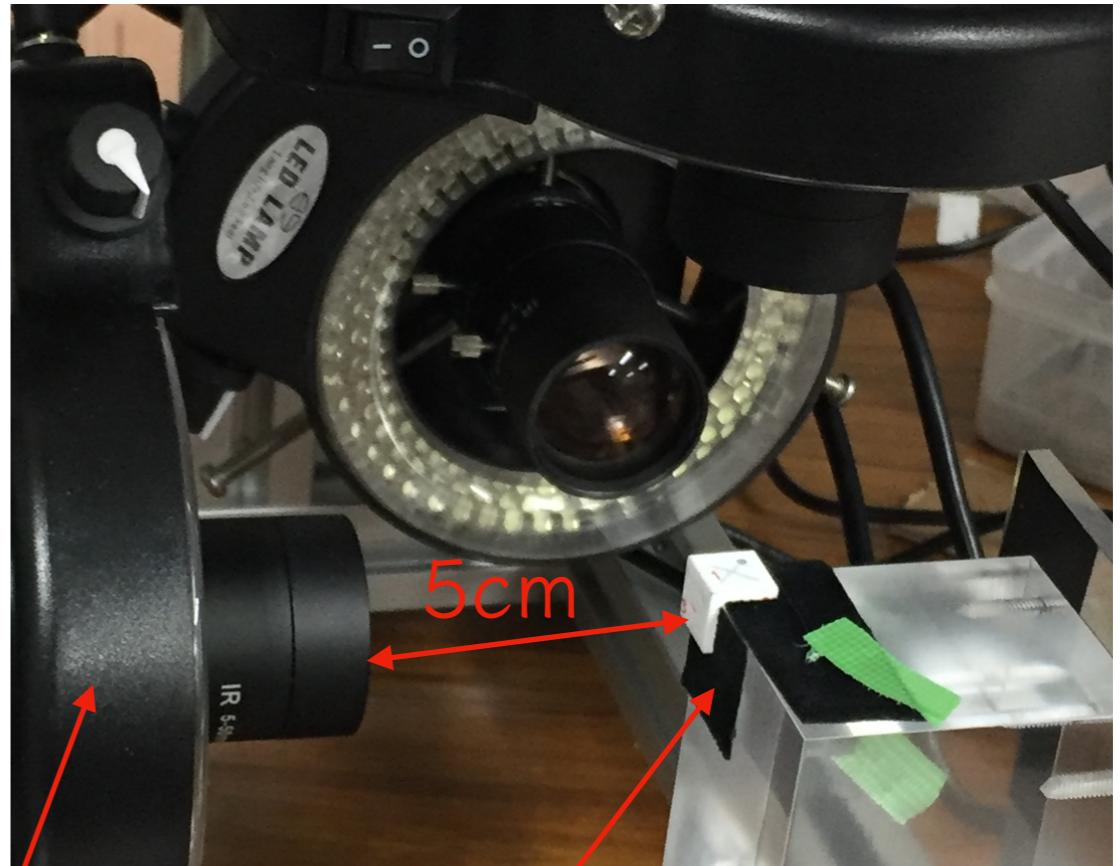
- 撮影時のキューブの傾きを解析コード内で補正
- より効率的なシンチレータキューブ撮影ジグの考案
- 今後の課題

キューブ撮影時の傾き補正

前回発表時の撮影ジグ

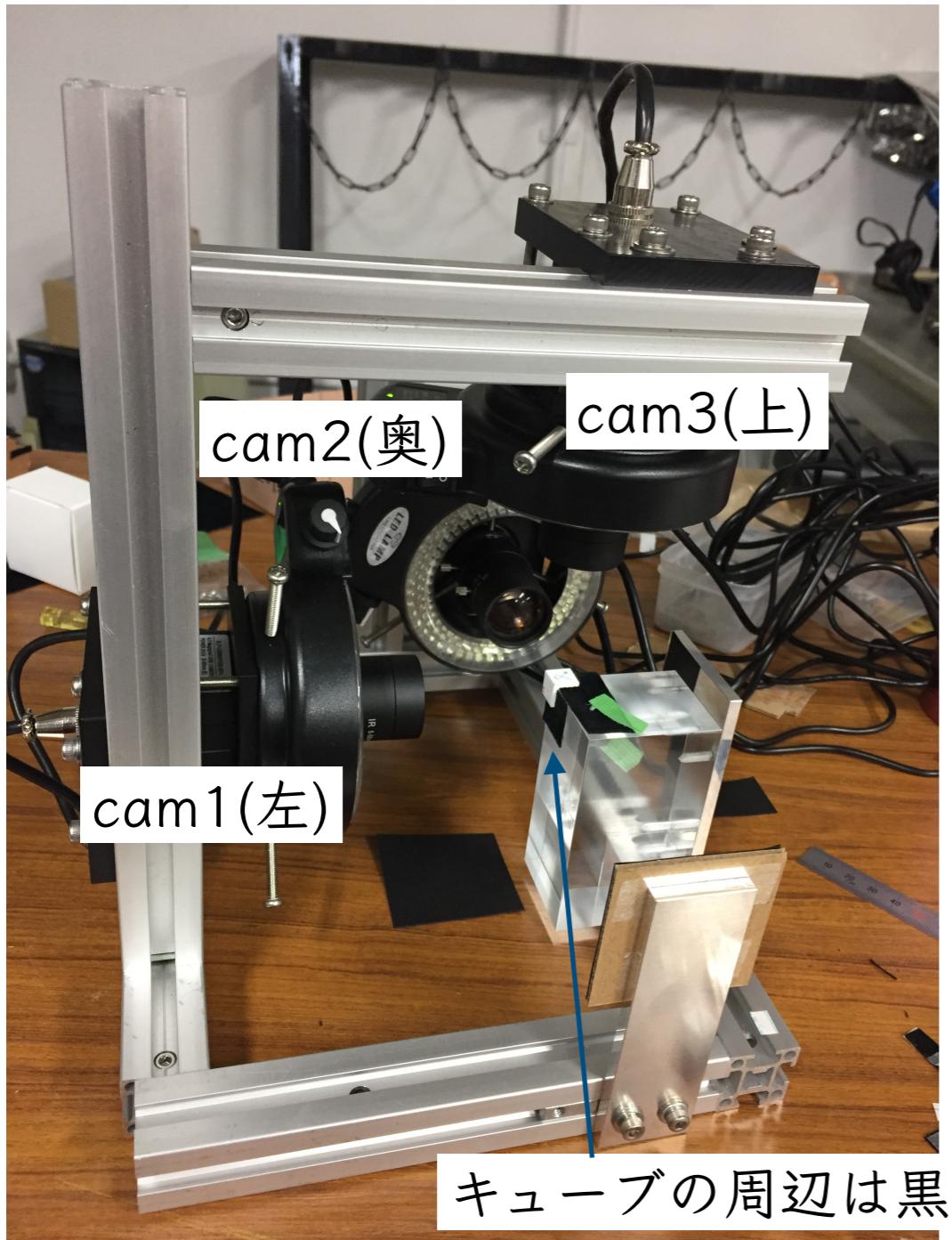


骨組み：太さ25mmのアルミフレーム



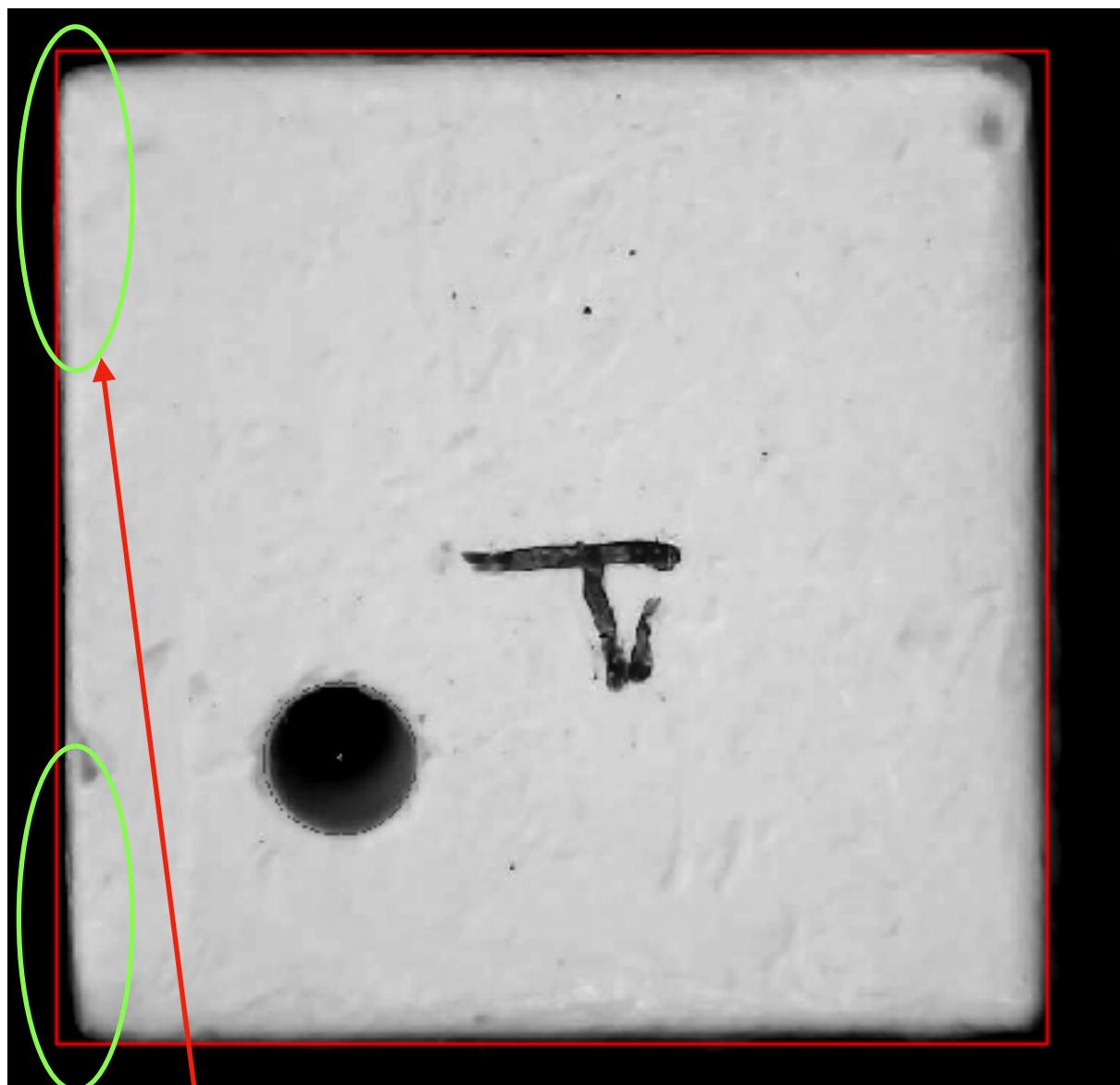
- キューブを設置→撮影→回転→撮影→選別の繰り返し
- 問題点：
 - 確実に回転しないと6面撮影できない
(人為的ミスの可能性)
 - **選別ミスの可能性**
 - 各キューブにつき上記操作の繰り返し：
時間がかかる

撮影テスト



- スズノ技研のジグを用いて3方向から撮影してみる。
- 理想的には、3つのカメラ全てで同じような撮影ができるくて欲しい。
- 実際には、
 - レンズ-キューブ距離
 - カメラの傾き・位置ズレ
 - キューブの置き方
 - ...
- などの微妙な違いによって写真は異なってしまう。**(今は目分量でカメラの位置を決定)**
- 撮影時の違いを画像解析によって補正する。

キューブの傾き補正

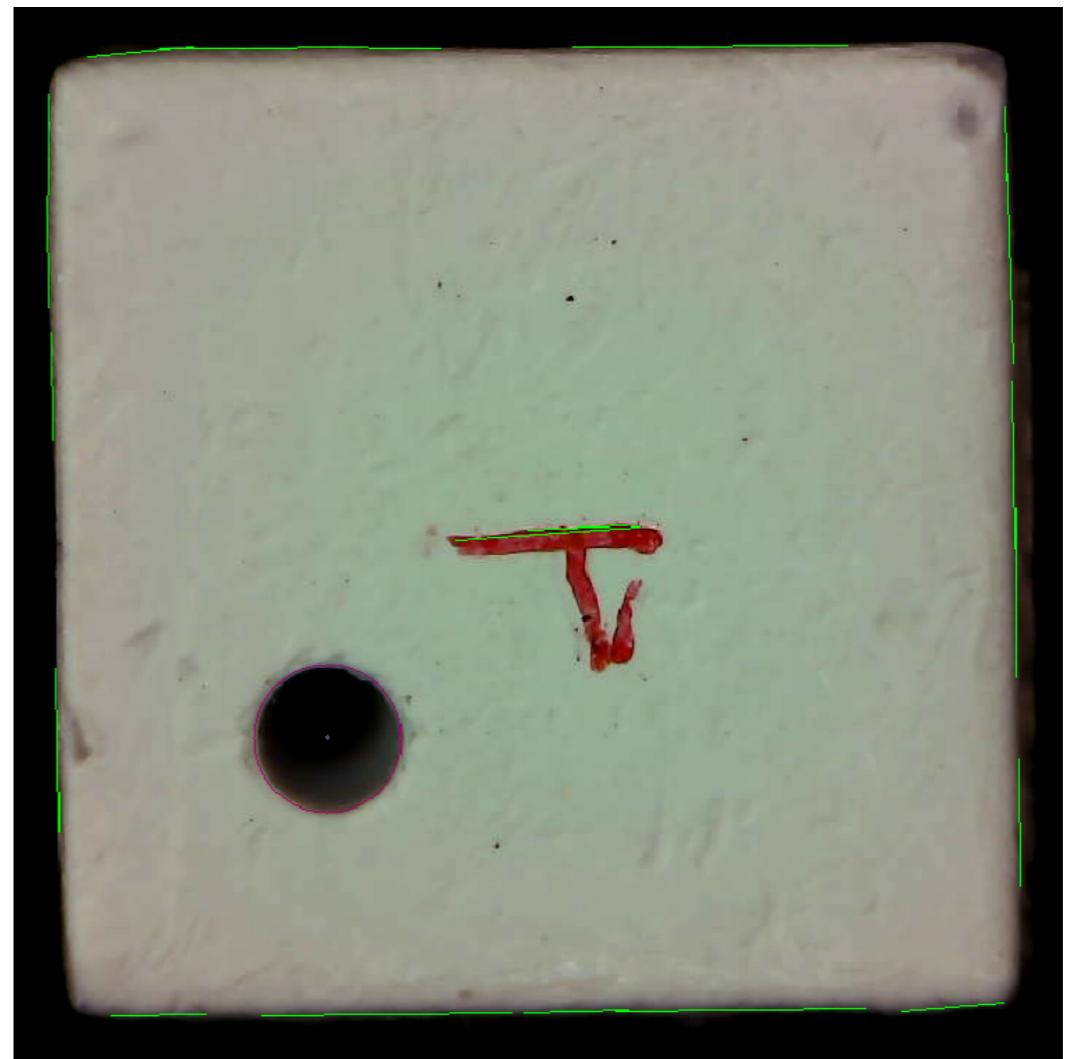


緑の丸：キューブが傾いているので、下の方では赤線とキューブ辺の間に隙間ができるている

- キューブの辺検出：
- 輪郭検出・直線検出によってキューブの各辺に複数の座標を得て、最も外側の点をその辺の実効位置としていた。
- 最も外側の4点を通る四角を書くと、左のように傾いていることが多い。
→大きさを正しく取れない
- 穴位置は辺からの相対位置として定義
→穴の位置にも影響



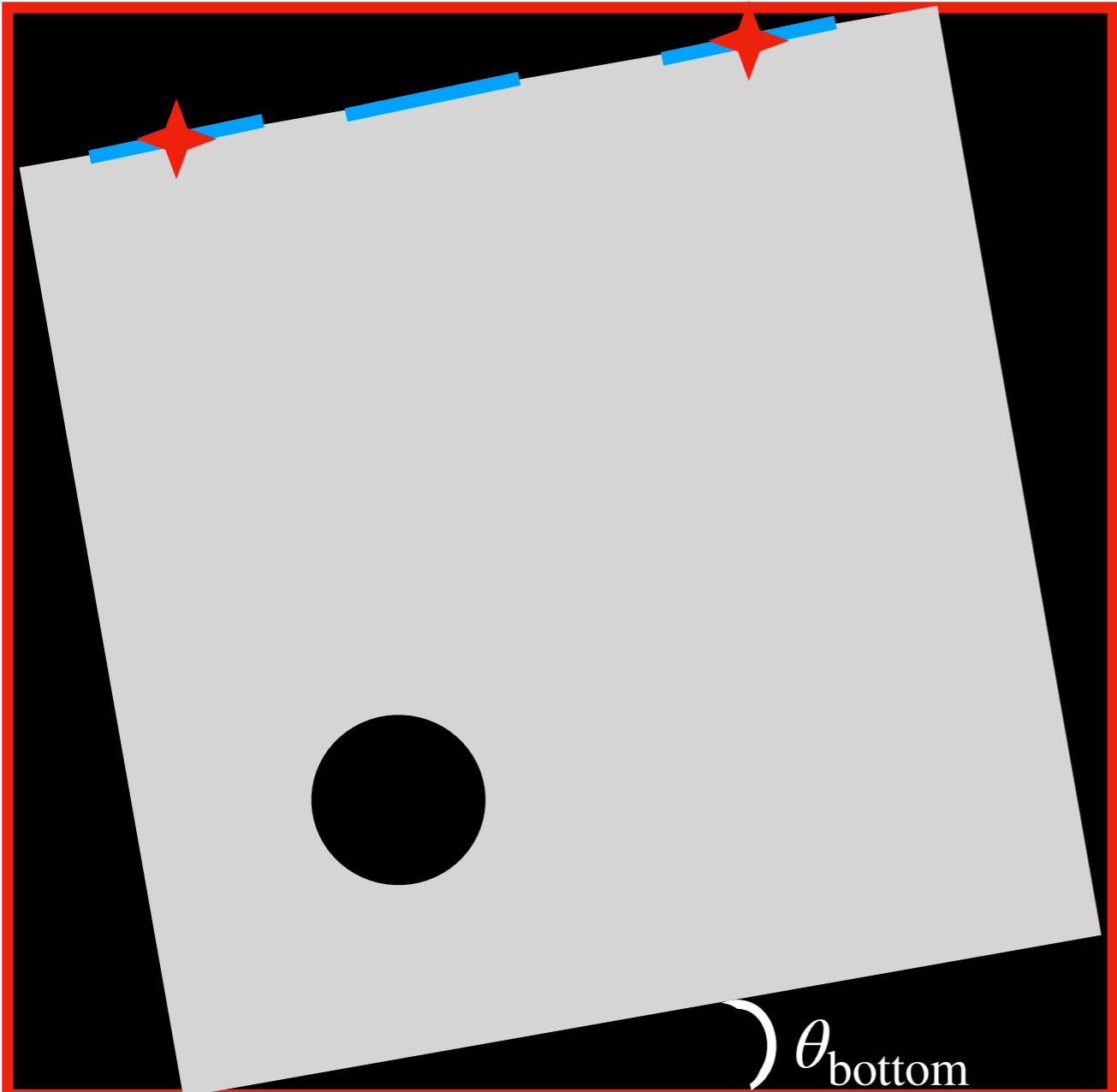
輪郭検出の様子



直線検出の様子（緑の線が検出された直線）

- 1つのカメラでの撮影時は、画面が傾いていたらカメラ・キューブの置き直して対処できていたため、キューブの各辺はx軸・y軸に平行と仮定していた。
- 複数カメラでの撮影はジグを動かすと他のカメラへ影響を与えるので、解析で補正する必要がある。

キューブの傾き補正



下辺の傾き = $|\tan \theta_{\text{bottom}}|$

$$\bar{T} = \frac{1}{N_{\tan \theta}} \sum_i |\tan \theta_i|$$

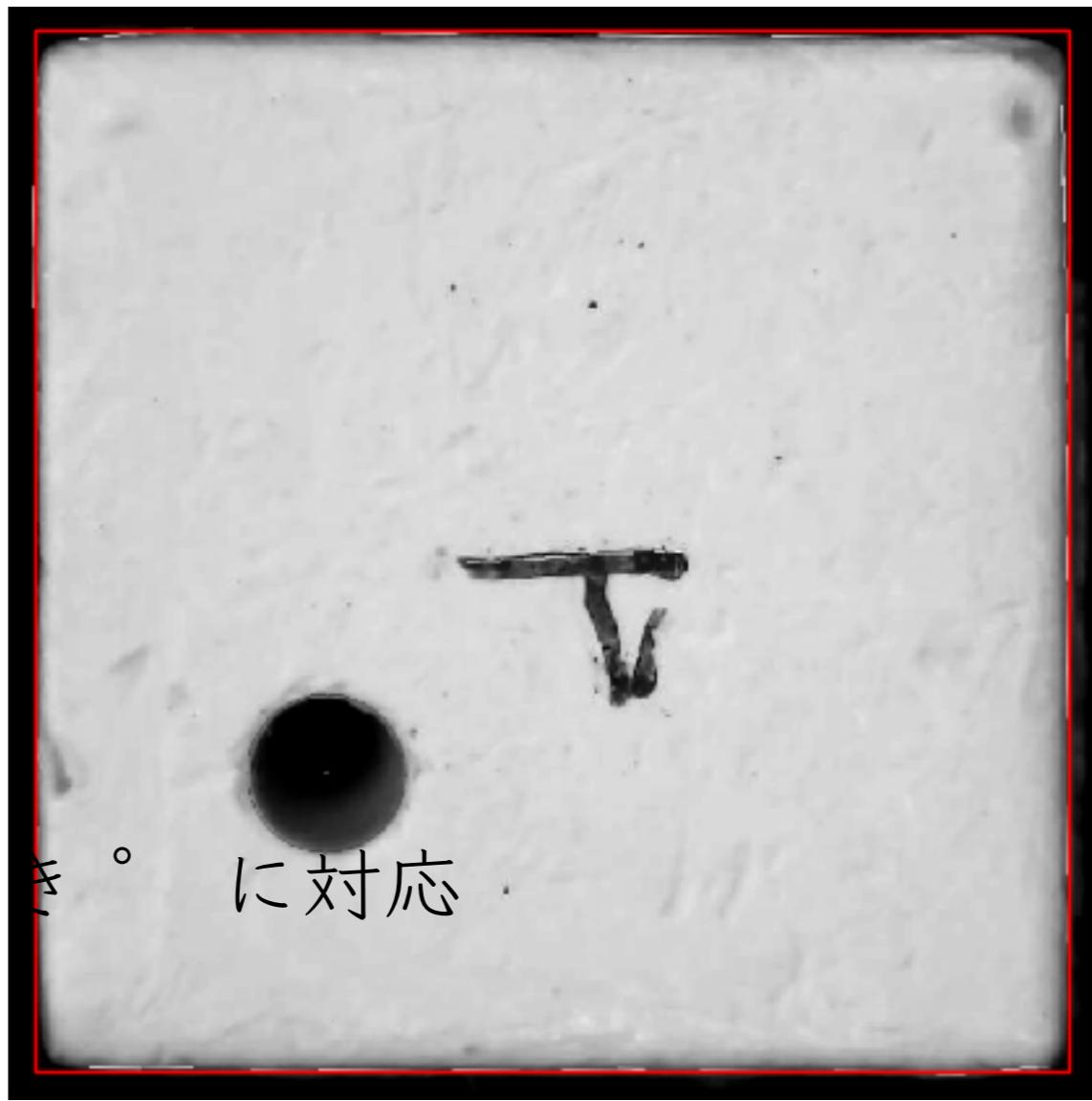
- キューブの各辺について、複数の直線が検出されている。最も遠い2直線の中心 を通る直線の傾きを求める。
- 4辺で求めた傾きの平均をそのキューブの傾き \bar{T} とする。
- 四角の中心を回転中心とし、角度 $\arctan \bar{T}$ だけ、
 1. 辺・輪郭点・穴中心の座標 (x, y) を回転
 2. 画像そのものを回転
- 位置座標の決定については1. を用いて、バンプ検出など色の情報が必要なときは2. を用いる。

補正の様子



$\bar{T} = 0.0139, \bar{\theta} = 0.796^\circ$ 平均の傾き。

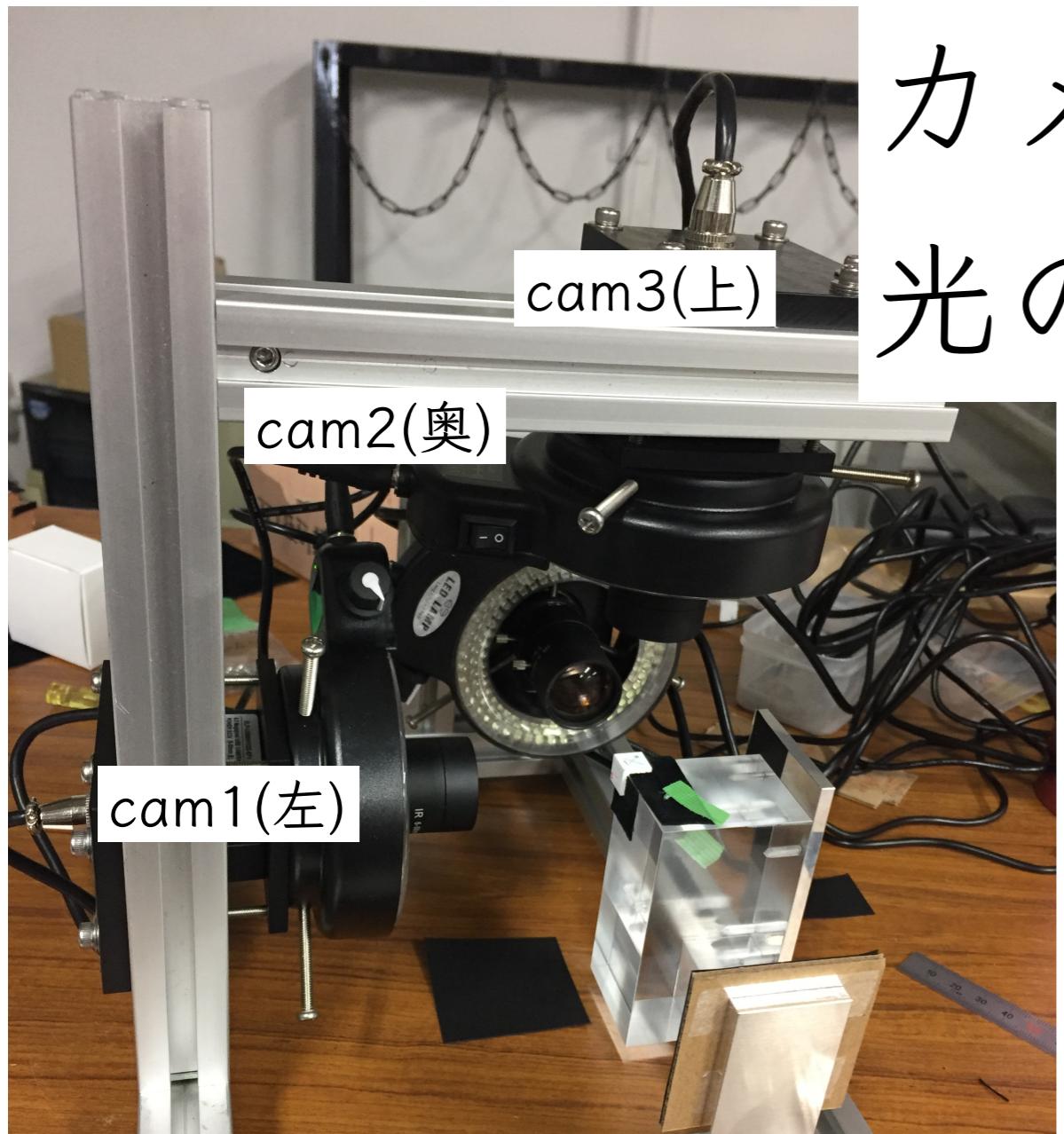
補正前



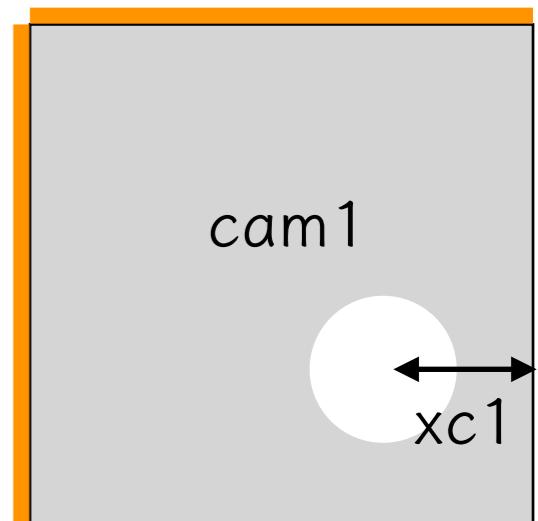
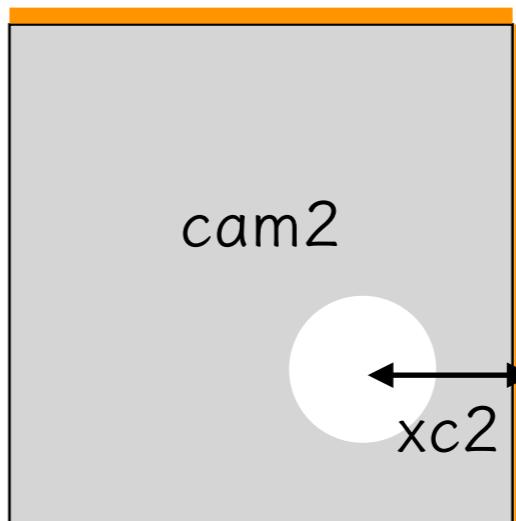
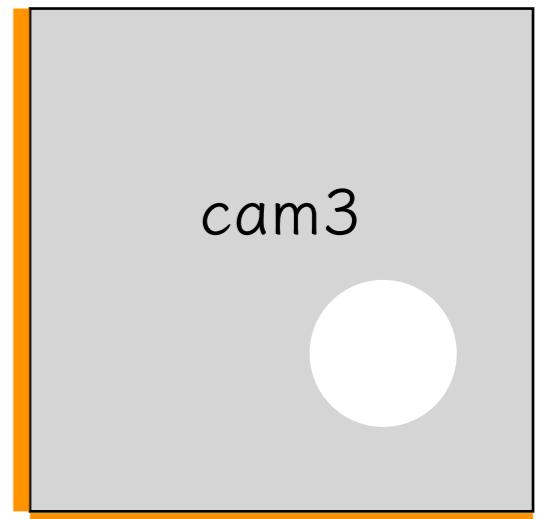
平均の傾き。 $\bar{\theta} = 0.796^\circ$ に対応

補正後(赤い四角は補正後に書き直した)

- 図の面の場合：各辺の傾き (左) (上) (右) (下)
 $\tan \theta = 0.0144, 0.0101, 0.0208, 0.0101$
- 平均の傾き $\bar{T} = 0.0139, \bar{\theta} = 0.796^\circ$ に対応
- 今後：補正後のデータを用いて、光の当たり方の違いによる穴位置の変動などを調べる。

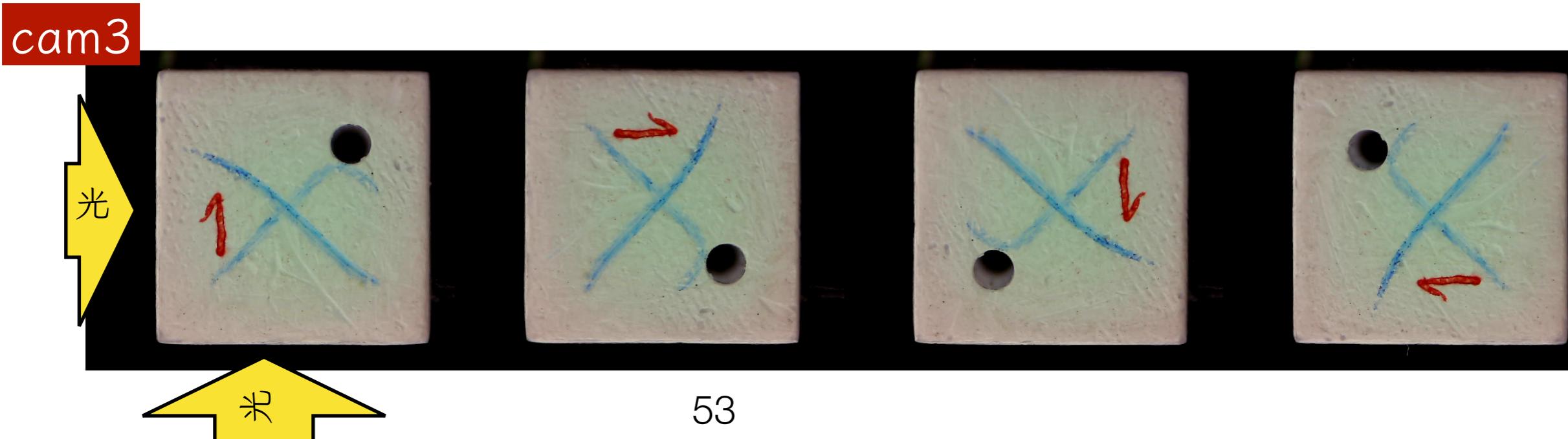
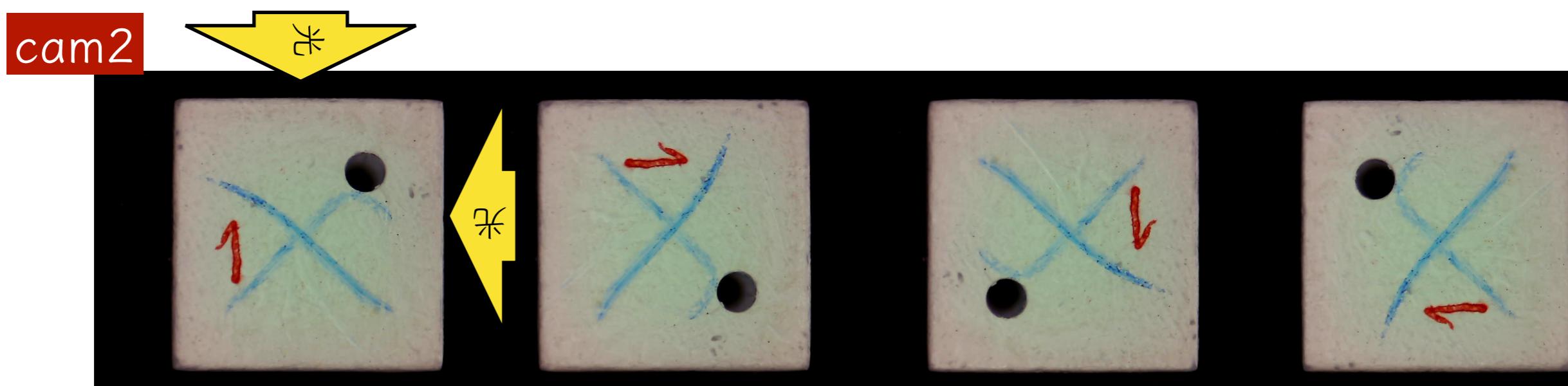
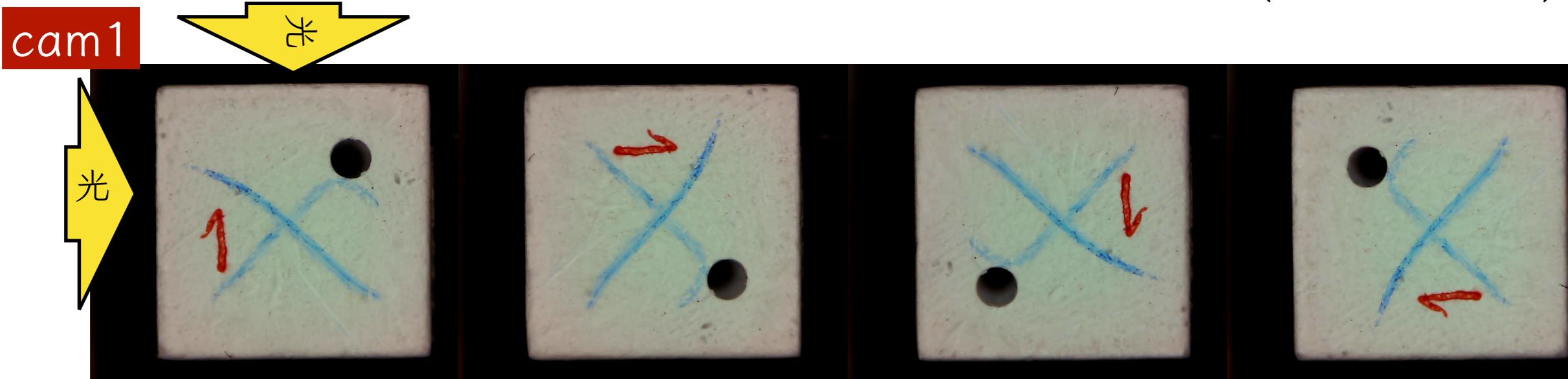


カメラ毎の 光の当たり方



- オレンジの部分に、別方向からの光が当たる。
- 穴の辺からの位置が撮影状況によって変わってしまう
(本来は $xc1=xc2$ のはずなのに、 $xc2$ のほうが大きく見えてしまう)
- 同一の面をそれぞれのカメラで撮影、同一の辺長・穴位置について光の有無によって有意な違いがあるか確認。

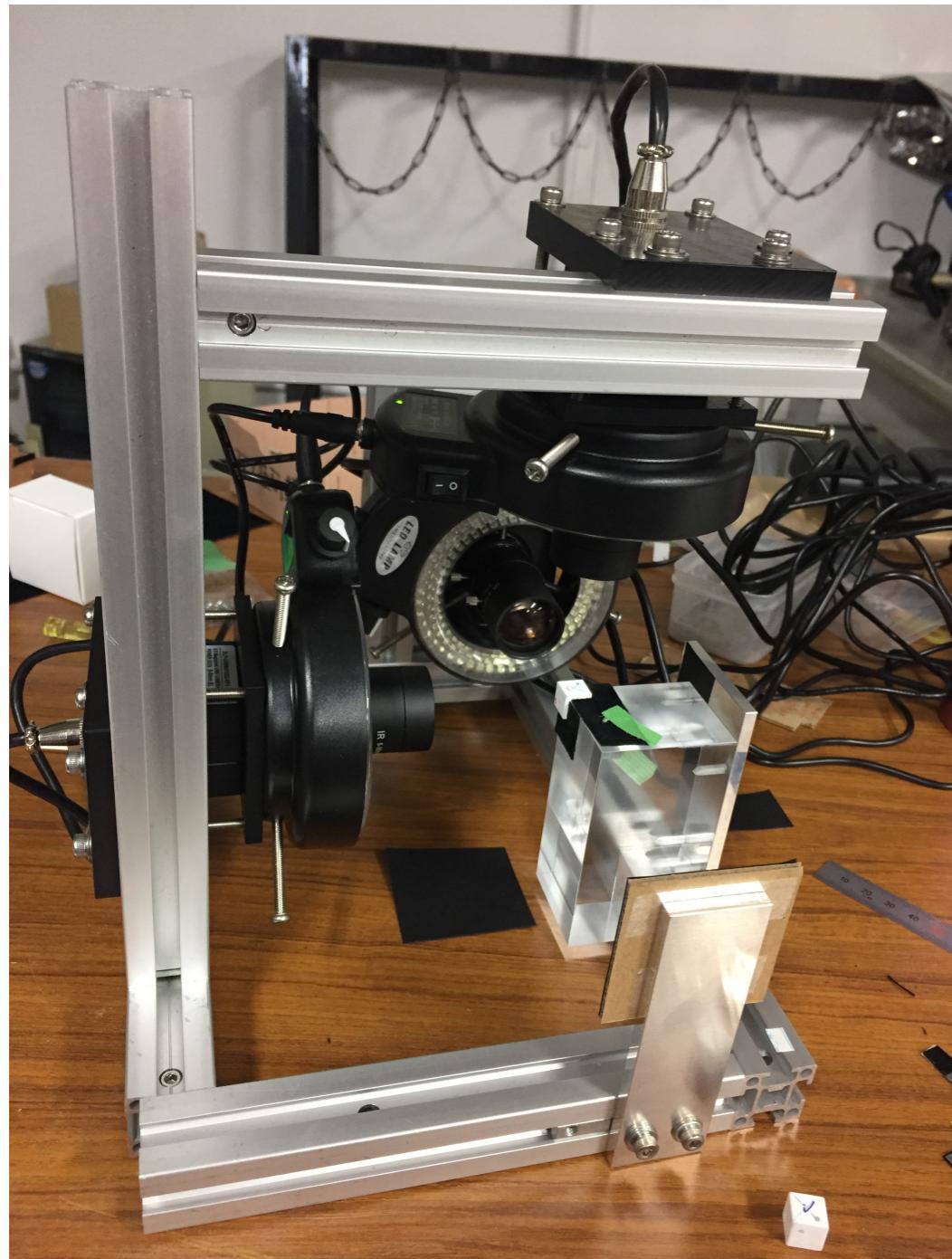
同一キューブの同一面の見え方の違い(回転補正前)



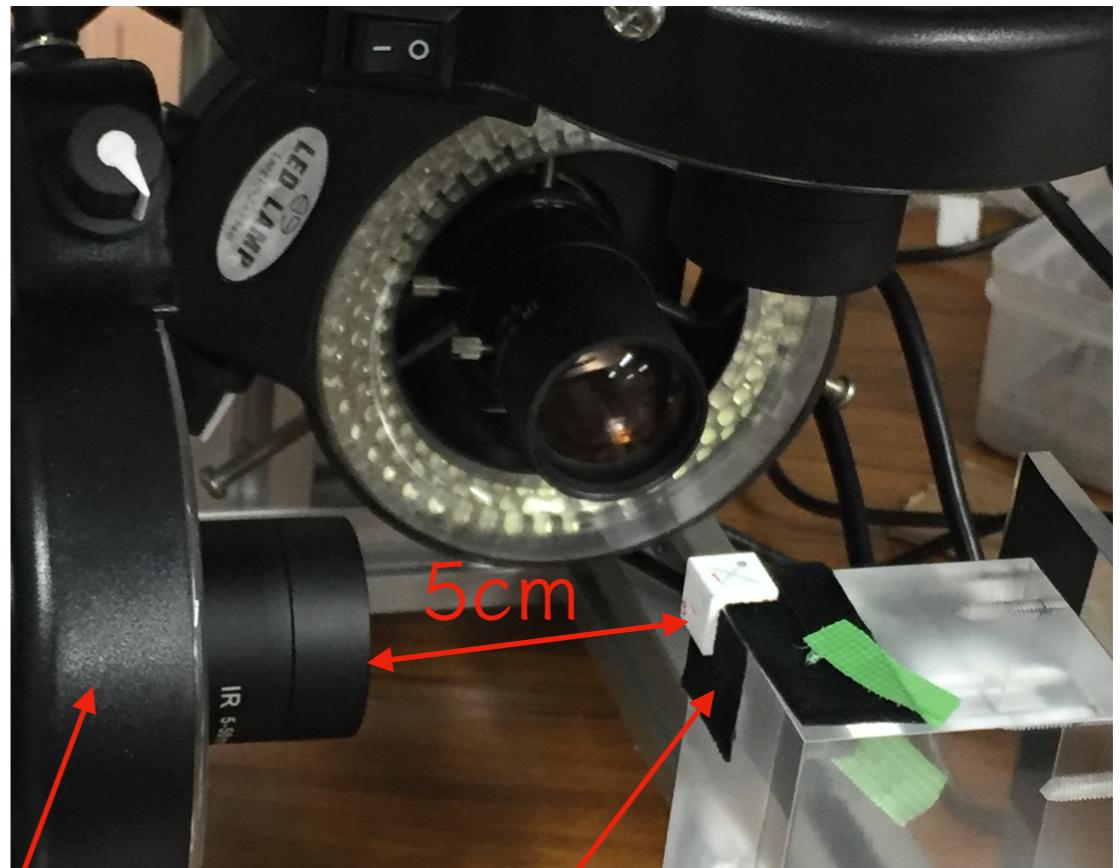
撮影ジグの考案

special thanks to 市川さん、木河さん

前回発表時の撮影ジグ



骨組み：太さ25mmのアルミフレーム



3方向にカメラ+
リングライト設置

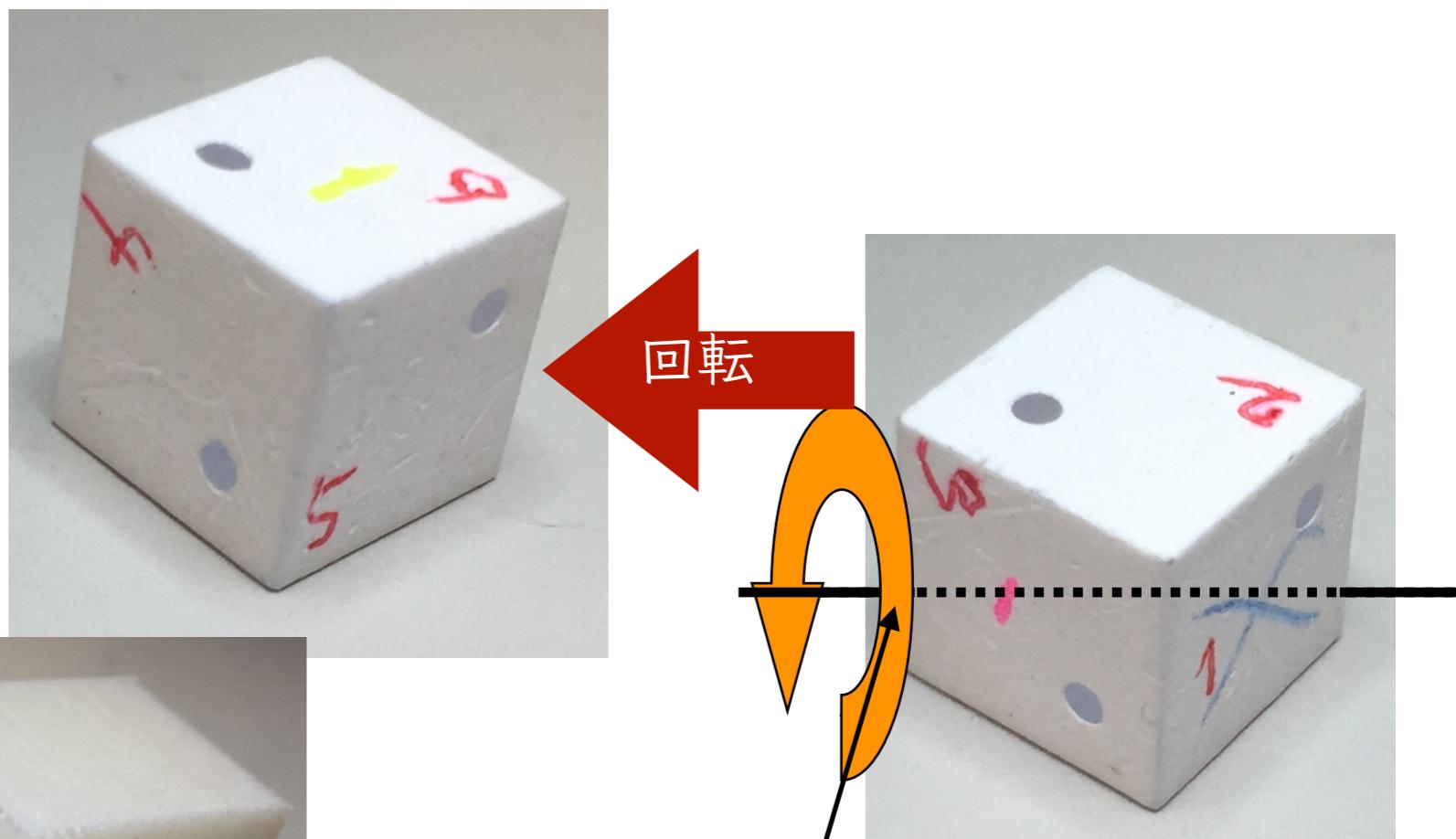
キューブ台座 (アクリル、スズノ技研)
不要箇所は黒い布を貼る

- キューブを設置→撮影→回転→撮影→選別の繰り返し
- 問題点：
 - 確実に回転しないと6面撮影できない
(人為的ミスの可能性)
 - **選別ミスの可能性**
 - 各キューブにつき上記操作の繰り返し：
時間がかかる

新しい撮影ジグ（案）

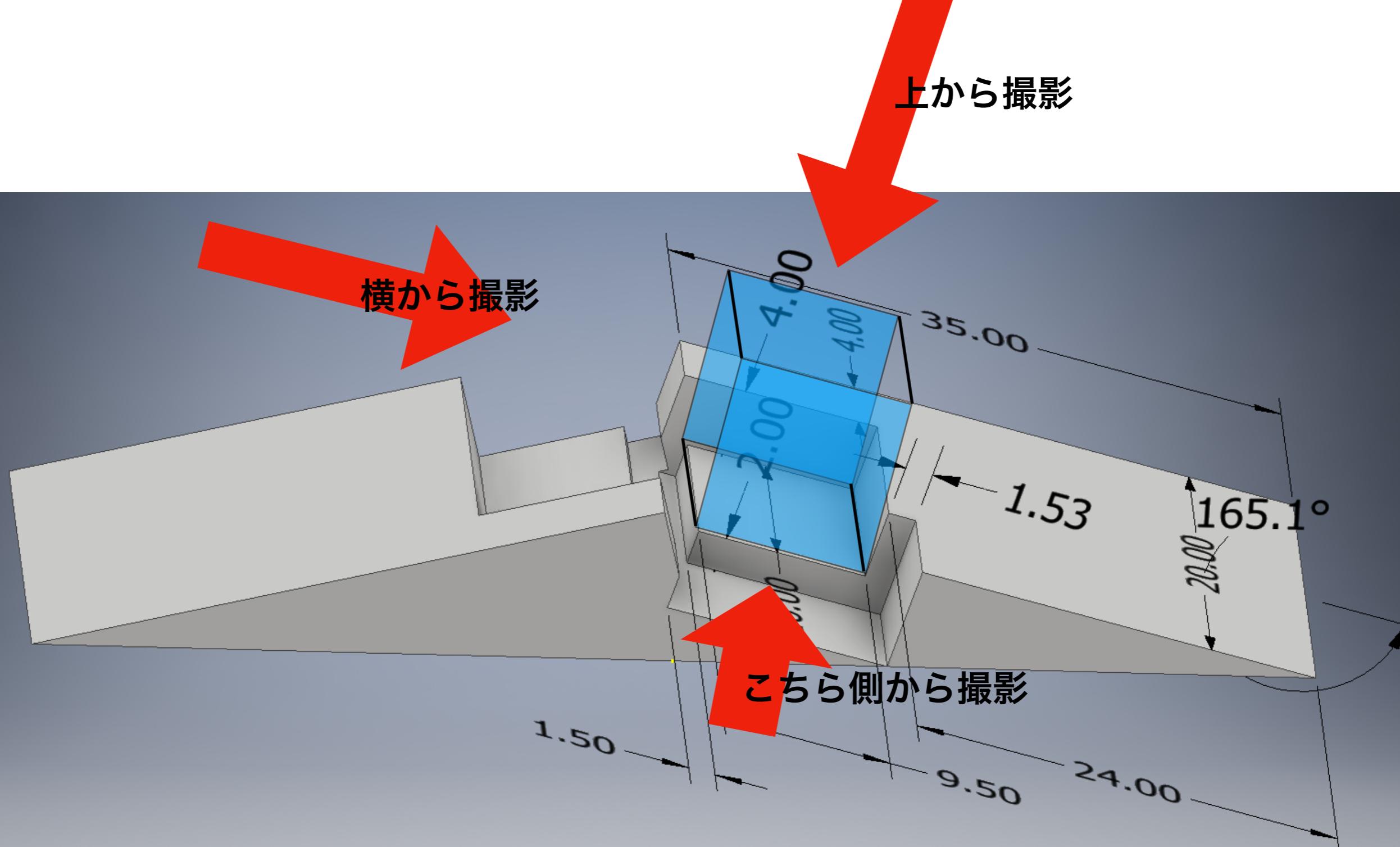
- ・ キューブの回転機構の工夫・改善
- ・ 手でやると間違える？
- ・ 新しい回転の方法

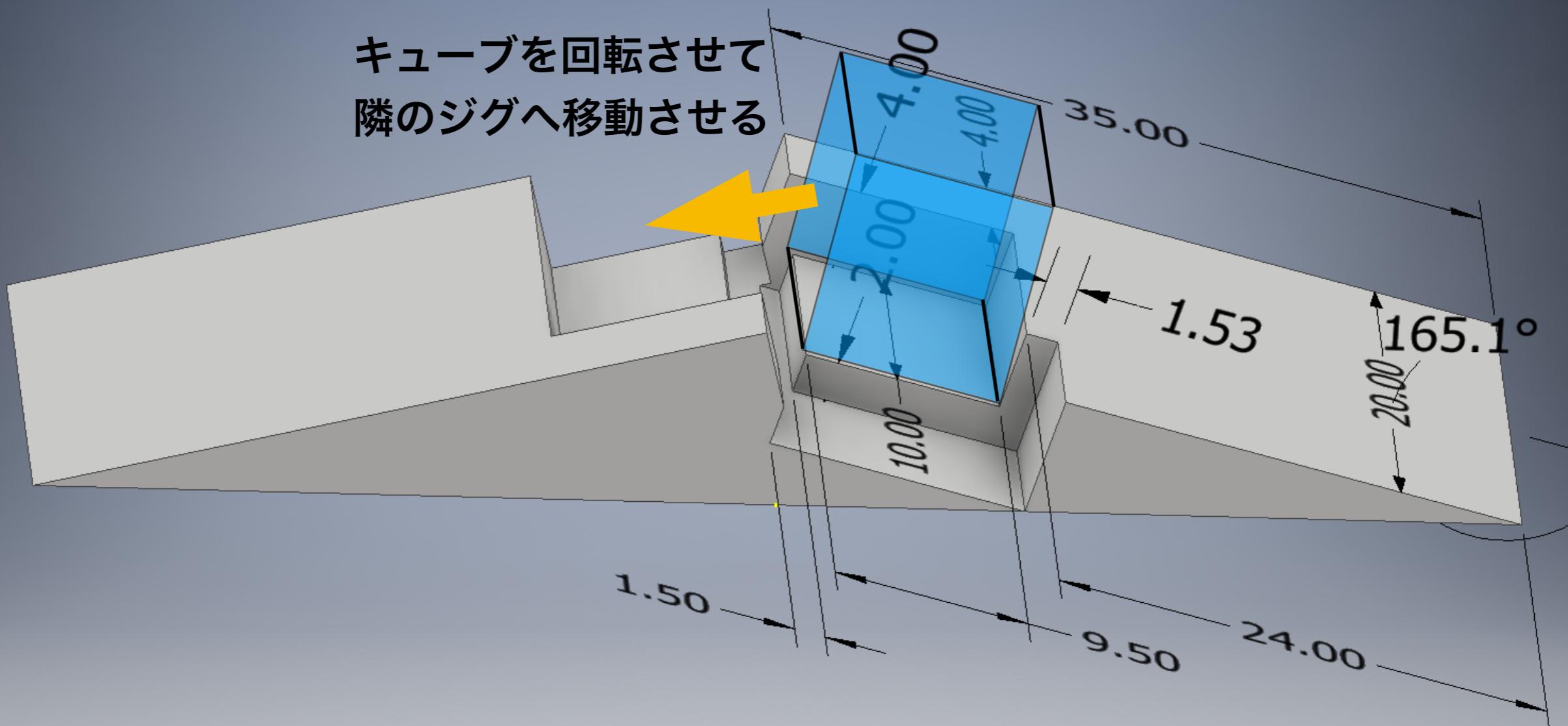
新しいキューブ台座案

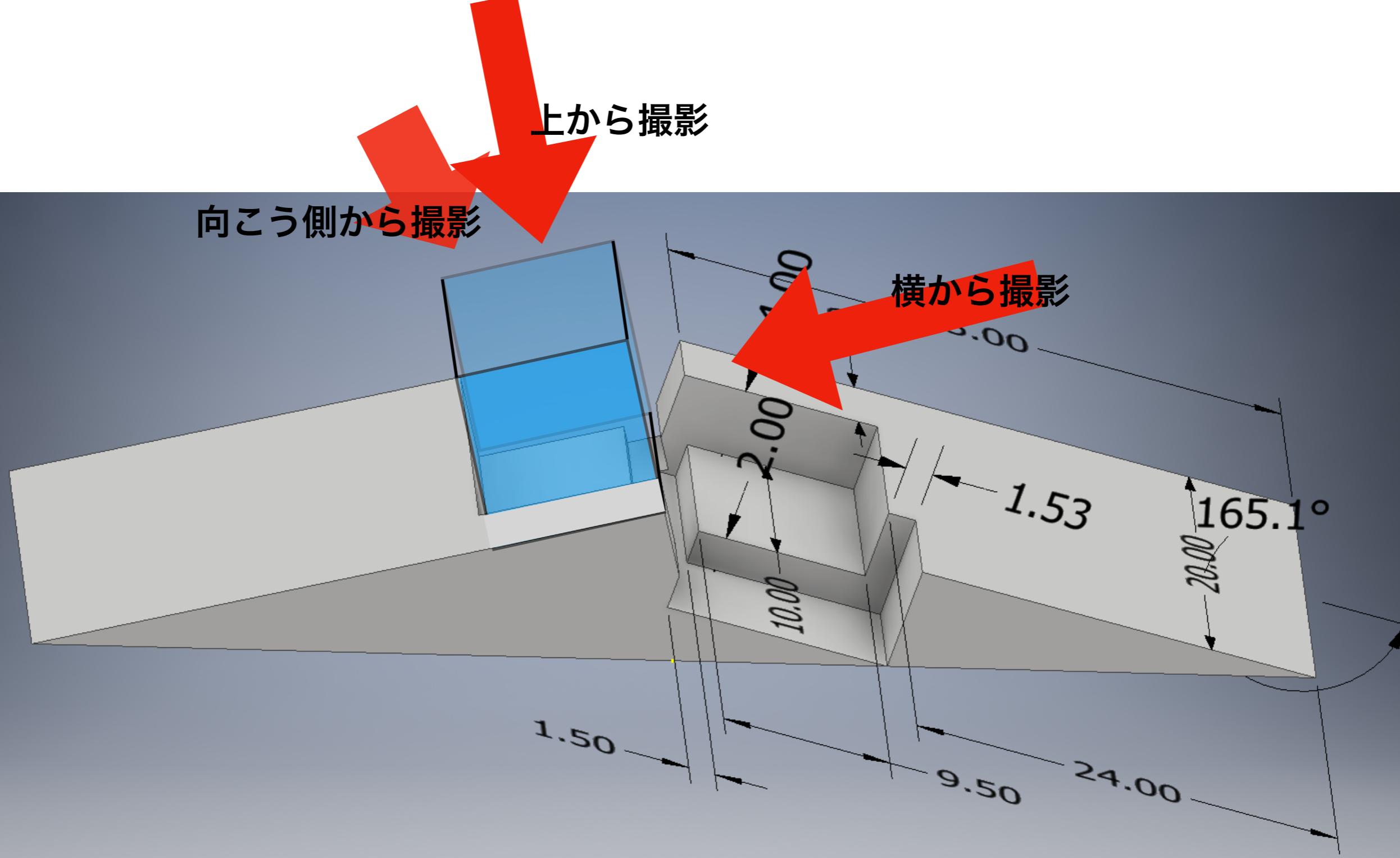


以前の機構では、この軸での回転が必要

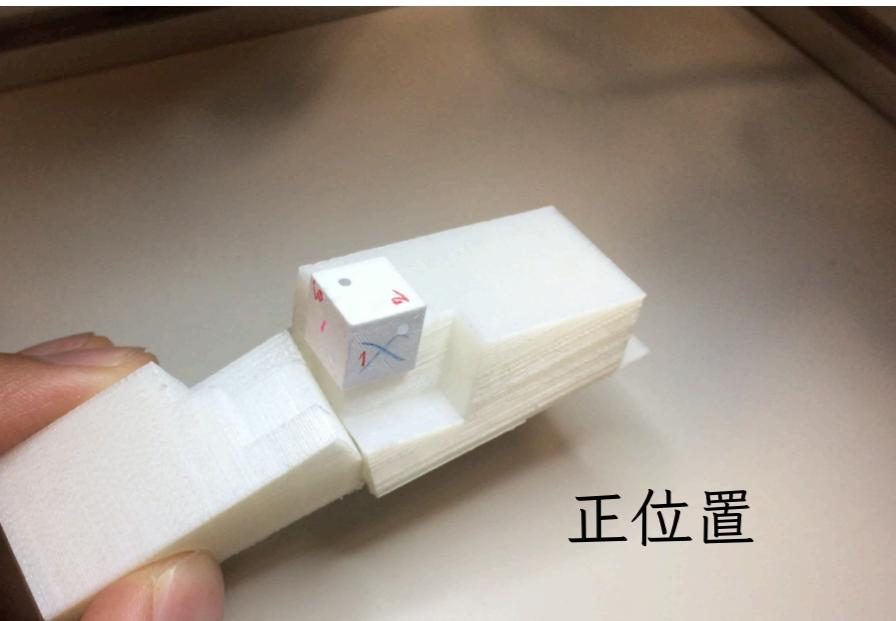
2つのジグが向かい合う形
J-PARC の3Dプリンタで作成





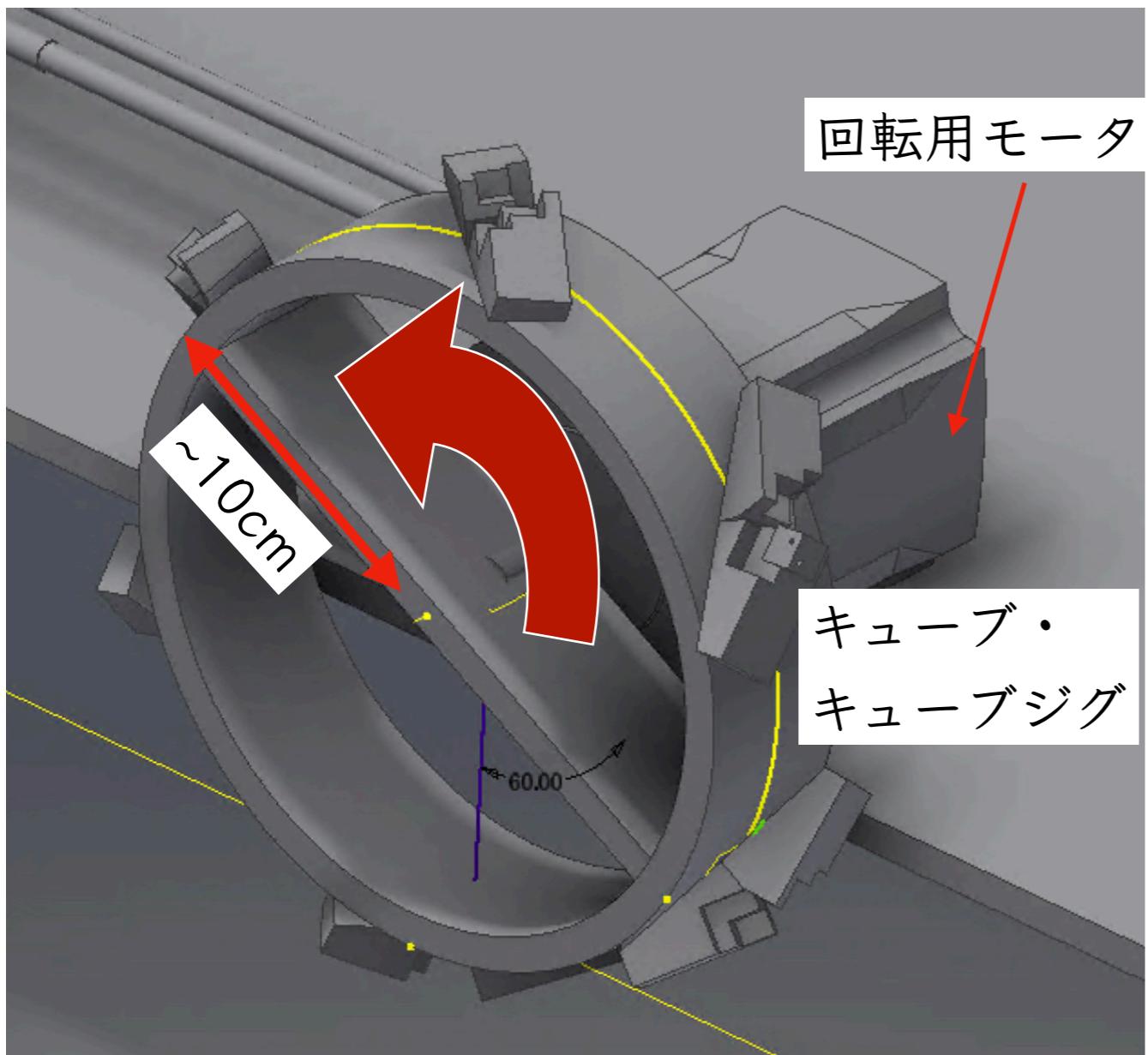


<https://www.dropbox.com/s/a3jbbdx26dxbrja/99F4A1A9-BA9A-4848-95CF-973A4DA556FE.mov?dl=0>

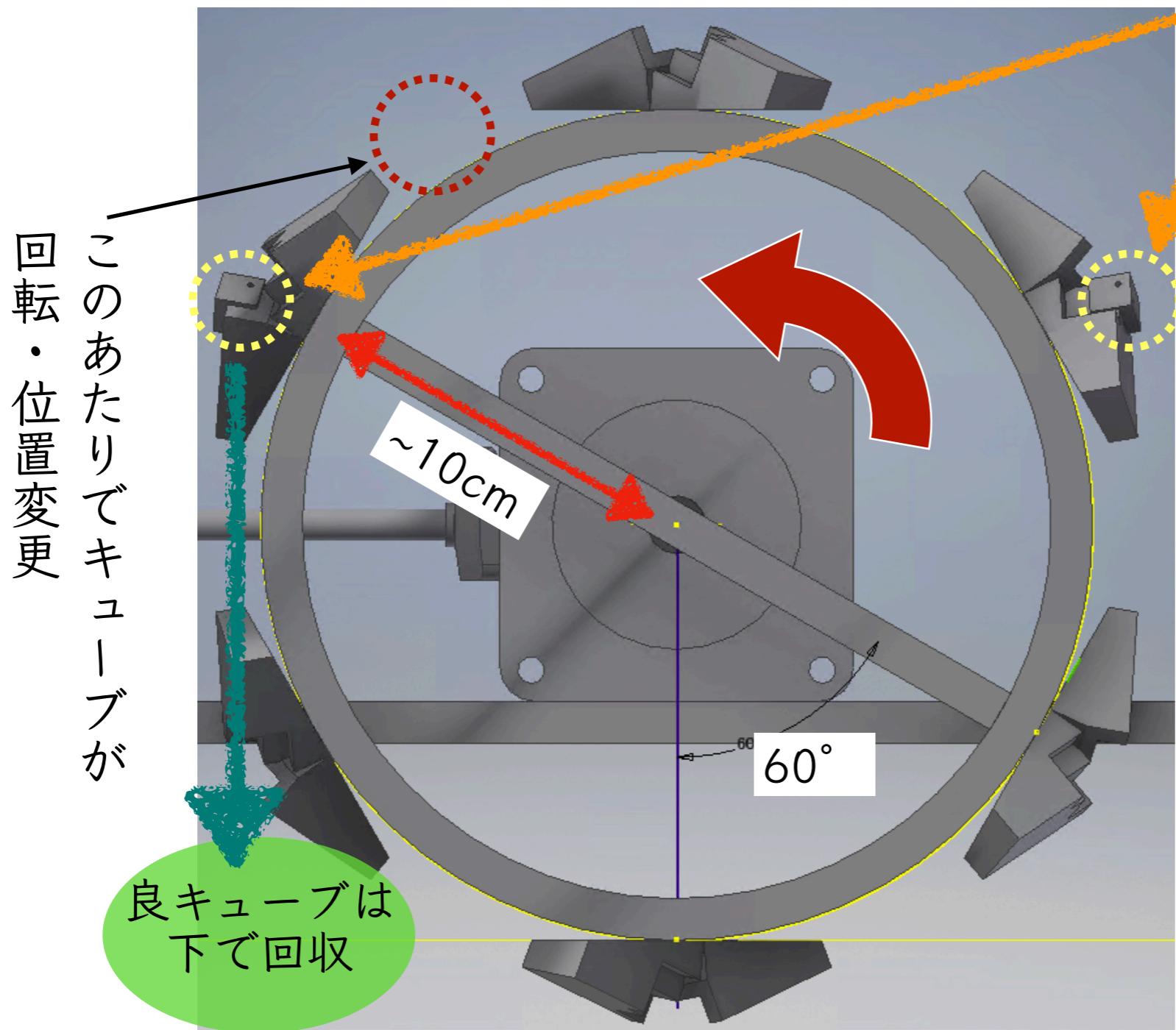


新しい撮影ジグ（案）

- キューブの回転を自動化できるか？
 - 円周上に複数のジグを用意
 - カメラでの撮影位置を2箇所用意（3面+3面）
 - ステッピングモータを用いたPCからの回転制御を予定



・キューブの新し自動化撮影ジグ（案）



キューブの撮影（2箇所）

悪キューブは2度目の撮影点ではじく

良キューブはそのまま下へ落下→後で箱に詰める

手元の PC 操作で、
回転→静止→撮影→
回転→静止→撮影…
を行いたい

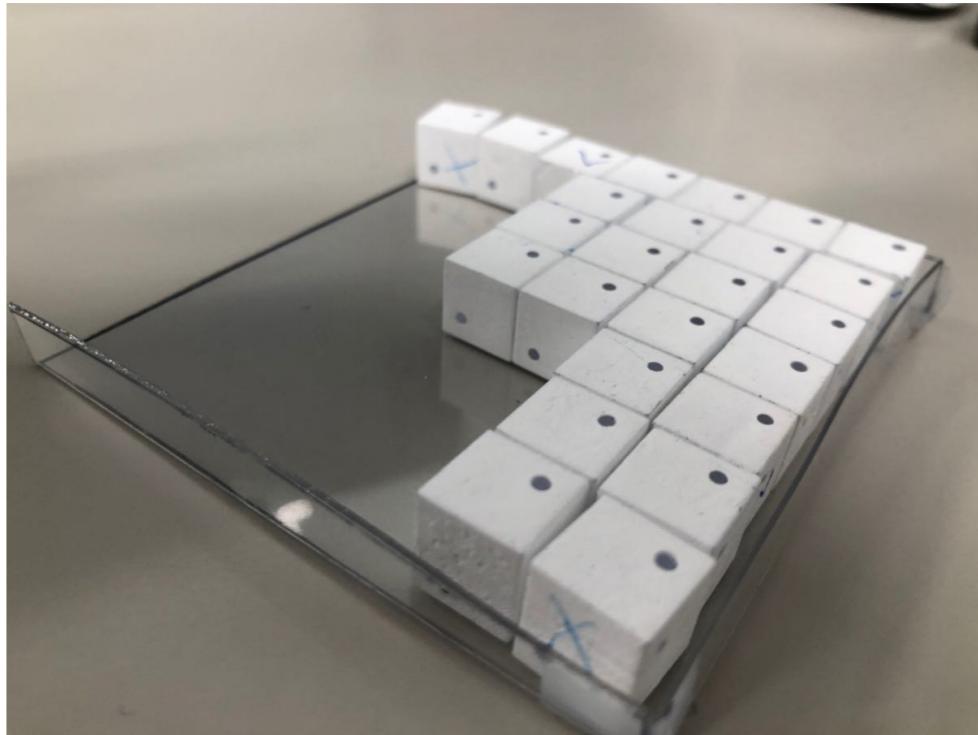
課題：微妙な角度でのカメラの固定方法を考える必要あり

今後の課題



ロシアでの梱包が終わった（松原さん）。
3/10（今日？）J-PARC に到着予定
その後京都に転送していただく

- ロシアから12000個のキューブを購入・溶着案での superFGD プロトタイプを組み立て予定。
→組み上げの前にQCを行う。
- 光の当たり方依存の問題解決後、（今のジグで）大量撮影を行い、分布を作ってみる。
- モータ（3/11納品予定）の制御の勉強
- 新しいジグ案の確立（カメラの置き方）



QC 後のキューブは溶着しやすいよう、
7×7 ずつトレイに入れて梱包予定。