QAserver

Release 1.0

Denis Brojan

Dec 01, 2019

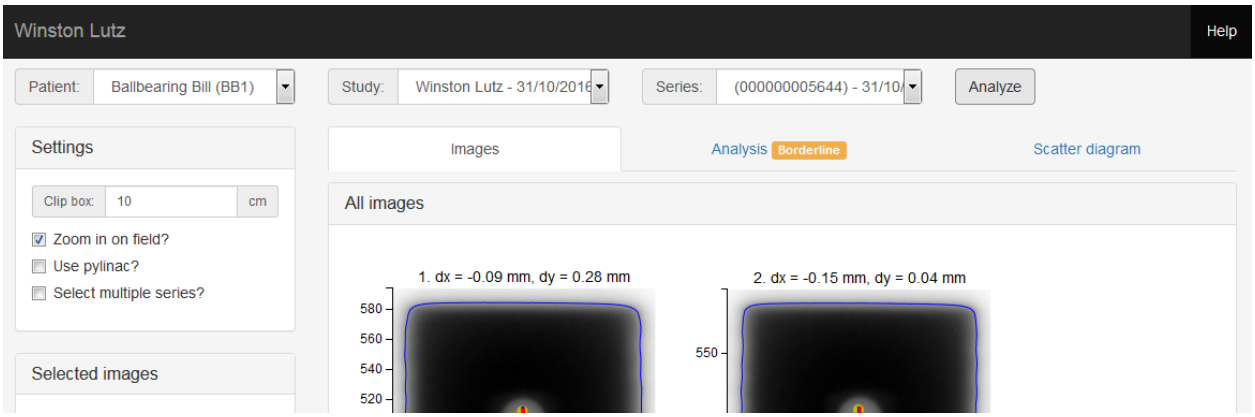
CONTENTS

CHAPTER

ONE

ABOUT

QAserver connects Pylinac and Orthanc with a simplistic web interface. It was built to speed up the image analysis process with Pylinac, without having to switch computers or work manually with dicom files. The procedure is easy: acquire the image, send the image from the imaging computer directly to Orthanc, and then analyze it with Pylinac by using QAserver. Since all images are stored in Orthanc, you can re-analyze them anytime you want. You can do the analysis on any computer in your network.



Not all Pylinac's capabilities are implemented. Trajectory logs cannot be analyzed, the calibration module is missing, and there are some missing features in other modules. See the user guide.

QAserver contains little original code, just enough to connect Pylinac and Orthanc. Some dependencies are included in the distribution of QAserver, other dependencies must be installed separately. I tried to list included dependencies along with corresponding licenses. You can read about it in the bottom table.

You can contribute to this project with suggestions or bug reports: brjdenis2000@gmail.com.

Warning: QAserver may have serious errors. Always run the analysis with pure Pylinac to see what the result should be. For my part of the code, this license applies: *License*.

Software	Version	Used for	Licence
Resttoolbox	/	Communication with Orthanc	GPLv3
Bokeh	1.3.4	JS files for HTML	BSD 3-Clause
mpld3	0.3	d3 plotting	BSD 3-Clause
Bootstrap	3.4.1	Styles for HTML	MIT, (c)2019 Twitter
Bootstrap-datepicker	1.4.1	Datepicker for HTML	Apache License 2.0
Bootstrap-select	1.13.9	Select button sytle	MIT
Popper	2019	Tooltips, dropdowns...	MIT
jQuery	3.2.1	Javascript	MIT
Glyphicons	Halflings	Icons	MIT
d3.v3.min.js	mpld3	Plotting	BSD 3-clause
mpld3.v0.3.min.js	mpld3 v0.3	Plotting	BSD 3-clause
Pylinac	2.2.6	Picket fence modification	MIT
math.js	6.2.2	Image Review calculation	Apache License 2.0

1.1 A sense of how it works

Orthanc allows to query dicom files (images) via its RESTful interface. The RestToolBox.py module, created by the authors of Orthanc, is used (with some modification) to connect to the Orthanc database, to pick the right images, and then to transfer them to the QA computer. Once the files have been transferred to the computer where QAserver is running, they will be parsed to Pylinac for analysis. QAserver will collect the results of the analysis, delete the files locally and send the results back to the web app. And that is more or less it.

CHAPTER

TWO

INSTALLATION

QAserver is designed to work with Orthanc and its RESTful interface. It will not work without it.

Follow this instructions to properly install QAserver. I suggest that you reserve a computer for QAserver that is connected to your hospital (local) network. There are no special requirements, a normal desktop computer should be sufficient.

> **Warning:** Do not make QAserver available to the public domain. QAserver does not implement any safety features, so by making QAserver public you are at risk of exposing your data.

QAserver cannot be installed as a Python package. A special procedure must be followed. Each version of QAserver is compatible only with specific versions of dependent software, so you must pay attention to the version of software you install within your Python environment.

2.1 Version 1.0 compatibility

The table below gives the versions of three dependencies that must be observed. More recent versions of dependencies, especially Pylinac and Bokeh, will not work with QAserver.

Software	Version
Python	>= 3.7
Pylinac	2.2.6 or 2.2.7
bokeh	1.3.4

Here is a list of version of other dependencies that were used during development:

Software	Version
Bottle	0.12.17
json	2.0.9
json	2.0.9
sqlite3	2.6.0
numpy	1.16.5
scipy	1.3.1
matplotlib	3.1.1
scikit-image	0.15.0
scikit-learn	0.21.3
pydicom	1.3.0
httplib2	0.14.0
prettytable	0.7.2

2.2 Instaling Anaconda

Download the latest Anaconda. The version I am using in this user manual is Anaconda3 2019.10.

When you install Anaconda, chose to add Anaconda to your PATH environment variable. If you chose not to, then you may have problems running the servers with Windows Task Scheduler.

After you install Anaconda, you can either use the "base" environment, or make a new environment for QAserver. The instructions that follow cover both ways of doing the installation. Making a special environment is necessary if you are using much more recent versions of Anaconda/Python, but need to install older versions of dependencies.

2.3 Download QAserver

Download QAserver archive from: Link

Unzip it into a special directory. Give it the name "qaserver". After that open Command Prompt or Anaconda Prompt with Administrator privileges, and go to that folder and try to start the server with

```
python start.py
```

You should get import errors. The first one will probably be a missing "bottle" module. So basically, what you need to do is install all the missing packages until the server starts without giving errors. As said before, you have two options: installing packages in the base environment or create a special environment for QAserver.

2.4 Installing missing packages in the base environment

Warning: Using `pip` to install certain packages may destroy your python distribution, so follow this to install everything correctly. Pay attention to when `conda` is used instead of `pip`.

Start Anaconda Prompt with Admin privileges. Your base environment should be active by default.

- **Install bottle with** `pip install bottle`
- **Install httplib2 with** `pip install httplib2`
- **Install pylinac with** `pip install pylinac==2.2.7`
- **Reinstall bokeh with** `pip install bokeh==1.3.4`
- **Install paste with** `pip install paste`
- **Install prettytable with** `pip install prettytable`

Now you should be able to start the server with `python start.py`. Some additional steps are necessary to fix a couple of Pylinac's modules.

```
(base) E:\qaserver>python start.py
Bottle v0.12.17 server starting up (using PasteServer())...
Listening on http://127.0.0.1:80/
Hit Ctrl-C to quit.

serving on http://127.0.0.1:80
```

2.5 Installing missing packages in a new environment

Open Anaconda Prompt with Admin privileges and type this to create an environment called "qaserver" with a specific version of Python:

```
conda create -n qaserver python=3.7
```

Then active this environment with

```
activate qaserver
```

Within the new environment you must install missing packages (keep to the order as defined below):

- **Install bottle with** `pip install bottle`
- **Install scipy with** `conda install scipy`
- **Install httplib2 with** `pip install httplib2`
- **Install scikit-image and scikit-learn with** `conda install scikit-image`

  `conda install scikit-learn`
- **Install pylinac with** `pip install pylinac==2.2.7`
- **Install jinja with** `pip install jinja2`
- **Install bokeh with** `pip install bokeh==1.3.4`
- **Install paste with** `pip install paste`
- **Install prettytable with** `pip install prettytable`

Now you should be able to start the server with `python start.py`. Some additional steps are necessary to fix a couple of Pylinac's modules.

2.6 Fixing planar imaging module

Find the planar_imaging.py module of pylinac. Go to line 538 and change the `phantom_angle(self)` function so that the function returns a 45 degree angle. Like this

```
    @property
def phantom_angle(self):
    """The angle of the phantom.

    Returns
    -------
    angle : float
        The angle in degrees.
    """
    #return -np.rad2deg(self.phantom_ski_region.orientation)
    return 45
```

This fix is valid only for pylinac version 2.2.7.

2.7 Fixing the Catphan module

Find the ct.py module of Pylinac. Go to line 850 and edit the module so that it looks like this:

```
@property
def overall_passed(self):
    """Whether there were enough low contrast ROIs "seen"."""
    return sum(roi.passed_cnr_constant for roi in self.rois.values()) >= self.
↪tolerance
```

This fix is valid only for pylinac version 2.2.7.

2.8 Fixing the VMAT module

Find the vmat.py module of Pylinac. Go to line 423 and edit the module so that it looks like this:

```
@property
def passed(self) -> bool:
    """Return whether the segment passed or failed."""
    return abs(self.r_dev) < self._tolerance * 100
```

This fix is valid only for pylinac version 2.2.7.

2.9 Setting up Orthanc

Download Orthanc from their webpage, and install it in a separate folder. You can even install it on a different computer.

The folder should contain a special JSON configuration file. If the file is not already present, you can generate it by running this in Windows Command Prompt (after opening CMD go to the Orthanc folder):

```
orthanc --config=Configuration.json
```

Open the file in a text editor like Notepad++ (be careful, sometimes there is a problem with the encoding, and changing the file will make it unreadable by Orthanc). First, set the HTTP port for REST services and GUI.

```
"HttpServerEnabled" :  true
```

```
"HttpPort" :  8042
```

Next, set the dicom port to either 4242 or 104. Port 104 is usually used for dicom transport, but 4242 will also work.

```
"DicomServerEnabled" :  true
```

```
"DicomPort" :  104
```

When you configure your imaging workstation to send images to Orthanc, use these settings.

Change the UnknownSopClassAccepted to true:

```
"UnknownSopClassAccepted" :  true
```

Set RemoteAccessAllowed to true:

```
"RemoteAccessAllowed" :  true
```

Set AuthenticationEnabled to true:

```
"AuthenticationEnabled" :  true
```

Define credentials for accessing the server:

```
"RegisteredUsers" :  { "orthancuser" :  "orthancpass" },
```

Delete "//" in front of the line to uncomment it. Chose appropriate username and password of the orthanc user. At a later moment you will tell QAserver to use these credentials when connecting with Orthanc.

Save the configuration file and run the Orthanc server:

```
Orthanc configuration.json
```

If everything is OK, you should be able to access the GUI of the server at

http://127.0.0.1:8042.

A hint: If you want to stop the server, press CTRL + C.

Orthanc is independent from QAserver. You will tell QAserver what settings must be used to access it.

2.10 Configuring QAserver

All the settings of QAserver are contained in **config.ini**. **Do not leave empty definitions!**

- **INSTITUTION** Set the name of your institution. It will be printed in the menu page.
- **PLWEB_IP and PLWEB_PORT** The IP address and PORT of the computer that will be running QAserver. You can check it in your network settings. I suggest that you use PORT = 80.
- **ORTHANC_IP and ORTHANC_PORT** The IP address and port where Orthanc is running. If Orthanc is on the same computer as QAserver, then match the IP address with PLWEB_IP.
- **PLWEB_USERNAME and PLWEB_PASSWORD** The username and password you will be using to connect to QAserver. Only one username/password is possible.
- **ORTHANC_USERNAME and ORTHANC_PASSWORD** The Orthanc username and password that you defined in the Orthanc configuration file.

There are additional settings for each module. They are explained in a separate chapter.

2.11 Starting the servers as Windows Tasks

Once everything is set up, you can start the QAserver in Anaconda Prompt (or Command Prompt) with:

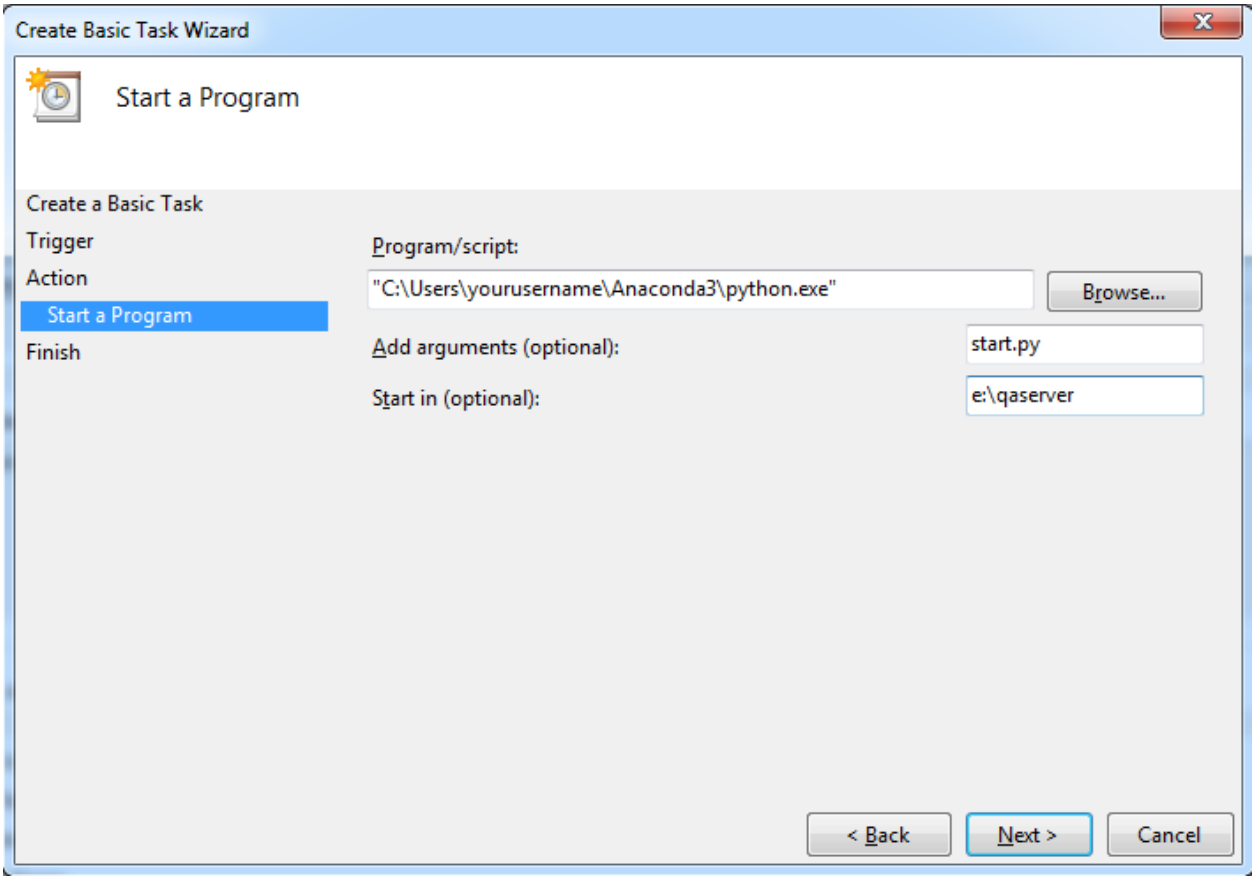
```
python start.py
```

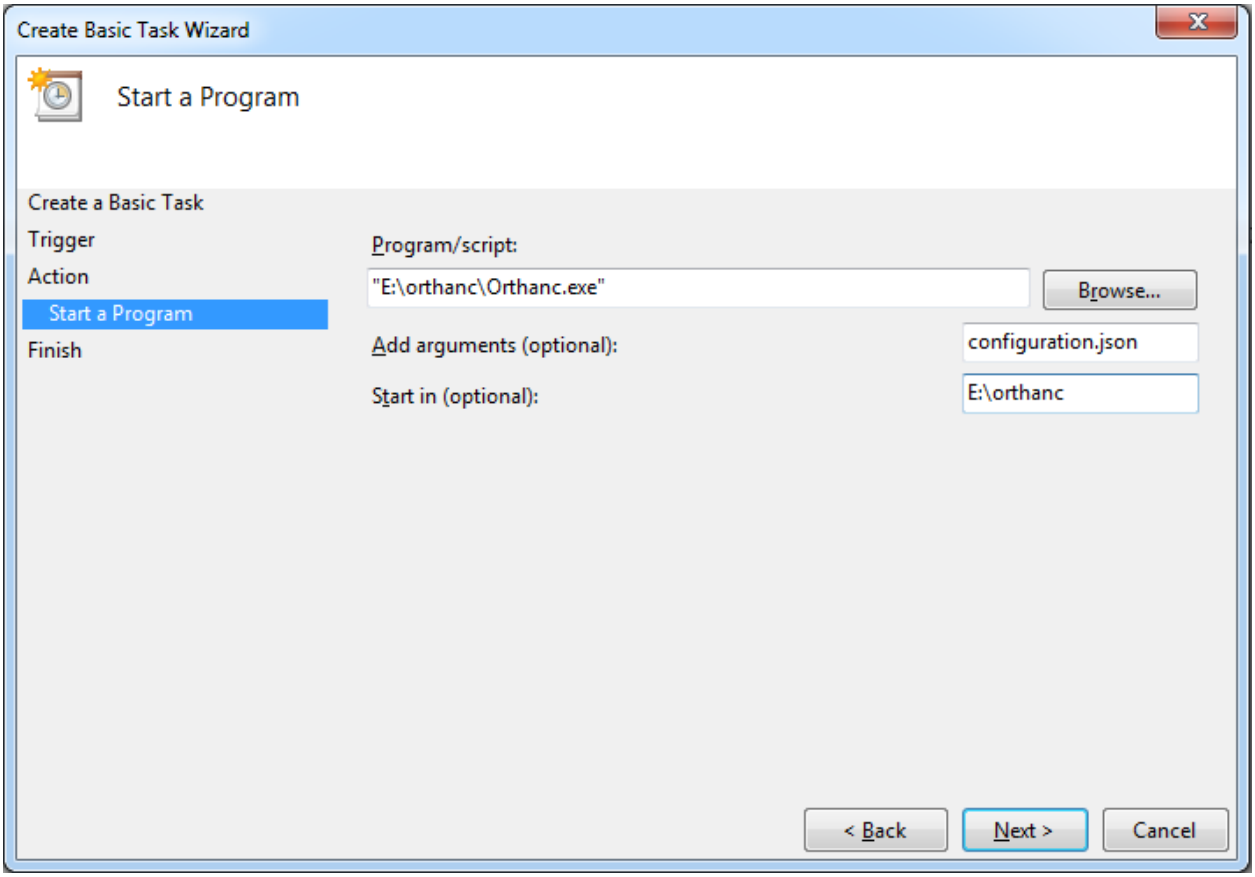
And Orthanc with:

```
orthanc configuration.json
```

If you are using one computer for both servers, then this should be it. Everybody in your network should be able to use QAserver and Orthanc. It is a good idea to check that Windows Firewall is not blocking Python.exe or Orthanc.exe, in case you have problems.

Anyway, having two Prompts open all the time may not be the optimal solution. You could create two Windows Tasks to be run anytime the computer is powered on. Open Windows Task Scheduler. Click on Create Basic Task. Input the info for QAserver and Orthanc. Images below will help you set up the action for both tasks. Make sure you you select "Run whether user is logged on or not", and "Run with highest privileges".


Create Basic Task Wizard
Start a Program
Create a Basic Task
Trigger
Action
Start a Program
Finish
Program/script:
"C:\Users\yourusername\Anaconda3\python.exe"
Browse...
Add arguments (optional):
start.py
Start in (optional):
e:\qaserver
< Back
Next >
Cancel



This will only work if you are using the base environment and if you added Anaconda to your system PATH.

CHAPTER

THREE

CONFIGURATION

QAserver can be configured by changing `config.ini`.

After you change the configuration file, you must restart the server.

3.1 Winston-Lutz

When you click Analyze, QAserver will fetch dicom files from Orthanc and temporarily store them into the *temp_dcm_archive* folder. Then Pylinac will analyze them. When the results are ready, temporary files will be deleted. If there was an error, files will remain in the temp folder.

PASS_RATE The radius (in mm) of the red circle in the "scatter diagram" section. At the same time the "failed" tolerance level.

SUCCESS_RATE The radius (in mm) of the green circle in the "scatter diagram" section. But also the "passed" tolerance level.

APPLY_TOLERANCE_TO_COLL_ASYM Chose whether the tolerance level will be applied to each analyzed image or to the section "Effect of collimator rotation". Possible values are True and False.

3.2 Starshot

The module works similarly to the Winston-Lutz module. If you are using tiff images, they will be stored into *temp_nondicom_archive* folder. The PDF report is stored in *temp_pdf_reports* folder.

TOLERANCE The tolerance for circle diameter (in mm).

GENERATE_PDF_REPORT Chose whether to generate a pdf report. Possible values are True or False.

3.3 Picket Fence

The module works similarly to the Winston-Lutz module. The PDF report is stored in *temp_pdf_reports* folder.

TOLERANCE The tolerance for the test in mm.

USE_ORIGINAL_PYLINAC Chose whether normal Pylinac should be used for analysis or a modified picket-fence.py module. The modified code gives you the option of using Elekta MLCs. Possible values are True and False.

> **Warning:** If you enable this feature, then a specially modified Pylinac module will be used for analysis. This may give unexpected results.

GENERATE_PDF_REPORT Chose whether to generate a pdf report. Possible values are True and False.

3.4 Planar Imaging

The module works similarly to the Winston-Lutz module. The PDF report is stored in *temp_pdf_reports* folder.

LOW_THRESHOLD The threshold for low contrast evaluation. See Pylinac for further details.

HIGH_THRESHOLD The threshold for high contrast evaluation. See Pylinac for further details.

QC3_MACHINES A list of reference images: subfolders that contain the reference images for each machine. The folders must be in the **reference_images/planar_imaging/QC3** folder. Example: say you have two machines, Linac1 and Linac2. Create two subfolders Linac1 and Linac2 and to each folder add a reference image called **First.dcm**.

Putting it in full form:

- reference_images/planar_imaging/QC3/Linac1/First.dcm
- reference_images/planar_imaging/QC3/Linac2/First.dcm

These reference images must have exactly this name and extension. Once you run the test, QAserver will analyze the current image as well as the reference image.

LEEDSTOR_MACHINES Exactly the same as in QC3_MACHINES, except that you must add the reference images to the folder **reference_images/planar_imaging/LeedsTOR**.

LASVEGAS_MACHINES Exactly the same as in QC3_MACHINES, except that you must add the reference images to the folder **reference_images/planar_imaging/LasVegas**.

LPPMM_QC3 A list of line pairs per mm for the QC3 phantom. You can read the values from your certificate. The list can contain only 5 elements in increasing order.

LPPMM_LEEDSTOR A list of line pairs per mm for the LeedsTOR phantom. You can read the values from your certificate. The list can contain only 9 elements in increasing order.

GENERATE_PDF_REPORT Chose whether to generate a pdf report. Possible values are True and False.

3.5 Catphan

After you click Analyze, the whole series is transferred from Orthanc to a temp folder. After the analysis is over, the folder is deleted.

503_MACHINES

A list of reference images. In fact, this is a list of subfolders that contain the reference images for each imager. The subfold

- reference_images/ct/Catphan503/Linac1_SFOV
- reference_images/ct/Catphan503/Linac1_MFOV
- reference_images/ct/Catphan503/Linac1_LFOV
- reference_images/ct/Catphan503/Linac2_SFOV

- reference_images/ct/Catphan503/Linac2_MFOV
- reference_images/ct/Catphan503/Linac2_LFOV

504_MACHINES Exactly the same as 503_MACHINES with one difference: add subfolders to **reference_images/ct/Catphan504**.

600_MACHINES Exactly the same as 503_MACHINES with one difference: add subfolders to **reference_images/ct/Catphan600**.

604_MACHINES Exactly the same as 503_MACHINES with one difference: add subfolders to **reference_images/ct/Catphan604**.

TOLERANCE_HU The tolerance for Hounsfield units and the uniformity module.

TOLERANCE_LCV The tolerance for low contrast visibility. Note that two definitions exist in QAserver. Using either of them can be controlled with USE_LCV2TOL.

TOLERANCE_SCALING The tolerance for geometric scaling in mm.

TOLERANCE_THICKNESS The tolerance for the difference between nominal and actual slice thickness in mm.

TOLERANCE_LOWCONTRAST The tolerance level for low contrast visibility. This is the number of ROIs that must be seen on the CTP515 module for the test to pass.

THRESHOLD_CNR The threshold for the low contrast ROI of the CTP515 module to be seen. It is measured in the same units as the CNR constant.

TOLERANCE_MTF The tolerance for the difference between reference and current 50% rMTF.

TOLERANCE_UNIFORMITYIDX The tolerance for the uniformity index. Note that two definitions exist in QAserver. Using either of them can be controlled with USE_UNIFORMITYIDX2.

USE_LCV2TOL If set to True, the tolerance will be applied to LCV, otherwise to LCV2.

USE_UNIFORMITYIDX2 If set to True, the tolerance will be applied to Uniformity index, otherwise to Uniformity index 2.

3.6 Dynalog

Before you can use the module, you must run the analyze_dynalog.py module to fill the dynalog database with records. The `config.ini` file is read by analyze_dynalogs.py, so no special edition of the latter should be necessary. The dynalog module can be configured to send emails after the analysis has been completed, but to begin you can turn this off by setting SEND_EMAIL to False.

Configure the module first, then run the **analyze_dynalogs.py** module from the command prompt by typing:

```
python analyze_dynalogs.py
```

Once you get familiar with this, you can schedule the module to start every day with Task Scheduler.

When run, the analyze_dynalogs.py module will read dynalogs from listed repositories (DYNALOG_REPOSITORIES). Dynalogs in the designated folders will be analyzed. There will be a short log about the analysis in \dynalog_database\log.txt. The results of the analysis will be entered into a special sqlite database (\dynalog_database\dynalog_database.db). When you search the records within the web interface, you are scrolling across this database. If a dynalog was analyzed successfully, it will be added to a zip archive located in \dynalog_database\ARCHIVE. The name of the archive is normally "YYYY_MM.zip". Each month a new archive will be created, so you do not have too large files. The zip archive and the sqlite database are linked.

Warning: Do not edit the zip archive or the sqlite database manually unless you really have to. Also, do not run the analysis at midnight.

When the analyze_dynalogs.py module analyzes dynalogs, it gives a special label to each record depending on the name of the repository from where the dynalog was taken. That is how you can separate dynalogs for different linacs. Labels are defined with REPOSITORIES_LABELS.

If there was an error while analyzing dynalogs, the problematic dynalog will be copied to the *dynalogs_with_errors* folder.

When you use the web interface to analyze one particular record, the corresponding dynalog is extracted from the zip archive and analyzed again with settings that you can define in the interface.

TOLERANCE_DTA The (gamma) distance-to-agreement tolerance for automatic analysis. In mm.

TOLERANCE_DD The (gamma) dose difference tolerance for automatic analysis. In %.

THRESHOLD The dose threshold for gamma analysis. In %.

RESOLUTION Resolution for gamma calculation in mm.

DYNALOG_REPOSITORIES A list of paths where dynalogs are located. Normally each linac (or MLC controller) stores dynalogs in a particular network folder. You can ask your engineers to set this up. Enter the full path of the folder. For many repositories make a list:

DYNALOG_REPOSITORIES = \network_path1\linac1\dynalogs, \network_path2\linac2\dynalogs

Of course the paths need not be network paths. If you wish to experiment, you can just use local folders with dynalogs.

REPOSITORIES_LABELS A list of names (labels) for each entry to DYNALOG_REPOSITORIES. Keep the same order as DYNALOG_REPOSITORIES.

SEND_EMAIL If set to True, after the analysis is over, QAserver will send a short summary of the results via email.

SMTP_SERVER The SMTP server address. For example: smtp.gmail.com.

SMTP_PORT The SMTP port. For example: 587.

SEND_FROM_USER Your email account for sending messages.

SEND_FROM_PASSWORD The password that you are using to access the account. Sorry, it is not encrypted.

SEND_TO A list of receivers. For example: person1@gmail.com, person2@gmail.com, etc.

3.7 VMAT

TOLERANCE The tolerance for the test. See Pylinac for further info.

3.8 Keeping it clean

From time to time check the folders for remaining files. In particular: temp_dcm_archive, temp_dynalog_folder, temp_nondicom_archive and temp_pdf_reports. The last one may be full of old pdf reports.

3.9 Sending data to Orthanc

Imaging workstation can be configured to send images directly to Orthanc via dicom transfer. This is particularly easy to do on Elekta's iView and XVI. If you do not feel competent to configure the export filters, ask your system administrator to do it.

CHAPTER

FOUR

WINSTON LUTZ MODULE

The module can be used in two ways:

- Acquire various images with different gantry, collimator and couch angles, and check the *Use Pylinac?* checkbox.
- Acquire a minimal set of gantry/collimator images (4 or 8 images in a predefined order), and uncheck the *Use Pylinac?* checkbox. Pylinac will analyze each image, but the results will be compiled independent from Pylinac. This way is good for doing the end-to-end test, particularly on Elekta linacs.

Examples of use are at the bottom of the page.

4.1 A minimal set of images

A minimal set of gantry/collimator images is a set of images that you must acquire in order to correctly determine the position of the MV isocenter. This must be done independently from potential collimator asymmetry or miss-calibration of MLCs or jaws. The standard set of images for Elekta and Varian is normally:

Gantry	180	180	270	270	0	0	90	90
Collimator	90	270	270	90	90	270	270	90

It is possible to use other angles, however, it is strongly suggested to use the ones listed above. You may change the collimator angle, if you wish. For Elekta, for example, you could use 0/180 instead of 90/270. But then the sag of the collimator will have a larger influence on the result at gantry 90/270.

Some people may need to use SRS cones for treatment. In this case the appropriate sequence is:

Gantry	180	270	0	90
Collimator	0	0	0	0

The field can be of any size, it doesn't even have to be symmetric. It is only vital that all beams use exactly the same MLC and jaw configurations! **The above mentioned sequences are used with "Use Pylinac?" checkbox unchecked.**

4.1.1 Options

Use pylinac? checked Full scale Pylinac analysis is performed. Once you tick the checkbox, a new table will show up underneath (*Angles for pylinac*). There you can enter angles for each image. If you are doing the analysis on a Varian linac, do not enter any values because Pylinac will read the angles from dicom tags. If, on the other hand, you are doing this test on an Elekta linac, then you *must* enter in angles, because dicom tags are not

present. You may image the BB with any gantry/collimator/couch configuration, as long as Pylinac will be able to analyze it.

Use pylinac? unchecked Now the QAserver will assume a specific acquisition order. Namely, the two sequences defined above. No dicom tags will be read, nor is it necessary to define specific angles. In this situation Pylinac will analyze each image, but the end results will be calculated independently of Pylinac.

Select multiple series? When you acquire images with different couch angles on a Varian linac, each image will be imported into Orthanc as a separate series. By ticking this box you have the option of selecting multiple series for analysis.

Zoom in on field? If checked, the image display will zoom-in on the field.

Clip box Sometimes images have unwanted artifacts at the edges that could disturb the calculation of field CAX. Here you can enter the size of the central portion of the image beyond which pixel values will be set to background signal. If you don't want to clip the image, put 0.

4.1.2 Things you should know

- If *Use pylinac?* is unchecked, the acquisition order is important:

> **Warning:** QAserver will assign gantry/collimator angles according to acquisition times. Before you use this software, make sure that QAserver is reading acquisition times correctly. Also, observe the gantry/collimator order.

- If *Use pylinac?* is unchecked, the analysis will take twice as long. The reason is that each image is analyzed twice because of the way QAserver uses Pylinac.
- If *Use pylinac?* is unchecked and neither 4 nor 8 images are contained in the series, only image-wise analysis will be returned. More results can be obtained if you check *Use Pylinac?* and enter the appropriate angles in the table *Angles for pylinac*.
- *Station* name is read from the first image of the series.

4.1.3 Interpreting results

Images

dx, dy Position of the center of the field with respect to the center of the BB.

Red cross Center of the BB.

Blue cross Center of the field.

Yellow dot Center of the image (EPID).

Blue contour 50%-edge of the field.

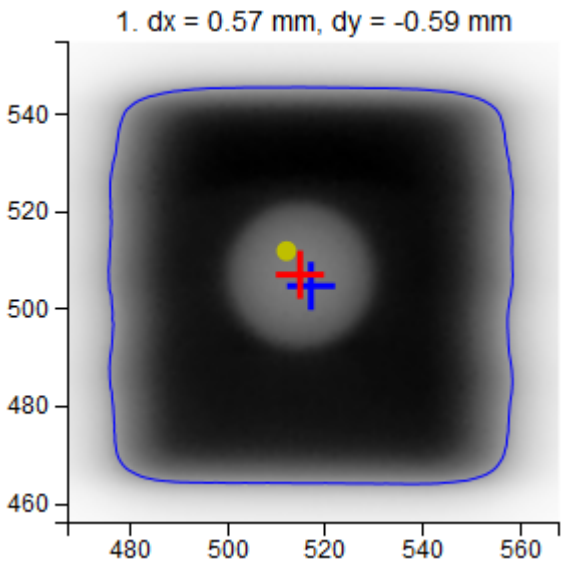


Image Analysis

CAX x, CAX y Coordinates of field center with respect to the EPID.

BB x, BB y Coordinates of the center of the BB with respect to the EPID.

$\Delta x, \Delta y$ The difference between CAX and BB.

R The length of the vector from BB to CAX.

Moving the BB into the isocenter - Use Pylinac? unchecked

If 8 images were acquired in the correct order:

$$\text{LAT} = \left(-\Delta x_1 - \Delta x_2 + \Delta x_5 + \Delta x_6\right)/4$$

$$\text{LONG} = \left(\Delta y_1 + \Delta y_2 + \Delta y_3 + \Delta y_4 + \Delta y_5 + \Delta y_6 + \Delta y_7 + \Delta y_8\right)/8$$

$$\text{VRT} = \left(\Delta x_3 + \Delta x_4 - \Delta x_7 - \Delta x_8\right)/4$$

If 4 images were acquired in the correct order:

$$\text{LAT} = \left(-\Delta x_1 + \Delta x_3\right)/2$$

$$\text{LONG} = \left(\Delta y_1 + \Delta y_2 + \Delta y_3 + \Delta y_4\right)/4$$

$$\text{VRT} = \left(\Delta x_2 - \Delta x_4\right)/2$$

Beam deviation - Use Pylinac? unchecked

It is possible to determine crossplane (A-B) deviations of the beam from the ideal position. Independently from the previous section we have by taking all images into account:

$$\text{deviation} = \left(\Delta x_1 + \Delta x_2 + \Delta x_3 + \Delta x_4 + \Delta x_5 + \Delta x_6 + \Delta x_7 + \Delta x_8\right)/8$$

$$\text{deviation} = \left(\Delta x_1 + \Delta x_2 + \Delta x_3 + \Delta x_4\right)/4$$

Or by just observing those at gantry 180/0 (those that have little effect of collimator sag):

$$\text{deviation} = \left(\Delta x_1 + \Delta x_2 + \Delta x_5 + \Delta x_6\right)/4$$

$$\text{deviation} = \left(\Delta x_1 + \Delta x_3\right)/2$$

This result is independent of the precision with which the BB is positioned into the isocenter.

Effect of collimator rotation - Use Pylinac? unchecked It only works with 8 images. Say your collimator aperture is slightly asymmetrical, or perhaps the axis around which the collimator spins is not so stable. When you rotate the collimator, you will notice that the field CAX will move from one point to the other. QAserver will give you a simple estimate of this displacement by averaging each image pair (1,2), (3,4), (5,6), (7,8) and calculating the shift from this average for each image.

Isocenter size - Use Pylinac? unchecked A simple evaluation of the size of the point cloud.

Gantry iso size is calculated as the length of the vector to the CAX point that is farthest from the center of the CAX cloud:

$\max_{i \in img} \sqrt{(\Delta x_i - \overline{\Delta x})^2 + (\Delta y_i - \overline{\Delta y})^2}$

Collimator iso size is $\max_{i \in img} \sqrt{(\pm\Delta x_i)^2 + (\pm\Delta y_i)^2}$,

where $\pm\Delta x, y$ is taken from the previous section (*Effect of collimator rotation*).

EPID movement - Use Pylinac? unchecked Say that the BB accurately represents the isocenter. Normally you would like your EPID to be positioned in such a way that the distance between the EPID center and the center of the BB does not exceed some tolerance (2 mm or 3 mm). The maximum EPID-BB distance is just:

$\max_{i \in img} \sqrt{(\mathrm{BB}x_i)^2 + (\mathrm{BB}y_i)^2}$

Scatter diagram The BB is in the center of the diagram. Blue dots represent field CAX, yellow dots the center of the EPID. The green circle corresponds to the normal tolerance, the red circle is the action tolerance. Both are defined in the configuration file.

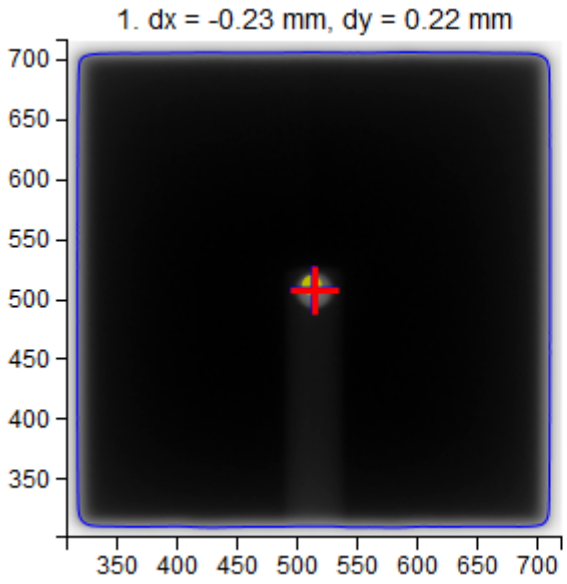
Status You can get either Passed, Borderline or Failed. If you are using 4 or 8 images, then you have the option of applying the tolerance to either each image in the series, or to the section *Effect of collimator rotation* where radius R is calculated by averaging over collimator pairs. See your configuration file.

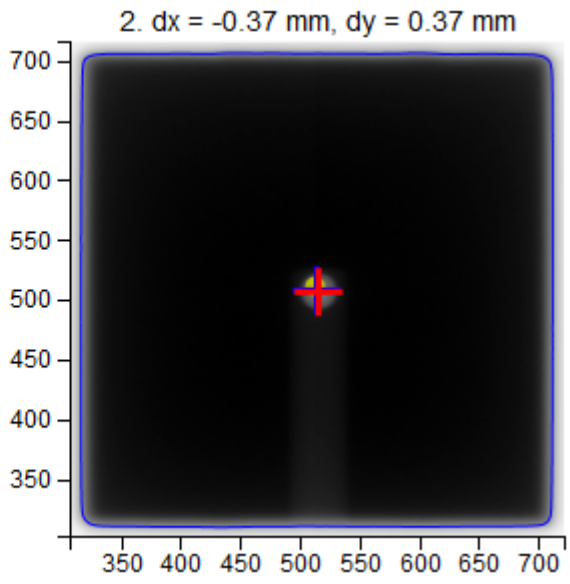
4.1.4 Examples

4.2 Example 1

Task: find the average MV isocenter of the linac and test the position of the couch axis of rotation for radiosurgery.

If you are doing this on an Elekta linac, you can use their flexmap calibration procedure. In any case, acquire a sequence of 8 images as stated at the beginning. Analyze them, and get the shifts. Apply the shifts to the BB and repeat the acquisition. Check that the residual shifts are small, say, smaller than 0.1 mm in any direction. If your BB does not have micrometer screws, you can use the robotic couch to make the shifts.





Moving the BB into the isocenter **First**

To move the BB into the isocenter the following shifts must be applied. Looking at the gantry from the foot of the couch:

LAT	LONG	VRT
0.31 mm → RIGHT	0.77 mm → IN	0.72 mm → DOWN

Moving the BB into the isocenter **Second**

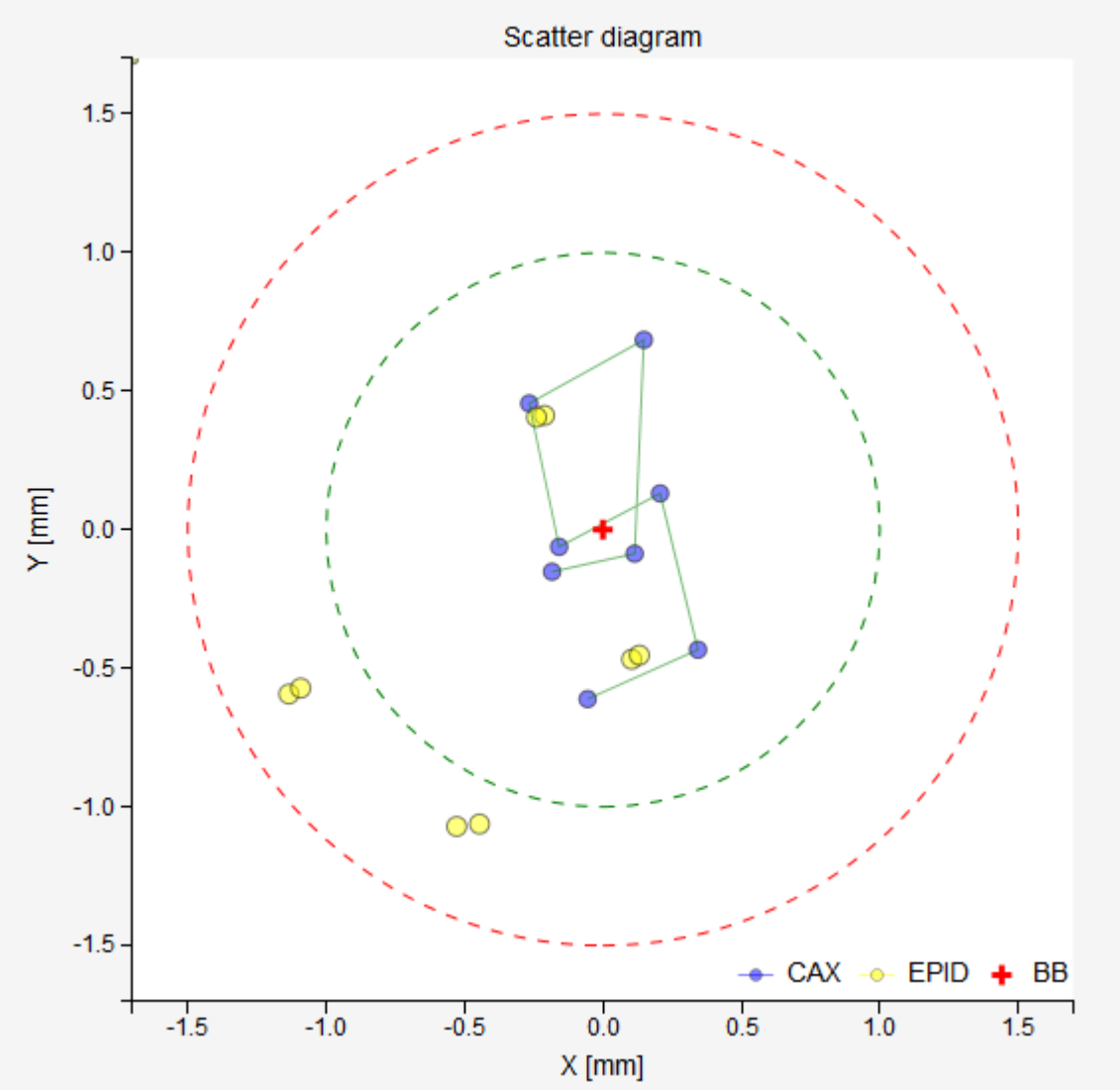
To move the BB into the isocenter the following shifts must be applied. Looking at the gantry from the foot of the couch:

LAT	LONG	VRT
0.1 mm → LEFT	0.01 mm → OUT	0.03 mm → UP

Image analysis

Deviations are measured with respect to the EPID center. Δ x and Δ y are calculated with BB as the origin.

Image	CAX x [mm]	CAX y [mm]	BB x [mm]	BB y [mm]	Δx [mm]	Δy [mm]	R [mm]
1	0.16	-1.02	0.21	-0.41	-0.06	-0.61	0.61
2	0.58	-0.84	0.24	-0.41	0.34	-0.43	0.55
3	0.1	0.6	-0.1	0.47	0.21	0.13	0.24
4	-0.29	0.39	-0.13	0.45	-0.16	-0.06	0.17
5	0.26	1.53	0.53	1.07	-0.27	0.46	0.53
6	0.59	1.75	0.45	1.06	0.15	0.68	0.7
7	1.25	0.51	1.13	0.59	0.11	-0.09	0.14
8	0.91	0.42	1.09	0.57	-0.18	-0.15	0.24



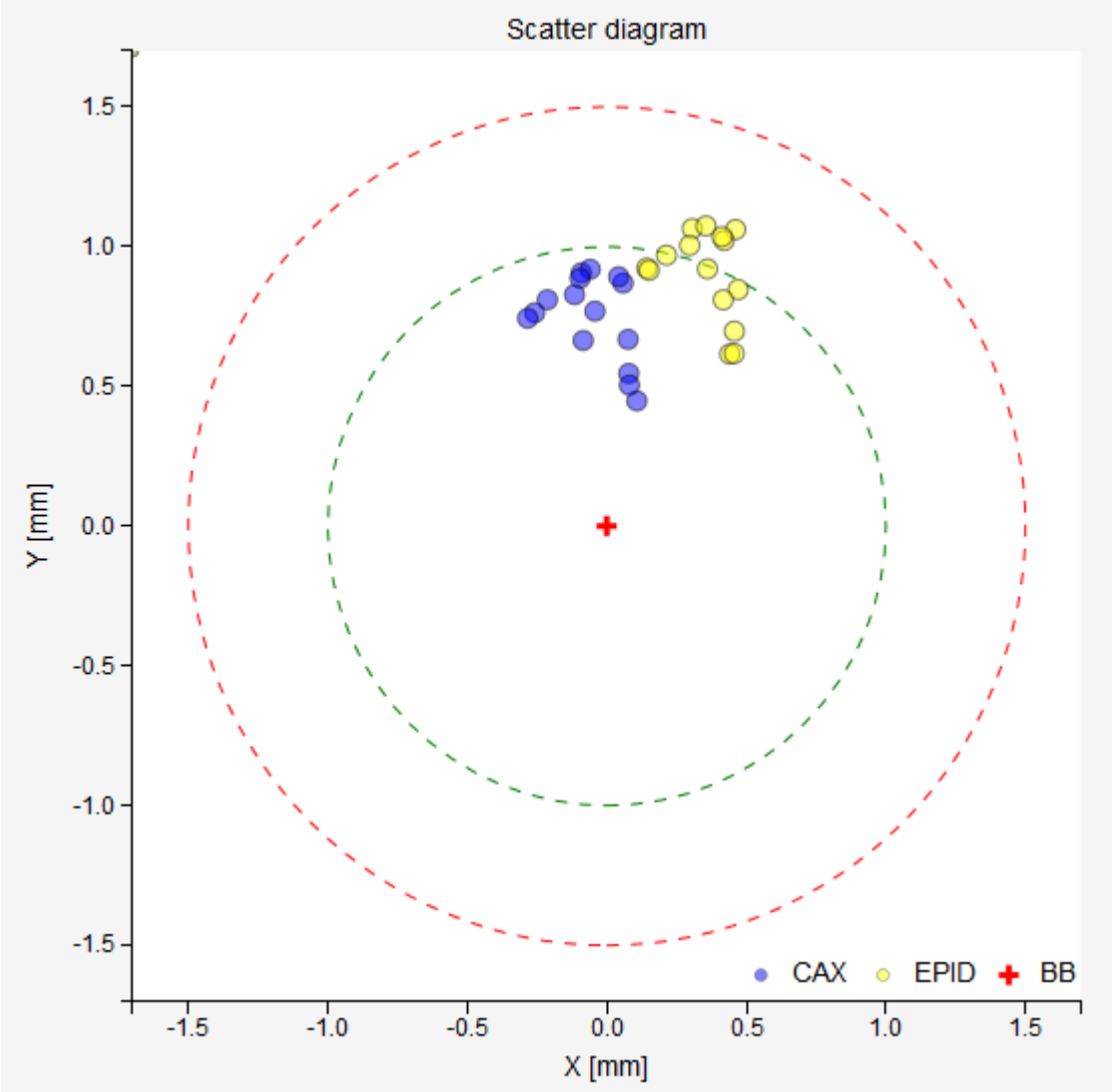
The results of the last analysis can be reviewed on the scatter diagram. You can see that the BB is well aligned with the average MV isocenter. There is a small collimator asymmetry noticeable at all gantry angles. And there is no beam deviation in the lateral direction. You will also notice a 1.3 mm gantry displacement in the longitudinal direction, which is perfectly normal for this type of linac.

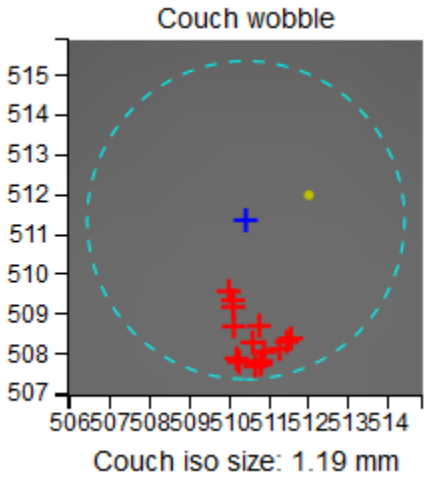
At this point you could make a CBCT scan of the BB and check whether the imager is showing that the BB is in the isocenter.

Testing the couch is easy. Put gantry and collimator to 0. Leave the BB exactly where it is. Then acquire images at several couch angles with the same gantry and collimator. You can use a series of angles like such:

Couch	0	30	60	90	60	30	0	330	300	270	300	330	0

You can see that moving the couch only slightly moves the BB away from the field CAX. Never does the distance between the BB and the CAX go beyond 1 mm. On the scatter plot you can also observe that while gantry and collimator are stationary, there is some movement of the EPID. This is because in this plot the BB is fixed in the center. But in reality it is the BB that is moving, not the EPID or the CAX. A better representation of what is going on is offered in the section Axis wobble (couch wobble).


Scatter diagram
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
Y [mm]
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
X [mm]
CAX
EPID
BB


Couch wobble
515
514
513
512
511
510
509
508
507
506
507
508
509
510
511
512
513
514
Couch iso size: 1.19 mm

4.3 Example 2

With the WL test you can determine if your beam has a lateral deviation from the ideal position. See an example scatter plot below. All blue CAX points are shifted to the right. This does not correspond to an error in the position of the BB, but to an error in the way the beam is targeting the BB. Indeed, the results show exactly this: the BB is perfectly aligned with the average MV isocenter, however there is a 0.5 mm lateral deviation of the beam.

Moving the BB into the isocenter

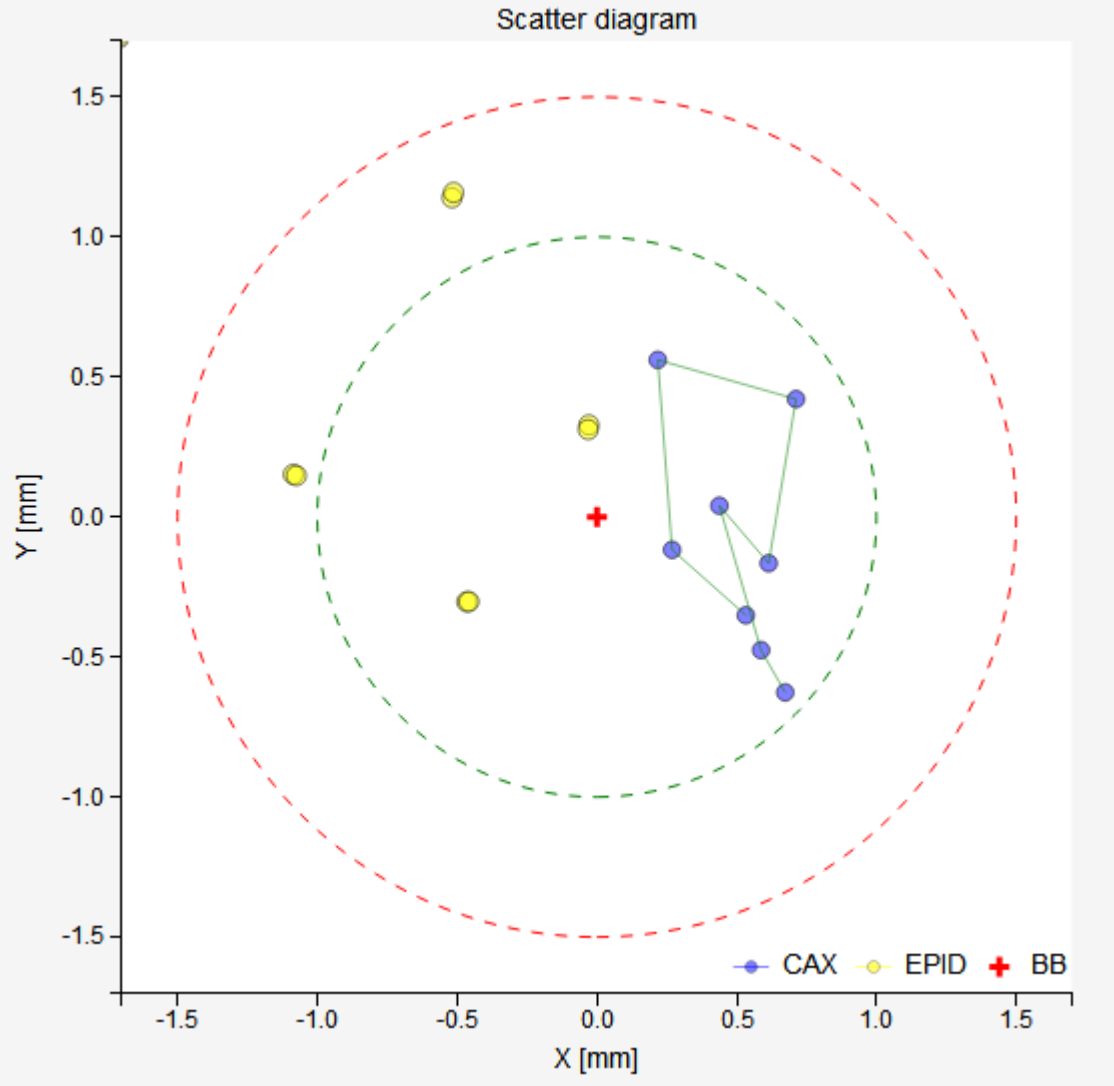
To move the BB into the isocenter the following shifts must be applied. Looking at the gantry from the foot of the couch:

LAT	LONG	VRT
0.08 mm → LEFT	0.09 mm → OUT	0.06 mm → UP

Beam deviation

Crossplane deviation of the beam from the ideal position. In beam's eye view:

Average over all images	Average over images at gantry 0 and 180
0.51 mm → RIGHT	0.55 mm → RIGHT



Note: Longitudinal deviations of the beam cannot be detected this way. Unless you take the 6 MV beam as the reference.

CHAPTER

FIVE

STARSHOT MODULE

5.1 How to start

Follow Pylinac's guidelines for image acquisition. The image can be a dicom EPID image imported into Orthanc, or a scanned tif image.

- Make sure the star is near the center of the image.
- The spokes should be sufficiently long.
- Try not to use jaws/diaphragms to shape the field, use only MLC leaves.
- If you are using a scanned tiff image, make sure that you use the correct scaling parameters DPI and SID.

5.2 Options

Clip box Sometimes images have unwanted artifacts at the edges. Here you can enter the size of the central portion of the image beyond which pixel values will be set to background signal. If you don't want to clip the image, put 0.

Radius The radius within which Pylinac will analyze the spokes. A value of 0.85 means that the length of the spokes is 85 % of image size. See Pylinac for further explanation.

Min peak height See Pylinac for further explanation.

Start point The approximate center of the star. Given in pixels. See Pylinac for further explanation.

SID Source-imager distance in mm. If the dicom image does not contain this tag, you have to enter it manually. The same goes if the tiff image does not contain this information.

DPI Dots-per-inch. If the dicom image does not contain this tag, you have to enter it manually. The same goes if the tiff image does not contain this information.

Set FWHM? See Pylinac for further explanation.

Recursive? See Pylinac for further explanation.

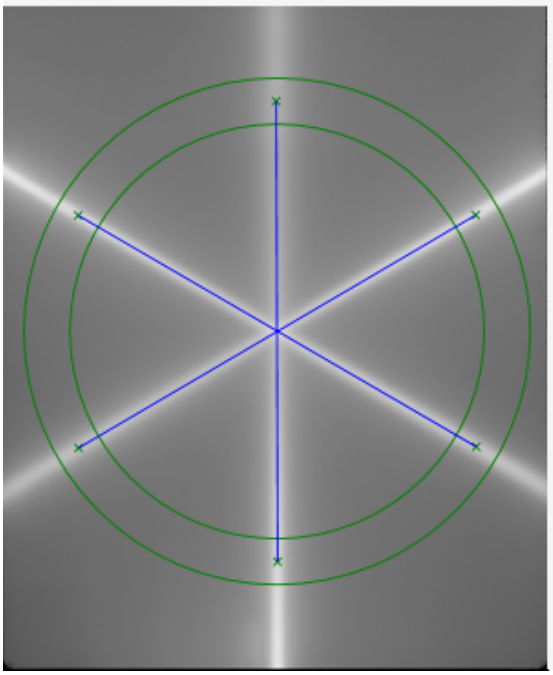
Invert image? Tick this box if you want to invert the image.

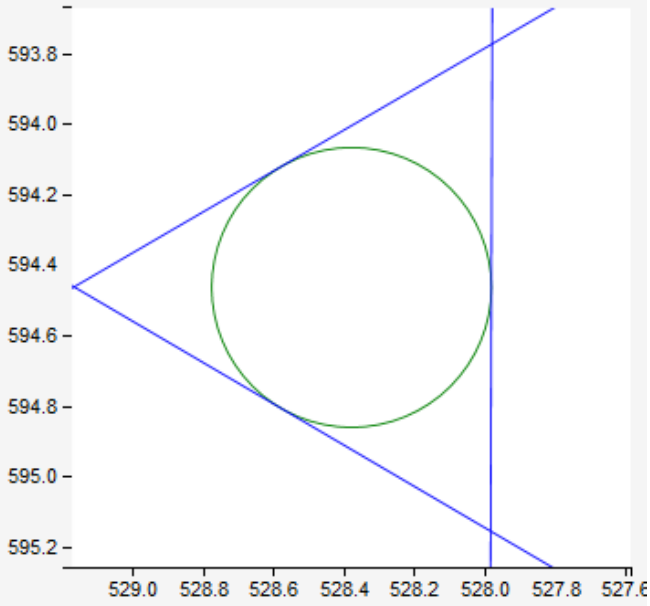
Load external image Here you can load an external tif image. Only .tif images are supported so far.

5.3 Results

You can define the tolerance in the configuration file. The status of the analysis will be Passed if the diameter of the circle is within tolerance.

Tolerance	2.0 mm
Radius	0.14 mm
Diameter	0.28 mm
Circle center	528.38 px, 594.46 px
Status	Passed





CHAPTER

SIX

PICKET FENCE MODULE

Warning: Before you start using this module make sure that it is working as you would expect. You should create a dynamic beam that simulates a picket fence with intentional errors, and see if Pylinac/QAserver detects them correctly.

6.1 Options

Clip box Sometimes images have unwanted artifacts at the edges. Here you can enter the size of the central portion of the image beyond which pixel values will be set to background signal. If you don't want to clip the image, put 0.

Filter Apply a median filter to the image. If 0, then no additional filtering is performed. See Pylinac and scipy.ndimage.median_filter for further information.

Num of pickets If too many pickets are detected, you can try to force the number of pickets beforehand. If 0, no number is defined. See Pylinac for further information.

Sag adjust If 0, no adjustment is made. Otherwise Pylinac will shift the MLCs laterally according to the value that you entered.

MLC type Select the appropriate MLC type. Elekta_80, Elekta_160 and Varian_80 are available if QAserver has been configured to use a modified Pylinac module, otherwise only Varian_120 and Varian_120HD will be available. See the configuration instructions.

> **Warning:** If QAserver has been configured to work with Elekta_80, Elekta_160 and Varian_80 MLCs, then a modified Pylinac module will be used for analysis. This may give unexpected results. Use it at your own peril. You can test whether you are using the unmodified Pylinac module by trying to run the analysis with, say, Elekta_160 MLCs.

Orientation If you have problems with the orientation of the pickets, you can force the orientation here.

Invert image? Check this box if you wish to invert the image.

Load machine log Not available in this version.

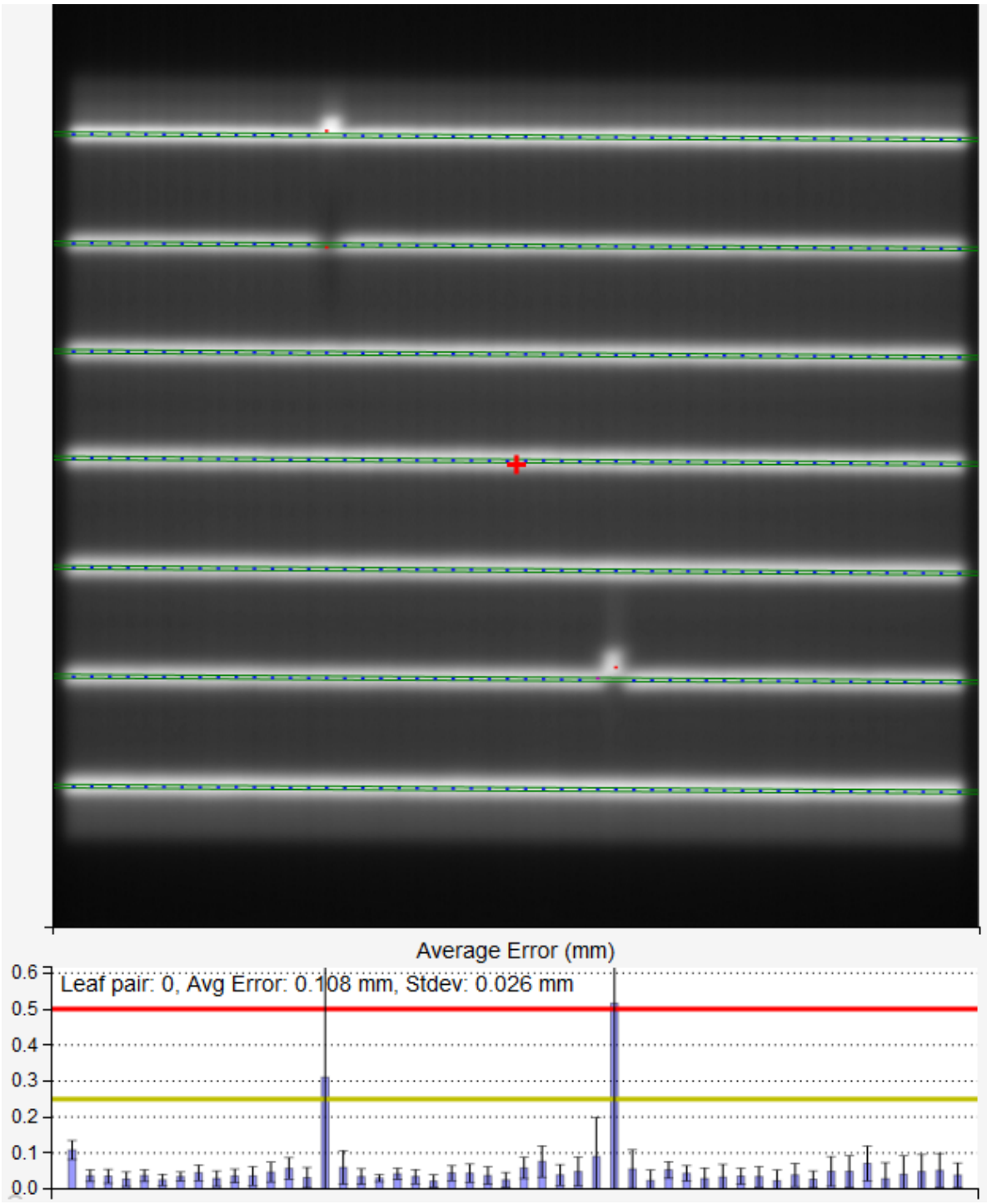
6.2 Results

Image Each picket is delimited with two green lines. The spacing between the lines corresponds to the tolerance level.

Within the lines dashes correspond to the center (fwhm) of the region formed by the MLC pairs. If the leaf pair has passed the tolerance, the dash will be colored blue. If the leaf pair has passed the tolerance, but not the action tolerance, the dash will be colored purple. If the leaf pair is not within tolerance, the dash will be colored red.

The error histogram shows the error and std of each leaf pair averaged over all the pickets. If you hover over the bars, additional information will be shown. The yellow line is the action tolerance, the red line is the tolerance.

The red cross on the image represents the image center.

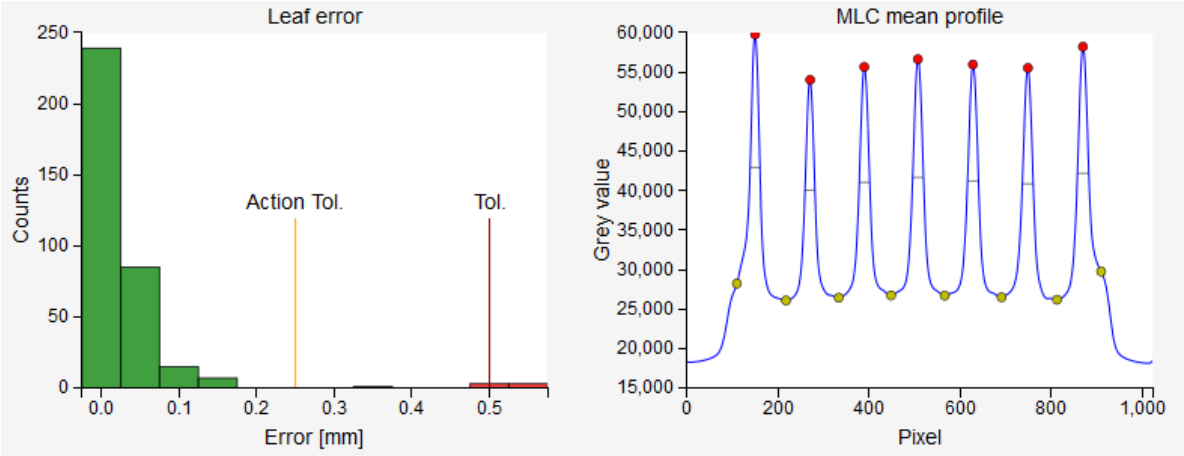


Results

- **Nr of picket** How many pickets were detected.
- **Pass prcnt** Passing percentage.
- **Max error** Maximum deviation of leaf pair fwhm that was detected.
- **Max error picket** Which picket line has the max error.
- **Max error leaf** Which leaf pair has the max error. Leaf pair are enumerated from left to right (or up-down), starting with 0.
- **Median error** The median of all errors.
- **Mean picket spacing** The average spacing between picket lines.
- **Mean FWHM** The average fwhm of all picket lines. Fwhm is evaluated from the mean MLC profile for each peak.
- **Offset from CAX** The distance of each picket line from the image center (red cross).
- **FWHM** The fwhm of each picket line evaluated from the mean MLC profile.
- **Passed** If the max error is within tolerance, the picket will pass the test.

Histogram shows the distribution of errors for all leaf pairs. If there are 7 pickets, each having 50 leaf pairs, then the histogram will have 350 counts altogether.

The MLC mean profile plot demonstrates the peaks for each picket line. Yellow dots signal the start and stop pixels between which peaks and fwhm are calculated. Horizontal dashes represent the fwhm. Because of the rounding effect, dashes may not be exactly horizontal.



Note: The calculation of FWHM is done independently of Pylinac.

CHAPTER

SEVEN

PLANAR IMAGING MODULE

This module is used to test the constancy of two-dimensional MV or kV image. All three phantoms recognized by Pylinac can be used:

- Standard Imaging QC3
- Leeds TOR 18
- Las Vegas

In order to be able to compare image quantifiers with the baseline, you have the option of saving a reference image into the QAserver folder structure. This image can then be analyzed at the same time as the currently acquired image.

Read[1] and[2] so that you will be able to understand the analysis.

7.1 Options

Image The acquired image. If *Analyze reference?* is unchecked, only this image will be analyzed.

Phantom A list of available phantoms recognized by Pylinac. Always pick the correct phantom.

Imager Here you can select the reference image of the phantom for a particular machine (imager). See "Configuration" for more details about how to add reference images to QAserver. If you don't have a reference image prepared, then skip this setting, and untick the *Analyze reference?* checkbox.

Clip box Here you can enter the size of the central portion of the image beyond which pixel values will be set to background signal. If you don't want to clip the image, put 0.

LeedsTor rot Here you can force the angle of the Leeds TOR phantom.

Show bounding box? If checked, a bounding box will be drawn around the phantom, and the image will be zoomed-in.

Invert image? Check this if you need to invert the image.

Use Pylinac? If checked, the analysis will give original Pylinac results. If unchecked, the analysis will be performed with Pylinac, however the results will be presented in a slightly different fashion (see further down for more explanation).

[1] Ronald T. Droege, *A practical method to routinely monitor resolution in digital images*, Medical Physics 10, 337 (1983)

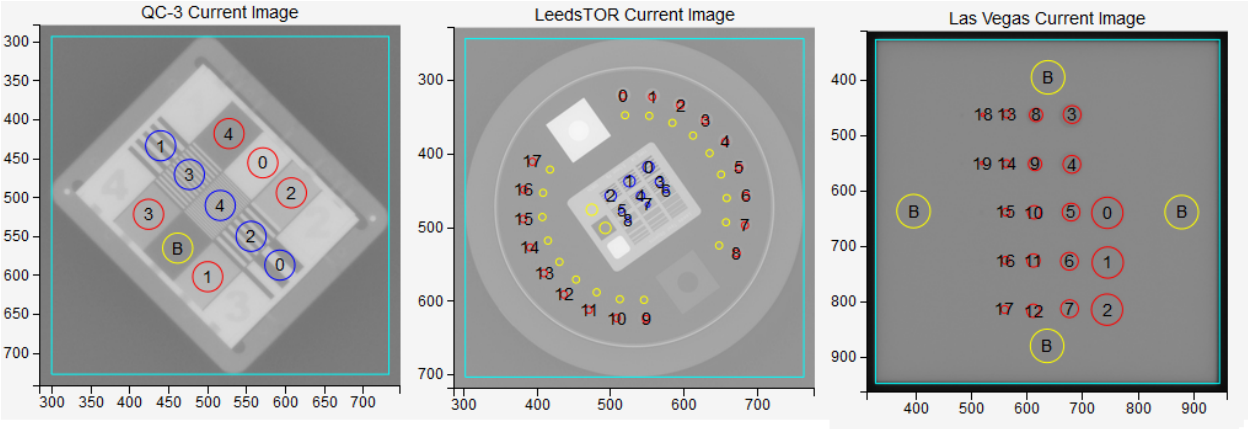
[2] R. Rajapakshe, K. Luchka, and S. Shalev, *A quality control test for electronic portal imaging devices*, Medical Physics 23, 1237 (1996)

7.2 How to use the module

- Acquire your image in such a way that Pylinac will be able to analyze it.
- To set a reference for comparison have your engineers optimize image quality, then acquire a good image that satisfies your clinical demands. This image can then be downloaded from Orthanc and stored into QAserver for future use.
- Select the appropriate Phantom. If the reference image is available, it will show up in the dropdown menu (Imager). If not, then you can work without them. In this case uncheck *Analyze reference?*.

7.3 Regions of interest (ROIs)

Pylinac will find several ROIs on each phantom. The following image shows ROIs for each phantom. Red circles mark those ROIs that are used for studying contrast (low frequency contrast resolution), blue circles are those ROIs that are used for studying MTF (high frequency contrast resolution) and yellow circles are the background ROIs.



When *Use Pylinac?* is checked, both low contrast and high contrast ROIs will be colored blue or red depending on whether they pass the threshold tolerance. If *Use Pylinac?* is unchecked, low contrast ROIs will always be colored red, and high contrast ROIs will always be colored blue (no passing tolerance is applied in this case). Background ROIs are always colored yellow.

We denote low contrast, high contrast, and background ROIs with

$$L, H, B_L, B_H$$

Where *L* stands for low frequency ROI, *H* for high frequency ROI and *B* is the background.

The QC3 phantom has 5 high frequency ROIs, 5 low frequency ROIs, and one background ROI:

$$H_0, H_1, H_2, H_3, H_4$$
$$L_0, L_1, L_2, L_3, L_4$$
$$B$$

The LeedsTOR phantom has 9 high frequency ROIs with 2 corresponding background ROIs, and 18 low frequency ROIs with corresponding 18 background ROIs.

$$H_0, H_1, \ldots, H_8$$
$$L_0, L_1, \ldots, L_{17}$$
$$B_{H1}, B_{H2}$$
$$B_{L0}, B_{L1}, \ldots, B_{L17}$$

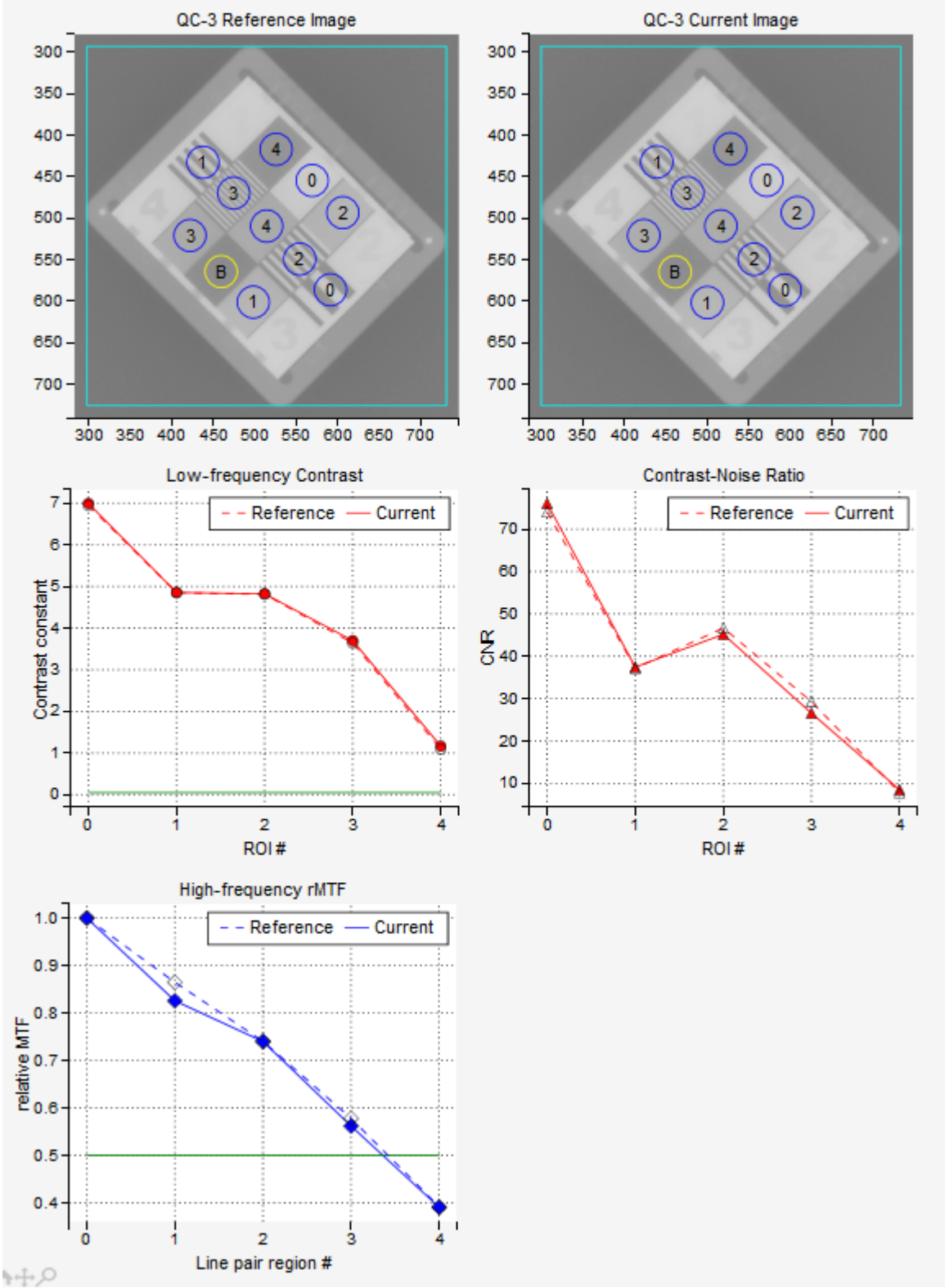
The Las Vegas phantom has 20 low frequency ROIs and 4 background ROIs.

$$L_0, L_1, \ldots, L_{19}$$
$$B_{L0}, B_{L1}, B_{L2}, B_{L3}$$

Once ROIs are defined, pixel values within can be extracted from the image. Random noise is not calculated according to [2], instead the standard deviation of pixel values within low contrast or background ROIs is used.

7.4 QC3 Analysis

7.4.1 *Use Pylinac?* checked



Low-frequency contrast for each *L* is calculated as the low-frequency constant, which is the product of contrast and the diameter of the ROI.

$$\text{contrast constant}\,(L_i) = \frac{\text{median}\,(L_i) - \text{median}\,(B)}{\text{median}\,(L_i) + \text{median}\,(B)} \times \text{diameter}\,(L_i)$$

The green line on the low-frequency plot represents the *low_threshold* defined in the configuration file. If the contrast constant is above the green line, the ROI is considered "visible".

Contrast-to-noise ratio is defined as

$$\text{CNR}\,(L_i) = \frac{\text{median}\,(L_i) - \text{median}\,(B)}{\text{std}\,(L_i)}$$

Relative modulation transfer function is calculated as

$$\text{rMTF}\,(H_i) = \frac{\max(H_i) - \min(H_i)}{\max(H_i) + \min(H_i)} \Big/ \frac{\max(H_0) - \min(H_0)}{\max(H_0) + \min(H_0)}$$

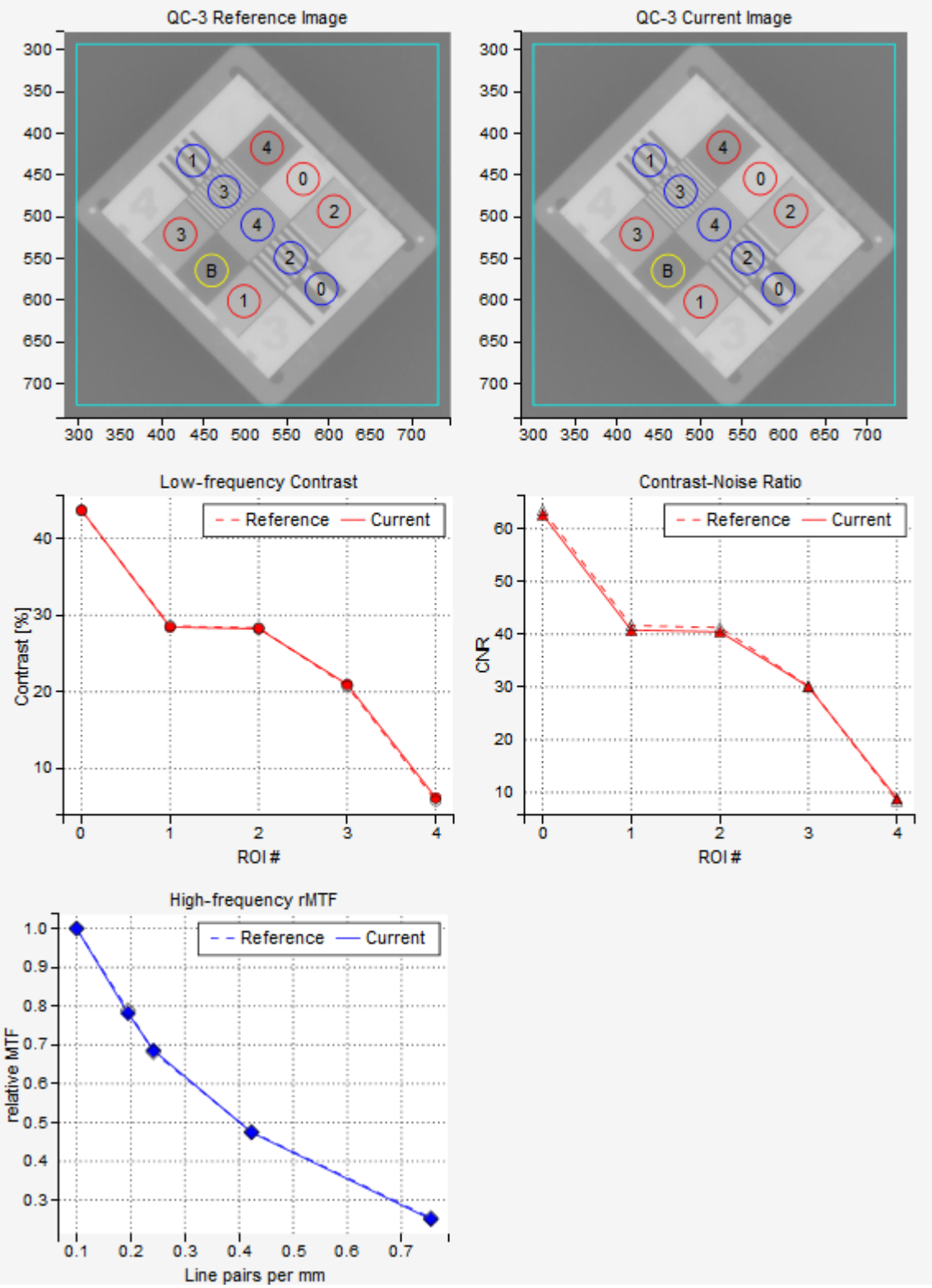
Note that it is normalized to 1 for the first high-frequency region. The green line on the plot represents the high_threshold defined in the configuration file. If the points are above the green line, then the lines in the corresponding region can be resolved.

Relative lppmm are calculated for 30%, 40%, 50% and 80% rMTF. The result is not in units of lppmm.

Median contrast is calculated as the median of contrast values for all low-frequency regions. Note that this is not the median of contrast constants.

Median CNR is calculated as the median of all CNR values.

7.4.2 *Use Pylinac?* unchecked



The analysis is similar, however these definitions are in use.

Low-frequency contrast:

$$\text{contrast}\,(L_i) = 100 \times \frac{\text{median}\,(L_i) - \text{median}\,(B)}{\text{median}\,(B)}$$

Relative modulation transfer function:

$$\text{rMTF}\,(H_i) = \sqrt{(\text{std}(H_i))^2 - (\text{std}(L_0))^2} \Big/ \sqrt{(\text{std}(H_0))^2 - (\text{std}(L_0))^2}$$

lppmm is returned in real units based on the calibration of line pairs in the configuration file.

Noise is the average value of standard deviations for all low-frequency regions and the background region:

$$\text{noise} = \frac{1}{6}\left(\sum_{i=0}^{i=4} \text{std}\,(L_i) + \text{std}\,(B)\right)$$

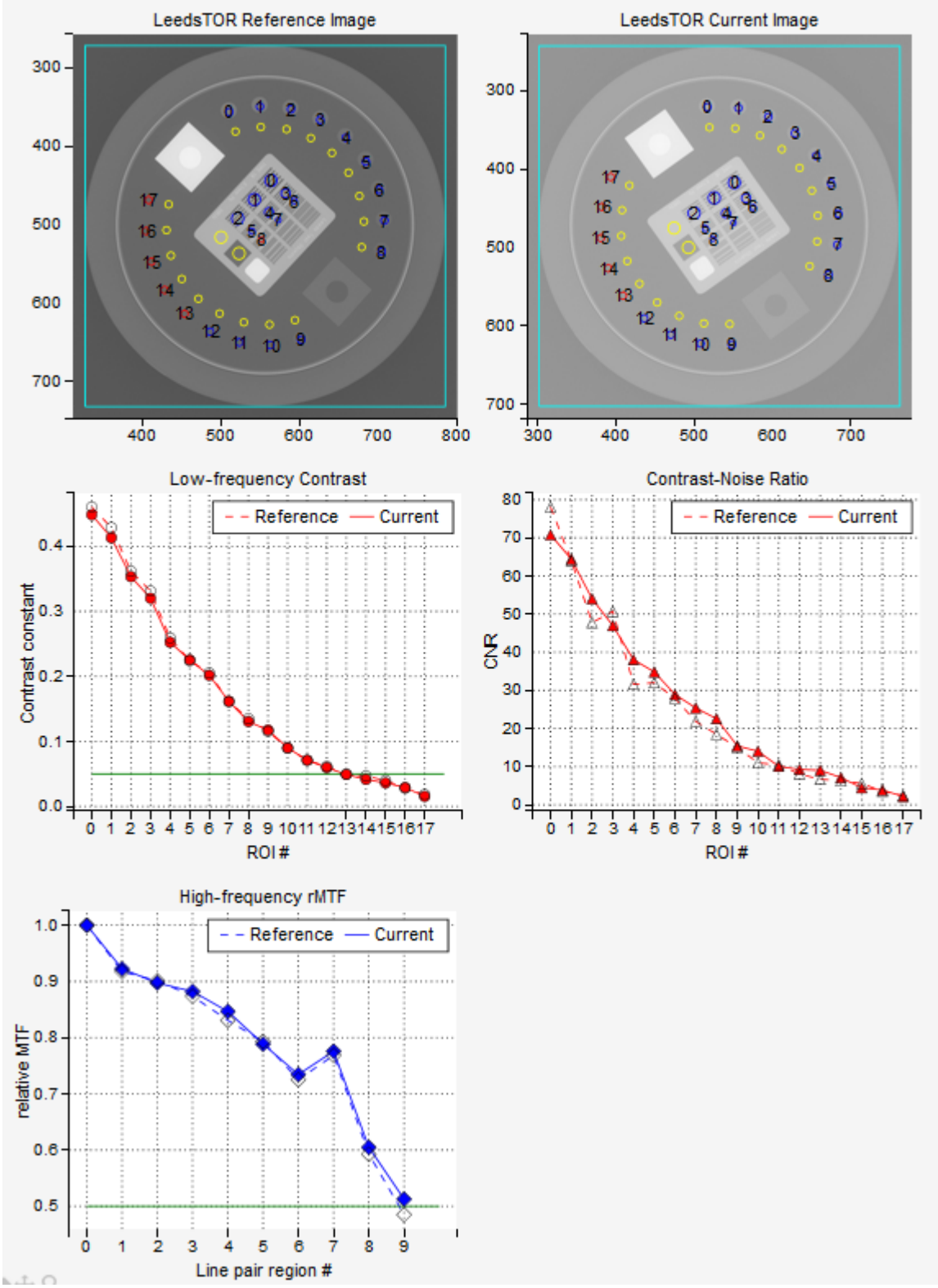
Contrast-to-noise ratio is calculated like this:

$$\text{CNR}\,(L_i) = \frac{\text{median}\,(L_i) - \text{median}\,(B)}{\text{noise}}$$

CNR is the CNR of the first low-frequency region.

7.5 Leeds TOR Analysis

7.5.1 *Use Pylinac?* checked



Pylinac uses a different background for each low-frequency ROI.

Low-frequency contrast is again the product of **contrast** and the ROI diameter:

$$\text{contrast constant}\,(L_i) = \frac{\text{median}\,(L_i) - \text{median}\,(B_{Li})}{\text{median}\,(L_i) + \text{median}\,(B_{Li})} \times \text{diameter}\,(L_i)$$

The green line on the low-frequency plot represents the low_threshold defined in the configuration file. If contrast-constant is above the green line, the ROI is considered "visible".

Contrast-to-noise ratio is defined as

$$\text{CNR}\,(L_i) = \frac{\text{median}\,(L_i) - \text{median}\,(B_{Li})}{\text{std}\,(L_i)}$$

Relative modulation transfer function is calculated like this:

$$\text{rMTF}\,(H_i) = \frac{\max\,(H_i) - \min\,(H_i)}{\max\,(H_i) + \min\,(H_i)} \Big/ \frac{\max\,(B_{H0}) - \min\,(B_{H1})}{\max\,(B_{H0}) + \min\,(B_{H1})}$$

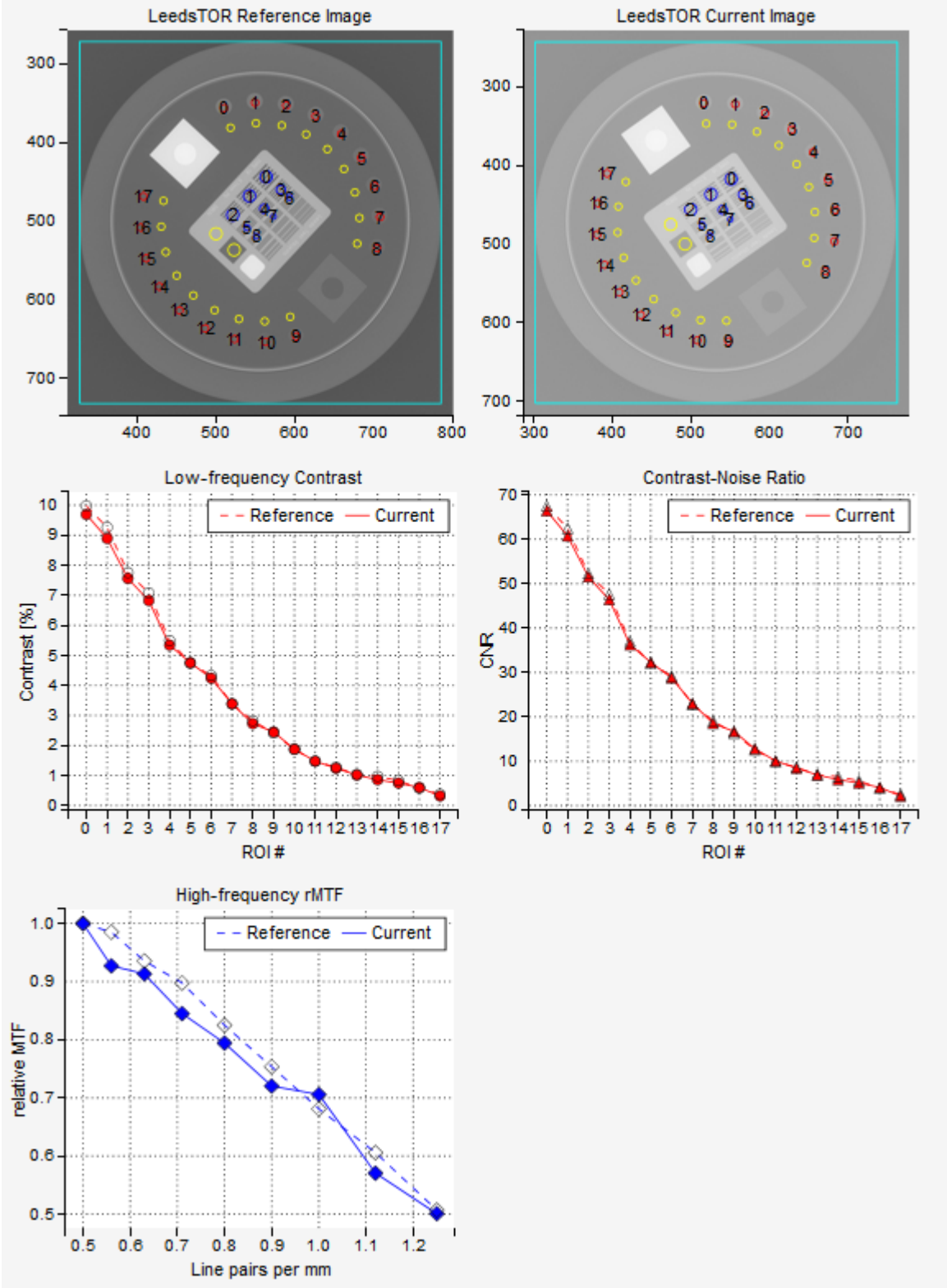
The function is normalized to 1 at the two background ROIs, if one is to take these two ROIs to form a starting line-pair region. Note that beause of this the plot is shifted to the right by one region. The green line on the plot represents the high_threshold defined in the configuration file. If the points are above the green line, then the lines in the corresponding region can be resolved.

Relative lppmm are calculated for 30%, 40%, 50% and 80% rMTF. The result is not in units of lppmm.

Median contrast is calculated as the median of contrast values for all low-frequency regions. Note that this is not the median of contrast-constants.

Median CNR is calculated as the median of all CNR values.

7.5.2 *Use Pylinac?* unchecked



Low-frequency contrast:

$$\text{contrast}\,(L_i) = 100 \times \frac{\text{median}\,(L_i) - \text{median}\,(B_{Li})}{\text{median}\,(B_{Li})}$$

Relative modulation transfer function:

$$\text{rMTF}\,(H_i) = \sqrt{(\text{std}(H_i))^2 - (\text{std}((B_{H0}))^2} \Big/ \sqrt{(\text{std}(H_0))^2 - (\text{std}(B_{H0}))^2}$$

rMTF is normalized to 1 at the max value, usually at H0.

lppmm is returned in real units based on the calibration of line pairs in the configuration file.

Noise is the average value of standard deviations for all low-frequency background regions:

$$\text{noise} = \frac{1}{18}\sum_{i=0}^{i=17}\text{std}\,(B_{Li})$$

Contrast-to-noise ratio is calculated like this:

$$\text{CNR}\,(L_i) = \frac{\text{median}\,(L_i) - \text{median}\,(B_{Li})}{\text{noise}}$$

CNR is the CNR of the first low-frequency region.

7.6 Las Vegas analysis

The Las Vegas phantom has only low-frequency regions.

7.6.1 *Use Pylinac?* checked

Low-fequency constant is the product of contrast and the diameter of the ROI:

$$\text{contrast constant}\,(L_i) = \frac{\text{median}\,(L_i) - \text{average}\,(B)}{\text{median}\,(L_i) + \text{average}\,(B)} \times \text{diameter}\,(L_i)$$

Where the average background is

$$\text{average}\,(B) = \frac{1}{4}\sum_{i=0}^{i=3}\text{median}\,(B_{Li})$$

The green line on the low-frequency plot represents the low_threshold defined in the configuration file. If contrast constant is above the green line, the ROI is considered "visible".

Contrast-to-noise ratio is defined as

$$\text{CNR}\,(L_i) = \frac{\text{median}\,(L_i) - \text{average}\,(B)}{\text{std}\,(L_i)}$$

Median contrast is calculated as the median of contrast values for all low-frequency regions. Note that this is not the median of contrast constants.

Median CNR is calculated as the median of all CNR values.

7.6.2 *Use Pylinac?* unchecked

Low-frequency contrast:

$$\text{contrast}\left(L_i\right) = 100 \times \frac{\text{median}\left(L_i\right) - \text{average}\left(B\right)}{\text{average}\left(B\right)}$$

Noise is the average of standard deviations for all low-frequency background regions:

$$\text{noise} = \frac{1}{4}\sum_{i=0}^{i=3} \text{std}\left(B_{Li}\right)$$

Contrast-to-noise ratio is calculated like this:

$$\text{CNR}\left(L_i\right) = \frac{\text{median}\left(L_i\right) - \text{average}\left(B\right)}{\text{noise}}$$

CNR is the CNR of the fourth low-frequency region.

7.7 Literature

CHAPTER

EIGHT

CT MODULE

Al four catphan models that Pylinac can analyze are implemented:

- Capthan 503
- Catphan 504
- Catphan 600
- Catphan 604

When scanning the phantom follow Pylinac's guidelines. Scan the whole phantom by placing the center of the phantom in the center of the image. Make sure it is well positioned, that is, aligned to the room lasers or other systems.

Note: The analysis of one single CT/CBCT scan with an axial length of 26 cm (approx. 130 slices with 2 mm slice thickness), and a resolution of 512x512 will take about 30 seconds to complete on a regular computer.

8.1 Options

Select phantom Select the phantom that you wish to analyze. Pylinac cannot determine the phantom model from the image, you must tell Pylinac which model you are using.

Select imager If you are using reference images, you can select the appropriate reference image for your imager. See the section "configuration" for more details about how to set the reference images.

Analyze reference? Chose whether you wish to analyze the reference image. If you chose to do so, the analysis will take twice as long.

Show HU Delta? If you check this box, the HU plot will show differences, and not absolute values of HU.

Tolerances Tolerances are defined in the configuration file.

8.2 Interpreting results

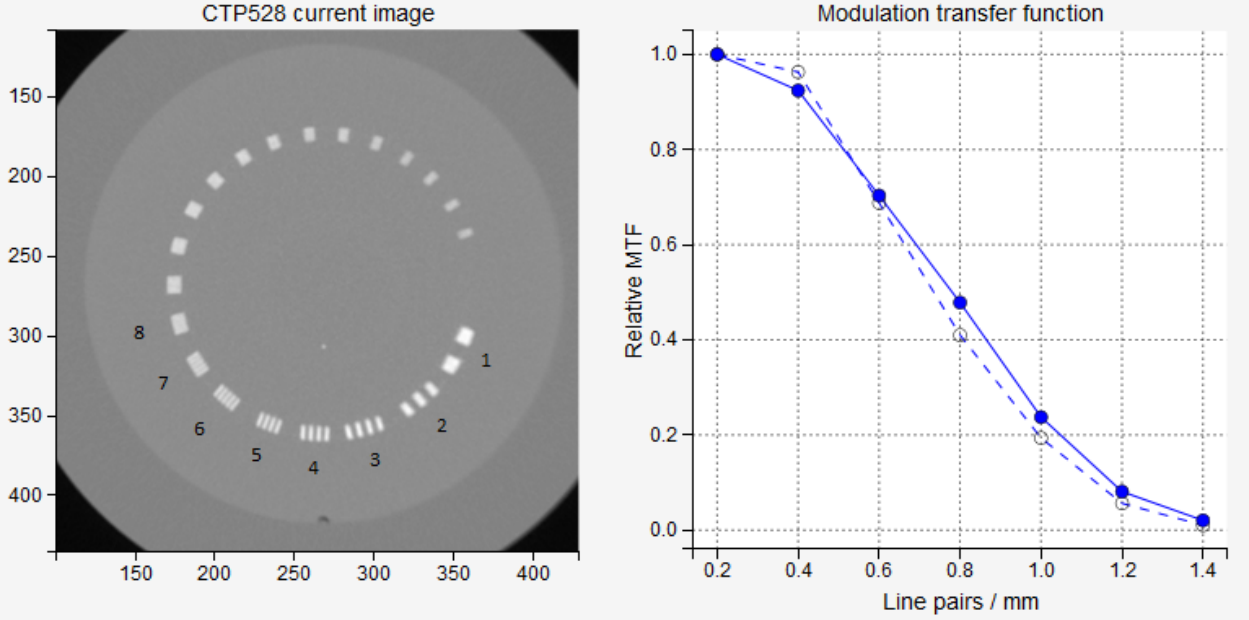
Each phantom model contains specific regions for studying different aspects of image quality. Models 504, 600 and 604 contain modules CTP404, CTP486, CTP528 and CTP515. Model 503 contains modules CTP404, CTP486 and CTP 528.

Note: The results of the analysis are gathered in tabs. Each module has its own tab. Next to the name of the module a **pass/fail** statement is given. A module gives general passing result if each submodule has passed the test. For example,

the CTP404 module gives a passing result if the HU, LCV, slice thickness etc. have passed.

8.2.1 CTP528

This module is used to test the high-frequency contrast of the image. Eight line-pair regions are analyzed.



The relative transfer modulation function is calculated similarly to that of the Planar imaging module. Within the circular band containing the line-pair region, profiles are drawn and averaged for each region. From the profiles min and max pixel values are calculated. For region 2, say, there are three peaks and two valleys, and hence three max values will be averaged into one, and two min values will be averaged in to one. At the end, the rMTF for this region will be:

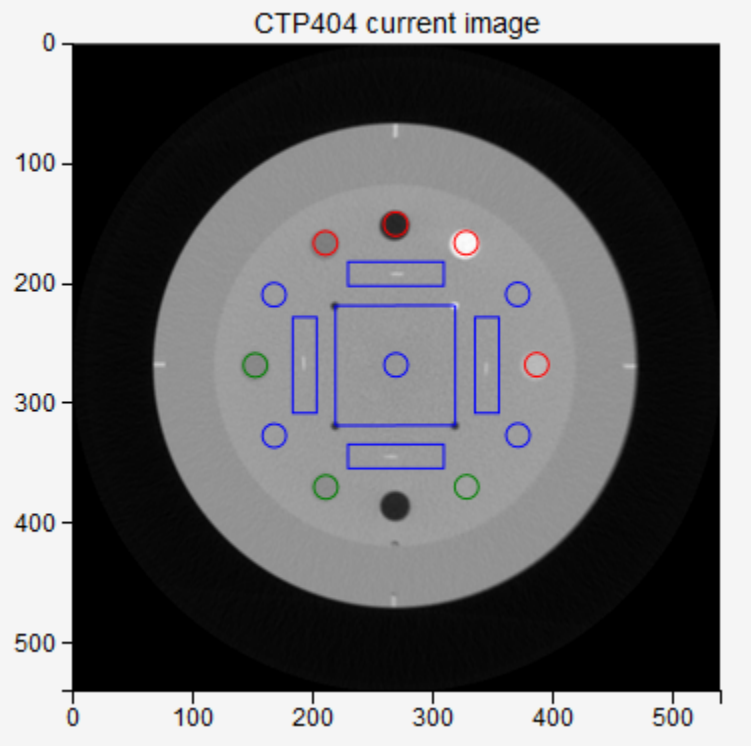
$$\text{rMTF}\left(H_i\right) = \frac{\max\left(H_i\right) - \min\left(H_i\right)}{\max\left(H_i\right) + \min\left(H_i\right)} \Big/ \frac{\max\left(H_0\right) - \min\left(H_0\right)}{\max\left(H_0\right) + \min\left(H_0\right)}$$

Where min and max are averaged min and max values.

If the difference between the reference and current rMTF (50)% is not greater than **MTF tolerance**, the test will pass.

8.2.2 CTP404

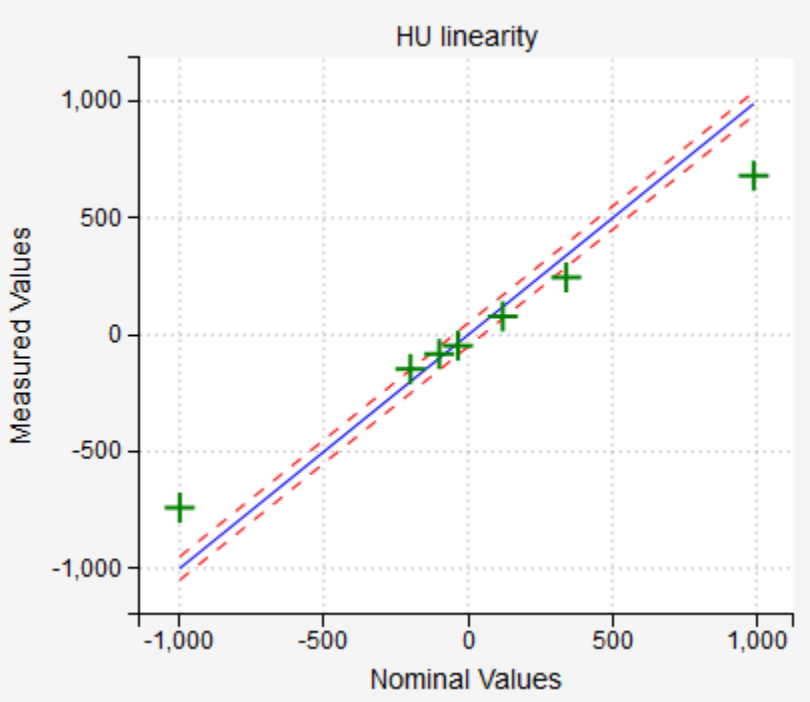
This module is used to test the low-frequency contrast, HU units and geometry of the image.



Note: Hover over the phantom image to reveal detected ROIs.

If a particular material has passed the HU test, it will be colored green, otherwise red. Blue circles are background ROIs.

A ROI has passed the test if the difference between nominal and current HU is less than **HU and uniformity tol.**.



Note: CBCT scans normally give a failed status for the HU units.

Low contrast visibility is calculated in two ways. The LCV is calculated according to Pylinac. Here are the formulas:

$$\text{LCV} = 2 \times \left| \frac{\text{median HU (LDPE)} - \text{median HU (Poly)}}{\text{std HU (LDPE)} + \text{std HU (Poly)}} \right|$$

$$\text{LCV2} = \left| \frac{\left[\text{nominal HU (Poly)} - \text{nominal HU (LDPE)}\right]/10}{2 \times \left[\text{mean HU (Poly)} - \text{mean HU (LDPE)}\right] / \left[\text{std HU (Poly)} + \text{std HU (LDPE)}\right]} \right|$$

The LCV will have a passing status if it is withing the **LCV tolerance**. In the configuration file you can chose whether this tolerance will be applied to LCV or LCV2.

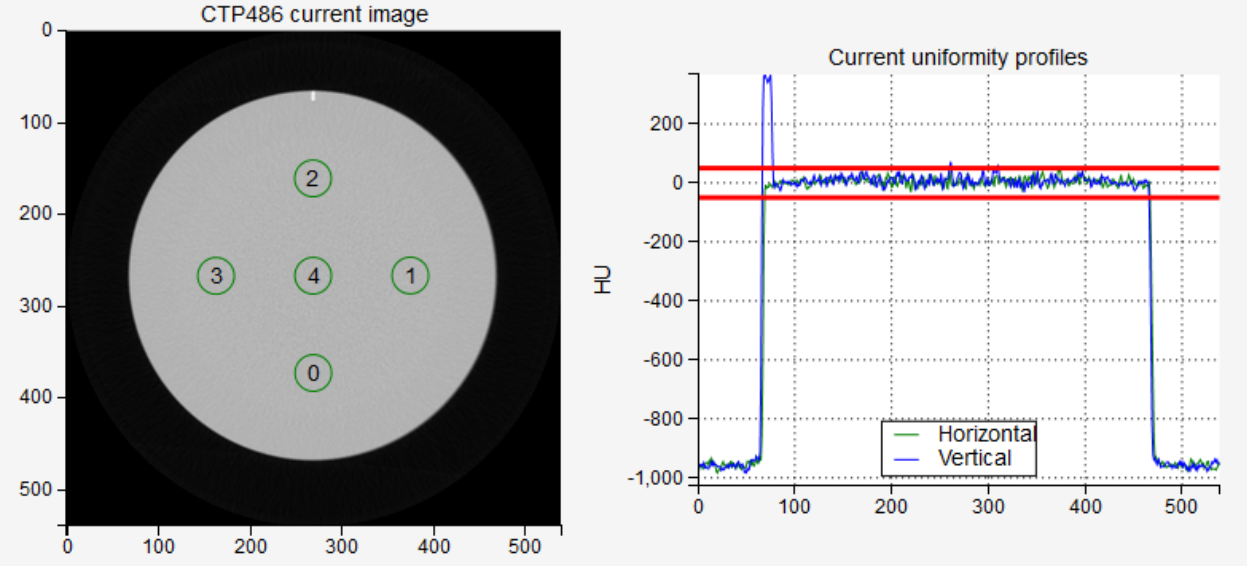
Slice thickness will pass the test if the difference between the nominal value and the measured value is within **Slice thickness tol.** If your image is very noisy or of low quality in general, it may happen that Pylinac will not calculate slice thickness properly.

Geometry scaling will pass the tolerance if all the lines (marked as Lines 1, 2, 3, 4) have a length of 50 mm plus-minus **Scaling tolerance**.

Phantom roll measures the axial rotation of the image. It has no tolerance, but you can use it to see if the image is axially well orientated. To do that, you must set the phantom accurately to well adjusted room lasers. Or use the spirit level. If you detect a phantom roll of, say, 1 degree, this means that your image is rotated. Which is bad.

8.3 CTP486

This module is used to test the uniformity of the image (HU).



The test will pass if the sampled HUs are close enough to the nominal values, the tolerance being **HU and uniformity tol.**.

The uniformity index is calculated in two ways:

$$\text{uniformity index} = 100 \times \max_{i=0,1,2,3} \left\{ \frac{\text{median HU (ROI}_i\text{)} - \text{median HU (center)}}{\text{median HU (center)} + 1000} \right\}$$

$$\text{uniformity index } 2 = 100 \times \frac{\max\left(\text{median PV(ROIs)}\right) - \min\left(\text{median PV(ROIs)}\right)}{\max\left(\text{median PV(ROIs)}\right)}$$

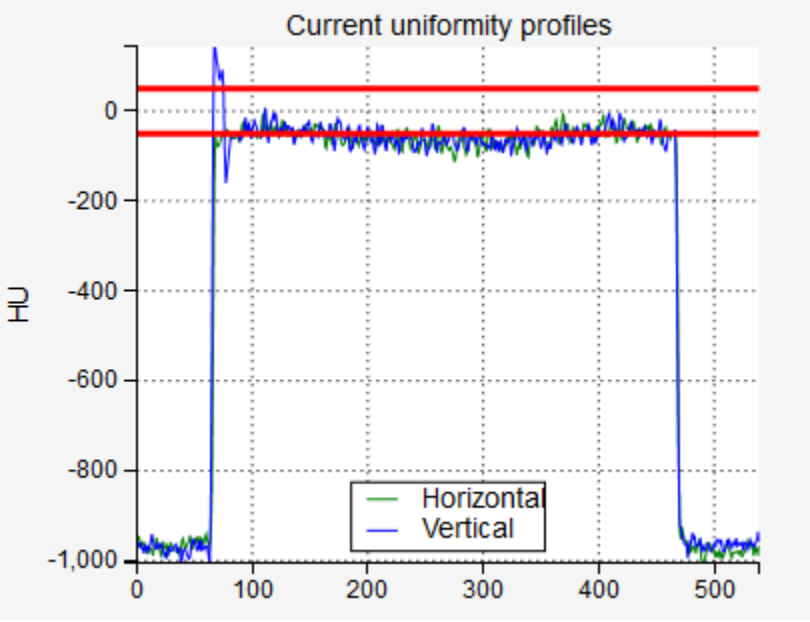
In the first formula, the maximum value is sought within a list of absolute values, however, the sign is preserved in the final presentation. In the second formula, median pixel values are evaluated not HUs. Both formulas give practically the same result.

The uniformity index will pass, if the HUs are within the **HU and uniformity** tolerance. The uniformity index 2 will pass, if it is within the **Uniformity index 2** tolerance (in percent).

The uniformity profiles may be uniform, but the test may still fail if the HUs are not close enough to the expected nominal values.

Note: For CBCT scans the uniformity index may fail for large FOV or in combination with some types of filters. This is not a reason to be alarmed, it is normal. Neither HUs or uniformity profiles are of great relevance in CBCT scans.

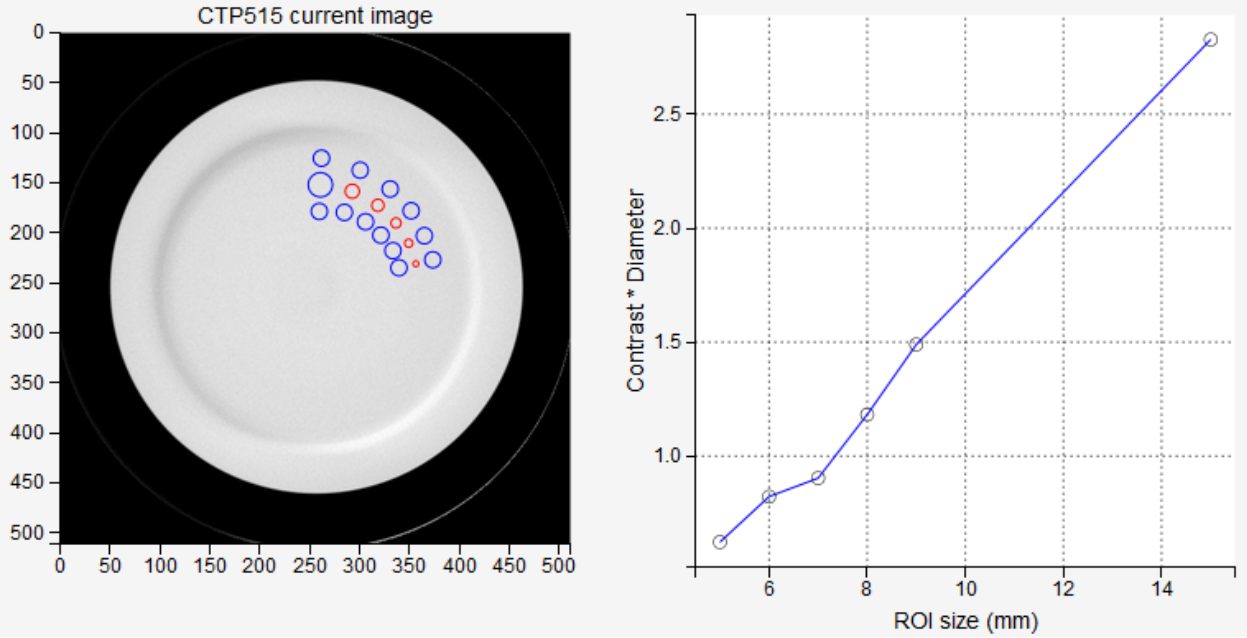
The image below gives an example of a perfectly normal CBCT scan with passing Uniformity index, but a failed CTP486 test.



8.4 CTP515

This module is used to study low-contrast resolution. Pylinac samples the image with several ROIs. The usual contrast constant is calculated and plotted as a function of circle diameter.

The test will pass if Pylinac detects at least a certain number of ROIs that are defined with **Low contrast tolerance** (number of ROIs that must be seen). In order for a ROI to be seen, the CNR constant must be greater than the **CNR threshold**. CNR constant is the product of CNR and circle diameter.



The CNR is defined as

$$\text{CNR}\left(L_i\right) = \frac{\text{median}\left(L_i\right) - \text{median}\left(B_i\right)}{\text{std}\left(L_i\right)}$$

Background Bi are the average of inner and outer background ROIs corresponding to Li.

CHAPTER

NINE

DYNALOG MODULE

This module is used to analyze Dynalogs. It is meant to be used for everyday tracking of MLC dynamics during treatment. To be able to use it, several steps must be followed.

Note: Trajectory log analysis is not available in this version of QAserver.

- The module **analyze_dynalogs.py** must be used to fill the database with dynalogs. This module reads dynalogs from a dedicated folder (local or network), analyzes them and saves them into a special QAserver database so that they can be reviewed later.
- You should set you MLC controller to make a copy of each dynalog to a network folder so that you will be able to access it remotely. It is best to do the analysis in the evening so that you collect all dynalogs of the day.
- You can configure QAserver to send you an email containing a short summary of the analysis.

Please read the configuration instruction carefully in order to set up the dynalog module.

9.1 How to start?

Collect a couple of dynalogs and save them to a local folder. In **config.ini** set this path to *DYNALOG_REPOSITORIES*. Define a name for your repository, say *REPOSITORIES_LABELS = Linac1*. Set *SEND_EMAIL = False*. Then run the analyze_dynalogs.py module.

After a successful run, you should have dynalogs in the database.

9.2 How to get data?

1. I want to get all patients/records for a particular date

Click Filter patients by date and chose the date. **Patient** dropdown will be populated. In the **Record** dropdown you can then select the record you want.

2. I want to get all the patients for a particular date and a particular folder (ie machine)

In the **select folder** dropdown chose the appropriate folder and repeat 1.

3. I want to get all the records for a particular patient

Search the patient with the **Filter patient list**. You can filter the results with **Filter patients by folder**.

4. I want to get all the patients from a particular folder (ie. machine)

Clear the **Filter patient list**, select the appropriate **Filter patients by folder** and click Go!

5. How to find out what records have been saved last?

Click the **Last upload** button.

6. How to get a summary of all the records for a particular date?

Select a date and click the **For selected date** button.

7. How to analyze one particular dynalog?

Chose the record and select the desired field (**Select field for analysis**). Select the appropriate MLC type (**Select MLC**) and then click **Analyze**.

9.3 Record data

Field The number of the field. If dynalog does not contain a record of the field id, the value will be "NA".

Gantry Gantry angle or arc starting angle.

Time Time of treatment.

Snaps Number of snapshots. Snapshots are 50 ms apart.

Holds The number of times that the MLC controller demanded beam hold.

Max RMS [mm] Maximum RMS over all snapshots and all leaves.

Max RMS2 [mm] Similar to MAX RMS, except that only snapshots when beam was ON and there were no holdoffs are included in the calculation.

Max DIFF [mm] Maximum difference between planned and measured leaf positions, calculated for all leaves and all snapshots.

Max DIFF2 [mm] Similar to Max DIFF2, except that only those snapshots are included when beam was ON and there were no beam holdoffs.

RMS avg [mm] Average RMS for moving leaves only (when beam was ON and there were no holdoffs). Leaf is moving if the standard deviation of its position over all snapshots is lower than 0.1 mm.

gamma avg Average gamma.

gamma<1 [%] Percentage of points that have gamma smaller than 1.

dd/dta/thresh/res Settings used for gamma calculation: dose difference, distance to agreement, threshold, resolution. See your configuration file.

Folder Name or label of the folder from where dynalogs were taken. Normally this would be the name of your linac.

9.4 Analysis

QAserver returns Pylinac's analysis with some additional features.

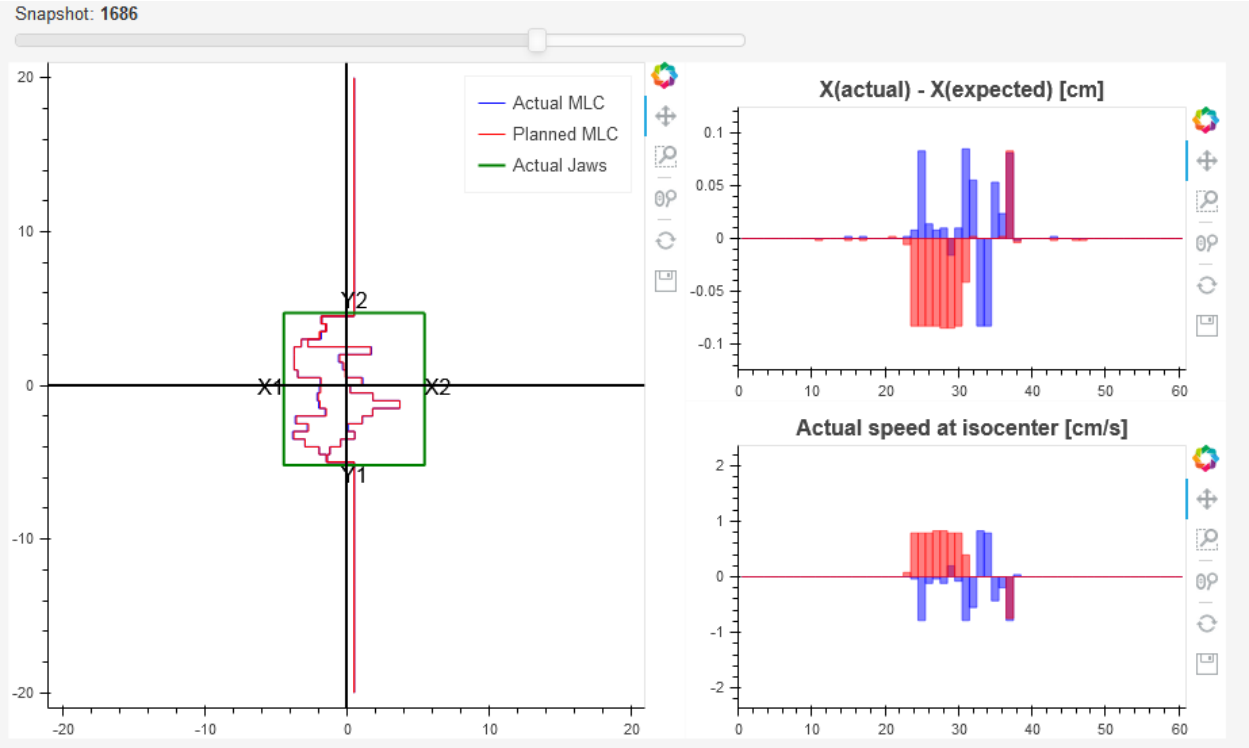
9.4.1 MLC

Here you can observe planned and actual MLC movements. If you don't see the animation, then there is a problem with your installation of Bokeh.

The speed of the leaves is calculated as the first time difference of leaf position. It is severely prone to noise. In future release it will be calculated more robustly.

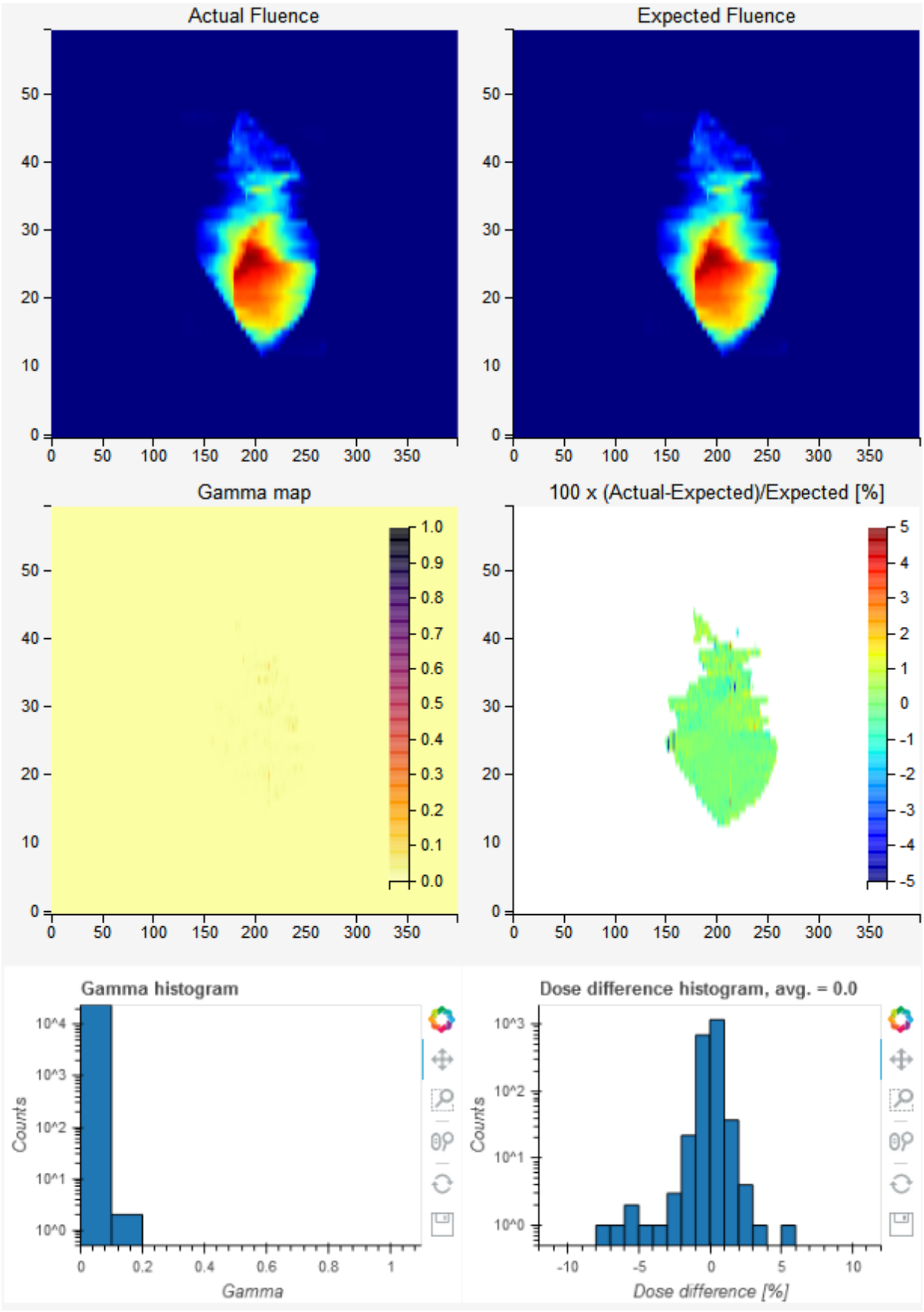
Red bars correspond to bank A and blue bars correspond to bank B.

Beam status has two signals: CONT/STOP and ON/OFF. The first one is the beam hold off, the second one is beam on or beam off.



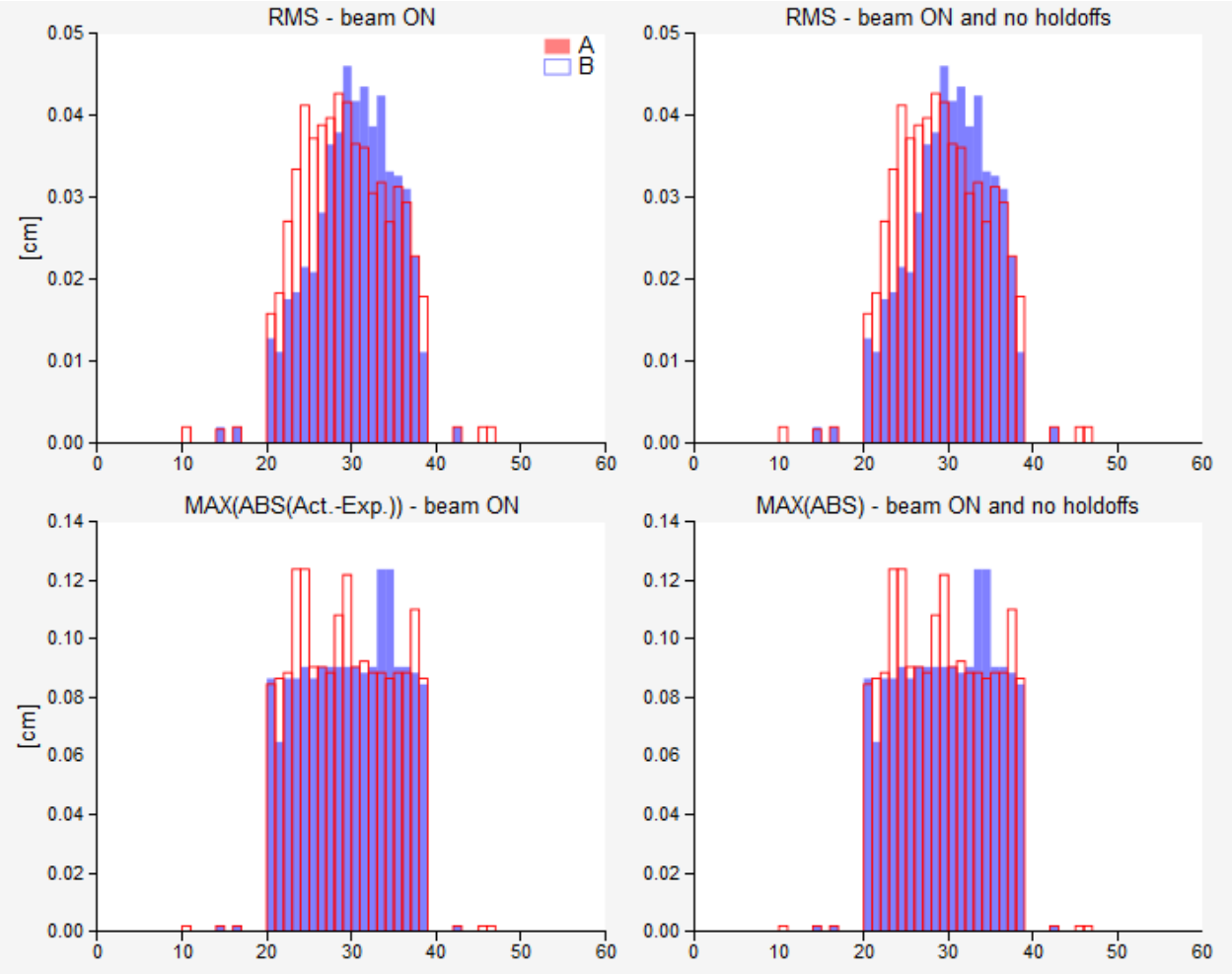
9.4.2 Fluence

Actual and planned fluence maps are presented. Besides the gamma map a dose difference map is also shown.

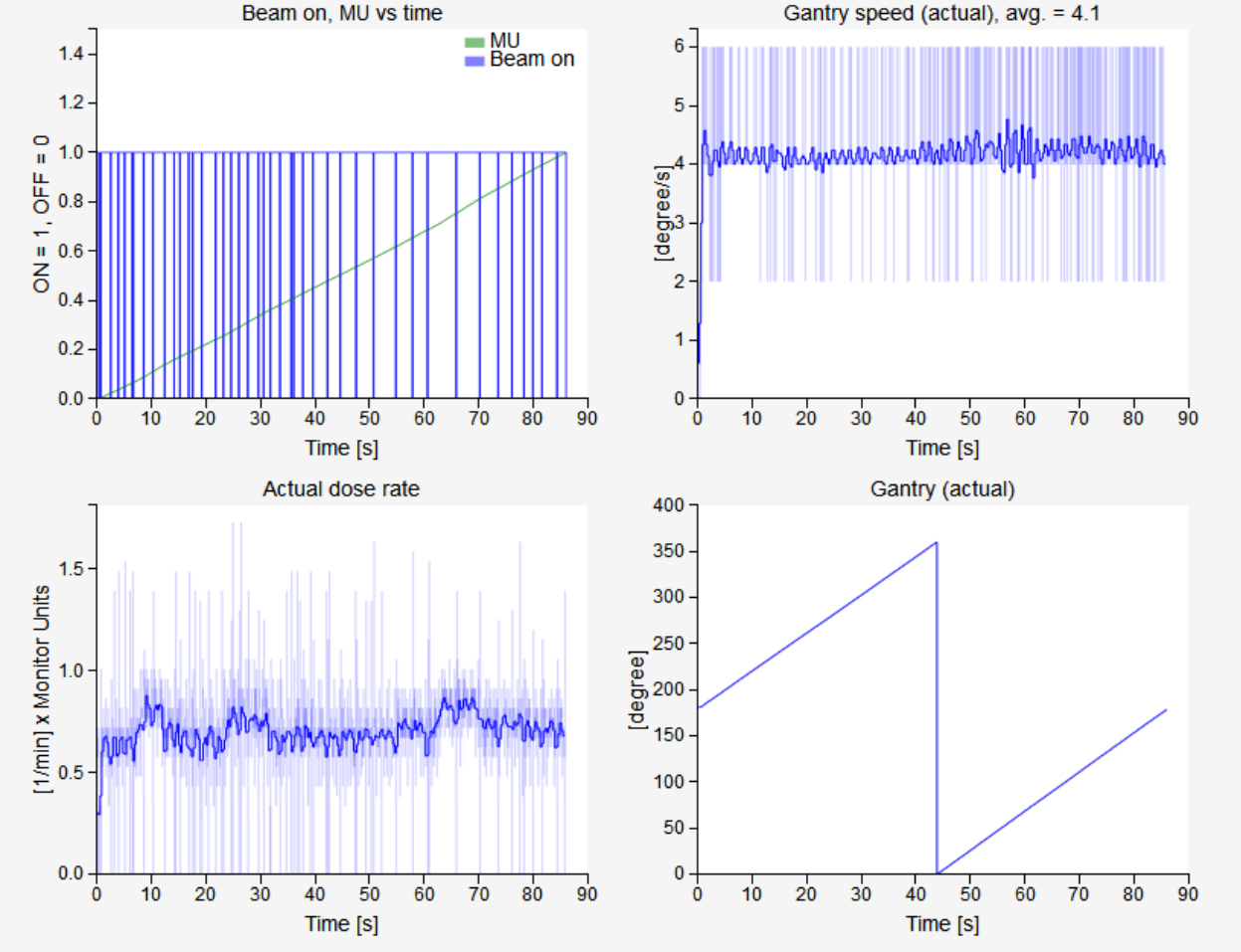

Actual Fluence
Expected Fluence
Gamma map
100 x (Actual-Expected)/Expected [%]
Gamma histogram
Counts
Gamma
Dose difference histogram, avg. = 0.0
Counts
Dose difference [%]

9.4.3 Error histogram

Four histograms are shown: RMS and RMS2, MAX DIFF and MAX DIFF2.



9.4.4 Dose rate



The first graph shows how beam status is changing over time and how meterset weight is adding up.

The second plot is the gantry speed. Gantry speed is not calculated with first-order difference. Instead, gantry angle as a function of time (snapshot) is filtered with the savgol filter using every other point out of the whole snapshot collection. Then the ordinary first order difference is calculated using every third point of the filtered function. This plot is shown in dark blue color. Underneath you will see a faint blue plot, that is the first-order difference of gantry position. It should be extremely noisy.

The dose rate graph is calculated similarly to gantry speed.

CHAPTER

TEN

FLATNESS/SYMMETRY MODULE

Warning: Flatness and symmetry measured with the portal imaging device is not the same as that measured in the water tank. Be sure you understand the results of this module.

10.1 Settings

Analysis definition: Pick the definition that suits you. See Pylinac for further info.

Center: Here you can set the point where profiles will be extracted from the image. This point only defines the horizontal and vertical line for the profiles, it does not define the center of the profiles.

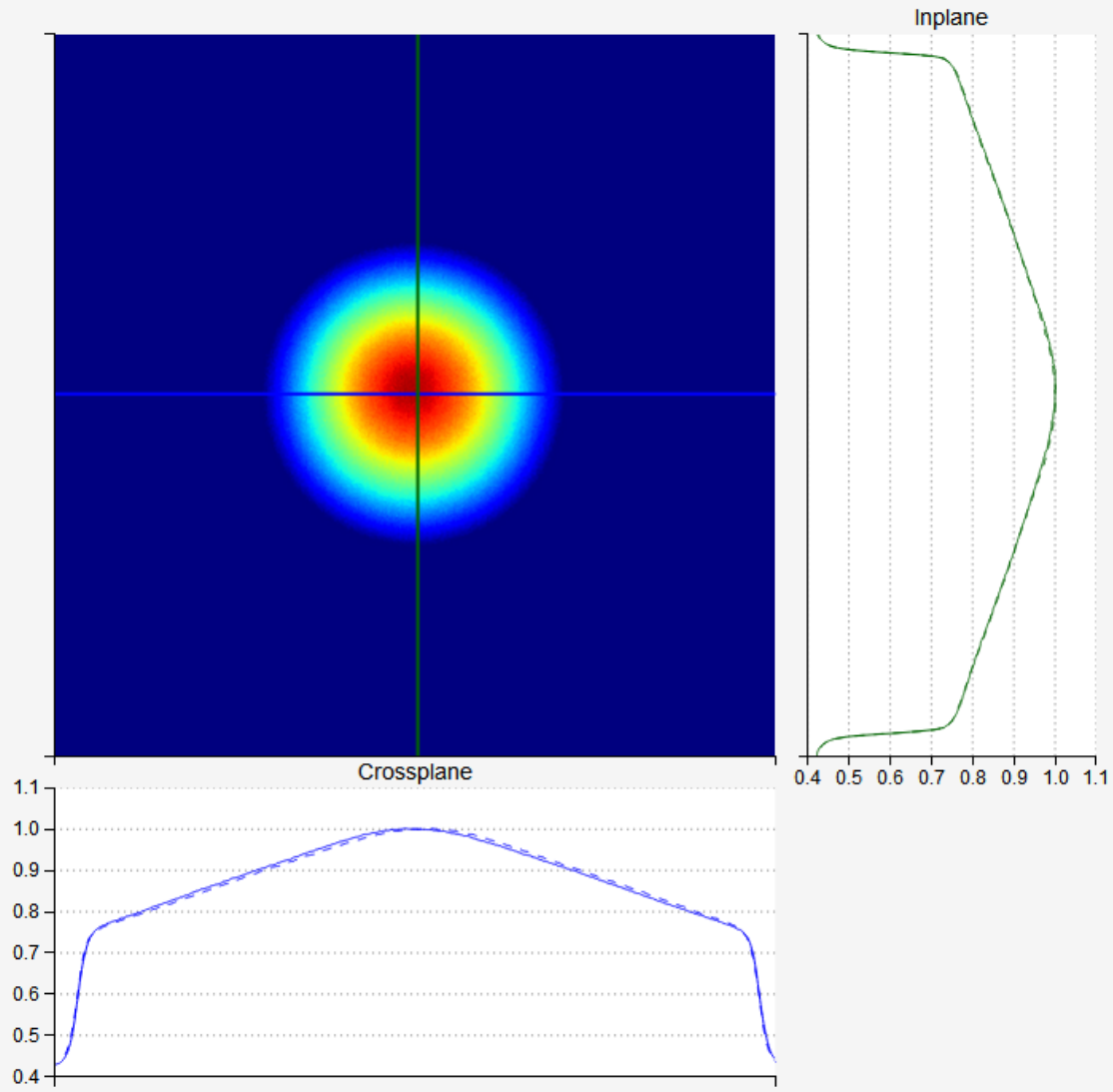
If set to Automatic, Pylinac will do it on it's own. If set to Manual, you can enter coordinates of the pixel. If set to CAX, a special function will try to find the center of the field. CAX method may not work well on non-flattened profiles.

If you wish to manually set the origin, then start with some guess, then on the plotted image use the mouse cursor to finely set the final point and enter it into the text box.

Invert image: Check this if you want to invert the image.

10.2 Results

	Horizontal	Vertical
Flatness	124.57	124.7
Symmetry	101.2	100.74



CHAPTER

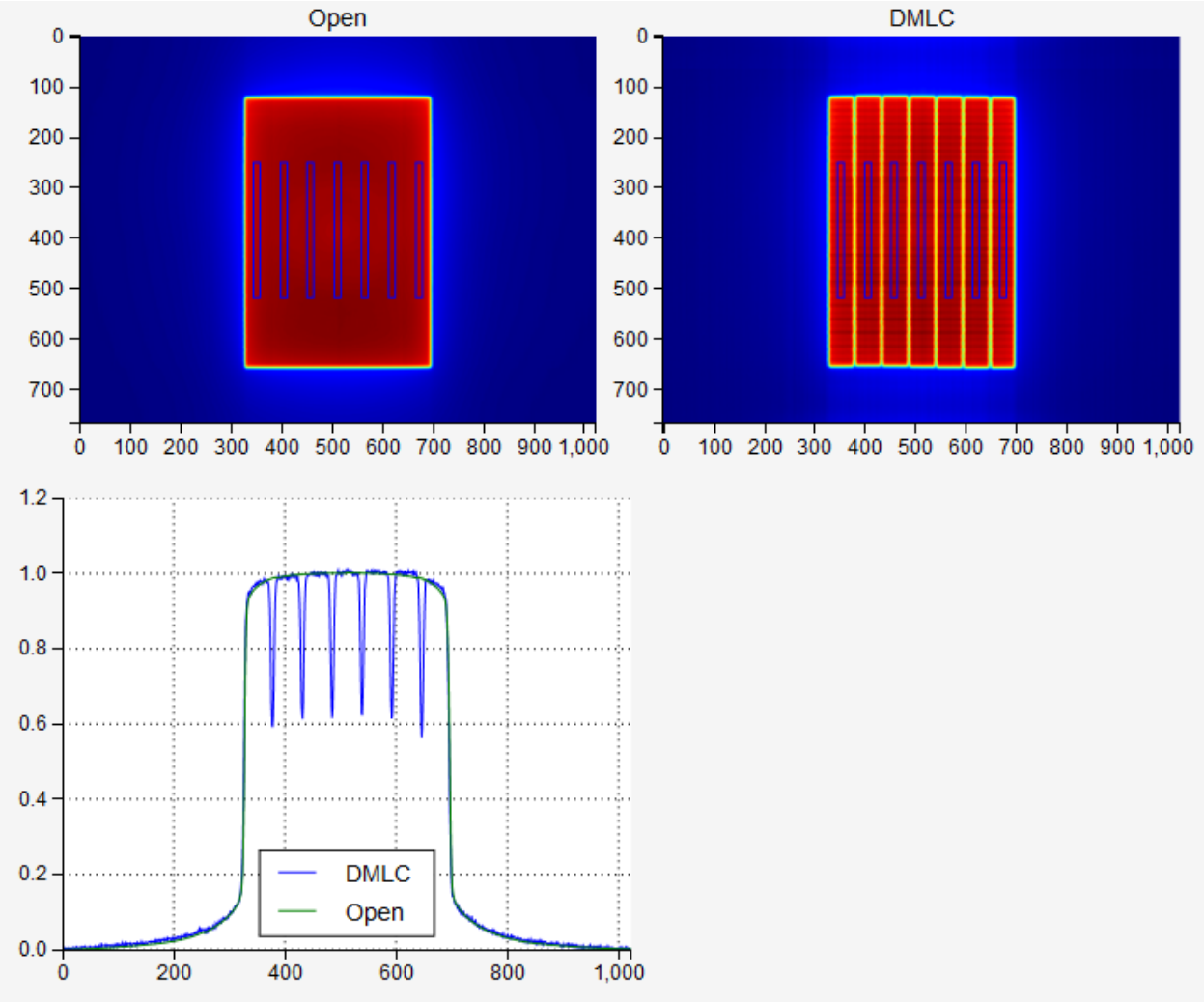
ELEVEN

VMAT MODULE

With this module you can analyze DRGS and DRMLC images according to Varian RapidArc QA. See Pylinac for full explanation.

Using the module is simple: chose image 1 and 2, one being DRMLC or DRGS, and the other an open field image. Click Analyze.

For each ROI, Rcorr and diff are calculated. If diff is within the tolerance, as defined in the configuration file, the ROI will pass the test. If all ROIs have passed, the test will have a passing status.



CHAPTER

TWELVE

FIELD SIZE MODULE

This module can be used to measure the coincidence of mechanical and radiation isocenters, as well as for measuring the positions of MLC leaves and jaws.

Note: The module is not part of Pylinac, but it uses Pylinac's methods. It has a known deficiency: the lateral positions of MLC edges are not accurate because non-radiation (mechanical) edges are assumed.

12.1 Options

Image 1 The image that determines the location of the mechanical isocenter.

Image 2 The image you wish to analyze.

MLC type Pick the right MLC leaves. This option defines the lateral spread of MLC sampling points. Note that the lateral spread uses physical dimensions of leaves, not radiation dimensions.

MLC direction Leaves can be in the left-right direction or up-down. If the collimator is not 0/90, chose something that works best.

Center The method to define the mechanical center. This applies to the first image.

Clip box The size of the central portion of the image beyond which pixel values will be set to background signal. If you don't want to clip the image, put 0.

MLC points The number of sampling points for each MLC leaf. Minimum values is 1.

Jaw points The number of points to sample jaws. Points are evenly spread.

mmpd Millimeter per dot. Only works if *Plate* is not chosen for *Set center*. If you wish Pylinac to determine the mmpd from dicom tags, leave the value at 0.

CAX point If *Manual* is chosen for *Set center*, this options allows to define the position of the mechanical center in pixels. The origin is in the bottom left corner of the image.

Invert image? Check this if you want to invert the image. This option should not be necessary because Pylinac will check for proper inversion.

12.2 How to use the module

The module needs two images. The first image is used to determine the location of the mechanical isocenter. Since we will not move the EPID while measuring, the position of the mechanical isocenter as seen by the EPID will not change, hence the pixel position of the isocenter can be transferred from the first image to the second. The second

image is a field of arbitrary size, it does not have to be square. Both images must belong to the same series. Try to have the field central on the EPID as well as at the right angle. Presence of couch in the beam should not cause problems.

The first image can be:

- A simple square field without any markers. In this case we will tell the software that we will use the field CAX as the mechanical isocenter.
- An image of Elekta Agility calibration plate accurately aligned to the optical crosshair.
- An image of the BB, aligned with the mechanical isocenter. When acquiring second, third image the BB may remain in position.
- Any image of a metal marker that shows the location of the mechanical isocenter. The location of the marker can be determined manually with a mouse and then entered into the software. You may leave the marker in place when acquiring subsequent images.

When acquiring image, make sure that the field is at least 0.5-1 cm away from the edges of the image.

12.2.1 Using *CAX*

1. Acquire an image of a simple square field, say 10 cm x 10 cm that you know is well center around the mechanical isocenter. You can even use SRS cones.
2. Acquire an image of a field you would like to measure, say 5 cm x 5 cm.
3. Set first image as Image 1, and second image as Image 2. For *Set center* use **CAX**. If you wish to use your own pixel size, you can enter it into mmpd. Also, set your MLC orientation.

The CAX point will be transferred from the first image to the second as the "mechanical isocenter".

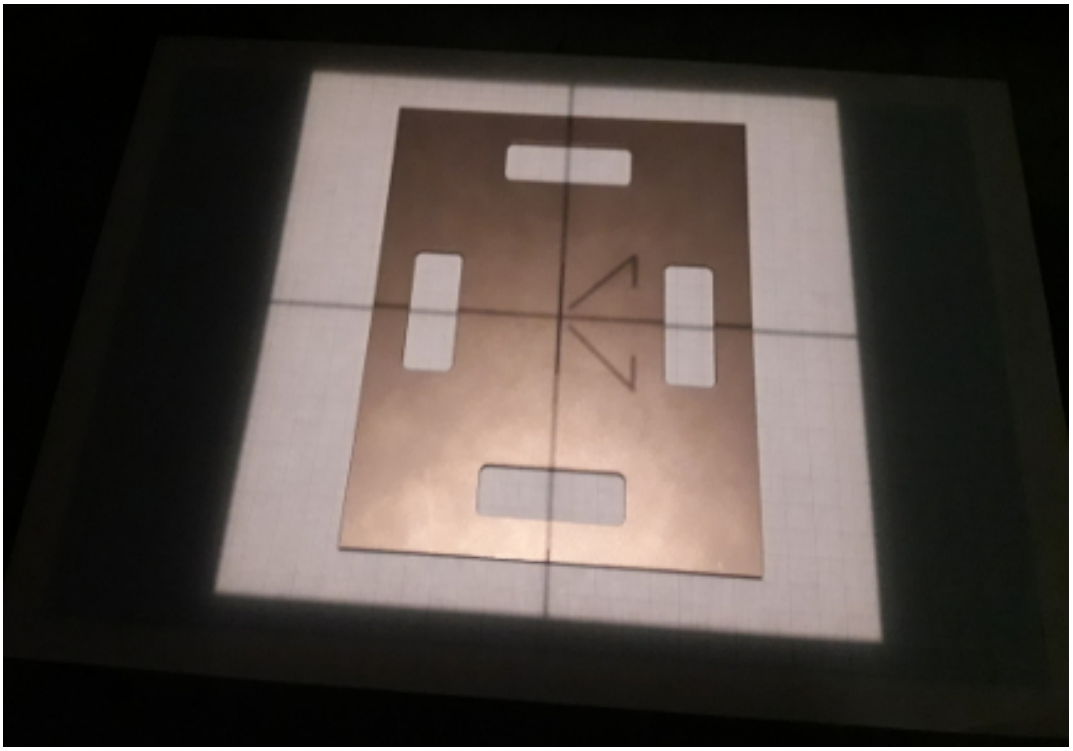
12.2.2 Using *BB*

1. Acquire an image of the BB. The BB should be well aligned to the mechanical isocenter (optical crosshair). You can use any field size. You can use SRS cones.
2. Acquire an image of a field you would like to measure, say 5 cm x 5 cm. The presence of BB in the image should not influence the analysis much, unless the BB is on a thick holder.
3. Set the first image as Image 1, and the second image as Image 2. For *Set center* use **BB**. If you wish to use your own pixel size, you can enter it into mmpd. Also, set your MLC orientation.

The center of the BB will be transferred from the first image to the second as the "mechanical isocenter".

12.2.3 Using *Plate*

1. Acquire an image of Elekta's plate. Accurately align the plate with the optical crosshair. Make sure that the optical crosshair is almost perfectly centered. After the plate is set, use a field size of 24 cm x 24 cm and 10-20 MU to make the image.

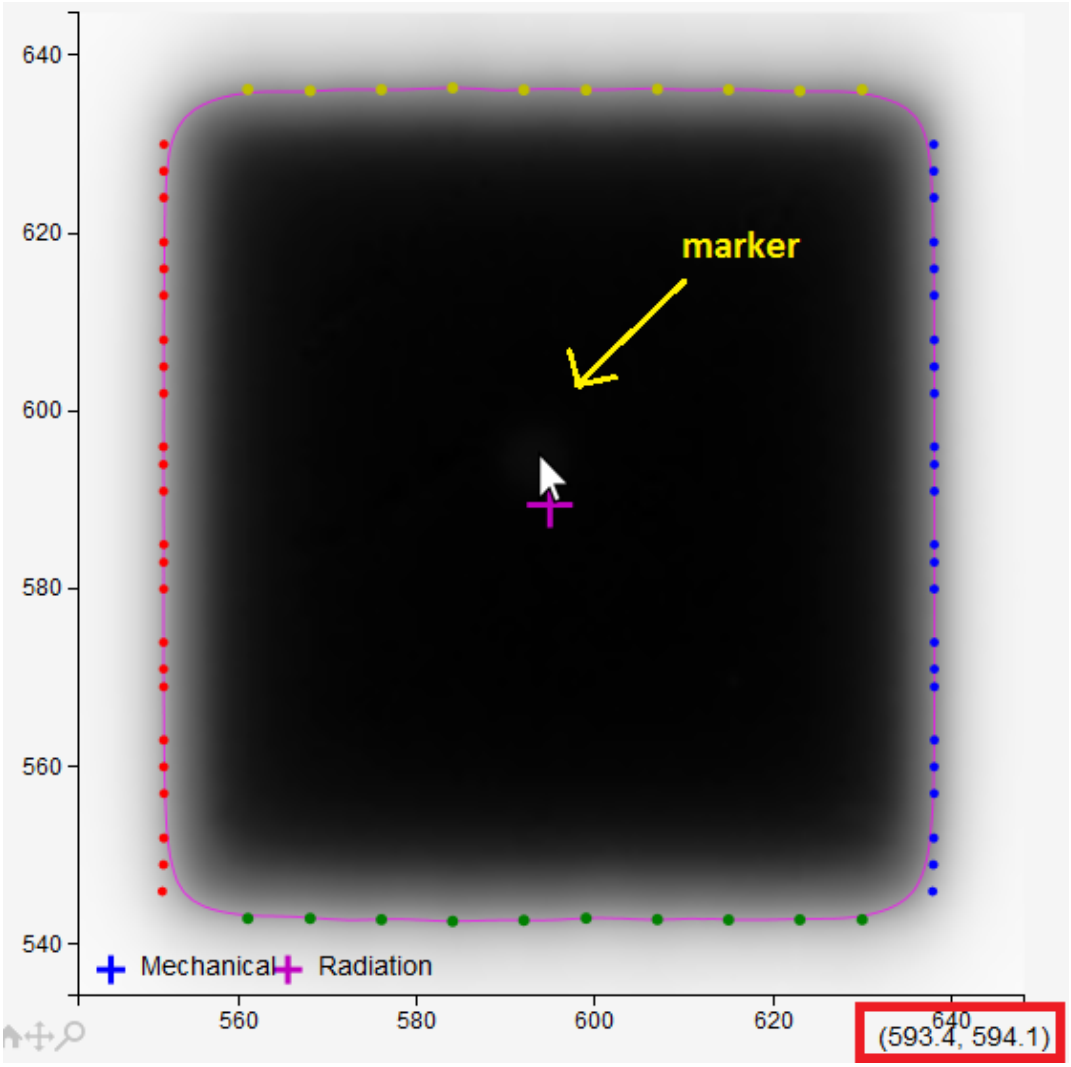


2. Acquire an image of a field you would like to measure, say 5 cm x 5 cm. It is possible to measure this field without removing the plate, but for larger fields it is recommended to remove the plate.
3. Set the first image as Image 1, and the second image as Image 2. For *Set center* use **Plate**. Set your MLC orientation.

The image of the plate will be analyzed and the center of it will be transferred to the second image as the "mechanical isocenter". Also, from the size of the windows in the plate, mmpd will be determined and used independently from dicom tags.

12.2.4 Using *Manual*

1. Put a small metal marker on the optical crosshair. Acquire an image of it.
2. You can leave the marker on and just acquire more interesting images of various fields.
3. Set the first image as Image 1, and the second/third image as Image 2. For *Set center* use **Manual**. Leave the *CAX point* as it is (510). Click Analyze.
4. Find the marker on the image and zoom in. With the mouse pointer go to the center of the marker and read the horizontal and vertical pixel position (shown in red square below). Enter the values into *CAX point*. Click Analyze once more.



12.3 Results

To determine leaf and jaw positions the image is sampled horizontally and vertically with multiple profiles. If MLC orientation is set to "X", then horizontal profiles will be clustered for each leaf, and vertical profiles will be evenly spread to find jaws. Lateral leaf dimensions are defined in a python module. Say Elekta_160 is chosen for MLC type. The leaf edges are distributed in the pattern: 0, -+5 mm, -+10 mm, -+15 mm, -20 mm … -+200 mm. The number of profiles that define the position of one leaf is set with MLC points. If 3, then a leaf will be sampled with 3 profile lines for each region between leaf lateral edges. The result in the table is the average over the three lines.

In the direction perpendicular to leaves profiles are extracted equidistantly. Their number is set with the parameter "Jaw points".

The position of one leaf is determined with the 50 % penumbra point. The average size of the field is calculated as the average over all widths excepts the first and the last which may be a bit off.

Leaf position

Leaf index	Left [mm]	Right [mm]	Width [mm]
-4	25.02	24.88	49.9
-3	24.94	24.81	49.75
-2	25.08	24.81	49.89
-1	24.95	24.85	49.8
1	24.9	24.87	49.77
2	24.99	24.85	49.83
3	25.01	24.86	49.87
4	24.95	24.82	49.76
5	25.01	24.89	49.9
Avg. without edge points	24.98	24.84	49.81

Field rotation can only be calculated for large fields with small collimator rotation. You can use it to check your collimator angle calibration if you trust the EPID position. Or vice versa. It is calculated simply from the slope of four lines fitted to the edges of the field.

Field rotation

Edge points (leaves) are not included in the calculation of field rotation.

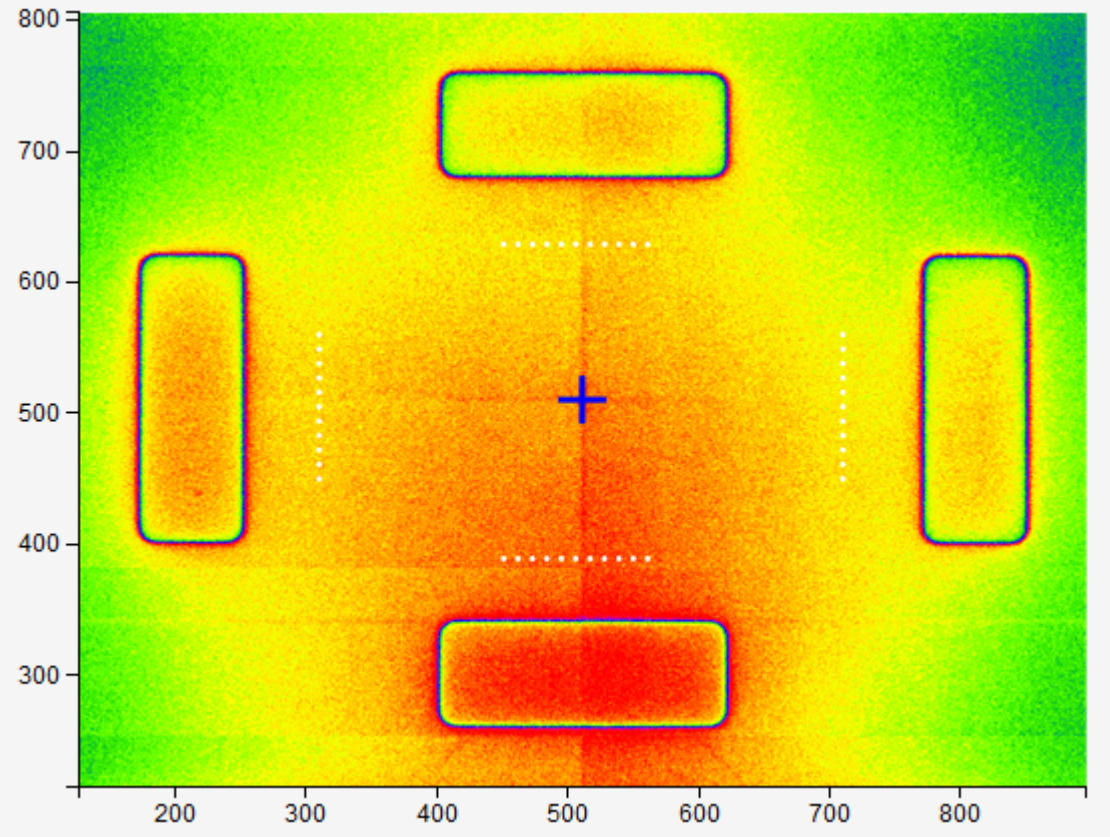
MLC left [°]	MLC right [°]	Jaw left [°]	Jaw right [°]
-0.04	-0.1	-0.06	0.06

Radiation center offset from mechanical center is the difference in position between the mechanical center defined by the first image and the CAX from the second image.

Radiation center offset from mechanical center

Δ X [mm]	Δ Y [mm]
-0.17	0.58

Image 1 is displayed in yellow colors. If you are using the plate to determine the mechanical isocenter, note the following. The plate has four windows cut out. Each window is well centered with respect to the markings on the plate, and has a width of 2 cm. This allows to accurately find the central point as well as pixel size at the isocenter. So the script has no need to use SDD from dicom tags.



Yellow dots on the picture of the plate show where the windows are sampled for horizontal/vertical inner penumbra points.

Note: Before analysis, 2 rows/columns of pixels are removed from the edges of each image.

12.3.1 A note on colors

- Violet cross and line: the center of field CAX and the 50-% isodose line.
- Blue cross: the mechanical center.
- Blue and red dots: MLC positions.
- Green and yellow dots: penumbra points where jaws are sampled.
- Red contour around BB: the region within which the BB is sought.

12.4 What you should know

So far I have not had the time to find a better way to define the latter spread of sampling points for MLC leaves. You will notice, if you shape an irregular field, which you can also analyze btw, that the script tends to miss the leaves as you move out from the isocenter. This will be fixed in future versions.

Warning: Also note that leaf and jaw positions are determined by simple penumbra point searching. Other software may give you a different result.

CHAPTER

THIRTEEN

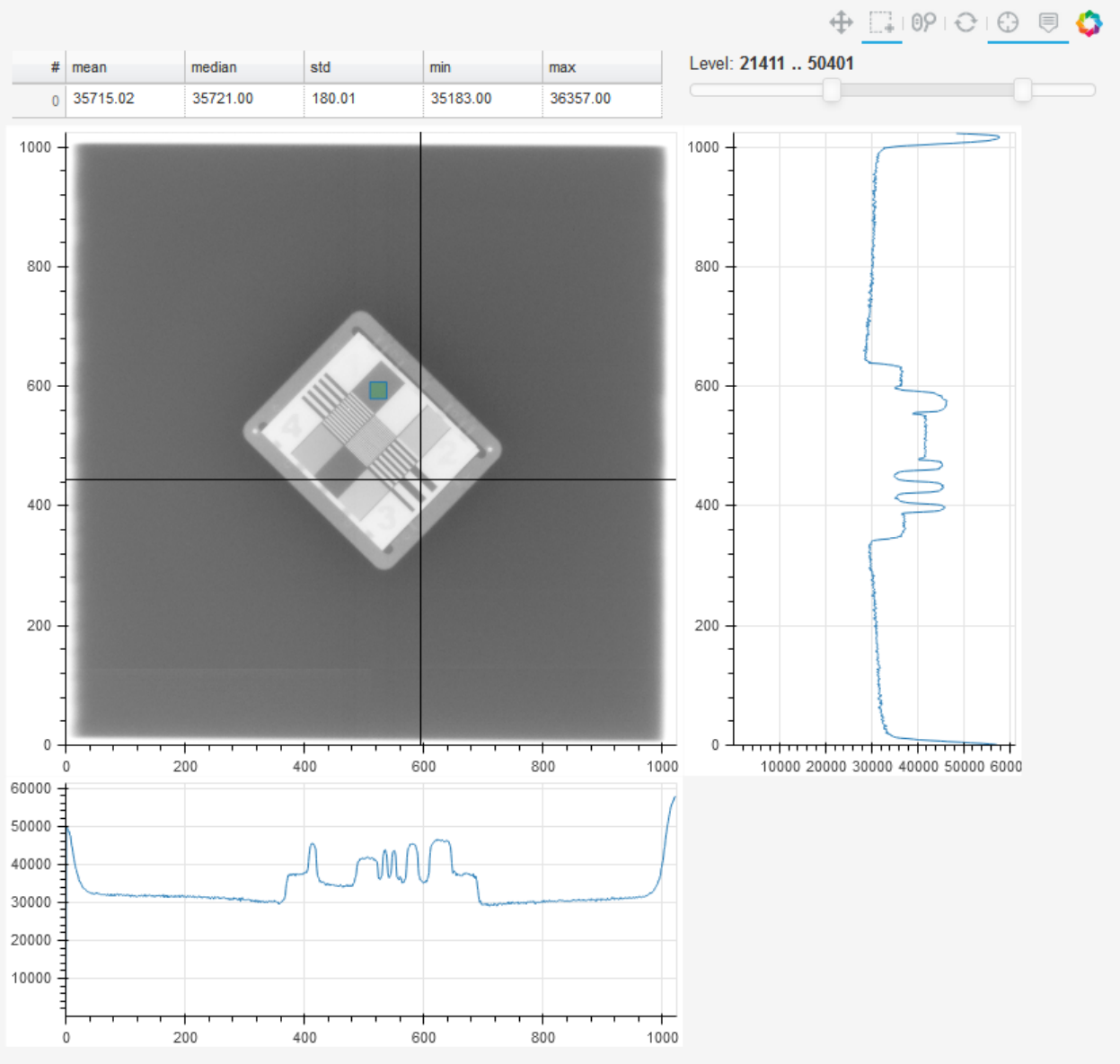
IMAGE REVIEW

This is a complementary module that can be used to extract image pixel data. It is not part of Pylinac.

Chose the image you wish to inspect and click Display. In the resulting diagram you can:

- use the crosshair to observe profiles,
- draw a rectangle with the select box to get pixel info,
- change the gray-scale display by moving slider level.

If needed, you can invert the image or convert HU back to pixel data.


#
mean
median
std
min
max
0
35715.02
35721.00
180.01
35183.00
36357.00
Level: 21411 .. 50401
1000
800
600
400
200
0
0
200
400
600
800
1000
10000 20000 30000 40000 50000 6000
60000
50000
40000
30000
20000
10000

CHAPTER

FOURTEEN

CHANGELOG

14.1 Version 1.0

The initial release.

CHAPTER

FIFTEEN

LICENSE

This software was put together from various sources that come with respective licenses. Please read the license files of that software. I tried wholeheartedly to document the license conditions of all the software that I used to construct QAserver. If I failed at some point, I can only ask for forgiveness.

For the rest, the MIT license applies.