シンチレータキューブの穴検出改善、エッジ検出

2020.2.4 京都大学 谷
collaborate with 小川さん (KEK)

目次

- キューブの穴検出
  - カイ二乗値の最小化
- キューブのエッジ検出・バンプ(表面の出っ張り)検出
  - 光の当て方とエッジの見え方
- 今後の予定

穴検出

-カイ二乗値の最小化-

カイ二乗値の最小化

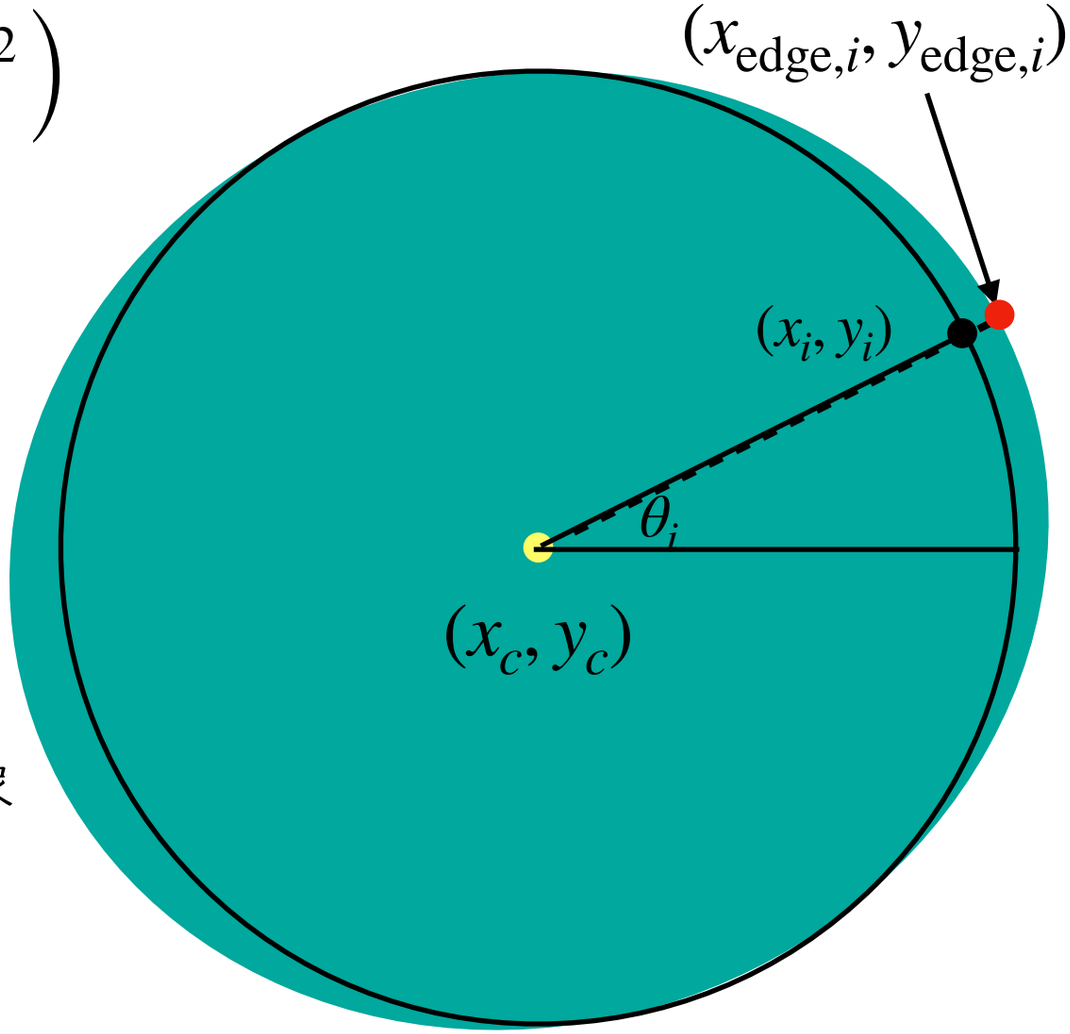
$$E_{\mathrm{sum}}(x_c, y_c, r) = \sum_i \left(|x_i - x_{\mathrm{edge},i}|^2 + |y_i - y_{\mathrm{edge},i}|^2 \right)$$

$$x_i = x_{\mathrm{c}} + r\cos\theta_i \ , \ y_i = y_c - r\sin\theta_i$$

Esum が最小となる (xc, yc, r) を求める。

- (xc, yc) は円の中心、r は円の半径
- $(x_{\mathrm{edge}}, y_{\mathrm{edge}})$ は穴のエッジ上の点
- (xc, yc, r) の初期値は以前やった、二値化画像から得た中央値を使う。
- 今回はエッジ上の16点で和をとった

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



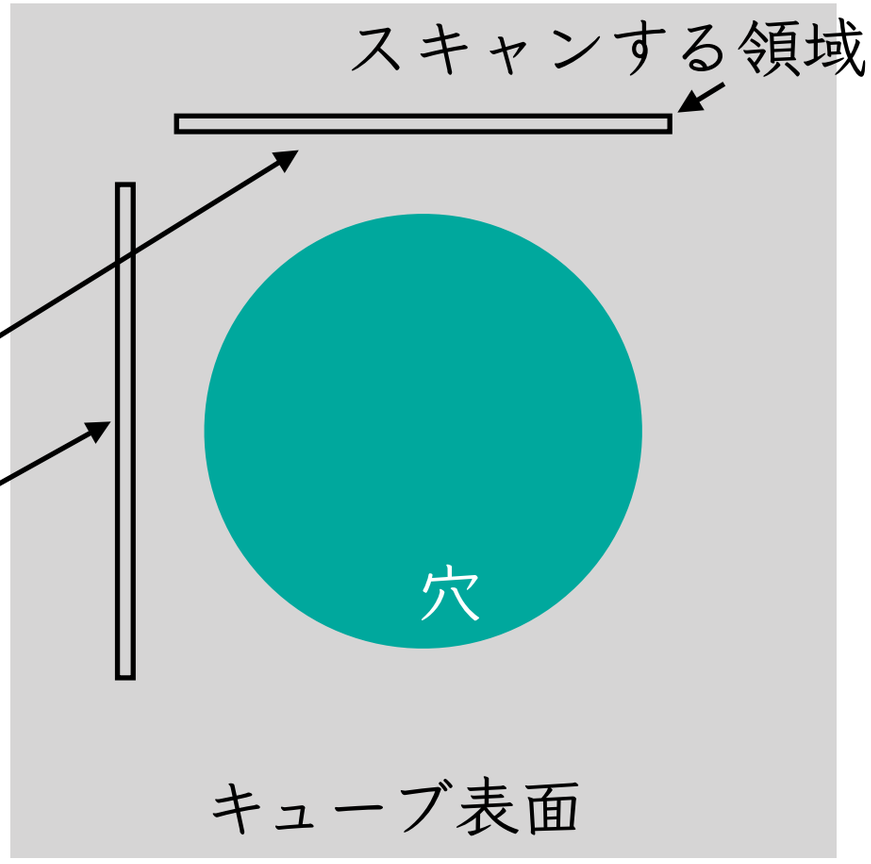
黒円:自分で描く円
青:キューブ穴

二値化の threshold は
各画像ごとに決定

画像の二値化

<u>二値化のための threshold の決め方</u>

- 穴の周囲の明るさ (0~255) 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、最も白いものを参照値として採用する。

今回は小川さんにご提案頂いた、
参照値の 30% カットの値を
threshold として用いた
例：参照値140 → threshold 98

最小化の手順

ある $\mathbf{x}_k = (x_k, y_k, r_k)$ での Esum の勾配

$$\nabla E_k = \left(\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}, \frac{\partial E}{\partial r} \right) \Bigg|_{\mathbf{x}=\mathbf{x}_k}$$

を求め、設定した learning rate η をかけて $\mathbf{x}_k$から引き更新する。

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \nabla E_k$$

この操作を指定ステップ数だけ繰り返す。

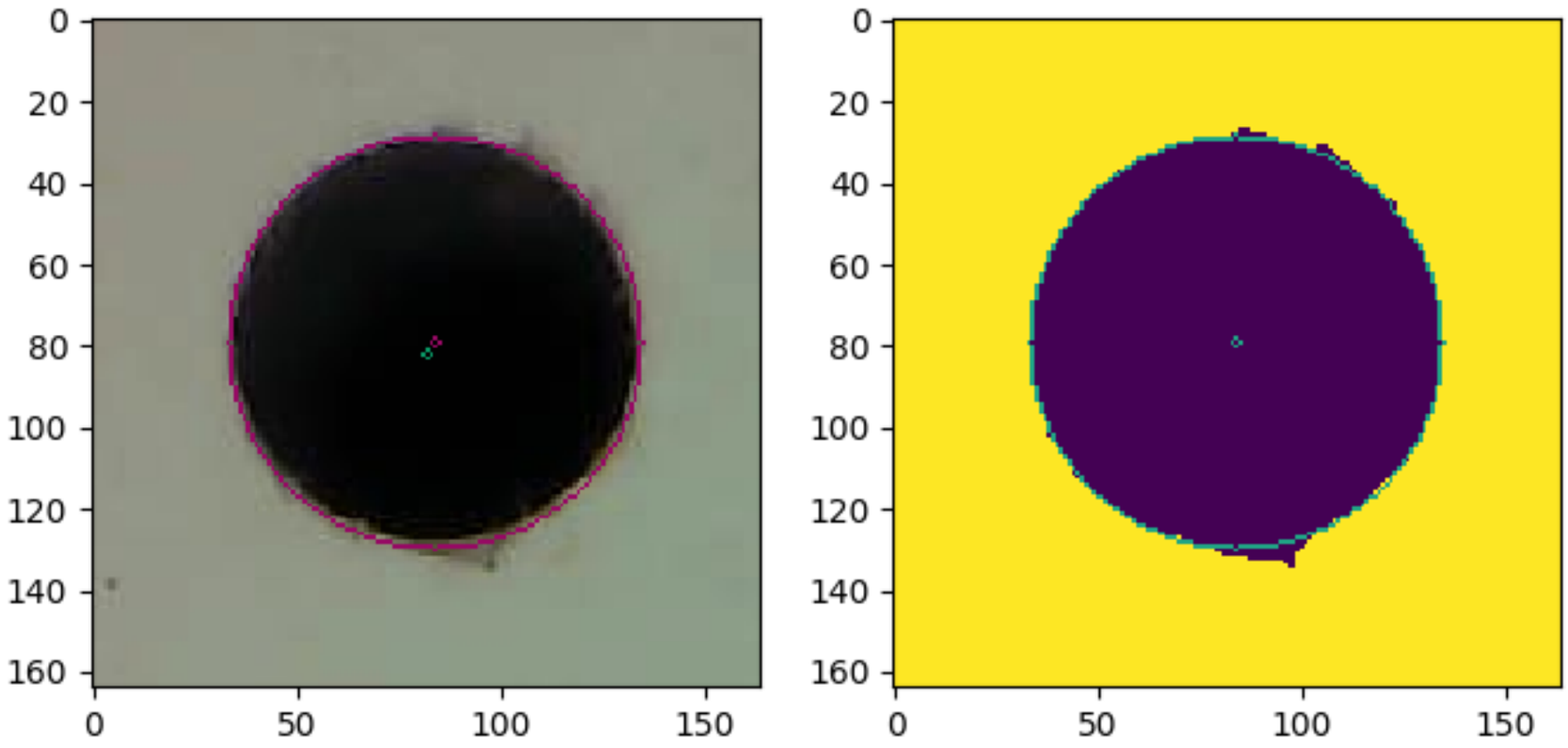
徐々に最小化されている様子がわかる（右の出力参照）。

時間もそれほどかかっていない。

```
learning_rate = 0.01, steps = 50
initial_position = [356.09999847, 244.09999847, 51.4]
はじめのEsumの値:294.19271881806037


gradient[0]=[   3.19995117   17.19995117 -127.8826313]
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   55.39633]

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

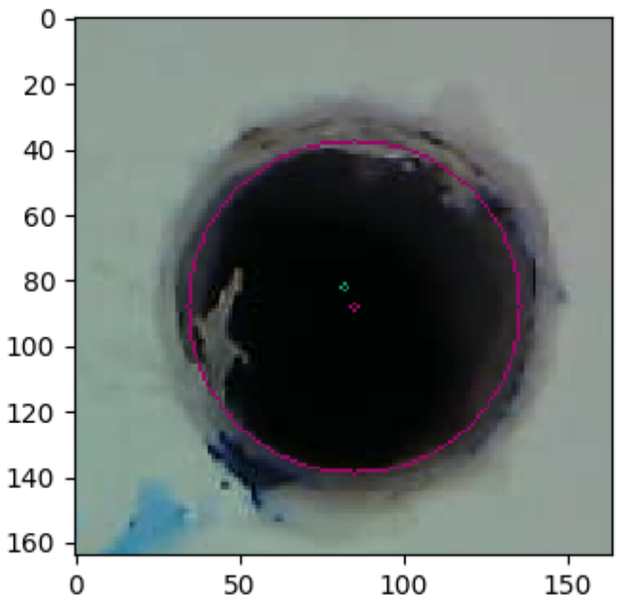
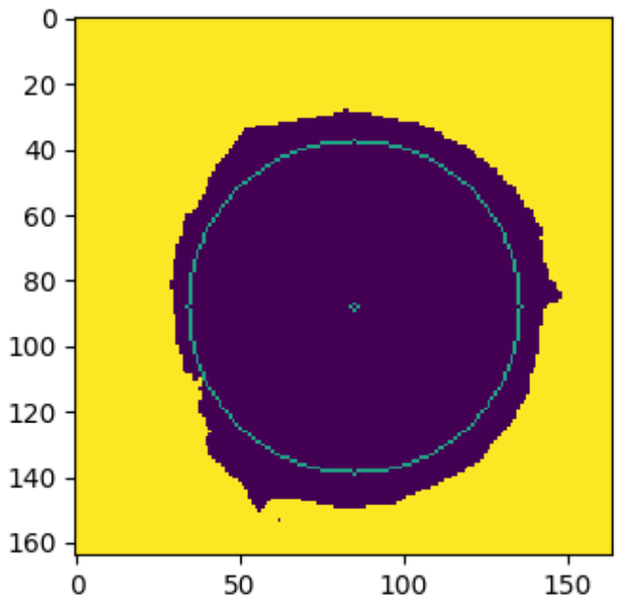
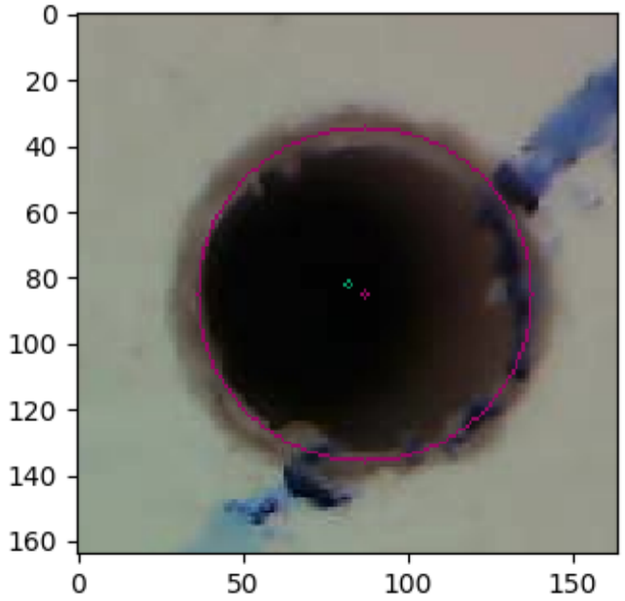
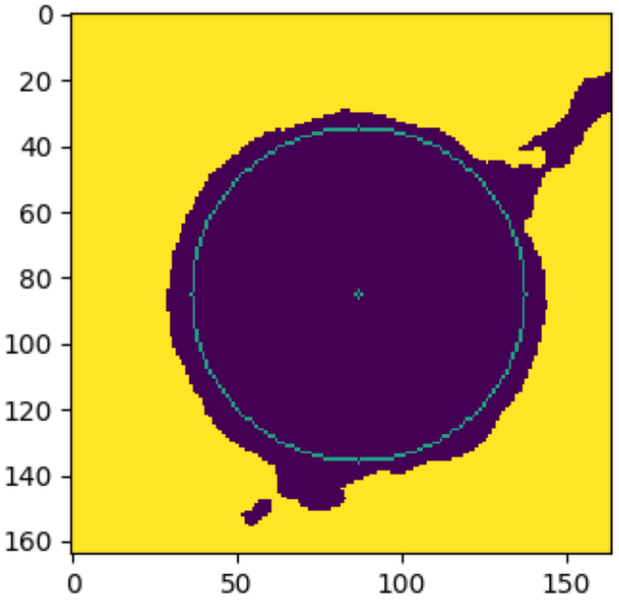



```
learning_rate = 0.01, steps = 50
initial_position = [364.0999984741211, 230.0999984741211, 52.4]
はじめのEsumの値:50.57412827491975
勾配法が見つけた最小値でのx: [363.8125     229.375       51.08272056]
time:0.043950080871582O3 sec
Esumの最小値:13.078075424774163
```

(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]
time:0.04877614974975586 sec
Esumの最小値:106.15383203261482
```

(x, y, r):(652.1, 254.1, 52.4)
→(652.1, 253.8, 58.1)

Esum : 626.5 → 106.1

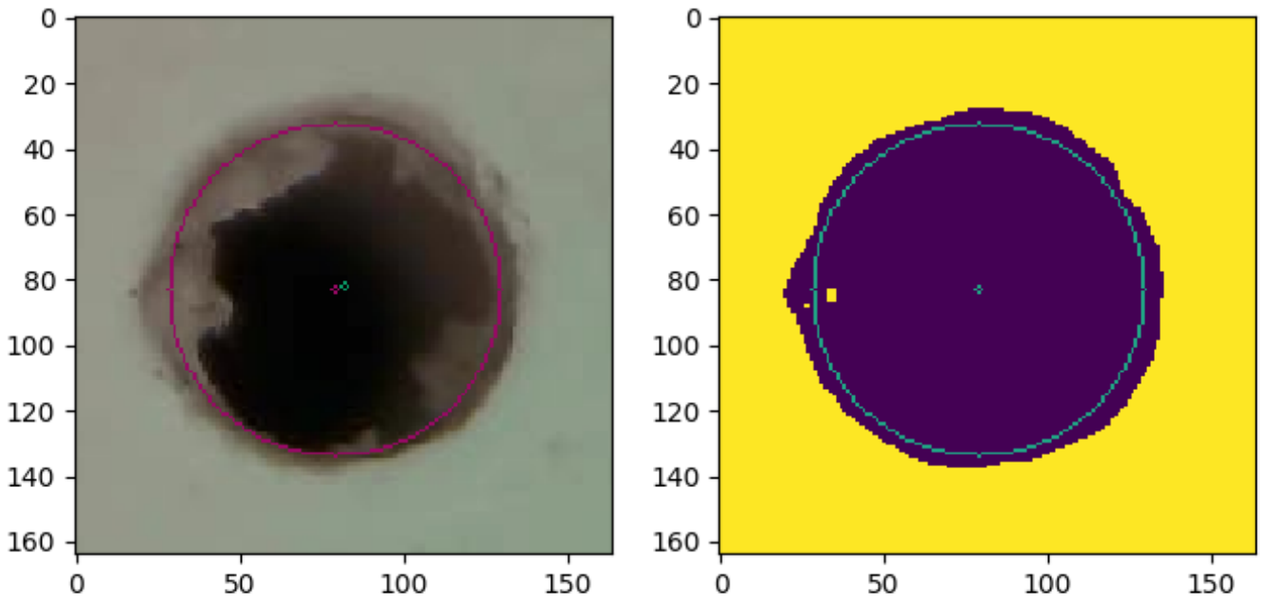
```
learning_rate = 0.01, steps = 50
initial_position = [659.09999847, 217.09999847, 52.4]
はじめのEsumの値:651.4675248444642
勾配法が見つけた最小値: [658.1875  217.625    57.347693]
time:0.044788360595703125 sec
最小値でのEsumの値:242.06030867727364
```

(x, y, r):(659.1, 217.1, 52.4)
→(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)

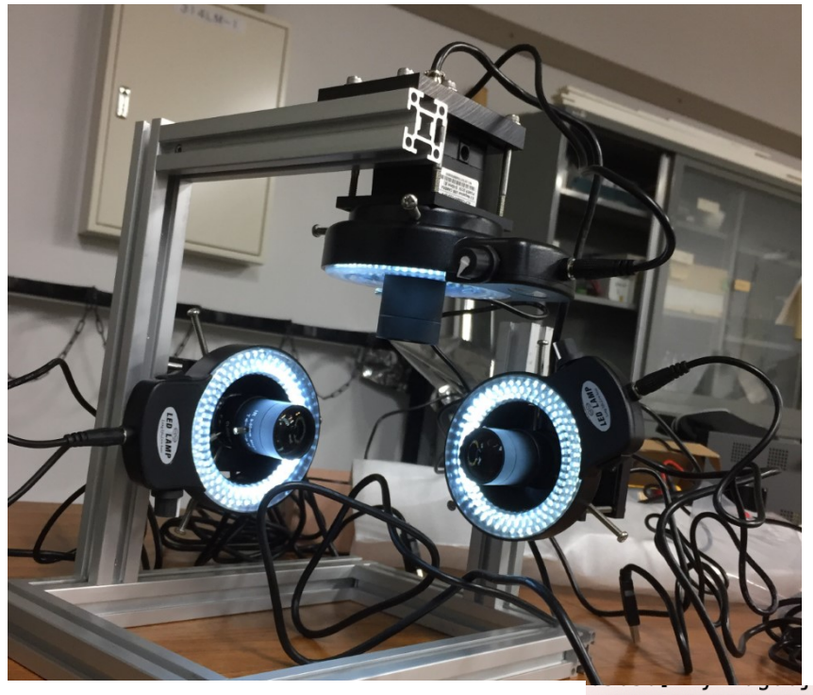
→(356.0, 243.6, 55.4)

Esum : 182.3 → 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。
- Esum の最小値についても good / bad 選別パラメータになるかもしれない

エッジ検出、バンプ検出

カメラ・照明



<u>撮影ジグの様子</u>

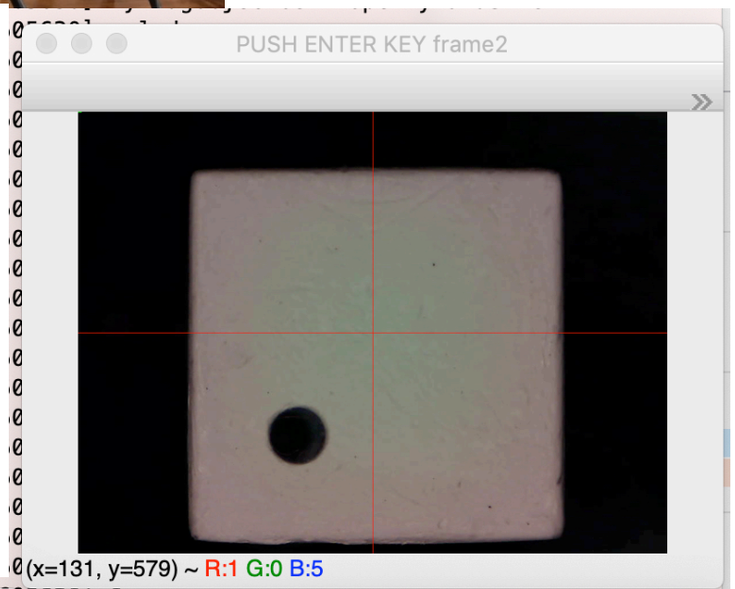
- カメラと照明を3方向に用意

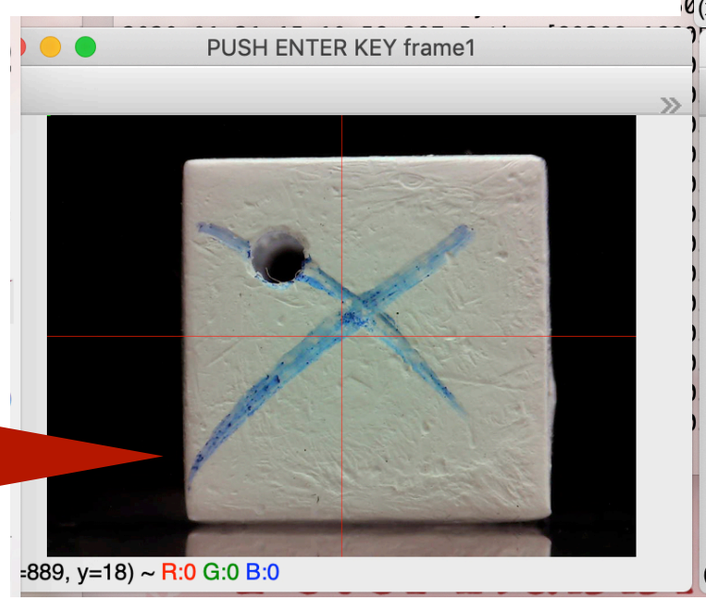
<u>3方向から光を当てる</u>

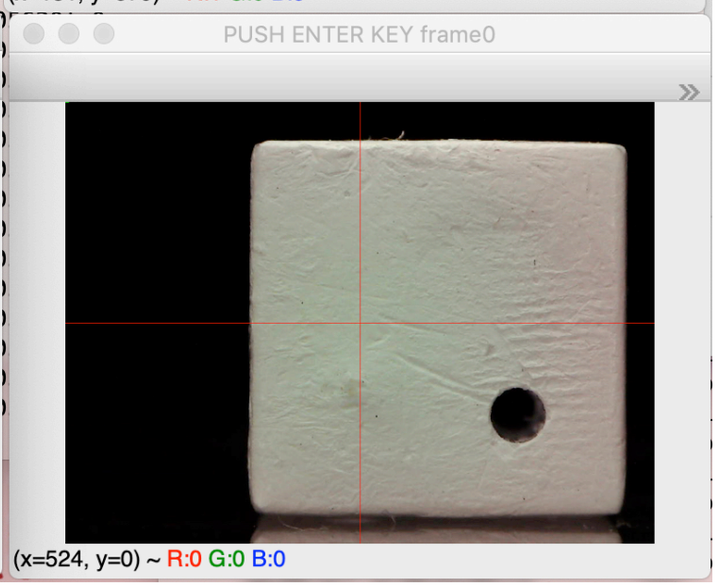
→ 撮影面に対し、2つの側面も照らされてしまう。

輪郭検出の threshold に影響が出るかもしれない。

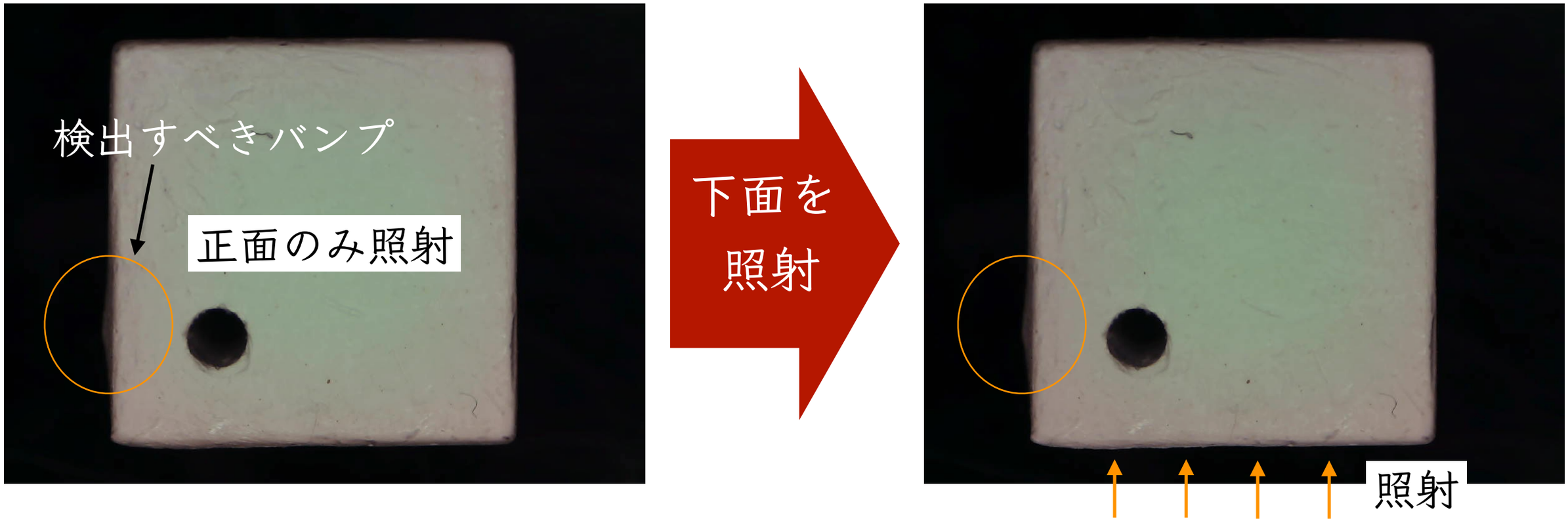
簡単なキューブ台座をつくり見え方を確認する（企業に依頼）。

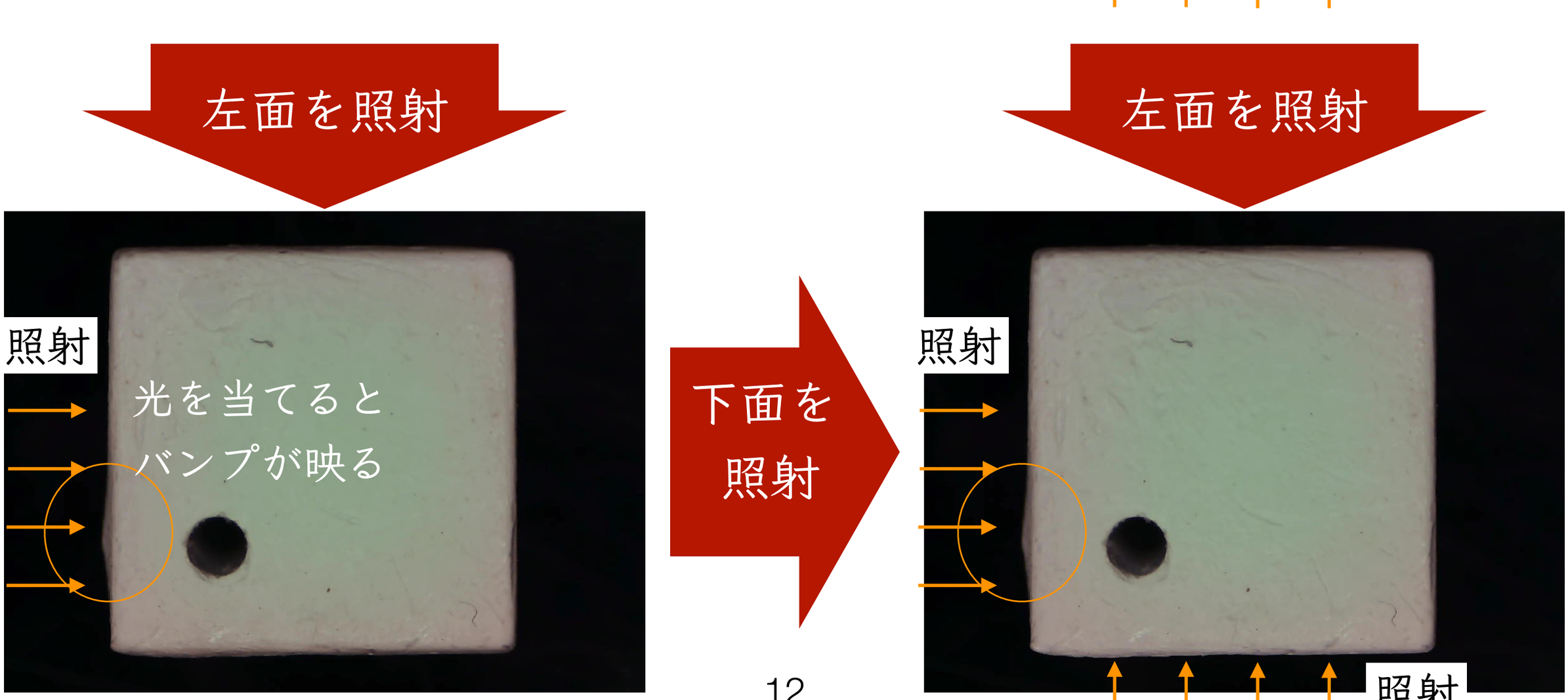






一見綺麗に写っている。
実際に影響がでるか不明。


検出すべきバンプ
正面のみ照射
下面を
照射
照射


左面を照射
照射
光を当てると
バンプが映る
下面を
照射
左面を照射
照射
照射

想定される点

- 穴の位置の決定：最も近いエッジからの距離で穴位置を定義していたが、改善すべきか？

- 光の当たらない辺が欠けているように見える（あるいは光の当たる辺が余計に出っ張って見える）。問題になるかもしれないし、相殺される可能性も考えられる。

- 全6面を撮影するので、バンプがある場合にはどこかしらで検出されるはず。

- <u>一度再現性のある台座での撮影・確認をする必要あり</u>

今後の予定

キューブの購入

- 松原さんがロシアからシンチレータキューブ12000個購入予定（3/10あたり）

- 4月頃にキューブチェック、その後溶着での組み立て

- 準備が万全なら一日6時間で二週間弱の計算

- バイトを雇う可能性？

まとめ

穴検出

- 穴辺上の点を円でフィット、カイ二乗値を最小化して高精度で穴検出できるようになった。

エッジ検出・バンプ検出

- 現行の照明で検出・選別に影響が出ないか、台座を作って確認する。

今後の予定

- 大量検査 (12000個)の予定が立った。学会までには準備を完了させる。