2D image calibration in digitizing process

Viktor Zerkin
International Atomic Energy Agency, Nuclear Data Section

EXFOR Workshop, 22 - 25 October 2018, Vienna, Austria

Interactive 2D-calibration for picture transformations on Web

A94	Zerkin Pikulina Chen JCPRG	Study problems in 2D calibration of original pictures, and process of approval of results of digitizing using plotting facilities.

2DX-calibration:

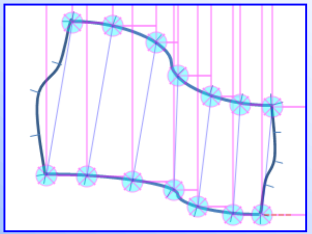
X-Axes bottom-top are marked by user; X{n} sent to ZVView which produces transformed picture.
Implemented in 2015-2016.

2DY-calibration:

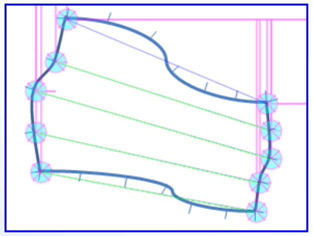
Y-Axes left-right are marked by user; Y{m} sent to Java-Servlet to produce 2DXY calibration points.
Done in 2017.

2DXY-calibration:

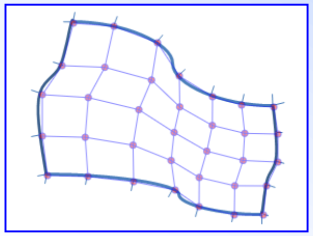
X{n} and Y{m} calibration points are processed to produce grid of XY{n×m} calibration points.
Done in 2017.



×



→



Goal-2017:

1.Distort output image according to 2DXY calibration: XY{n×m} grid.

Plan-2017:

~~Extend ZVView to transform output picture according to 2DXY calibration grid: XY{n×m} calibration points.~~

Development in 2018: new algorithms, new technology, new goal

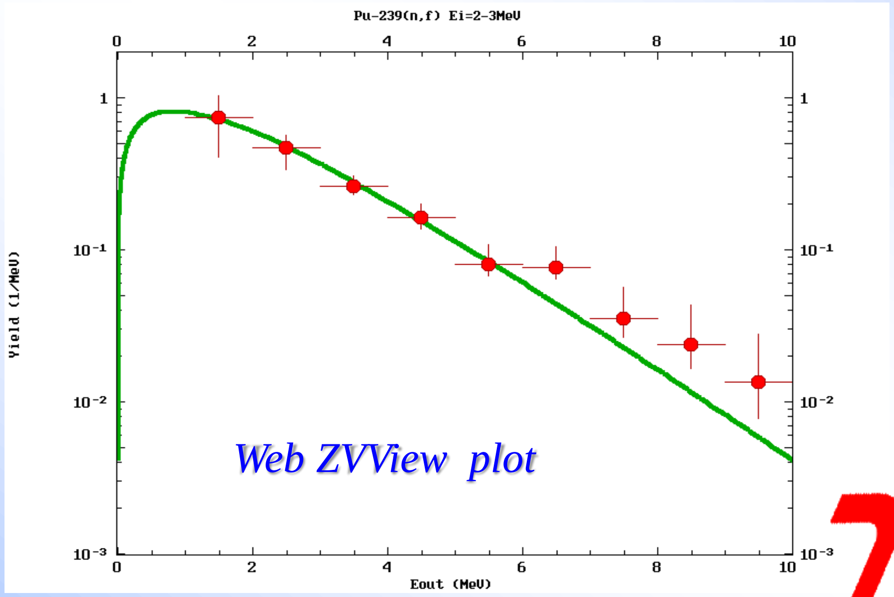
Goals-2018:

1.Distort output image...
2.Recover original input picture according to 2DXY calibration: XY{n×m} grid.

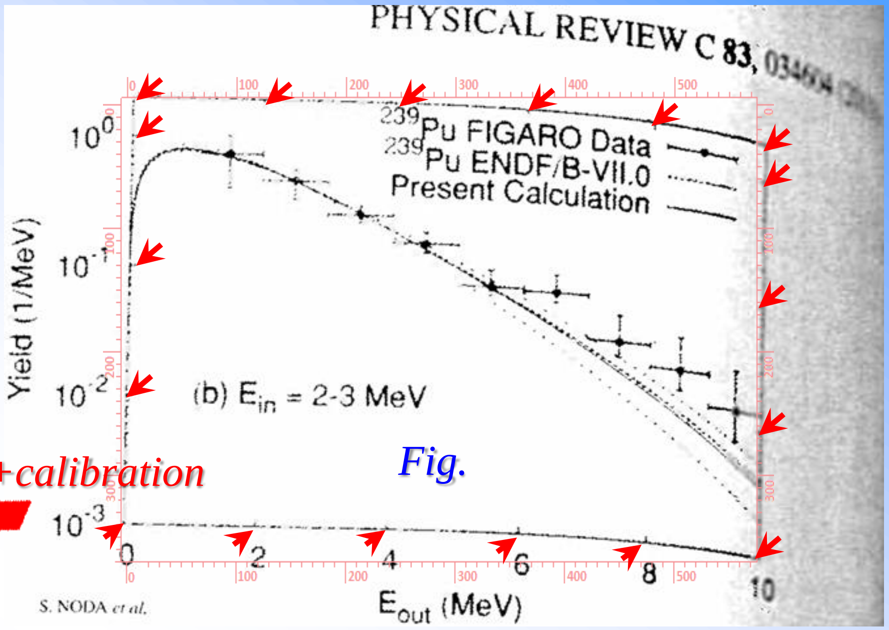
Todo:

Extend 2D calibration tool to deal with more difficult cases, e.g. missing points, missting top and right exes, etc. Logarithmic scales.

1. Distort plot for checking result of digitizing



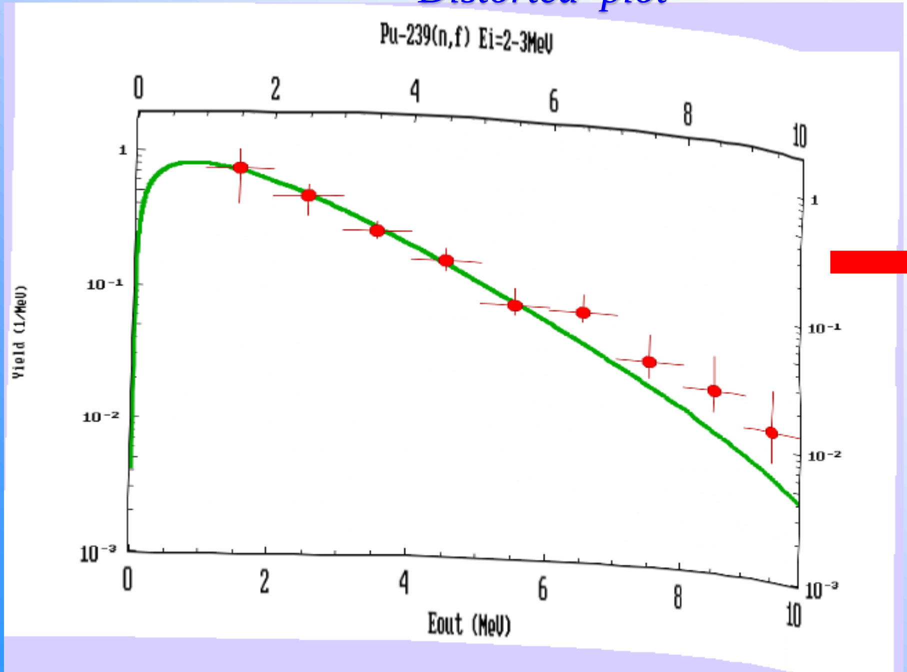
Web ZVView plot



+calibration

Fig.

Distorted plot



Distorted plot: mono color

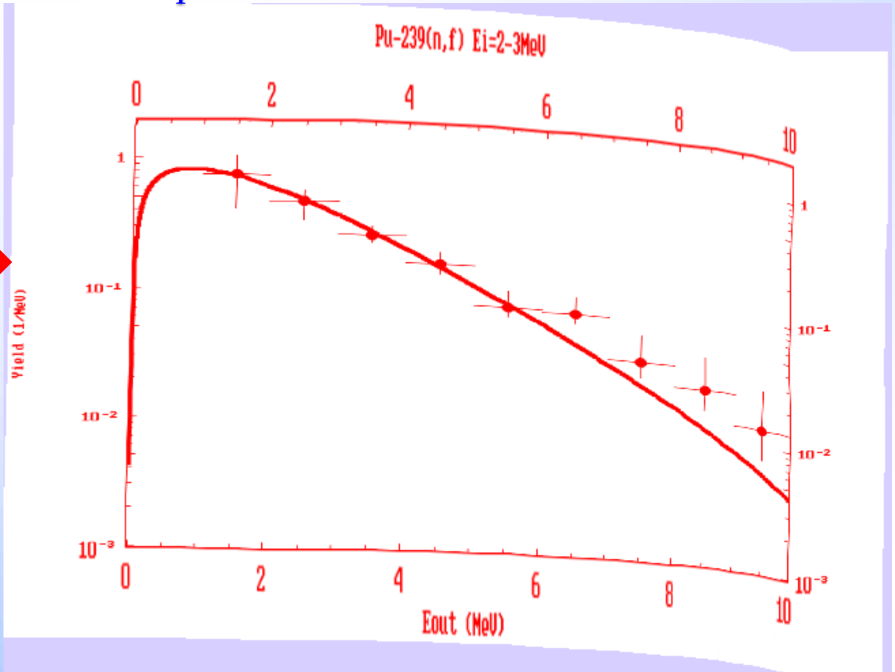
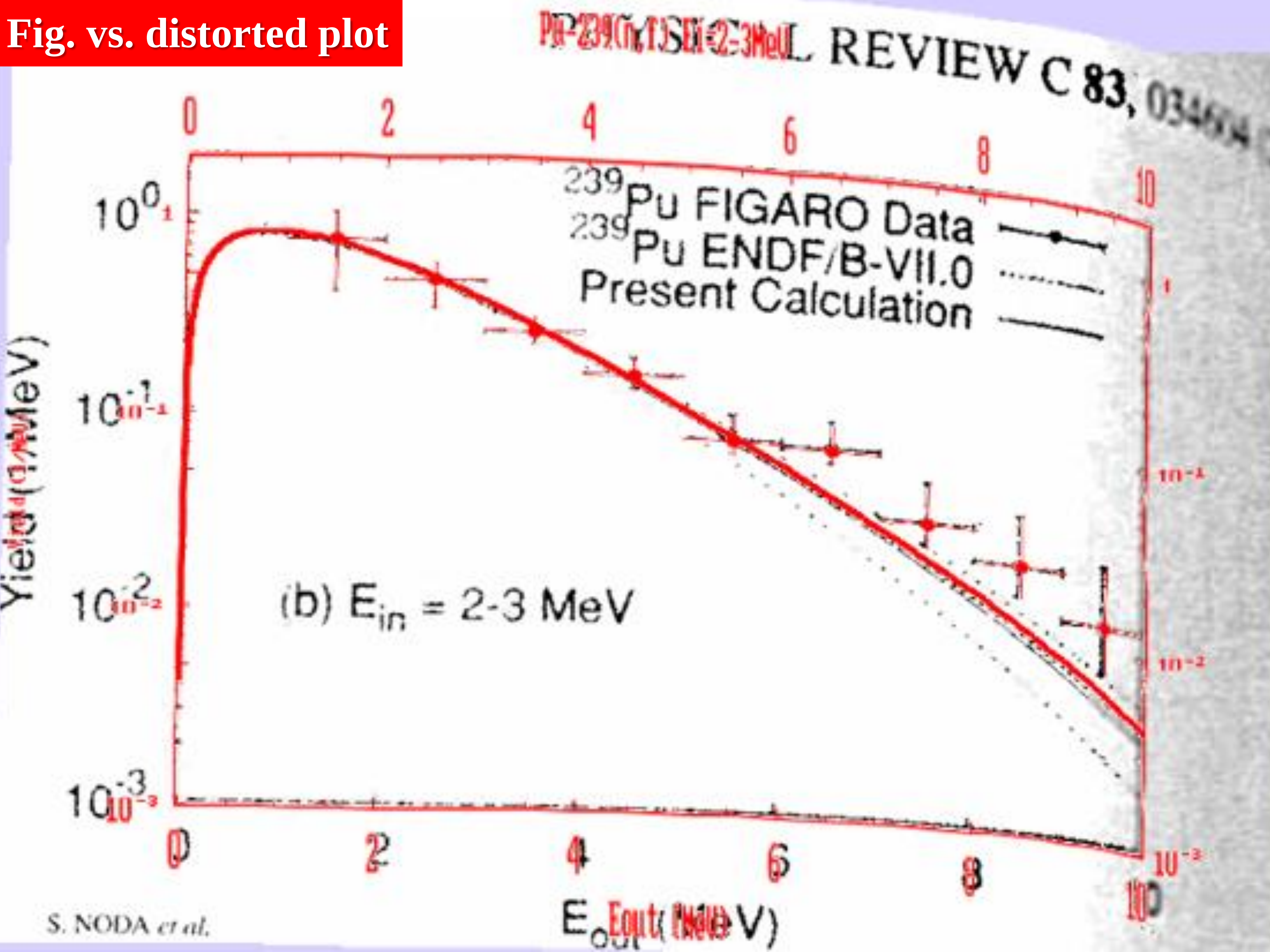


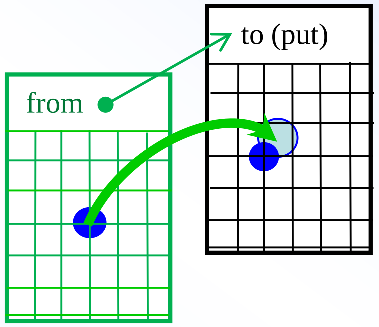
Fig. vs. distorted plot



Algorithms of distorting picture

Algorithm-1 (2014)

loop:	pixel(x_0,y_0) //from
calculate:	x_1=(int)$F_1(x_0,y_0)$; y_1=(int)$F_2(x_0,y_0)$; //int(int)
	$color_1=color_0$
set:	pixel(x_1,y_1):$color_1$

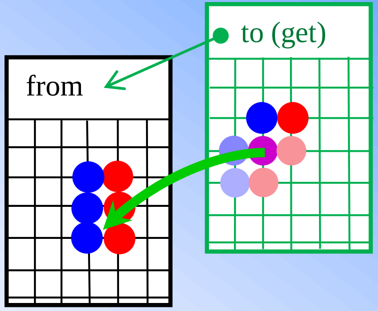


Done: by ZVView during plotting

Output: image/gif

Algorithm-2 (2018)

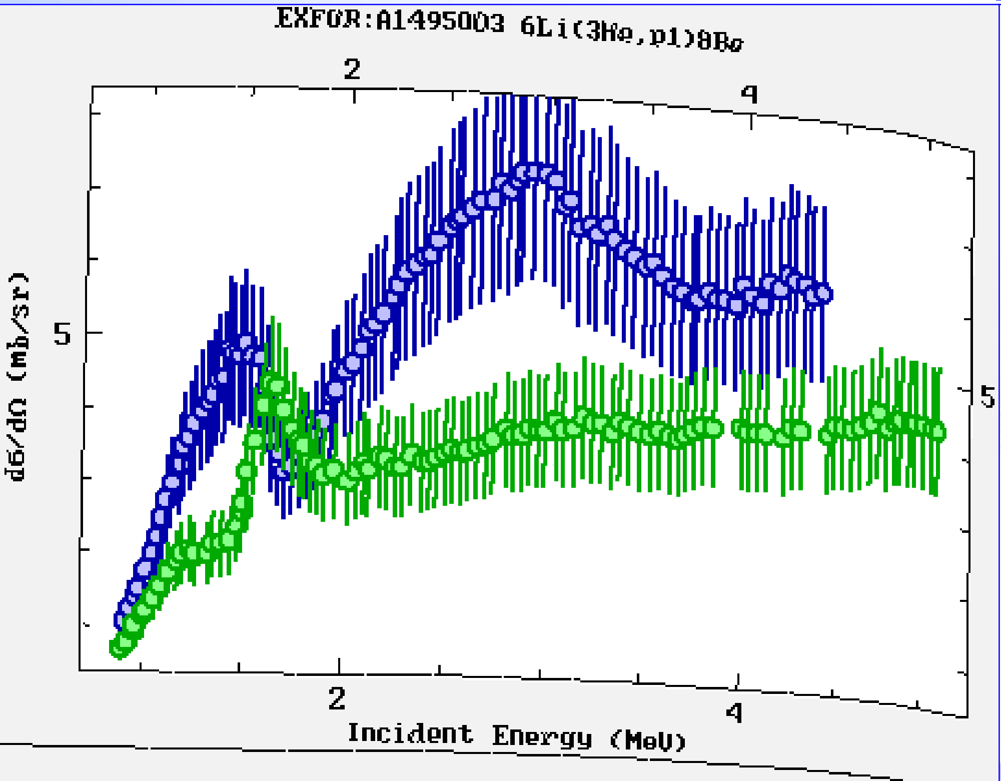
loop:	pixel(x_1,y_1) //to
calculate:	$x_0=G_1(x_1,y_1)$; $y_0=G_2(x_1,y_1)$; //double(int)
	$color_1=G_3(x_0,y_0,colors[])$ //averaged
set:	pixel(x_1,y_1):$color_1$

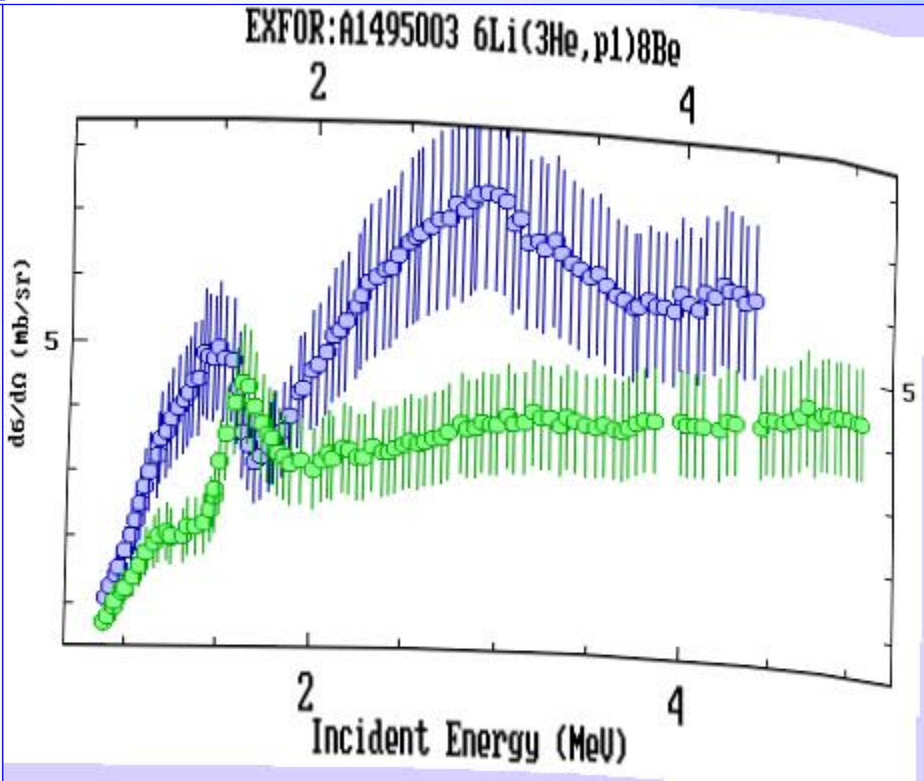


Done: by Java-code

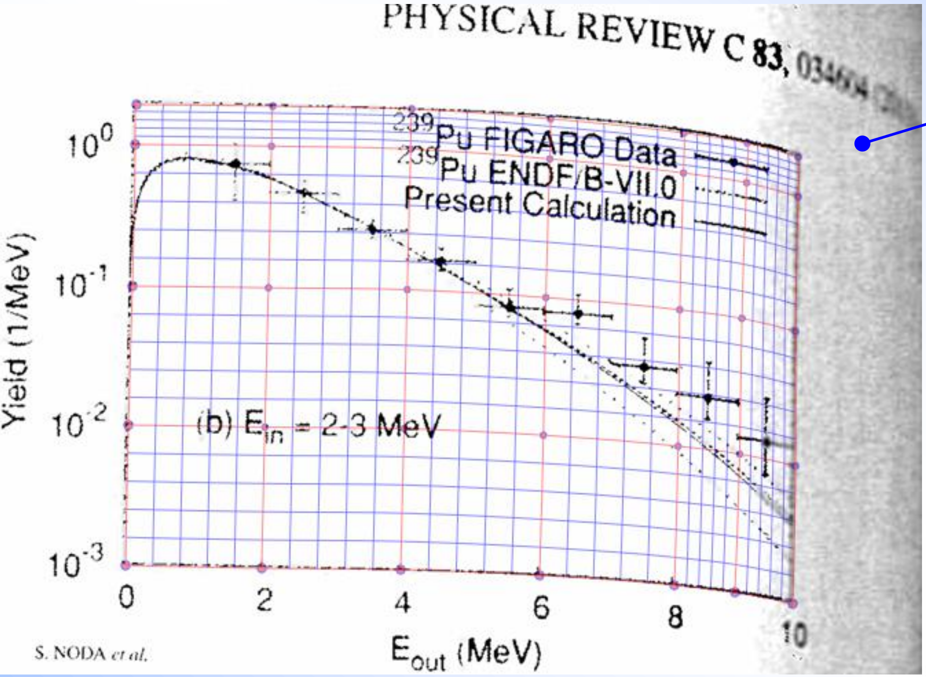
Input: image:gif/png/jpeg

Output: image/png





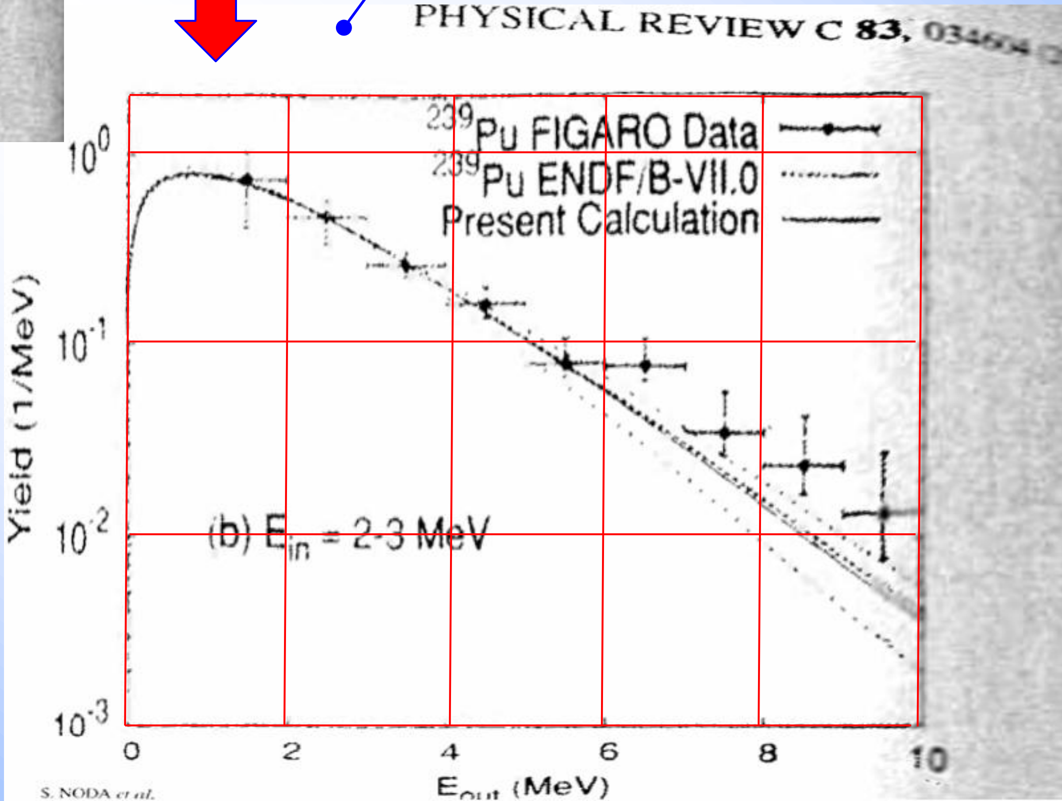
2. New task: recover “bad” image //2018



“Bad” image + calibration

Processing image with 2D calibration

Improved image



We can try to solve inverse problem: to improve image before digitizing

2D-calibration: operations

2X-calibration //2015

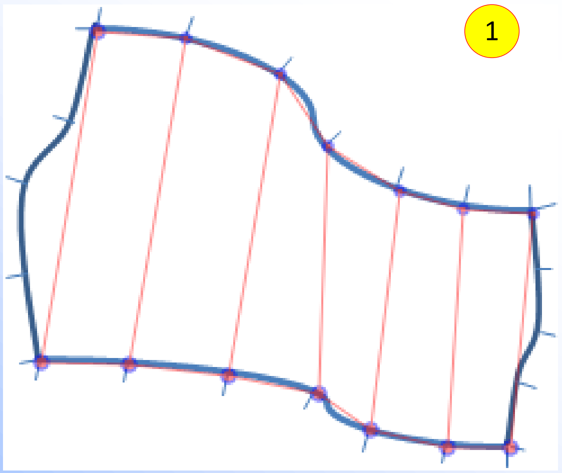
X-Axes bottom-top are marked by user: X{2n}

2Y-calibration //2017

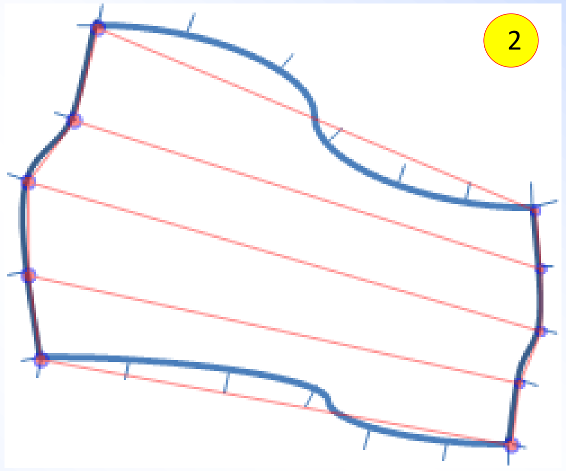
Y-Axes left-right are marked by user: Y{2m}

2XY-calibration //2017

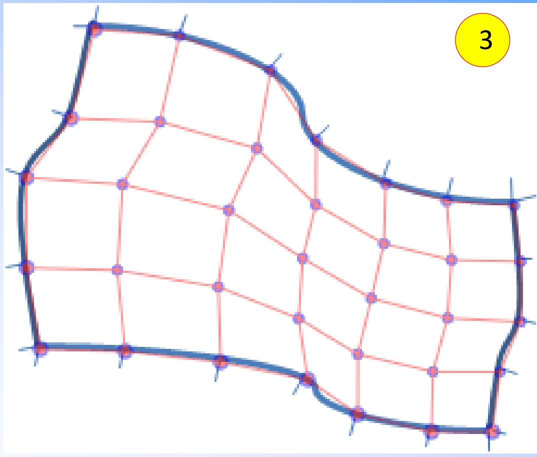
Processing X{2n} and Y{2m}...
Result: grid of XY{n×m} points



×

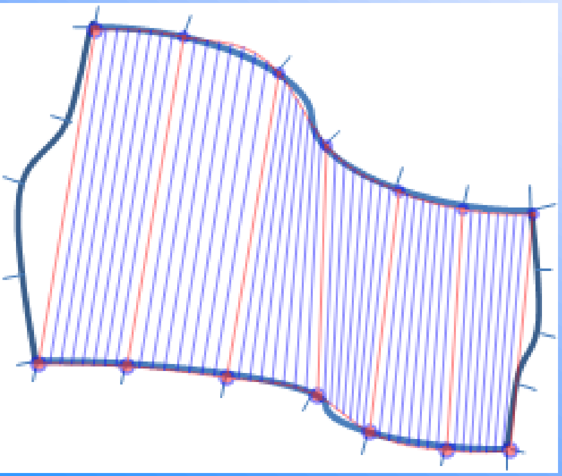


→

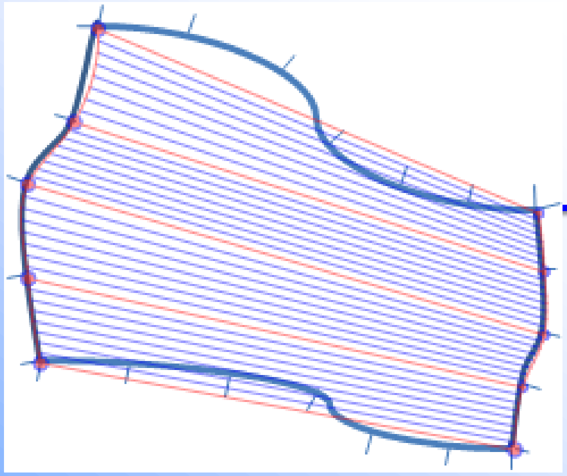


4 Smoothing 2XY-calibration //2018

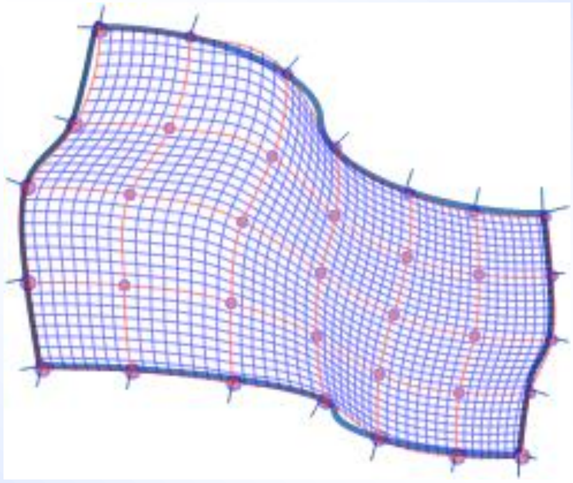
User defines steps for interpolation between initial calibration points on X and Y axes. Implemented in 2018.



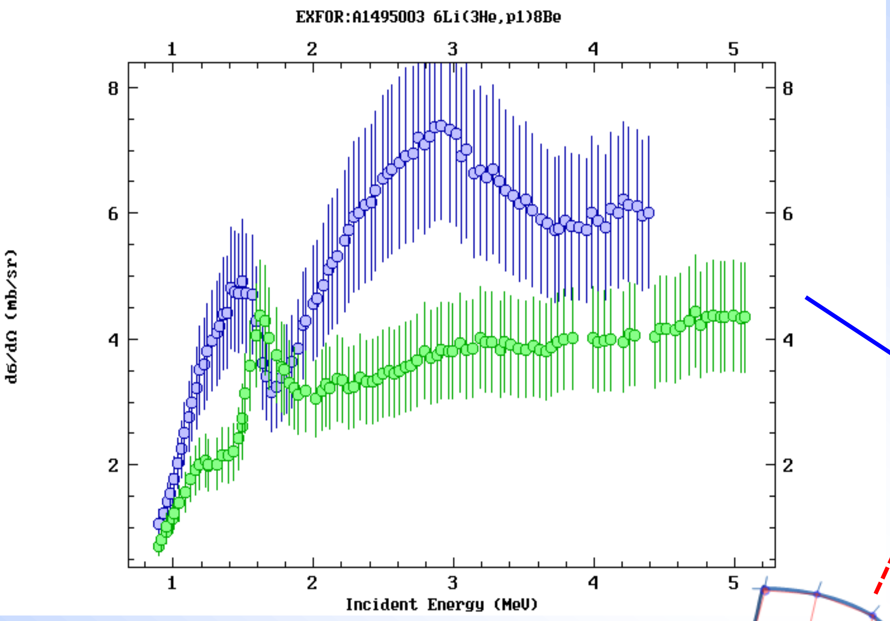
×

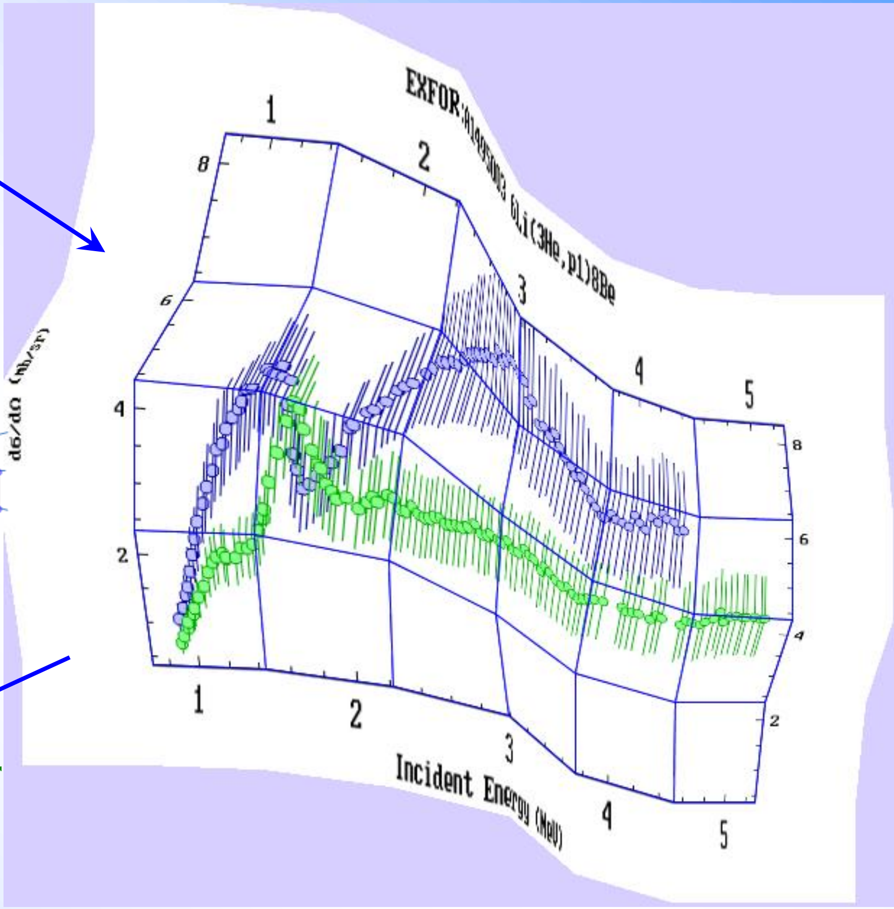


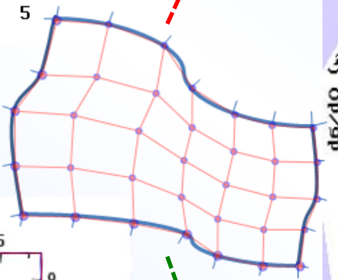
→

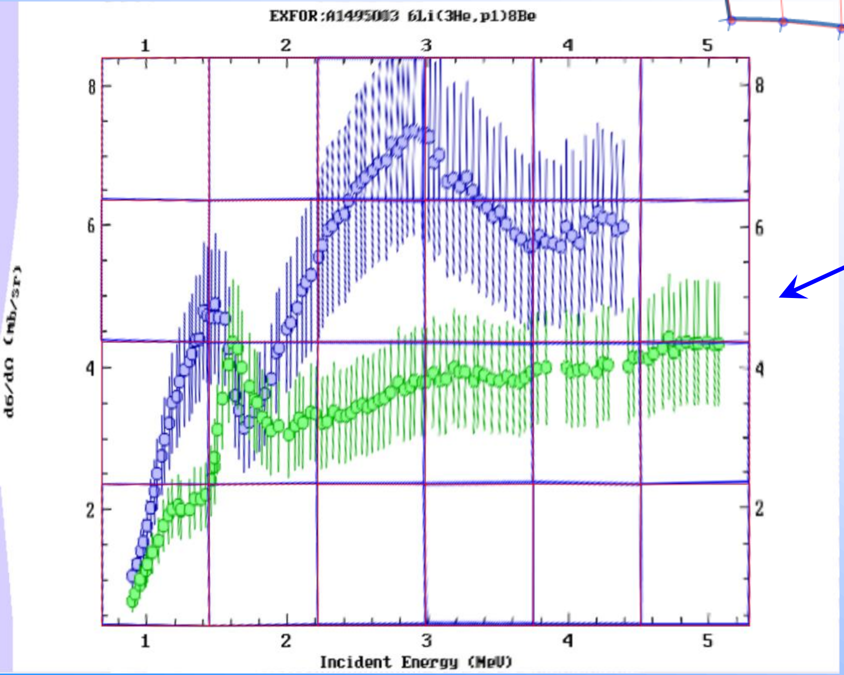


Self test-1

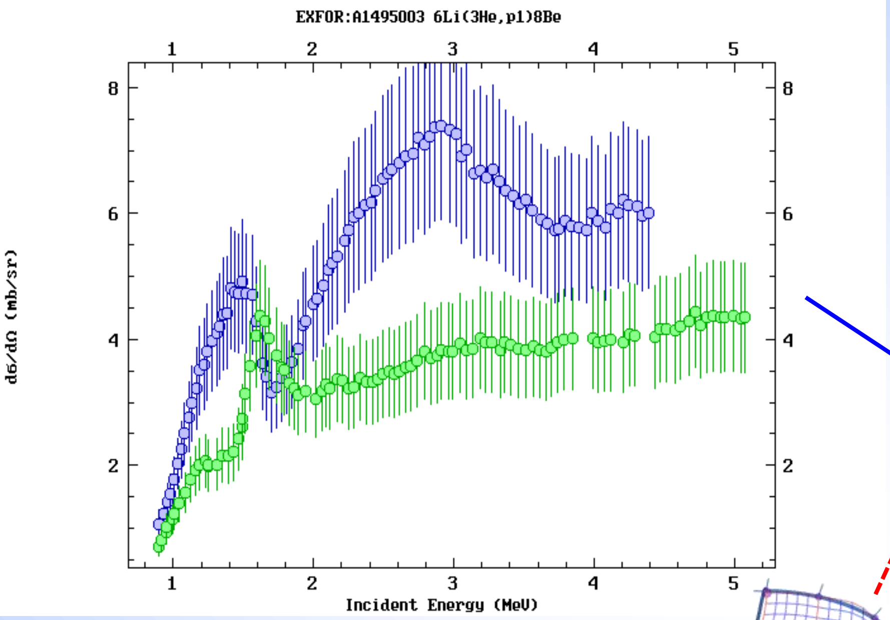

EXFOR:A1495003 6Li(3He,p1)8Be
dσ/dΩ (mb/sr)
Incident Energy (MeV)


EXFOR:A1495003 6Li(3He,p1)8Be
dσ/dΩ (mb/sr)
Incident Energy (MeV)

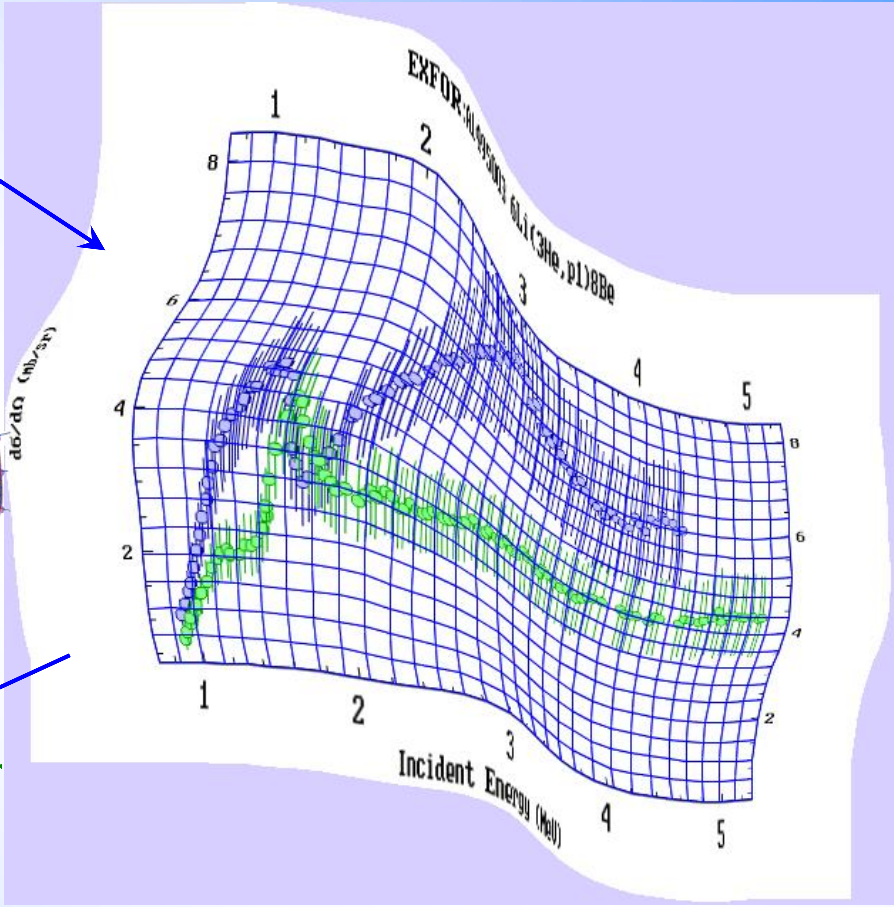

Distort
1
2D-calibration
2
Recover


EXFOR:A1495003 6Li(3He,p1)8Be
dσ/dΩ (mb/sr)
Incident Energy (MeV)

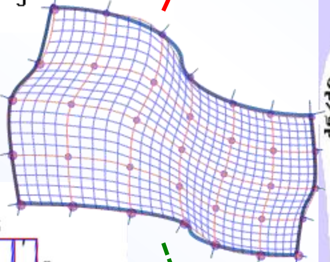
Self test-2


EXFOR:A1495003 6Li(3He,p1)8Be
dσ/dΩ (mb/sr)
Incident Energy (MeV)

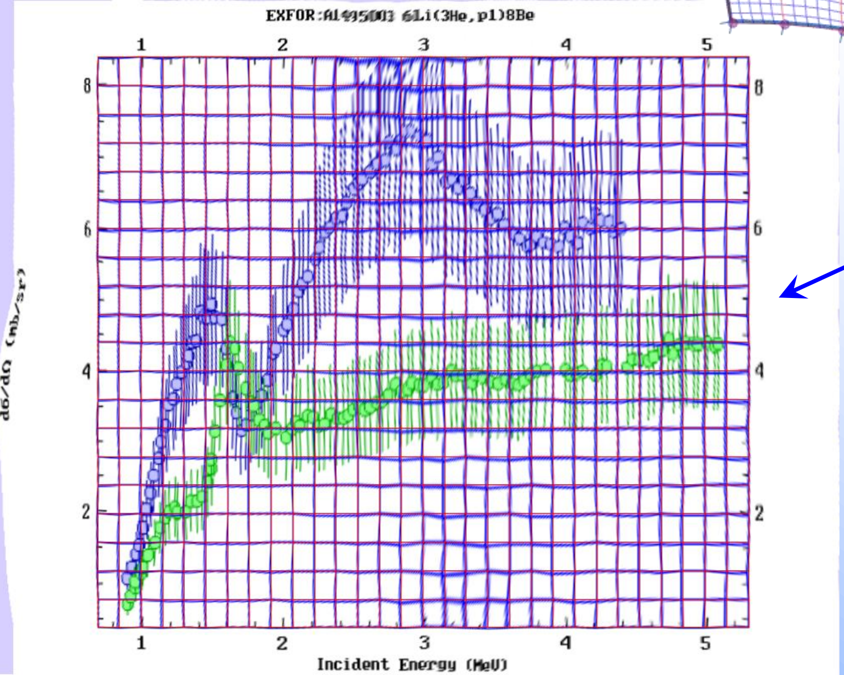
Distort
1


EXFOR:A1495003 6Li(3He,p1)8Be
dσ/dΩ (mb/sr)
Incident Energy (MeV)

2D-calibration



2
Recover


EXFOR:A1495003 6Li(3He,p1)8Be
dσ/dΩ (mb/sr)
Incident Energy (MeV)

Concluding remarks

1. 2D calibration algorithms (mathematics) and software were completely redesigned and significantly improved
2. New web-tool built on the 2D calibration can make it practically useful to improve quality of digitization
3. Next tasks:
   a) to extend 2D calibration tool to deal with more difficult cases, e.g. missing points, absent top and right exes, etc.
   b) improve format to save/restore calibration points
   c) Log scales

Thank you.