ROI extraction from CDMAM 4.0 DBT volumes and image quality evaluation with DL-based observer

A. Makeev

January 22, 2025

Extracting images for DL-based performance testing

Regions of interest (ROIs) can be cropped from DBT volumes (DICOM) in two ways, "manually" with minimal one-time intervention, and semi-automatically, but with potential need of adjusting blob detector algorithm parameters. If the user followed data collection instructions and ensured that the CDMAM placement wasn't altered between acquisitions, manual way is preferred due to its simplicity.

Manual mode

Here we assume that the CDMAM position was unchanged with respect to the x-ray detector during image acquisition (i.e. phantom assembly was fixed on the breast support plate with glue dots, electrical tape, etc., for the set of scans with given PMMA thickness). In this case a user can open one of the DBT volumes with ImageJ viewer, navigate to the central slice (in-focus plane) to localize and write down four corner coordinates of the CDMAM grid $(A, B, C, D) = (x_A, y_A,\ \ x_B, y_B,\ \ x_C, y_C,\ \ x_D, y_D)$ as indicated in Figure 1 below:

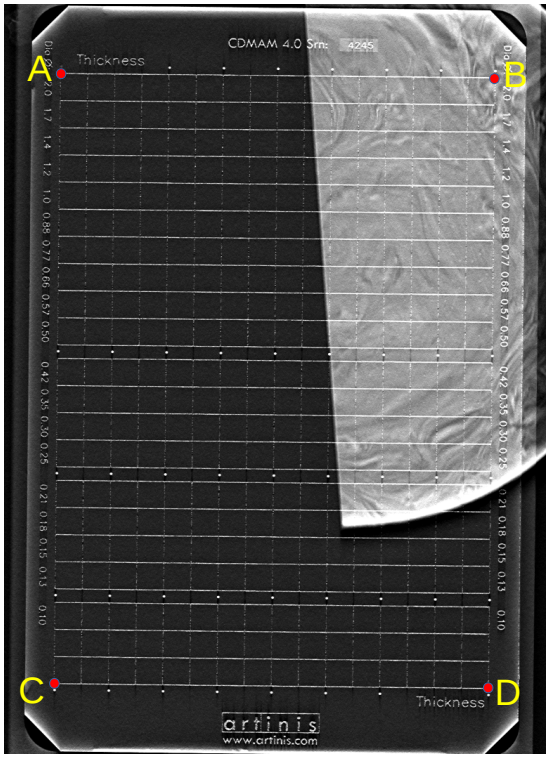


Figure 1: CDMAM grid corners (A, B, C, D). *Both* manual and auto modes use (A, B, C, D) for ROI extraction. Small white dots are CDMAM fiducial markers (N_{markers} =42).

Having this information one can proceed to perform ROI extraction in manual mode, by

Contact: andrey.makeev@fda.hhs.gov

entering the following command:

```
>>> python3 extract_cdmam_rois.py -ctr_slc 24 <path_to_dicom_files> -m
"(x_A, y_A, x_B, y_B, x_C, y_C, x_D, y_D)"
```

where `ctr_slc` is the central slice number counted from zero (for instance, if in ImageJ the central slice is #25 counted from 1, than `ctr_slc` should be set to 24), `<path_to_dicom_files>` is the disk location with DICOM images, and the integer values in the parentheses are (x, y) coordinates of the CDMAM grid four outer corners.
If CDMAM phantom placement was inadvertently altered with respect to the x-ray detector (while collecting data for a particular PMMA thickness) user can still use the manual mode, but would need to provide a new set of (A, B, C, D) coordinates for the scans affected.

(Semi-)automated mode

ROI extraction can also be done (semi-)automatically by employing small disc-like fiducial markers in the CDMAM v4.0 image, Python computer vision (CV) algorithm for localizing them, and subsequent calculations to determine ROI center $(x, y)-$coordinates. This method was tested to work well, but may require additional parameter tuning. As implemented, the method is scale-invariant, i.e. it should work with image data from any DBT system, regardless of x-ray detector dimensions, pixel size, magnification factor, etc. To perform ROI extraction in auto mode, enter the following command:

```
>>> python3 extract_cdmam_rois.py -ctr_slc 24 <path_to_dicom_files> -a
```

The script accepts the central slice number `ctr_slc` and a path to DBT (DICOM) images `<path_to_dicom_files>` as in manual mode. CDMAM circular marker localization is done by `detect_blobs.py` module (invoked from the main script) that employs the `cv2.SimpleBlobDetector` algorithm for small blob detection. Both `extract_cdmam_rois.py` and `detect_blobs.py` scripts should be placed to the folder where DBT reconstruction files are located. An important pre-processing step, called *top-hat* enhancement, is applied to the image before blob detection. Its result is stored in the intermediate file tophat.png. This image enhancement technique boosts blob-like structures of the specified size while suppresses irrelevant background, hence greatly improving blobs contrast. A single parameter of the top-hat operation is the kernel size

```
kernel= cv2.getStructuringElement(cv2.MORPH_RECT, (21, 21)),
```

that needs to match expected blob size. For example, if for a given DBT reconstruction CDMAM fiducial markers sizes are measured to be 18 pixels in diameter, a 20×20 px^2 kernel would be appropriate. `SimpleBlobDetector` itself has a number of adjustable parameters, which may require some tweaking for the algorithm to detect all or most of the fiducial markers. Listed below are `SimpleBlobDetector` parameters that may need to be adjusted for proper fiducial marker registration. Parameter values are inferred from analyzing the tophat.png image in ImageJ viewer. For example, they can be as such (not guaranteed to

Contact: andrey.makeev@fda.hhs.gov

work, but may be used as a starting point in finding appropriate settings):

```
minThreshold= 1
maxThreshold= 190
thresholdStep= 2
minArea= 50
maxArea= 90
minCircularity= 0.70
minConvexity= 0.70
```

Most of these are intuitive, however due to nonlinear nature of the thresholding algorithm in `SimpleBlobDetector` threshold-related parameters may require additional experimentation. In some instances disabling circularity and convexity criteria, or commenting out `thresholdStep` may be helpful. An important parameter that will vary for different thicknesses of added PMMA is minimal horizontal distance between blobs `min_x_spacing`. Unfortunately there is no universal set of parameter values that will work for all situations. A user may need to do some experimentation by examining blobs.png output (described below) and tweaking blob detector parameters after each trial. Examples of the above quantities and how they are measured in tophat.png using ImageJ are demonstrated in Fig. 2-4.

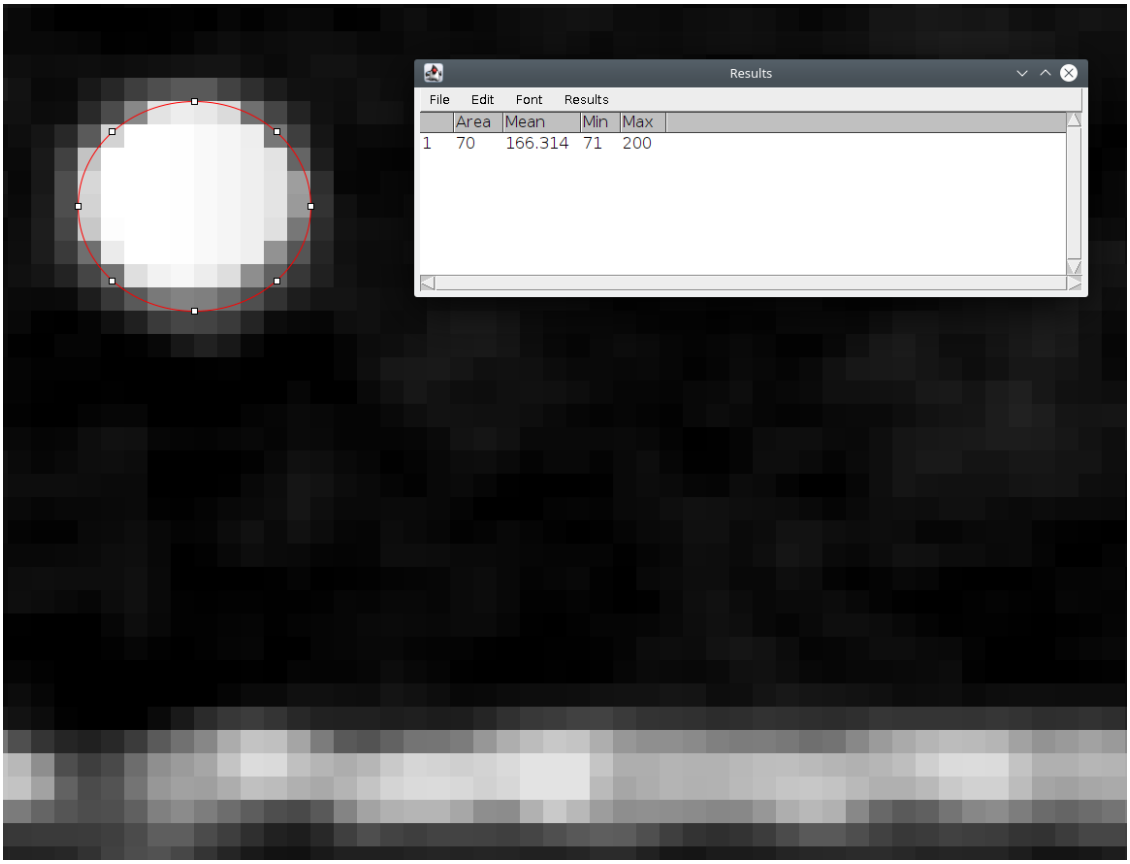


Figure 2: Area of the blob measured in ImageJ: 70 px^2.

Contact: andrey.makeev@fda.hhs.gov

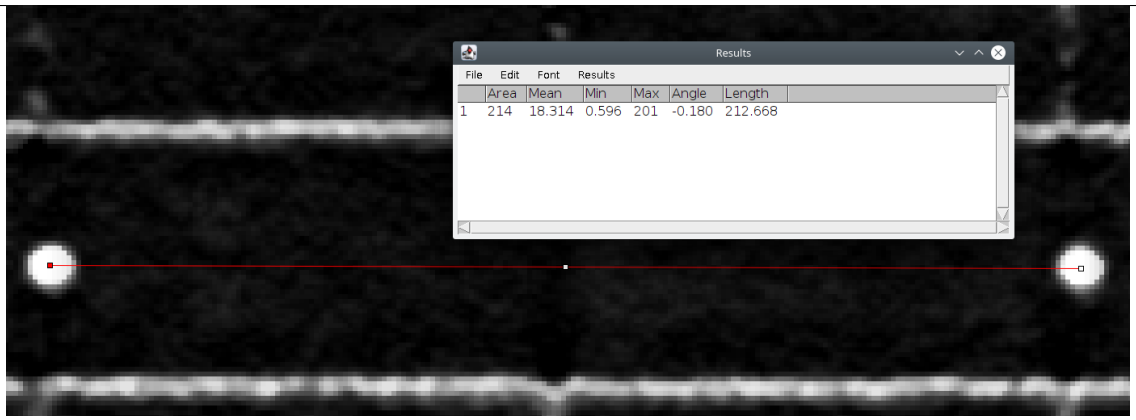


Figure 3: Distance between the blobs: 213 px.

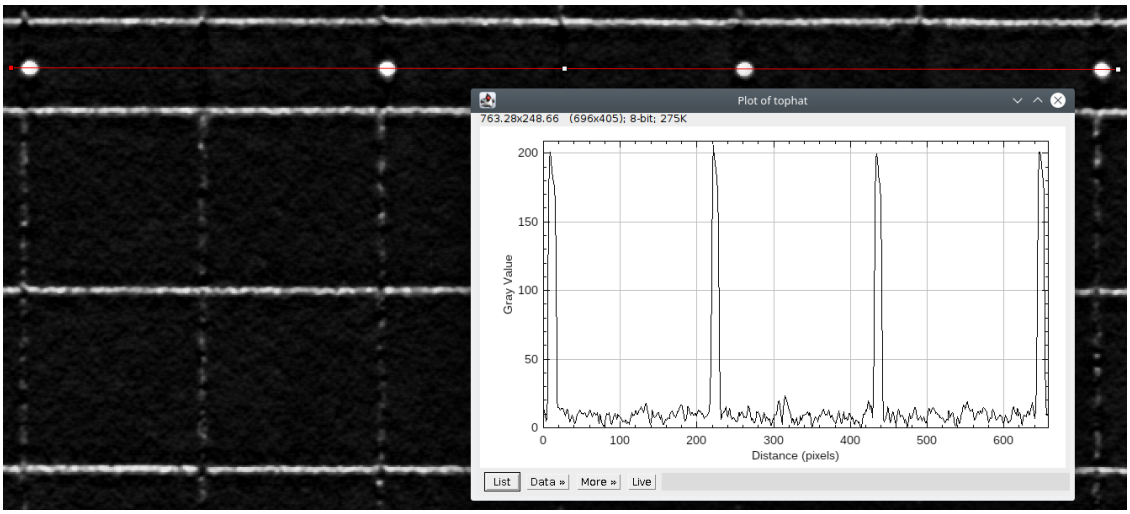


Figure 4: Profile through the blobs (`Ctrl+k` in ImageJ) for determining thresholds min/max range.

Ideally, `extract_cdmam_rois.py` should identify all **42** fiducial markers, however only *the four outermost circular markers* will be used to calculate (x, y) coordinates of the four corners (A, B, C, D) in the CDMAM cell grid as shown in Figure 5.

Note 1: because we are extracting ROIs from the four right columns and fourteen top rows of the CDMAM 4.0, it is possible to use only top 34 markers, discarding the bottom eight. Extraction script should still correctly localize ROIs centers with using 34 upper markers. As an example, when collecting data from vendors for the baseline DL model, we have encountered data with the phantom stack positioned too close to the "left" edge of the x-ray detector, which for a wider angle DBT acquisition geometry resulted in a notably lower contrast of the eight bottom markers, thus making it difficult to come up with a uniform set of parameters for `SimpleBlobDetector`. For those scans the bottom row of markers was

Contact: andrey.makeev@fda.hhs.gov

disregarded, and only upper 34 circles were used for extracting ROIs.

Note 2: in some scans one or a few "inner" fiducial markers can be challenging to register due to low contrast or masking effect of the BR3D background, requiring time-consuming blob detector parameter tweaking. Since the extraction code relies on the coordinates of the four outer markers, their registration allows the user to bypass the need for detecting missing "inner" points. Figure 5 shows the results of successful blob detection, with the image saved as blobs.png file. This visual output should be used when adjusting `SimpleBlobDetector` parameters.

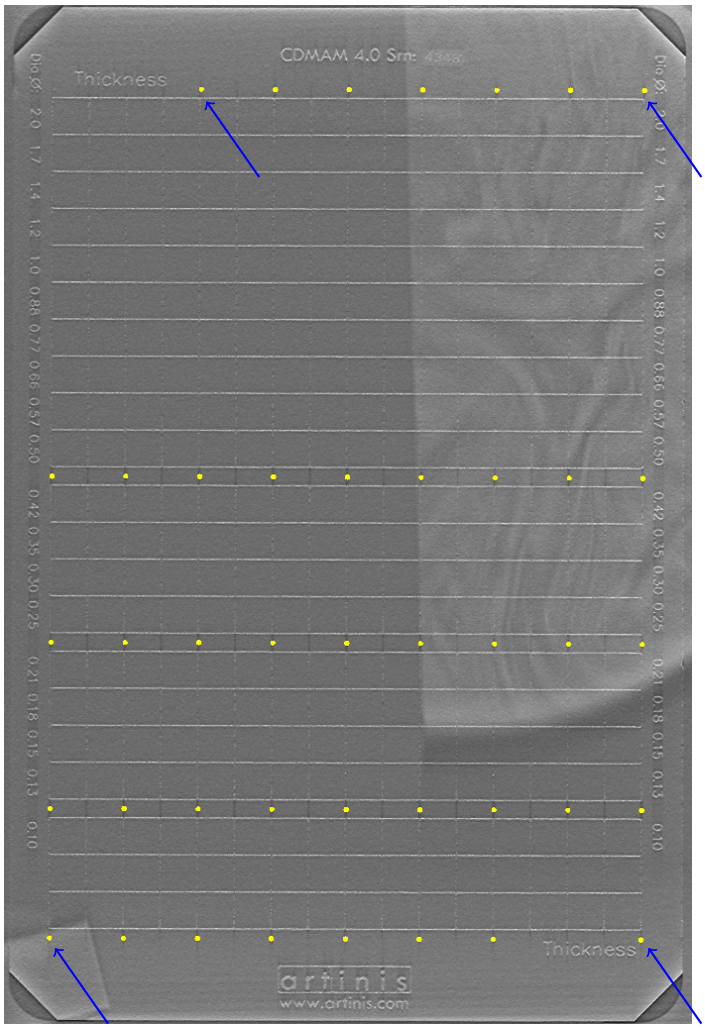


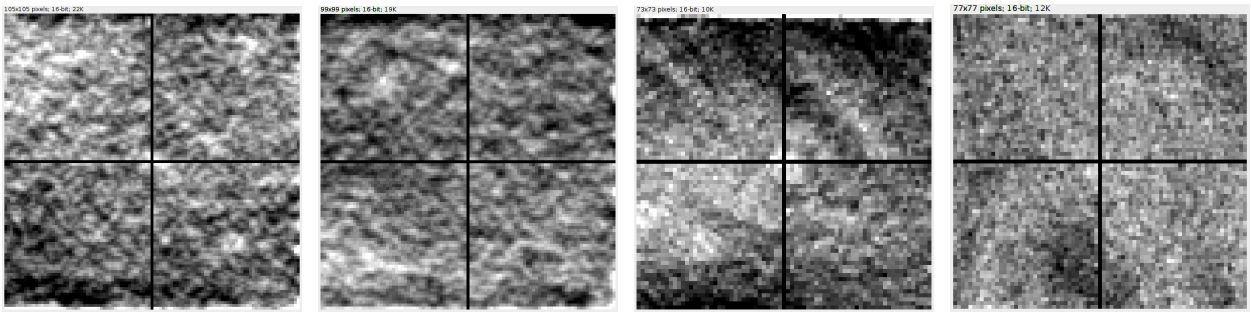
Figure 5: CDMAM 4.0 fiducial markers identified and highlighted (yellow) in the blobs.png output. Arrows indicate the four markers needed for ROI extraction. If these markers are detected properly other markers are not required. False positive markers, however, need to be purged.

Contact: andrey.makeev@fda.hhs.gov

Note 3: it should be pointed out that the top-left marker in the CDMAM 4.0 is not well-visible and is not detected, so it is normal. This has been observed with at least two different CDMAM 4.0 phantoms, so it must be by design. The software extrapolates this "missing" marker position from the other detected markers.

ROI quality inspection after extraction step

Visual inspection of the cropped ROIs is essential to ensure that the extraction program worked as expected, regardless whether manual or automated mode was used. Each input DBT volume with CDMAM phantom will produce 56 ROIs that will be used for the DL-model fine-tuning. Note, that extraction software adds two zero-value 1 px−wide lines (vertical and horizontal) to each ROI to break it into four quadrants to assist the DL model in learning the 4-AFC task. A slight tilt may also be present if the CDMAM was placed at an angle with respect to the detector pixel array. Good quality ROIs should contain eccentric detail *approximately* in the center of the quadrant and not contain the CDMAM cell-separating lines crossing through (but at the edges 2-4 pixels in), text, or fiducial markers. Examples of good ROIs are shown in Figure 6.



(a) Signal: bottom-right (b) Signal: top-left (c) Signal: bottom-left (d) Signal: top-right

Figure 6: Examples of good CDMAM 4.0 ROIs that can be used for fine-tuning. There is a disc-like detail in one of the four corners and a similar sized detail at the ROI center (approximately) at separator lines intersection.

Let's assume that we've collected 18 scans with the [CDMAM + 20 mm of PMMA + BR3D background] assembly. Running `extract_cdmam_rois.py` on this dataset will generate 1,008 PNG images named like "roi_vendor_name_20mm_zkyHVKr_0.660_0.130_q0.png". Using ImageJ's `File→ Import→ Image Sequence...` it is possible to load the image set into a stack to conveniently browse it with a slider and inspect for possible problems. If insufficient or too many (false positive) markers were detected the program logic will break and produce obviously unacceptable patches, as illustrated in Figure 7 below. False positive marker locations may sometimes occur when the blob algorithm registers symbols like dots in the "i" characters in the CDMAM, sporadic blob-like noise spikes, and so on. Additional `SimpleBlobDetector` parameter tuning, may be required to filter those out. In rare occasions there could be persistent false-positive blobs, preventing the user from obtaining

Contact: andrey.makeev@fda.hhs.gov

a quality ROI dataset. In such circumstances one can directly edit the DICOM slice to clear the offending false signal and save the new DICOM image using "Bio-Formats" plugin "Export" feature in ImageJ.

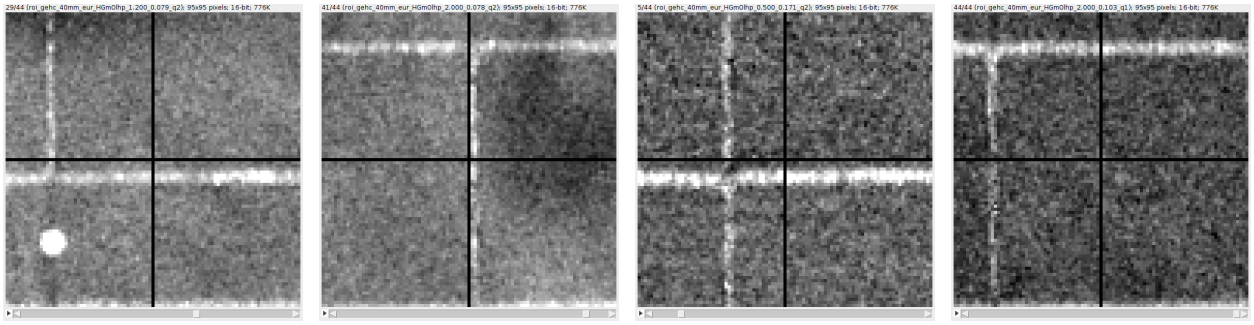
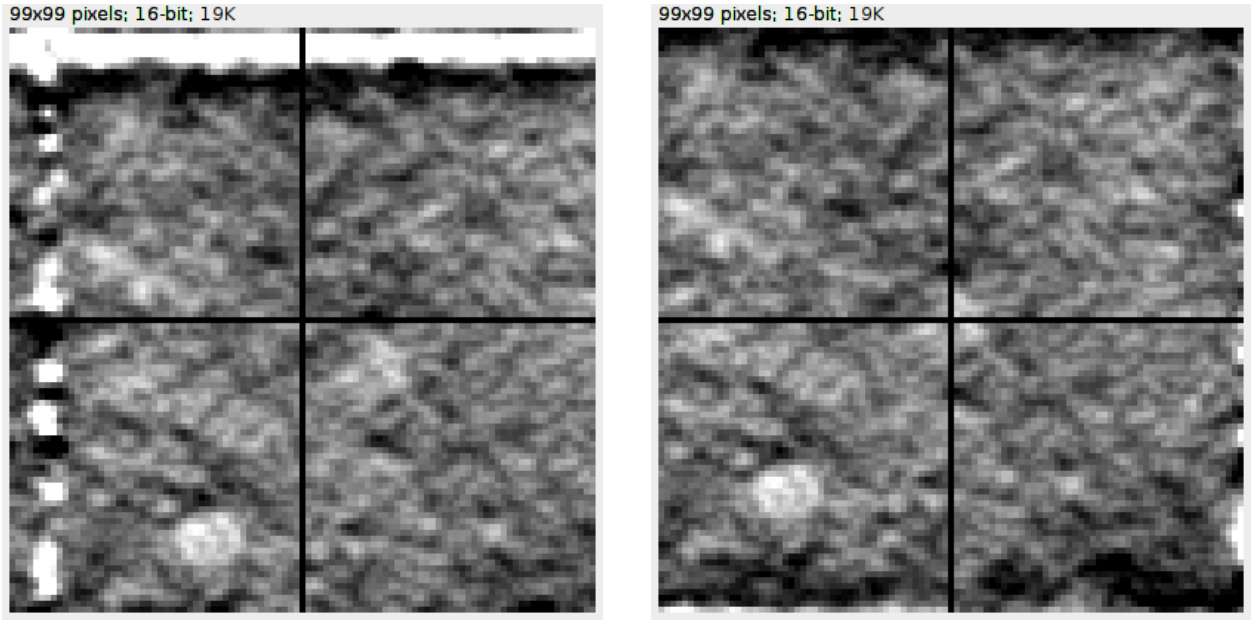


Figure 7: Examples of unacceptable ROIs that should not be used for fine-tuning. DL model would not be able to learn from such data.

The main script `extract_cdmam_rois.py` has two constants `roi_ctr_off_hor` and `roi_ctr_off_ver` that control systematic offsets of the ROI center positions in x- and y-directions which can be adjusted after visually inspecting the output. Example of using these offsets is shown in Fig. 8



(a) Cropped ROI may look like this. (b) Same ROI after applying offsets.

Figure 8: Example of using `roi_ctr_off_hor` and `roi_ctr_off_ver` parameters to adjust systematic offsets in ROI centers.

Successful execution of the program will be indicated by printing a total number of blobs

Contact: andrey.makeev@fda.hhs.gov

found in the image and the number of "detected" (i.e. good) blobs, which ideally should be 42 as shown in the Figure 9. As was mentioned previously the number of detected blobs can be less than 42, as long as missing markers are not the corner markers (highlighted in Fig 5 with blue arrows). Program will print the number of detected blobs in red color if it is less than 42 to warn the user to check the blobs.png output.

```
makeev@shrek~/work/projects/cdmam/swirl_cdmam_03/18_select_dcm_for_testing_6cm_COPY$  python3 extract_cdmam_rois.py -ctr_slc 60 $PWD -a
-------------------------------------------------------------------------------------------------
Mode: automatic
Central slice: 60
Path to DICOM files: /home/makeev/work/projects/cdmam/swirl_cdmam_03/18_select_dcm_for_testing_6cm_COPY
-------------------------------------------------------------------------------------------------
1.2.840.113681.3232246842.1711111762.4392.10575.1_73200000_000536_171328996000a4.dcm
total blobs: 45
detcd blobs: 42
1.2.840.113681.3232246842.1711111762.4392.13039.1_73200000_000664_17133965450104.dcm
total blobs: 44
detcd blobs: 42
1.2.840.113681.3232246842.1711111762.4392.11422.1_73200000_000580_171329839200c5.dcm
total blobs: 45
detcd blobs: 42
1.2.840.113681.3232246842.1711111762.4392.12885.1_73200000_000656_171339638400fd.dcm
total blobs: 42
detcd blobs: 41
1.2.840.113681.3232246842.1711111762.4392.11807.1_73200000_000600_171330409100d4.dcm
total blobs: 45
detcd blobs: 42
1.2.840.113681.3232246842.1711111762.4392.13578.1_73200000_000692_17133971190119.dcm
total blobs: 43
detcd blobs: 41
```

Figure 9: Expected `extract_cdmam_rois.py` screen output. Visual inspection of blobs.png is advised if the number of detected blobs is less than 42.

Cross-validation performance testing

The purpose of this test is to use a pre-trained, fixed deep learning model, which is part of the RST, as a starting point for fine-tuning the new observer model with images collected using the subject device. This process evaluates the device's performance through cross-validation. The fixed "baseline" model was trained by the FDA using approximately 75,000 ROIs and includes images from several major DBT manufacturers, collected with assistance from the AdvaMed association. The model represents a variety of DBT systems featuring different x-ray detector types, resolutions, reconstruction algorithms, post-processing techniques, scanning geometries, and more.

For performance testing of a new system (to be submitted to FDA for pre-market application) the vendor is asked to collect a number of scans (18 as of now) of the [CDMAM 4.0 + PMMA + BR3D background] phantom assembly for each of the three PMMA thicknesses (20 mm, 40 mm, and 50 mm). Each scan will produce 56 CDMAM ROIs with unique swirl background realizations, or 1,008 ROIs per given PMMA thickness. The program `run_cv.py` takes paths to the baseline model and to the 1,008 test ROIs as an input and performs a 10-fold cross-validation (CV) to calculate the proportion of correct responses in a 4-AFC task, or PC hereafter:

Contact: andrey.makeev@fda.hhs.gov

```
>>> python3 run_cv.py -m <baseline_model> -d <test_dataset_path>
```

In order to average effects of random splits and random mini-batches CV is repeated four times. The final mean PC value and its standard error of the mean (SEM) are calculated from 40 CV folds and reported as the device's performance metric.

From our preliminary testing we have observed that using ∼1000 ROIs (1,008 ROIs for 18 scans) typically allows the model to achieve stable performance, i.e. a state when PC score reaches saturation and is not improving with more data. To check whether this convergence has been achieved CV is repeated three more times with progressively smaller number of images used. The suggested decrement step is 200, e.g. CV with 808, 608, and 408 images are performed after the first CV pass with a full dataset. We define performance "saturation" when PC score is not increasing by more than 3% from one data point to another. If a visible positive trend of PC vs. #ROIs is still observed with existing 1,008-ROI set, it is possible that more images are needed for achieving stability. In this situation it is suggested to acquire additional 5−10 scans and demonstrate that the model performance eventually reaches its limit that does not improve with more data. When completed the `run_cv.py` script summarizes PC values for all CV folds for the four data points and writes out a report text file named like cv_results_2024-12-04_13-14-29_test.txt. Sample contents of such output is shown in Figure 10 below.

```
makeev@tyan~/ml/keras/cdmam_dlmo$ cat cv_results_2024-11-21_01-00-50.txt
CV using 1008 ROIs
0.782 0.723 0.842 0.772 0.822 0.743 0.752 0.822 0.810 0.850
0.851 0.792 0.752 0.762 0.743 0.792 0.812 0.822 0.770 0.800
0.782 0.881 0.782 0.782 0.802 0.782 0.802 0.743 0.740 0.780
0.733 0.772 0.822 0.772 0.752 0.733 0.812 0.911 0.770 0.750
CV using 808 ROIs
0.716 0.741 0.753 0.790 0.765 0.877 0.790 0.802 0.812 0.775
0.716 0.765 0.778 0.840 0.827 0.790 0.889 0.815 0.750 0.738
0.790 0.741 0.778 0.753 0.790 0.778 0.765 0.827 0.800 0.775
0.691 0.802 0.815 0.827 0.827 0.778 0.728 0.802 0.762 0.700
CV using 608 ROIs
0.754 0.770 0.754 0.885 0.836 0.836 0.836 0.803 0.767 0.733
0.787 0.754 0.820 0.672 0.787 0.770 0.852 0.754 0.750 0.800
0.770 0.836 0.820 0.836 0.820 0.869 0.770 0.820 0.667 0.783
0.787 0.770 0.754 0.787 0.770 0.803 0.820 0.754 0.817 0.783
CV using 408 ROIs
0.683 0.732 0.805 0.829 0.805 0.829 0.756 0.854 0.725 0.800
0.707 0.732 0.854 0.756 0.732 0.780 0.756 0.780 0.875 0.750
0.854 0.732 0.780 0.732 0.707 0.805 0.878 0.780 0.800 0.900
0.707 0.854 0.780 0.829 0.707 0.854 0.756 0.829 0.925 0.775
```

Figure 10: Results of running `run_cv.py` on test images. For each dataset size 10-fold cross-validation is repeated 4 times. Mean PC score and SEM for each data point are calculated and plotted using another script.

This data can be visualized using `plot_pc_vs_nroi.py` script as:

```
>>> python3 plot_pc_vs_nroi.py -f <cv_results_file.txt> -o <output_png_filename>
```

Example of the final performance report graph is shown in Figure 11. For regulatory submis-

Contact: andrey.makeev@fda.hhs.gov

sion purposes, FDA proposes demonstrating non-inferiority of the subject device compared to the predicate device by conducting analysis as described above. Specifically, performing three tests with varied PMMA thicknesses using an old system and three tests using a new system, each test producing a plot similar to the one in Figure 11.

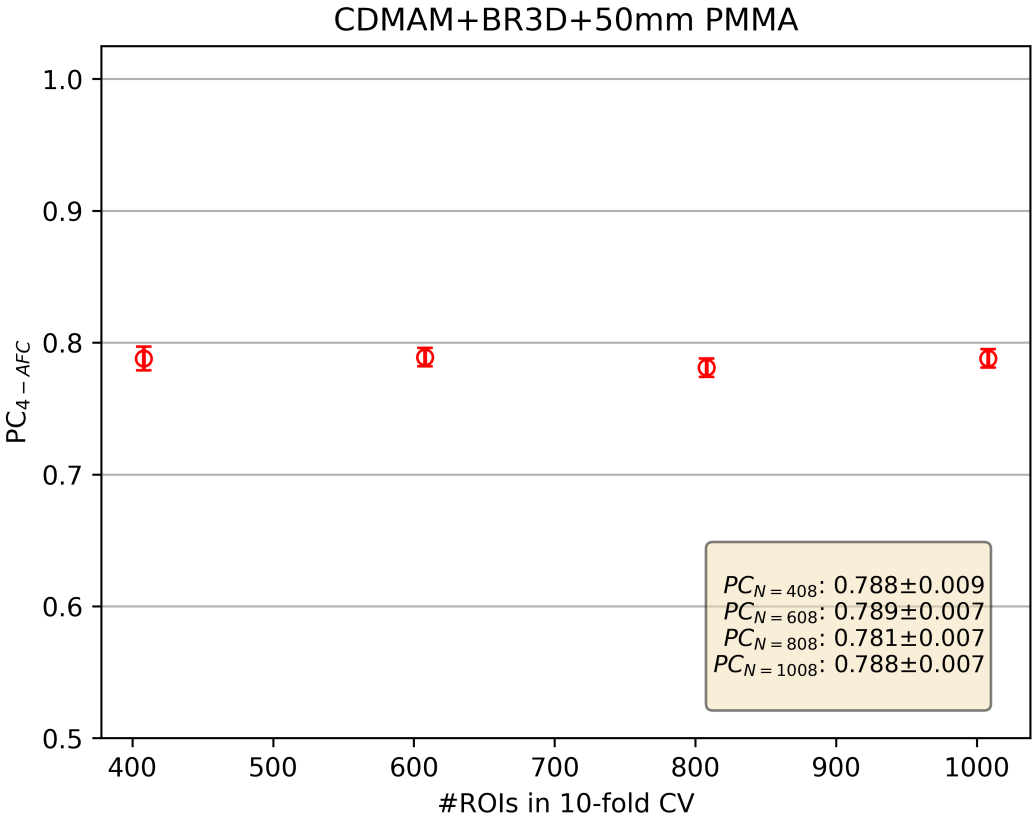


Figure 11: PC performance as a function of number of ROIs used in cross-validation. Notice how PC is not changing with increased CV dataset size.

Contact: andrey.makeev@fda.hhs.gov