ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA099

XCHEM LABORATORY PUCK SCANNER – ALGORITHM AND RESULT VISUALISATION

U. M. Neuman*, J. D. O'Hea, Diamond Light Source Ltd, Didcot, UK
K.Ward, I. Rey, Tessella, Abingdon, UK

Abstract

Macromolecular Crystallography (MX) facilities are known for using many samples and require software tools which can scan, store and help to track samples' Data Matrix codes and to maintain the correct sample processing order. An open source Data Matrix code scanning program, Puck Scanner, developed at Diamond Light Source (DLS) is introduced, its scanning algorithm explained and the continuous visualisation of results presented. Scanned codes are stored together with date, time, and the number of valid codes within a puck. This information is crucial for researchers as it allows them to match the sample with X-ray scanning results. The software is used in Diamond's XChem laboratory on a day to day basis and has started to be adopted by other facilities.

INTRODUCTION

An MX beamline experiment collects the diffraction patterns produced by an X-ray beam sent through a crystal sample [1]. Crystal samples are prepared in a laboratory, placed in a sample holder and moved to the beam. Both in-house and external laboratories may be used to prepare the samples. The scope of this paper is the DLS [2] internal sample preparation laboratory XChem. The XChem laboratory sample preparation process is described in [3]. The process is assisted by various software tools: Shifter [4], PanDDA [5], XChemExplorer [6] and the discussed Puck Scanner.

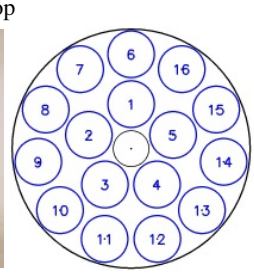
Puck Scanner is used to scan the Data Matrix [7] codes on top of each of the sample pins and on the side of the 'puck' in which the pins are stored. The decoded Data Matrix codes are stored on a disk by Puck Scanner together with the date, time and the number of successfully scanned codes before the sample puck is sent to the beamline. Scanning pucks before sending them allows researchers to track the pucks. Not all codes need to be successfully scanned for reserchers to identify a puck.

COMPONENTS

The XChem laboratory uses standard sample holders - unipucks (universal pucks) (Fig. 1) - produced by companies like Crystal Positioning Systems [8] and MiTeGen [9] with an additional Data Matrix on side. This side Data Matrix code is a crucial addition. Before this was introduced researchers experienced lots of problems related to mismatching the pins and pucks. There are 16 slots in each puck arranged in two rings. Each slot has a number assigned to it. A characteristic cut in the edge of the puck - the notch - is visible in the right bottom corner of Fig. 1a.

(a) Unipuck top

(b) Unipuck side



(c) Puck template

Figure 1: Unipuck top 1a and side1b, and puck template 1c.

Standard SPINE [9, 10] sample pins with Data Matrix codes printed on top are used by the XChem laboratory. Crystals are held on the pins using small loops attached to the tips. Fig. 2 shows a pin from two different perspectives, the side and the top with the Data Matrix codes visible.

A laboratory stand designed and 3D printed at DLS supports the puck scanning procedure (Fig. 3). It has two camera slots - one for a camera pointing at the side code and one for a camera pointing at the top of the puck. The notch ensures that the puck is positioned correctly and the side code is visible to the camera. Without the notch there would be no guarantee this side code is seen by the side camera. The cameras are connected to the laboratory PC using USB 2.0. Basic microscope pencil shaped cameras by Supereyes are used by XChem [11].

* urszula.neuman@diamond.ac.uk

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA099

Figure 2: Pin side (left) and top (right) view.

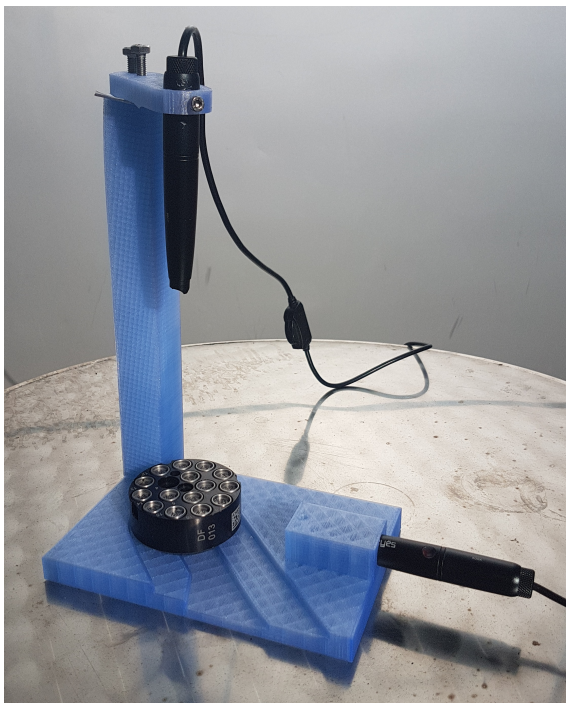


Figure 3: Laboratory stand used in the XChem laboratory, designed at DLS.

AUTOMATED CODE DETECTION

Data Matrix codes are two dimensional codes. Each Data Matrix code contains two adjacent solid borders which create an 'L' shape pattern (Fig. 4). This pattern appears to be relatively easy to find and can therefore be used to initially detect the codes in an image.

When Puck Scanner is running on a computer it attempts to detect a code in the field of view of the side camera. Once it is successfully detected and scanned the view switches to the top camera and Puck Scanner attempts to read as many codes as it can from the top image. The camera continuously feeds frames to Puck Scanner. This constant updating helps to get better results as sometimes the light conditions change between frames. The scan is complete when all of the detected codes are successfully scanned or when the scan times out. Sometimes the pins' tops get rusty or damaged. The scanner therefore tries to read the codes for a certain amount of time (currently 5 s) and stops when it times out. Samples

Figure 4: Data Matrix code.

have to be kept at a very low temperature and when not being scanned the pucks are submerged in liquid nitrogen. It is important to minimise the time they are kept at room temperature. Not all of the sample pins have to be scanned. As long as the puck code and several pins are successfully scanned the researcher will be able to track the puck.

The automated Data Matrix code detection must be able to detect and read the codes visible in the image (Code Locator) and secondly identify the orientation of the puck so that an appropriate position in the puck can be assigned to each pin code (Puck Locator). Both of the problems are resolved using image processing supported by simple optimisation methods.

Code Locator

Code Locator is a part of the scanner algorithm which consists of two steps. Firstly 'L' shapes in the image from the Data Matrix codes are detected using OpenCV image processing methods. The precise location is then fine tuned using separate optimisation steps.

The algorithm assumes that the size of the Data Matrix is specified by the user. All the image processing operations are applied on a gray scale version of the input image.

Image Processing The image processing part of Code Locator make uses of several image processing methods implemented in OpenCV. The steps involved in this algorithm are identified in Fig. 5 (top part). The algorithm constructs a set of polygons (using threshold and contour detection methods); all 'L' shaped polygons are then extracted from this set.

Optimisation Once the 'L' shapes are detected in an image a small image fragment with an area twice as big as the size of the code is cut out from the original image around the each of the 'L' shapes. The steps involved in the optimisation process are illustrated in Fig. 5 (bottom). The Data Matrix codes are squares partially filled with black colour. These small fragments consist mainly of a light silver

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA099

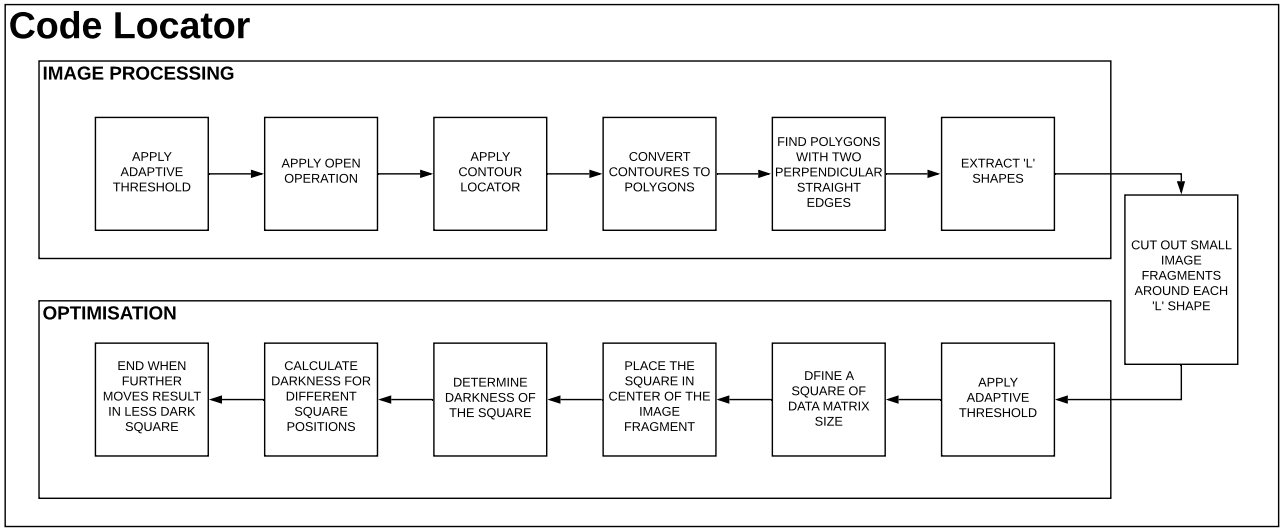


Figure 5: Code Locator - steps. Adaptive Threshold [12] and Morphological Open Operation [13] are well established image processing methods implemented in OpenCV [14].

background on which a dark code sits. The Data Matrix area is thus easily detectable.

Puck Locator

The locations of the pins on the puck are found either using image processing techniques or (if this fails) optimisation methods. Puck Locator determines the slot number for each code visible in an image of the top of a puck. The Puck Locator uses the positions of the center of the codes (as determined by the Code Locator) as input. There must be at least six slots in use for the optimisation algorithm to succeed; the preferred primary process is therefore image processing.

Image Processing The steps involved in image processing are illustrated in Fig. 6. The image processing algorithm aims to find the notch at the edge of the puck which sits between slots 11 and 12 (see the puck template in Fig. 1c). Once the notch position is established the template can be used to establish the boundaries of each slot. A slot number is assigned to a code if its boundary encloses the center of the code. This method works even if only one pin is present in the whole puck.

Optimisation Sometimes the notch cannot be found by the image processing procedure due to factors such as light condition. When this happens Puck Scanner uses an optimisation procedure instead to identify the location of the puck. The optimisation process determines the size, the position and the orientation of a puck based upon the positions of the codes identified by Code Locator. Each code is assumed to be printed on the top of a sample pin and roughly in the center of a puck slot. The puck has no rotational symmetry, so if the approximate position of the center of each slot is known, the unique orientation and position of the puck can be determined. The procedure is illustrated in Fig. 6 (bottom part).

The puck radius is calculated according to Eq. 1 where R is the puck radius and r the outer layer radius.

$$R = \frac{r}{0.79} \tag{1}$$

When the puck center and size are known the puck orientation is found by rotating the puck template in two degree increments to a number of different positions and seeing which angle best fits the codes identified by Code Locator. For each angle a total error is calculated and the angle with the smallest error is indicated as the correct position. For each code, at a given angle, the error is simply the distance from the point to the nearest template slot center. The total error for the angle is then the sum of the errors of every code.

RESULTS AND VISUALISATION

Graphical User Interface

Puck Scanner comes with a graphical user interface (GUI) which allows users to interact with the program. The GUI contains the following elements: a menu bar, scan record table, start/stop scan button, code table, image frame, message box and a progress bar (Fig. 7).

The GUI can be used to configure and test the cameras (‘Options’ on menu bar). The scan record table holds all the records kept in the store (left hand side of the GUI). Scan button allows to start or stop the scan (below the scan record table). It also helps to indicate whether the scanner is running or not - its layout and colour changes accordingly. The code table displays the list of codes from a puck (middle of the GUI). Notice the puck code is displayed above this table. Both the scan record table and the code table are colour coded. For the scan record table red means none of the codes in a puck could be read, green means all of the

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA099

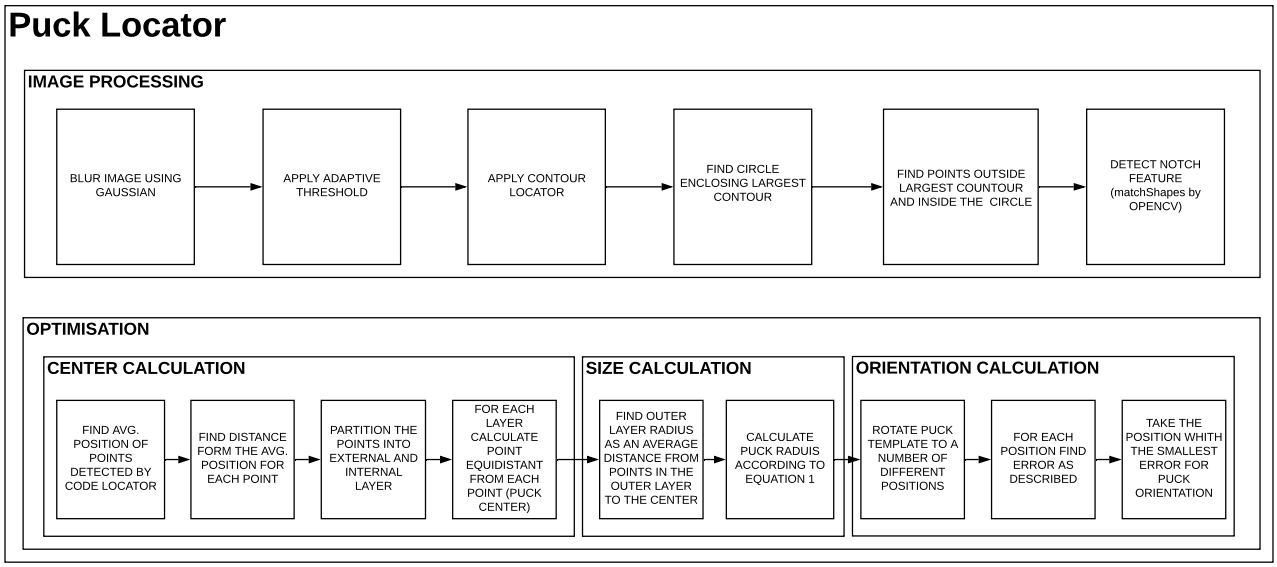


Figure 6: Puck Locator - steps. Adaptive Threshold and Open Operation are image processing methods implemented in OpenCV.

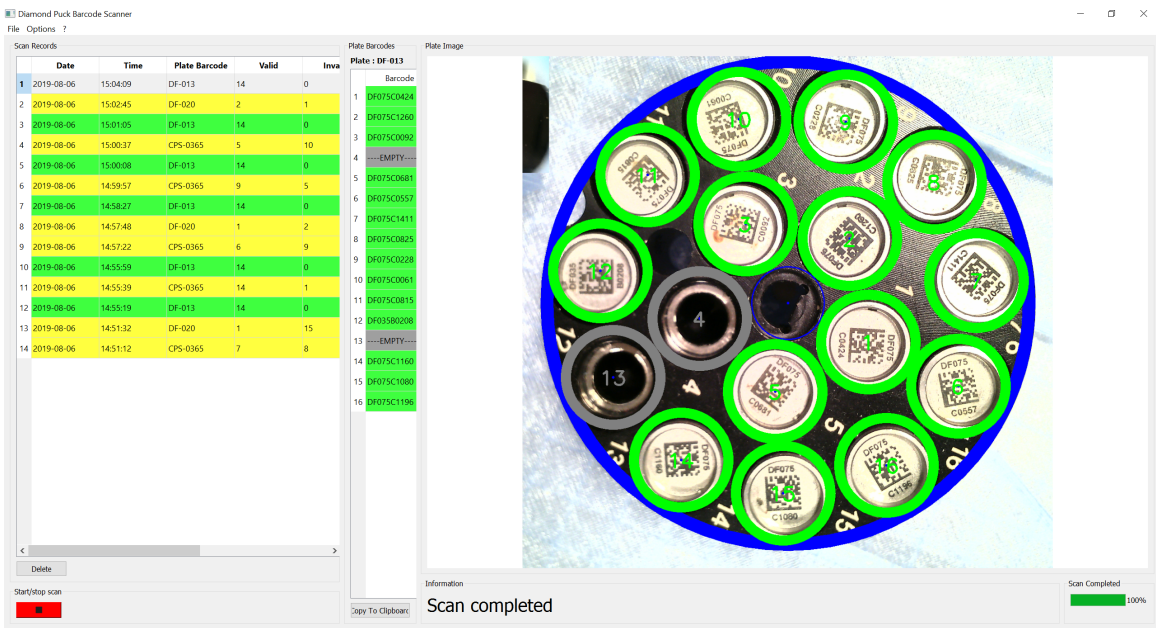


Figure 7: Graphical User Interface - scan record table visible on the left hand side, start/stop button in the bottom left corner, code table in the middle, image frame on the right hand side and progress bar in the right bottom corner.

codes visible were successfully read and yellow means that only some but not all codes were read. For the code table red means the scanner fails to read the code, green scan was successful and gray means no code detected. The image frame shows the current scanning scene (when the scanner is running) or the image from the store linked to the record chosen in the scan record table (right hand side of the GUI). The message box and progress bar are primarily used during scanning to inform the user about the status of the scanning process (under the image frame).

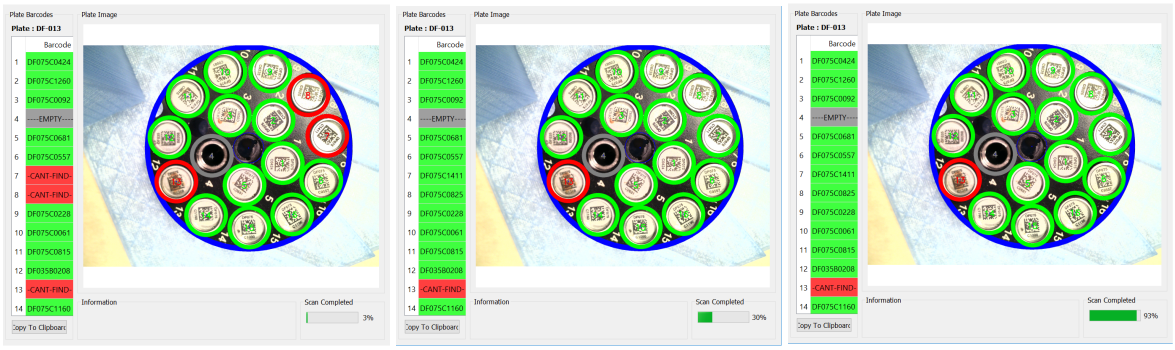
Progress Visualisation

Fig. 8 shows how the code table, the image frame and the process bar of the GUI change while a puck is being scanned. The timeout for this scan was set to 20 seconds.

20 seconds is quite a long timeout compared to 5 seconds used currently by XChem. It was set this high for illustration purposes. It can be observed that most of the codes are registered at the very beginning of the scan - within the first few seconds.

Fig. 8(a) shows the result obtained at the very beginning of the scanning process, when just 3% of the set time has

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA099



(a) Scan completed in 3% of the set time elapsed

(b) Scan completed in 30% of the set time elapsed

(c) Scan completed in 93% of the set time elapsed

Figure 8: Scanning in progress.

elapsed. It can be observed that even after such short time most of the pins are successfully scanned, only three failed. The empty slot was identified correctly. Fig. 8(b) shows the result after 30% of the elapsed time. All the readable codes are successfully read. Fig. 8(c) shows the result at the very end of the process - after 93% of the elapsed time. The result hasn't changed between Fig. 8(b) and Fig. 8(c) as the last pin is covered with rust and is very difficult to read. Although this one code is not scanned the result is still valid and usable for tracking purposes.

CONCLUSION

Thanks to Puck Scanner, researchers don't have to scan pins manually one by one in order to track their samples. Puck Scanner makes the laboratory workflow smoother and saves a lot of time (it takes only a few seconds to scan a whole puck). It is used on an everyday basis by the XChem laboratory at Diamond Light Sources. Every month around 400 pucks are scanned in the laboratory and the scanner has been constantly in use for the last two years. It is also used by two external laboratories.

Puck Scanner is an open source implementation. The stand can be 3D printed, the cameras are non-expensive so any laboratory that needs a puck scanner could easily and cheaply use this one. All the code and documentation can be accessed online [15]. It is still a work in progress and any support in the code development would be very welcomed.

ACKNOWLEDGMENTS

The authors would like to thank researches from the XChem laboratory: Alice Douangamath, Romain Talon, Frank von Delft and Anthony Aimon for their positive support and very useful feedback.

REFERENCES

[1] G. Winter and K. E. McAuley, "Automated data collection for macromolecular crystallography," *Methods*, vol. 55, no. 1, pp. 81–93, 2011, Methods in Structural Proteomics, ISSN: 1046-2023. DOI: 10.1016/j.ymeth.2011.06.010. http://www.sciencedirect.com/science/article/pii/S1046202311001265

[2] Diamond Light Source. https://www.diamond.ac.uk/

[3] P. Collins *et al.*, "Achieving a good crystal system for crystallographic x-ray fragment screening," *Preprints*, Sep. 2018. DOI: 10.20944/preprints201809.0383.v1.

[4] Shifter, Oxford Lab Technologie. http://www.oxfordlabtech.com/

[5] N. Pearce *et al.*, "A multi-crystal method for extracting obscured crystallographic states from conventionally uninterpretable electron density," *Nature Communications*, vol. 8, p. 15 123, Apr. 2017. DOI: 10.1038/ncomms15123.

[6] T. Krojer *et al.*, "The xchemexplorer graphical workflow tool for routine or large-scale protein–ligand structure determination," *Acta Crystallographica Section D Structural Biology*, vol. 73, Mar. 2017. DOI: 10.1107/S2059798316020234.

[7] ISO/IEC16022:2006. https://www.iso.org/standard/44230.html

[8] Crystal Positioning Systems. https://www.crystalpositioningsystems.com/

[9] MiTeGen plates. https://www.mitegen.com/

[10] Spine standard pins. https://hamptonresearch.com/product-CrystalCap-SPINE-HT-445.html

[11] Supereyes. https://www.supereyes-store.com/collections/microscopes

[12] J. J. Sauvola, T. Seppänen, S. Haapakoski, and M. Pietikäinen, "Adaptive document binarization," in *Proceedings of the 4th International Conference on Document Analysis and Recognition*, ser. ICDAR '97, Washington, DC, USA: IEEE Computer Society, 1997, pp. 147–152, ISBN: 0-8186-7898-4. http://dl.acm.org/citation.cfm?id=646270.685453

[13] P. Soille, *Morphological Image Analysis: Principles and Applications*, 2nd ed. Berlin, Heidelberg: Springer-Verlag, 2003, ISBN: 3540429883.

[14] OpenCV. https://opencv.org/

[15] Puck Scanner Github. https://github.com/DiamondLightSource/PuckBarcodeReader