Supplementary Material S1: Methodology Details

The first supplementary material is split into sections and provides methodological detail into the (1) VS200 calibration, (2) image analysis pipeline, (3) SEM-BSE recolouring, (4) Fourier spectral modelling, and (5) future work recommendations.

1. VS200 Calibration

The calibration of the VS200 followed standard procedures explained in the calibration manual, and once calibrated, the settings were found to be robust. Changes in the hardware, optical components, and illumination were isolated and made specific to each microscopy modality or observation method group (PPL, XPL and BF RL), mostly requiring calibration of the illumination variations (exposure, white balance, shading). Note that accidental manual changes to the camera, light paths (transmitted and reflected), or ASW software configuration by local users can have unexpected negative consequences, requiring careful logging of all uses of the instrument. For example, the 5 Mpx VS-264C colour camera (CMOS sensor) is sensitive to movement if not properly adjusted in the C-mount with a micron-level precision. Users can use 'live' mode to establish that the slides present themselves as desired for the observation method before triggering a batch scan.

The calibrating steps are simple for BF and PPL modalities. An extended calibration is expected when there are dozens of different 'observation methods' and components to configure, which is typical for multi-pol petrography. Basically, the system components are varied embedded systems containing the required optics and electronics (camera, stage, nosepiece, polarizer, etc.) that are better integrated than in binocular microscopes of previous generations. The chain of the calibrating wizard uses the VS200 calibration slide and is sequentially organized as follows:

1. Stage limits: the algorithm checks the extend and coordinates of the X and Y stage movement. The Z movement can be recalibrated depending on the desired slide thickness given that there is a constant security distance (= -200 μm) to protect the higher magnification objectives when the nosepiece turret spins.
2. Camera adapter (parfocality): controls that the C-mount produces an automatic focus distance difference lower than 20 μm between the 40X and 2X objectives. The parfocal distance is given as a measurement from the top of the objective (outside the socket) to the focusing point, measured in an pre-warmed system.
3. Koehler illumination: manual routine that ensures conoscopic illumination for >10X objectives that requires aligning the condenser lens centring and finding an adequate distancing with the bottom of the sample. It requires adjusting the field stop ring until sharpening its outline image in both transmitted and reflected light. The factory assembly is close to the perfect positioning.
4. Camera-to-stage rotation: The algorithm measures the rotation between the coplanar camera and stage coordinate systems to adjust it within [-1; 1°].
5. XY objective shift (parcentricity) and parfocality: measures the offset between the objective centring in the micron range and that they are correctly seated in their sockets within +/- 4 μm (10X, 20X, 40X) and +/- 20 μm (2X).
6. Lens correction: uses specific checker-board patterns on the calibration slide to find corner features and to calculate metrology data at various standoff distances that allow modelling the geometric optical distortion.
7. Shading correction (flat-field illumination): requires an empty field of view without dust or scratches. The estimated model typically consists of the subtraction of the dark-field pixel values (without light beam) and division by the calculated flat-field. Therefore, it is advised to slightly defocus the surface in reflected light (and label scan) and pre-focus for an empty pocket (at low magnifications) in transmitted light for running the calibration. This ensures not having artefact whitened areas (divided by zero). Note that in XPL, the model is specific to the

rotation of the motorized polarizing pair and the thickness of the sample, which can be verified opening the shading model images.

8. White balance (WB): applies multiplying factors to each image channel (RGB) to approximate the true or natural light colours observed in the histograms. This is required since each LED lamp and optic component modifies this balance, and the wizard allows running the routine during shading correction.
9. Magnification test scan: uses the concentric circles of the calibration slide (or high contrast sample) to calibrate the objective magnification (µm/px) and the residual rotation of the camera against the stage within [-0.1; 0.1°].

The WB camera calibration is specific to each observation method but in the same modality it can be generalized for the multiple polarization orientation angles. The ASW interface in 'manual control' allows clicking an 'illuminant' value (true grey pixel or ROI) in 'live mode', for which we selected grey quartz or plagioclase crystals with the correct orientation. It is important to perform this on a properly oriented crystal and is easiest achieved with a slide containing multiple quartz and/or plagioclase crystals in different orientations. The resulting channel multiplying factors are found in a single modality browsing the camera settings and copy-pasting across the settings of the entire modality under calibration until R-G-B histograms perfectly correspond. In the latest software version (ASW v3.4) WB settings are preserved across modalities. In earlier versions (e.g., up ASW v3.3), the user needed to create a 'fake' fluorescence routine, where BF RL was copied two times ('RL_ppl' and 'RL_xpl') to preserve the modality WB during multiplexed scanning. Within the 'Edit settings' sub-menu, 'RL_ppl' and 'RL_xpl' were included in the image stack by activating 'use for focusing only' option.

It was found beneficial to perform WB calibration both before and after shading correction. The XPL shading calibration requires a special rock (garnet-biotite gneiss) in a stage pocket next to the VS200 calibrating slide and under normal operating conditions. For instance, biotite in PPL turns dark brown when oriented E-W (// polarizer) and light pale when N-S (// analyser) under normal operation dynamic range. Therefore, exposure times must be pre-configured using manual exposure levels for all observation methods and our normal settings try to generalize scanning most rock types. In addition, RL shading calibration requires placing a silicon semi-conductor wafer for achieving a maximal and pristine reflection on the entire field of view at different magnifications.

2. Image Analysis Pipeline

The acquisition image files are stored in a local server and downloaded within the local intranet through the OlyVIA software. Alternatively, remote users can request temporary credentials to access the network or be provided with an external drive containing multi-gigapixel files (data trees). The projects are opened using open-source software and dedicated scripts in MatLab (Option A) and Python (Option B) that work with the full-resolution images. This avoids losing resolution of the original files or non-reliable overnight downloads.

The two main products of this section (Option A/B) and the workflow in section 3 are manually registered ray tracing image stacks and recoloured SEM-BSE (multi-Otsu thresholding). For opening the files, the whole-slide slide imaging outputs are converted into open-source pyramidal images for enabling the analysis of 'patches' (2D image stacks) sliced from precursor image pyramid 'blocks'.

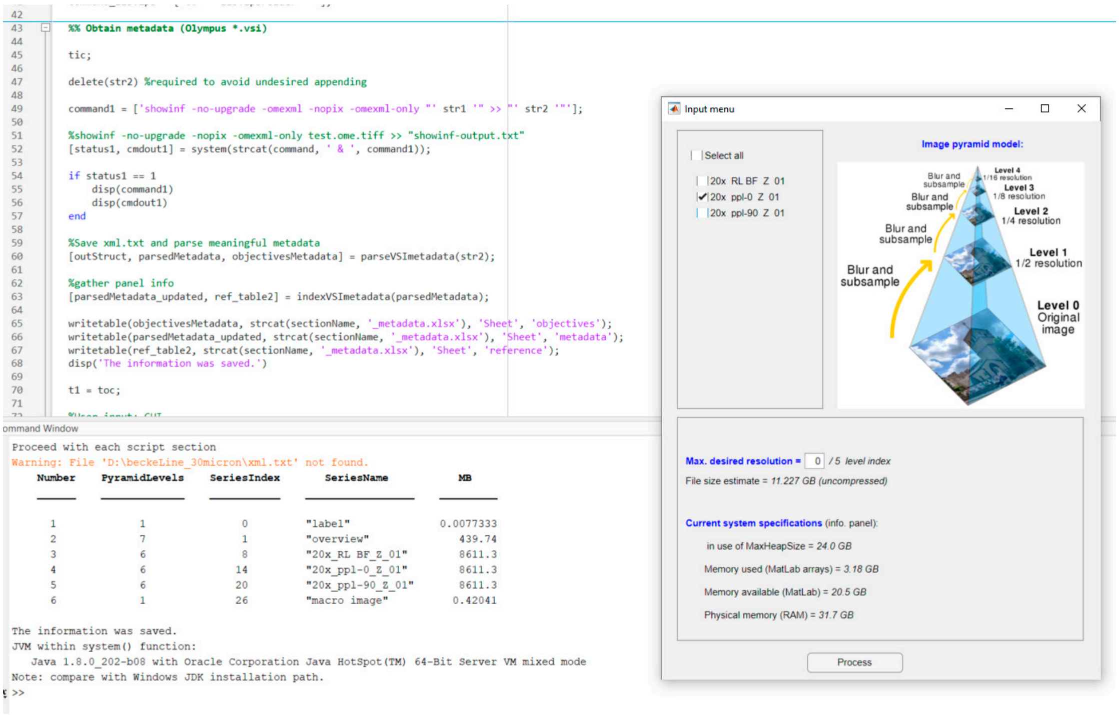
We have used an offline PC with adequate multi-threating capabilities for this task, i.e., image processing and analysis. The Alienware X15 R1 (i9 11900H @up to 4.9 GHz 8-core, 32 GB DDR4 RAM 3200 MHz) gaming laptop saves data to two M.2 PCIe NVMe SSDs that are a 1 TB system disk (Gen3 SK Hynix) and 2 TB secondary disk (Gen4 KIOXIA). An additional 1 TB SanDisk SDXC MicroSD card was used for easy dataset shipping (1 or 2 slides) such as raw images, intermediate pipeline products, and ray tracing inputs for image segmentation.

Overall, managing storage is crucial as the processes do not have progress bar indicators and the files should not be moved until they are fully written to disk. The additional programming criteria have been parallelizing the processes and enabling the high-level code to be modified for usage flexibility, where created functions (commented script sections) were not defined in separate files or thoroughly documented to ease code management. No part of the pipeline has been compiled.

2.1. Overall Workflow

The Open Microscopy Environment (OME) developed the Bio-formats Java library to translate image pyramid areas from ~140 proprietary formats (https://docs.openmicroscopy.org/bio-formats/6.10.0/supported-formats.html (accessed on 19 January 2023)) and saving them in a widely compatible open-source format known as OME-TIFF (http://ome.github.io/design/OME005/ (accessed on 19 January 2023)), keeping knowledge of the Olympus file format (*.vsi) since 2012. Translation is required to standardize collecting and reporting of 2D microscopy datasets within the client-server software (e.g.: OMERO).

This work uses raw files (*.vsi) from a script ('combineScript_parallel.m') that calls 'bftools' v6.8.1 (Bio-formats command-line interface; https://docs.openmicroscopy.org/bio-formats/) for reading and translating the contained image pyramid blocks. The initial script section parses the metadata and opens an informative graphical user interface (GUI) that allows easy image pyramid page and resolution selection (Figure S1). This also allows predicting the file sizes to be saved in view of the constraints of the physical RAM memory available. The next script section performs multi-thread exportation with parallel 'system()' calls to save flat images for further processing. Java language (JDK 64-bit Server 11.0.12) has to be pre-configured in the Windows path to allow the process to use a larger share of the RAM setting the 'MaxHeapSize' to <75% of physical memory and enable higher flexibility using Bio-formats.



```
%% Obtain metadata (Olympus *.vsi)

tic;

delete(str2) %required to avoid undesired appending

command1 = ['showinf -no-upgrade -omexml -nopix -omexml-only "' str1 '" >> "' str2 '"'];

%showinf -no-upgrade -nopix -omexml-only test.ome.tiff >> "showinf-output.txt"
[status1, cmdout1] = system(strcat(command, ' & ', command1));

if status1 == 1
    disp(command1)
    disp(cmdout1)
end

%Save xml.txt and parse meaningful metadata
[outStruct, parsedMetadata, objectivesMetadata] = parseVSImetadata(str2);

%gather panel info
[parsedMetadata_updated, ref_table2] = indexVSImetadata(parsedMetadata);

writetable(objectivesMetadata, strcat(sectionName, '_metadata.xlsx'), 'Sheet', 'objectives');
writetable(parsedMetadata_updated, strcat(sectionName, '_metadata.xlsx'), 'Sheet', 'metadata');
writetable(ref_table2, strcat(sectionName, '_metadata.xlsx'), 'Sheet', 'reference');
disp('The information was saved.')

t1 = toc;
```

ommand Window

```
Proceed with each script section
Warning: File 'D:\beckeLine_30micron\xml.txt' not found.
    Number    PyramidLevels    SeriesIndex       SeriesName          MB
    ______    _____________    ___________    _________________    _________

      1             1               0         "label"              0.0077333
      2             7               1         "overview"              439.74
      3             6               8         "20x_RL BF_Z_01"        8611.3
      4             6              14         "20x_ppl-0_Z_01"        8611.3
      5             6              20         "20x_ppl-90_Z_01"       8611.3
      6             1              26         "macro image"          0.42041

The information was saved.
JVM within system() function:
   Java 1.8.0_202-b08 with Oracle Corporation Java HotSpot(TM) 64-Bit Server VM mixed mode
Note: compare with Windows JDK installation path.
>>
```

☐ Select all

☐ 20x RL BF Z 01
☑ 20x ppl-0 Z 01
☐ 20x ppl-90 Z 01

Max. desired resolution = 0 / 5 *level index*

File size estimate = *11.227 GB (uncompressed)*

Current system specifications (info. panel):

in use of MaxHeapSize = *24.0 GB*
Memory used (MatLab arrays) = *3.18 GB*
Memory available (MatLab) = *20.5 GB*
Physical memory (RAM) = *31.7 GB*

Figure S1. Developed MatLab interface showing the script with functions to parse and index *.vsi images. The GUI shows a schematic diagram of an image pyramid and the selectable thick boxes.

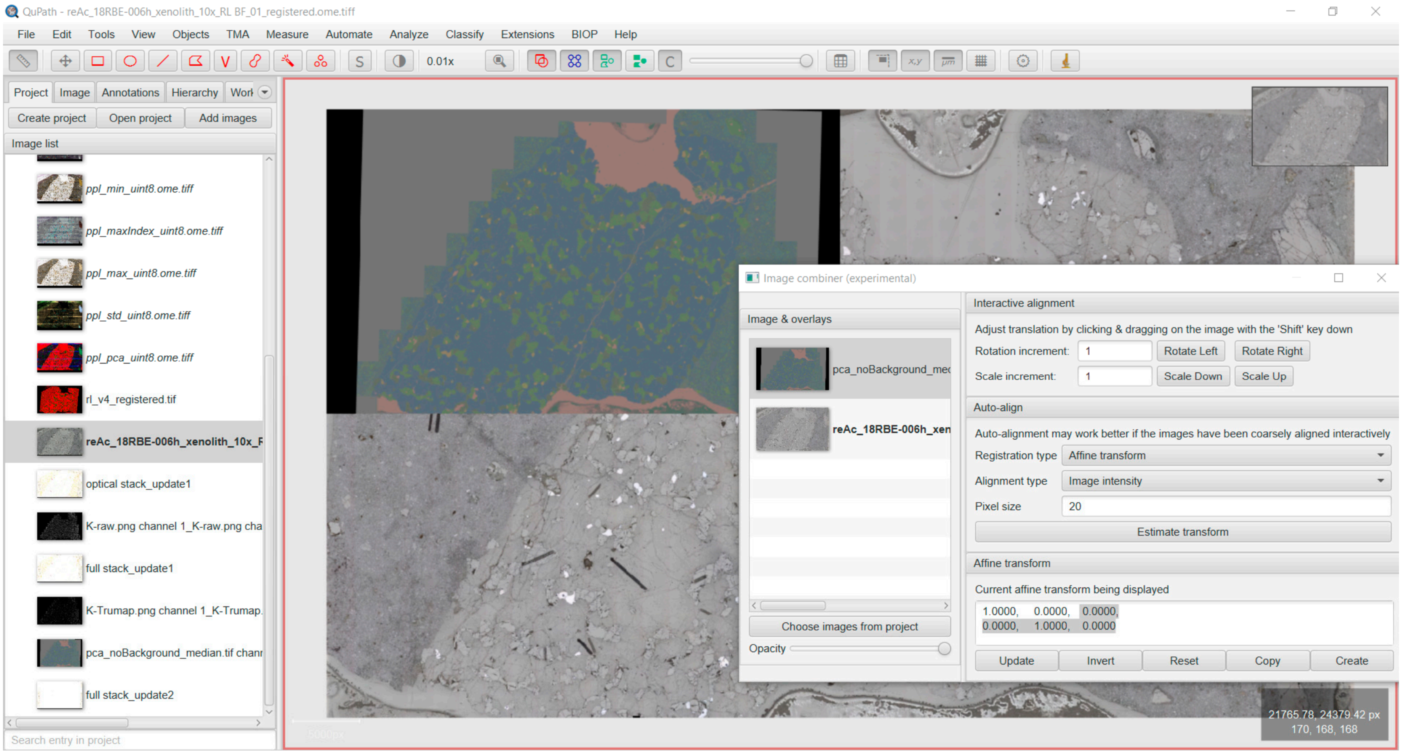
Within the MatLab R2022a installation used here the following toolboxes were available: Computer Vision 10.2, Mapping 5.3, Wavelet 6.1, Deep Learning 14.4, Image

Processing 11.5, GPU Code 2.3, Parallel Computing 7.6, and Curve Fitting 3.7. These combined with external packages covered most scripts dependencies. Notably, because MatLab presents an easier package management than Python, it is recommended for starting development projects for non-specialist groups.

2.2. Option A (MatLab)

The Option A file sizes approximately are: the original file (18 GB), translated images (~140 GB), ray tracing products in blocks (~240 GB), stitched flat (~140 GB) and pyramidal (~70 GB) versions, and QuPath projects with the segmentation (~1.5 GB), summing up to ~610 GB. Therefore, if wanting to recover all intermediate products and/or do all possible calculations, the file size readily approaches 1 TB, e.g., the capacity of a large MicroSD card for physical file transfer.

Initially, the QuPath software [1] interface allows creating a new project in an empty folder and opening *.vsi files. All the images can be loaded to a single project, which internally uses the Bio-formats Java interface for translation (allowing a few argument options). Next, 'Image aligner' (contributed by Peter Bankhead) or 'combiner' (contributed by Peter Haub) are used to calculate inter-modality image transforms (rigid and affine). The tools generate virtual image stacks that reside in the QuPath project and do not require to be saved to disk (permanent memory). They allow automatic estimation of the phase shift with a limited amount of rotation calculated in the Fourier space between a fixed and a moving image under a 'downsampled' resolution (practically ~10 × 10 pixels for 10-sec estimations). The resulting affine transform is shown on the GUI and can be copy-pasted for scripting elsewhere. In all cases, the registration works best when done in progressive steps of increasing complexity and accuracy.



Adjust translation by clicking & dragging on the image with the 'Shift' key down

Rotation increment: 1

Scale increment: 1

Auto-alignment may work better if the images have been coarsely aligned interactively

Registration type: Affine transform

Alignment type: Image intensity

Pixel size: 20

Current affine transform being displayed

```
1.0000,   0.0000,   0.0000,
0.0000,   1.0000,   0.0000
```

Figure S2. QuPath 'combiner' routine showing the pairwise comparison between the RL (fixed image) and SEM-EDX principal component analysis (moving image) of lower spatial resolution (preregistered in BigWarp to the second pyramid level, scale of 2). The 'combiner' submenu shows the options for interactive alignment and the current affine transform matrix that can be edited. Therein, the moving image only needs upscaling by a factor of 2.

The 'combiner' allows dragging transparencies of the moving images (PPL-0 and XPL-0) on top of the fixed image (e.g., RL γ-boosted). We noted that, although for most cases RL and PPL have a converging phase-correlation registration, this usually requires minor adjustments that can be picked up assessing the pixel offsets in the whole-slide, as these can be biased due to the image montage distortions (including stage drift). Next, the specific modality series (e.g.: PPL-0, -18, -36, -54, -72, -90) are automatically aligned and the transform matrices are copied to a text file, typed individually as input to the image registration section of the MatLab script ('combineScript_parallel.m'). Therein, the serial transforms are composed with the multi-modal transform matrices to obtain the pair-wise registration of all images with respect to the fixed image.

The script uses subsequent command line calls to 'vips' (CLI) that input the previously found transform matrices within the built-in 'affine' function. Note that CLI is slower than 'pyvips' (called from Python) as each operation runs as a separate process without chaining. The output is saved to disk in a destination folder (Figure S3). The script allows multi-threading 'vips' via a 'parfor' loop that reaches >95% of CPU utilisation. The entire process is controlled using regular expressions to find files belonging to a specific magnification (10X), and modality (PPL and XPL).

Next, the flat images are sliced into Deep Zoom image pyramid blocks ('dzsave' function) that are saved in a 'deep zoom' folder structure for only 1 pyramid level (full resolution) containing 4096×4096 image patches named according to "(?<x>\d*)_(?<y>\d*).tif" expression. The name sequence is in a row-major and top-down comb pattern comprising blocks with no overlap. The intermediate flat image products (*.tif) after translation and registration (suffix: *_registered.tif) can be deleted to minimise the permanent memory footprint (disk) and the recommended extension (*.v) was not used.

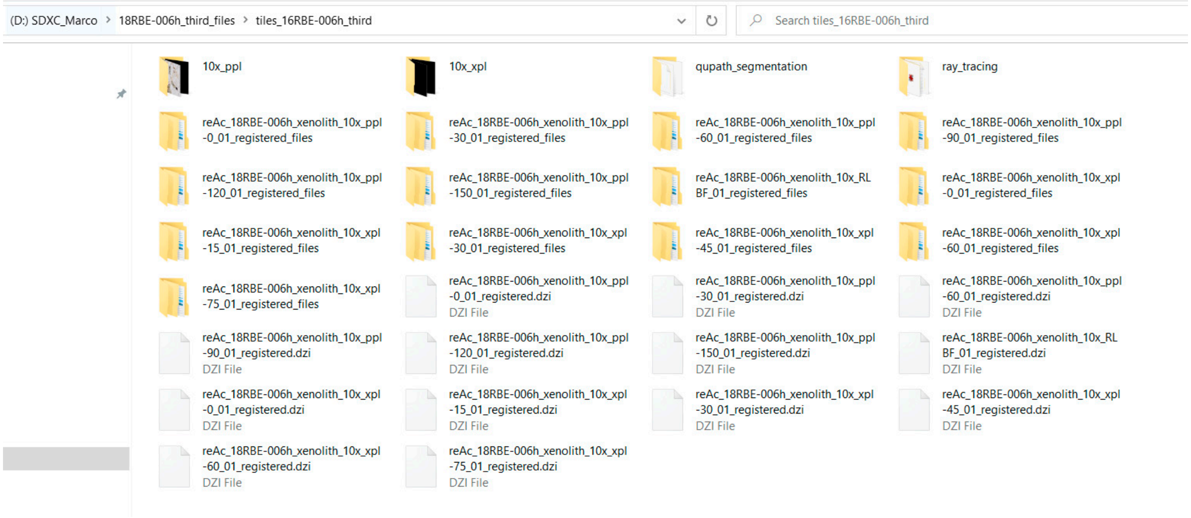
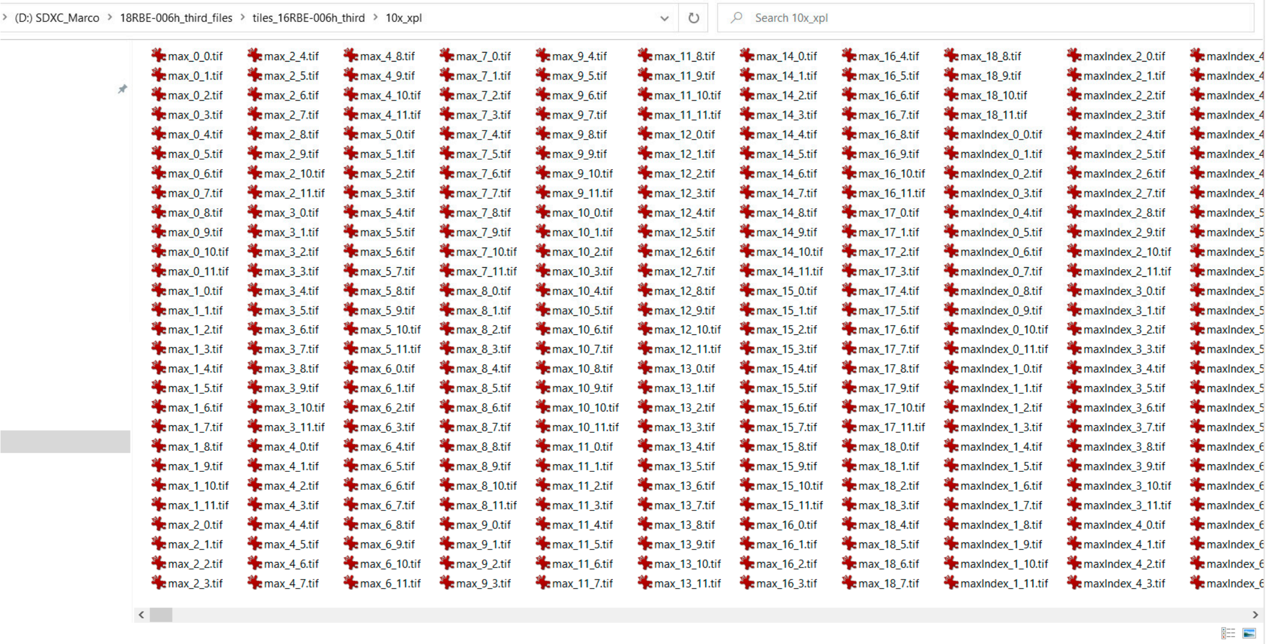

(D:) SDXC_Marco > 18RBE-006h_third_files > tiles_16RBE-006h_third
Search tiles_16RBE-006h_third
10x_ppl
10x_xpl
qupath_segmentation
ray_tracing
reAc_18RBE-006h_xenolith_10x_ppl-0_01_registered_files
reAc_18RBE-006h_xenolith_10x_ppl-30_01_registered_files
reAc_18RBE-006h_xenolith_10x_ppl-60_01_registered_files
reAc_18RBE-006h_xenolith_10x_ppl-90_01_registered_files
reAc_18RBE-006h_xenolith_10x_ppl-120_01_registered_files
reAc_18RBE-006h_xenolith_10x_ppl-150_01_registered_files
reAc_18RBE-006h_xenolith_10x_RLBF_01_registered_files
reAc_18RBE-006h_xenolith_10x_xpl-0_01_registered_files
reAc_18RBE-006h_xenolith_10x_xpl-15_01_registered_files
reAc_18RBE-006h_xenolith_10x_xpl-30_01_registered_files
reAc_18RBE-006h_xenolith_10x_xpl-45_01_registered_files
reAc_18RBE-006h_xenolith_10x_xpl-60_01_registered_files
reAc_18RBE-006h_xenolith_10x_xpl-75_01_registered_files
reAc_18RBE-006h_xenolith_10x_ppl-0_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_ppl-30_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_ppl-60_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_ppl-90_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_ppl-120_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_ppl-150_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_RLBF_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_xpl-0_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_xpl-15_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_xpl-30_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_xpl-45_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_xpl-60_01_registered.dzi DZI File
reAc_18RBE-006h_xenolith_10x_xpl-75_01_registered.dzi DZI File

Figure S3. Snapshot of folder containing all image processing pipeline files. The 'dzsave' function exports are shown from the second row down.

The image patches corresponding to the same area in the slide are parsed by the folder names within a subsequent MatLab script that begins describing the pyramid blocking in a structure variable (WxH, number of channels, number of blocks in X and Y, number of images per modality, whole-slide WxH, incremental PCA eigenvectors, etc.). The incremental principal component analysis (PCA) follows a MatLab function (contributed by Wai J.) following [2]. It consists of reducing the dimensionality of the input to 3 principal components (PC) for each channel, achieving similar results as using all the montage pixels simultaneously. This is initially done block-by-block and reading all pixels, which allows recalculating the mean and singular value decomposition (economy-size SVD) as new batches of data are presented at each update of the eigen basis. The retrieved information includes the PC eigenvectors, ranked SVD eigenvalues, mean, and number of samples seen (pixels). Then, the blocking structure and the image addresses are input to a function that performs the ray tracing and saves the calculated patches (Figure S4) in the corresponding folders (e.g., '10x_ppl', top row in Figure S3). The contents are shown in the image below for some options (max, maxIndex, min, minIndex, pca, std) following a tiling convention (e.g., 'max_0_0.tif'), and the user can freely add other useful calculations (e.g.: edge detection).



› (D:) SDXC_Marco › 18RBE-006h_third_files › tiles_16RBE-006h_third › 10x_xpl

Search 10x_xpl

max_0_0.tif
max_0_1.tif
max_0_2.tif
max_0_3.tif
max_0_4.tif
max_0_5.tif
max_0_6.tif
max_0_7.tif
max_0_8.tif
max_0_9.tif
max_0_10.tif
max_0_11.tif
max_1_0.tif
max_1_1.tif
max_1_2.tif
max_1_3.tif
max_1_4.tif
max_1_5.tif
max_1_6.tif
max_1_7.tif
max_1_8.tif
max_1_9.tif
max_1_10.tif
max_1_11.tif
max_2_0.tif
max_2_1.tif
max_2_2.tif
max_2_3.tif
max_2_4.tif
max_2_5.tif
max_2_6.tif
max_2_7.tif
max_2_8.tif
max_2_9.tif
max_2_10.tif
max_2_11.tif
max_3_0.tif
max_3_1.tif
max_3_2.tif
max_3_3.tif
max_3_4.tif
max_3_5.tif
max_3_6.tif
max_3_7.tif
max_3_8.tif
max_3_9.tif
max_3_10.tif
max_3_11.tif
max_4_0.tif
max_4_1.tif
max_4_2.tif
max_4_3.tif
max_4_4.tif
max_4_5.tif
max_4_6.tif
max_4_7.tif
max_4_8.tif
max_4_9.tif
max_4_10.tif
max_4_11.tif
max_5_0.tif
max_5_1.tif
max_5_2.tif
max_5_3.tif
max_5_4.tif
max_5_5.tif
max_5_6.tif
max_5_7.tif
max_5_8.tif
max_5_9.tif
max_5_10.tif
max_5_11.tif
max_6_0.tif
max_6_1.tif
max_6_2.tif
max_6_3.tif
max_6_4.tif
max_6_5.tif
max_6_6.tif
max_6_7.tif
max_6_8.tif
max_6_9.tif
max_6_10.tif
max_6_11.tif
max_7_0.tif
max_7_1.tif
max_7_2.tif
max_7_3.tif
max_7_4.tif
max_7_5.tif
max_7_6.tif
max_7_7.tif
max_7_8.tif
max_7_9.tif
max_7_10.tif
max_7_11.tif
max_8_0.tif
max_8_1.tif
max_8_2.tif
max_8_3.tif
max_8_4.tif
max_8_5.tif
max_8_6.tif
max_8_7.tif
max_8_8.tif
max_8_9.tif
max_8_10.tif
max_8_11.tif
max_9_0.tif
max_9_1.tif
max_9_2.tif
max_9_3.tif
max_9_4.tif
max_9_5.tif
max_9_6.tif
max_9_7.tif
max_9_8.tif
max_9_9.tif
max_9_10.tif
max_9_11.tif
max_10_0.tif
max_10_1.tif
max_10_2.tif
max_10_3.tif
max_10_4.tif
max_10_5.tif
max_10_6.tif
max_10_7.tif
max_10_8.tif
max_10_9.tif
max_10_10.tif
max_10_11.tif
max_11_0.tif
max_11_1.tif
max_11_2.tif
max_11_3.tif
max_11_4.tif
max_11_5.tif
max_11_6.tif
max_11_7.tif
max_11_8.tif
max_11_9.tif
max_11_10.tif
max_11_11.tif
max_12_0.tif
max_12_1.tif
max_12_2.tif
max_12_3.tif
max_12_4.tif
max_12_5.tif
max_12_6.tif
max_12_7.tif
max_12_8.tif
max_12_9.tif
max_12_10.tif
max_12_11.tif
max_13_0.tif
max_13_1.tif
max_13_2.tif
max_13_3.tif
max_13_4.tif
max_13_5.tif
max_13_6.tif
max_13_7.tif
max_13_8.tif
max_13_9.tif
max_13_10.tif
max_13_11.tif
max_14_0.tif
max_14_1.tif
max_14_2.tif
max_14_3.tif
max_14_4.tif
max_14_5.tif
max_14_6.tif
max_14_7.tif
max_14_8.tif
max_14_9.tif
max_14_10.tif
max_14_11.tif
max_15_0.tif
max_15_1.tif
max_15_2.tif
max_15_3.tif
max_15_4.tif
max_15_5.tif
max_15_6.tif
max_15_7.tif
max_15_8.tif
max_15_9.tif
max_15_10.tif
max_15_11.tif
max_16_0.tif
max_16_1.tif
max_16_2.tif
max_16_3.tif
max_16_4.tif
max_16_5.tif
max_16_6.tif
max_16_7.tif
max_16_8.tif
max_16_9.tif
max_16_10.tif
max_16_11.tif
max_17_0.tif
max_17_1.tif
max_17_2.tif
max_17_3.tif
max_17_4.tif
max_17_5.tif
max_17_6.tif
max_17_7.tif
max_17_8.tif
max_17_9.tif
max_17_10.tif
max_17_11.tif
max_18_0.tif
max_18_1.tif
max_18_2.tif
max_18_3.tif
max_18_4.tif
max_18_5.tif
max_18_6.tif
max_18_7.tif
max_18_8.tif
max_18_9.tif
max_18_10.tif
max_18_11.tif
maxIndex_0_0.tif
maxIndex_0_1.tif
maxIndex_0_2.tif
maxIndex_0_3.tif
maxIndex_0_4.tif
maxIndex_0_5.tif
maxIndex_0_6.tif
maxIndex_0_7.tif
maxIndex_0_8.tif
maxIndex_0_9.tif
maxIndex_0_10.tif
maxIndex_0_11.tif
maxIndex_1_0.tif
maxIndex_1_1.tif
maxIndex_1_2.tif
maxIndex_1_3.tif
maxIndex_1_4.tif
maxIndex_1_5.tif
maxIndex_1_6.tif
maxIndex_1_7.tif
maxIndex_1_8.tif
maxIndex_1_9.tif
maxIndex_1_10.tif
maxIndex_1_11.tif
maxIndex_2_0.tif
maxIndex_2_1.tif
maxIndex_2_2.tif
maxIndex_2_3.tif
maxIndex_2_4.tif
maxIndex_2_5.tif
maxIndex_2_6.tif
maxIndex_2_7.tif
maxIndex_2_8.tif
maxIndex_2_9.tif
maxIndex_2_10.tif
maxIndex_2_11.tif
maxIndex_3_0.tif
maxIndex_3_1.tif
maxIndex_3_2.tif
maxIndex_3_3.tif
maxIndex_3_4.tif
maxIndex_3_5.tif
maxIndex_3_6.tif
maxIndex_3_7.tif
maxIndex_3_8.tif
maxIndex_3_9.tif
maxIndex_3_10.tif
maxIndex_3_11.tif
maxIndex_4_0.tif
maxIndex_4_1.tif
maxIndex_4_2.tif
maxIndex_4_3.tif
maxIndex_4
maxIndex_4
maxIndex_4
maxIndex_4
maxIndex_4
maxIndex_4
maxIndex_4
maxIndex_4
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_5
maxIndex_6
maxIndex_6
maxIndex_6
maxIndex_6
maxIndex_6
maxIndex_6
maxIndex_6
maxIndex_6

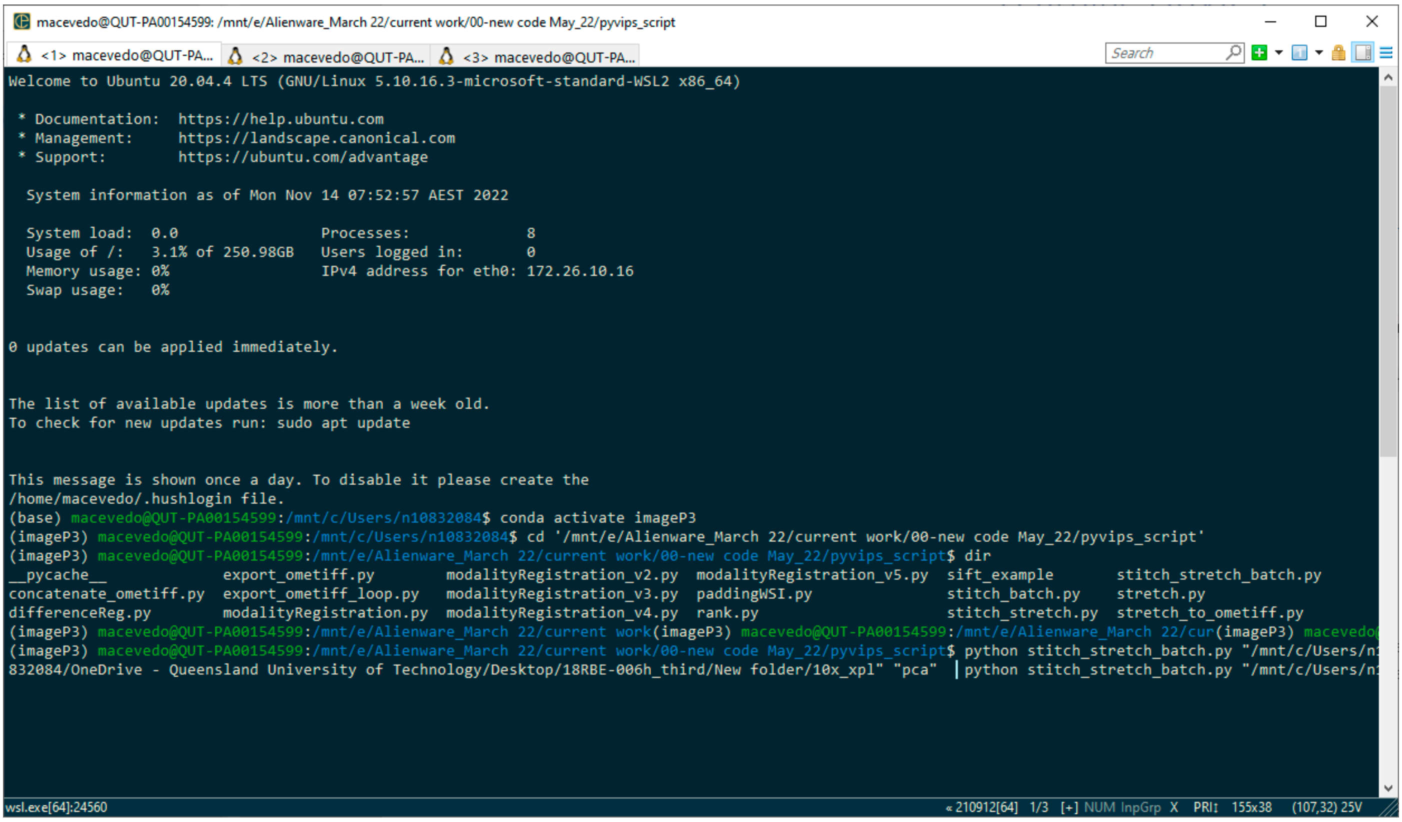
Figure S4. Snapshot of sub-folder structure containing all XPL ray tracing products exported from MatLab. There are a total of 1369 RGB image patches (4096 × 4096) that are stitched in the second part of Option B.

2.3. *Option B (Python)*

Option B is a branch of the pipeline that has a minimal permanent memory footprint, and it is preferred for future developments. The written protype code shows consistent results – comparing the transform matrices with 'combiner'. Option B was not used in this work for stack registration since the limiting factor for automating this part of the process is the intrinsic deformation within WSI (see main text) that can be corrected using tools from the Fiji big data ecosystem [3].

Regardless of image perfection, it remains a challenge to do multi-modal (RL, PPL, XPL) registration automatically and precisely given the dissimilarity between illumination intensities, altering control point feature descriptors that cannot be easily matched with algorithms such as SIFT. Despite this, laser-engraved fiduciary marks may one day to be used at the slide corners for straightforward manual alignment.

The python script series is used from Windows 10 with a command-prompt installation of Windows Subsystem for Linux (WSL2 x86) having Ubuntu 20.04.4 LTS (GNU/Linux 5.10.16). The Linux terminal has an easier syntax than Windows for programmers wanting to use image paths and function calls in their scripts. ConEmu terminal emulator was chosen to allow booting the system and activating Anaconda Distribution environments. This is installed as Miniconda (bootstrap version) that creates an environment having a Python version (3.8.5) of overall compatibility, where the following libraries are required: os, sys, glob, re, time, pyvips 2.1.8, numpy 1.22.4, and scikit-image 0.19.2. Then, ConEmu easily manages multiple tabs, where we call the scripts at convenient intervals for parallel processing for not freezing the Alienware X15 (usually, 4 tab-calls at a time are fine).


macevedo@QUT-PA00154599: /mnt/e/Alienware_March 22/current work/00-new code May_22/pyvips_script
<1> macevedo@QUT-PA...
<2> macevedo@QUT-PA...
<3> macevedo@QUT-PA...
Search
wsl.exe[64]:24560
« 210912[64] 1/3 [+] NUM InpGrp X PRI: 155x38 (107,32) 25V

```
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.10.16.3-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Nov 14 07:52:57 AEST 2022

  System load:  0.0                Processes:             8
  Usage of /:   3.1% of 250.98GB   Users logged in:       0
  Memory usage: 0%                 IPv4 address for eth0: 172.26.10.16
  Swap usage:   0%


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


This message is shown once a day. To disable it please create the
/home/macevedo/.hushlogin file.
(base) macevedo@QUT-PA00154599:/mnt/c/Users/n10832084$ conda activate imageP3
(imageP3) macevedo@QUT-PA00154599:/mnt/c/Users/n10832084$ cd '/mnt/e/Alienware_March 22/current work/00-new code May_22/pyvips_script'
(imageP3) macevedo@QUT-PA00154599:/mnt/e/Alienware_March 22/current work/00-new code May_22/pyvips_script$ dir
__pycache__            export_ometiff.py        modalityRegistration_v2.py  modalityRegistration_v5.py  sift_example       stitch_stretch_batch.py
concatenate_ometiff.py  export_ometiff_loop.py   modalityRegistration_v3.py  paddingWSI.py               stitch_batch.py    stretch.py
differenceReg.py        modalityRegistration.py  modalityRegistration_v4.py  rank.py                     stitch_stretch.py  stretch_to_ometiff.py
(imageP3) macevedo@QUT-PA00154599:/mnt/e/Alienware_March 22/current work(imageP3) macevedo@QUT-PA00154599:/mnt/e/Alienware_March 22/cur(imageP3) macevedo
(imageP3) macevedo@QUT-PA00154599:/mnt/e/Alienware_March 22/current work/00-new code May_22/pyvips_script$ python stitch_stretch_batch.py "/mnt/c/Users/n1
832084/OneDrive - Queensland University of Technology/Desktop/18RBE-006h_third/New folder/10x_xpl" "pca"  |python stitch_stretch_batch.py "/mnt/c/Users/n1
```

Figure S5. Snapshot of Linux system installed in Windows ('WSL2'). The written commands activate the coda environment, change the working directory to the folder containing the Python scripts, list the scripts (several options available), and show an example of a function call. At the top, multiple tabs can be simultaneously opened for multi-threading.

'pyvips' is the python binding for the 'libvips' image processing library (API mode). The library has a syntax that normally consists of input/output image paths, a desired function, and its optional arguments. The core [4] is written in C++. These are sequentially streamed through an algorithmic chain for operating on the source. Therefore, it can process GB- and TB-images in parallel and with low memory needs, as explained in the documentation (https://www.libvips.org/API/current/ (accessed on 19 January 2023)).

For each modality, the script successfully finds the file name patterns (magnification, PPL or XPL, and corresponding angle), reads the top-left image corner (cropped 4000 × 4000), converts to a 'numpy' array, recolours as greyscale during the estimation loop, and uses the transforms in the image registration loop. In the first script version, RL is the fixed image and two techniques are optionally used, namely feature- and intensity/frequency-based registration. Assuming >70% overlap between the fixed and moving images, the first and second techniques respectively use:

- The Scale Invariant Feature Transform (SIFT) algorithm for feature extraction and description, followed by putative control point matching, and Random Sampling Consensus (RANSAC) for robustly estimating the respective affine (Euclidean or rigid) transform between corresponding point clouds. Trusting the VS200 stage repeatability of the input, it has not been possible to simply implement feature-based registration for translation transform (X-Y freedom), as scikit-image RANSAC package minimally supports Euclidean or rigid (X-Y-rotation freedom) and affine transforms (X-Y-rotation-scale-shear freedom) that would provide unnecessary refinement.
- The Phase correlation algorithm to find the best frequency matches between the pixel intensities, filter maximum peaks, and outputs the translation transform and error estimates of the optimal image alignment.

Both processes are often found best using a double input, first a downscaled level of the image pyramid (see pyramid translation step) for transform estimation and later upscaling the initial XY-shifts to match the equivalent number of pixels of a full-resolution (level 0) input for image registration. In the second script version, the fixed image should be allowed to change from the top to the bottom of the stack to propagate the composed affine transforms from top (0°) to bottom (90°), as there is more similarity between PPL or XPL that are angularly closer, increasing the registration success rate. Currently, the final step is saving the registered blocks as an image pyramid folder with 'dzsave', see Option A.

The third leg of the python workflow – terminating either in Option A or B – uses 'pyvips' bandsplit, arrayjoin, set_type, and tiffsave functions. There are two main steps:

- In the first script ('stitch_stretch_batch.py'), the ray tracing blocks (usually 'max', 'min' and 'std') are read from the same source folder (e.g.: '10x_ppl') and montaged (with 'arrayjoin') as a single flat image file for each calculation. The RGB channels are optionally rescaled within the (1-99) percentile to avoid outliers in the full range of the unsigned integer 8-bit output. Notably, a special script ('stitch_batch.py') is used for the index images ('minIndex' and 'maxIndex') with values from 1 to 6 (six rotations) that should not be rescaled.
- The second script reads all interesting images in the same folder and 'bandsplit' their 3 channels that are added to a python list for vertical concatenation using 'arrayjoin'. Then, a copy of the virtual image is described with custom metadata using 'set_type' as an 'uint8' OME-TIFF file. Finally, 'tiffsave' saves the new files as *.ome.tiff with a block size of 512 × 512 and applies LZW compression (Lempel–Ziv–Welch algorithm) to original pixels. This differs from the recommended 256 × 256 blocks which were found to yield too many files. Normally, some trial and error is required to find the most performant pipeline block size and ensure that LZW is efficiently decoded in the destination software. Note that uncompressed TIFF format has the greatest support across other software that client the images.

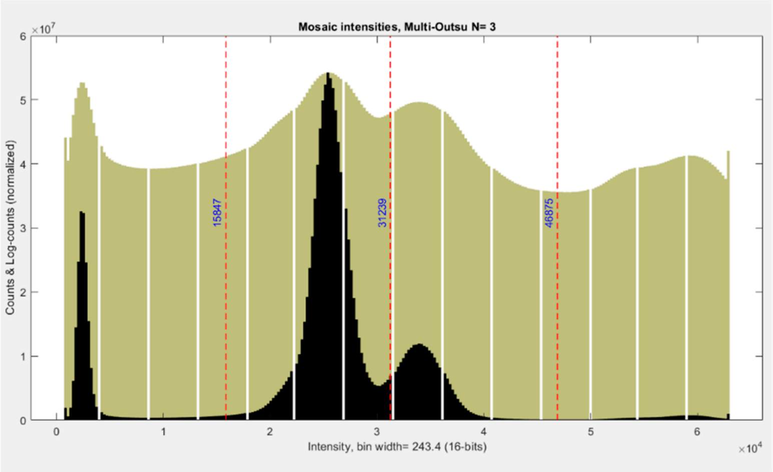
3. SEM-BSE Greyscale Recoloring by Multi-Otsu Thresholding

This RGB recolouring routine is applied to grey scale scanning electron microscopy (SEM) backscattered electron (BSE) images to facilitate better visualisation of mineral densities. The mineral density contrasts are visualised via red (lowest), green (medium) and blue (highest). The algorithm chain recalculates a 16-bit greyscale image as 3-channel 8-bit coloured image, where each channel was generated using the suggestion of the Multi-Otsu thresholding [5] to split the histogram in bands between a minimum and maximum range and with unequal interval bandwidths. Rolling up a SEM-BSE image in this fashion achieves multiple outcomes: (1) avoids pixel oversaturation of high-density phases when re-saving; (2) enhances faint density differences in more specific mineral groups; (3) highlights accessory phases crystal zonation; and (4) exposes stitching artifacts hidden with blending algorithms.

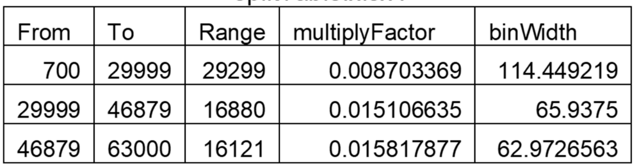
Initially, a script ('renamingSequence_v3.m') parses the file names and reads all the image tiles saved by the SEM-BSE control software and calculates descriptive statistics that represent the whole slide. This is done generalizing the values of all histograms in a column-wise summary and saving it as a structure ('mosaicInfo.mat'). Then, the montage histogram bottom and top thresholds are defined (manually or automatically) to focus on the inliers, and a bin width is calculated to perform general histogram counting on all the image tiles. The base-10 logarithm of the counts is useful to accentuate the different populations (valleys and crests) of mineral densities, specially at the right tail of the new montage histogram.

The function ('multiTH_extract.m'), modified from built-in 'multithresh()', takes the log-counts and number of desired thresholds (2 to 5) to return multi-class thresholds that minimise the intra-class intensity variance. This is done iteratively calculating the class probabilities and means through all possible thresholds from 0 to 1 (all population) and finding the maximum of the inter-class variance curve. Note that having larger intra-class variance than inter-class variance, dissimilar humps that are not normal distributions, or heavy noise can be detrimental to the global thresholding method. A more adaptative algorithm would enable full automation. The workaround has been using 5 thresholds and picking only 2 of them manually. These values are saved in an informative table ('splitTable.xlsx') before saving all tiles as RGB images. For instance, xenolith RBE-006h BSE (1 µm/px with 32 µsec dwell time) had 1302 tiles (1024 × 1024, 3 MB) with 20% overlap acquired on the SEM.

MatLab interface:



'splitTable.xlsx':



From	To	Range	multiplyFactor	binWidth
700	29999	29299	0.008703369	114.449219
29999	46879	16880	0.015106635	65.9375
46879	63000	16121	0.015817877	62.9726563

Figure S6. Multi-Otsu recolouring SEM-BSE greyscale as RGB. Top: Selecting the suggested thresholds based on log-counts. Bottom: Red, green, and blue channel information and bandwidths.

4. Fourier Spectral Modeling

Image patch stacks were extracted from certain scan projects for trial fitting of the spectra to a mathematical model. The benefit of having functions is that multi-paged scans (potentially ~14 to 360 rotations) can be reduced to a much smaller number of model parameters. A Fourier model with a 2-term equation was selected to better fit to the observation points (traced pixels), although a few areas require up to 3 terms for better correspondence.

$$f(x) = a_0 + a_1 \times \cos(x \times w) + b_1 \times \sin(x \times w) + a_2 \times \cos(2x \times w) + b_2 \times \sin(2x \times w), \quad (1)$$

where 'a' and 'b' are the coefficients of each term and 'w' (equal to $\frac{2\times\pi}{angle^{\circ}}$) is the period (fixed boundary condition).

Multi-pol scans of adequate step sizes are required to ensure convergence of the Fourier series modelling with non-linear least square fitting (MatLab Curve Fitter app). Calculating the model parameters preferably requires linearising the sRGB that are gamma-corrected for LCD monitors. Then, it is convenient to transform f(x) into polar coordinates for an easier estimation of the curve offset. This is done using the image luminosity (V) in the HSV colour space to reduce the computations and is easier to visualise, shifting PPL-max to zero by an offset (right of Figure S7), for better spectral comparison of different crystal orientations.

Overall, the model behaves well against camera noise (+/- 5 live intensity variation) since we experimented averaging pixels within a hovering region of interest with a MatLab callback function (script not provided). The lower detection limit of the sensor is noticeable in a few mineral curves such as in crystals oriented along the indicatrix circular section, i.e.: optic axis perpendicular to the sample plane, which have unusual darker colouring (e.g.: 'biotite 2' in basal orientation).

Further experiments were carried out for only 1 image tile PPL dataset of a half-rotation of the stage (180°) with 1° increments. These suggested that there was unnecessary redundancy in using 18° increments in co-angular PPL and XPL (e.g.: XPL-0 and XPL-90). Therefore, to make the most of 6 angle multi-pol image stacks, we have moved to performing the 6 PPL acquisitions in increments of 30° in PPL, while acquiring XPL in 15° in XPL. This setup captures the full periods better, particularly in PPL. Other improvements such as a freely rotatable motorized 'analyser' would enable 1° angular increments in XPL from which spectral libraries could be built (see Figure S6 for PPL). These could be compared with Electron Backscattered Diffraction (SEM-EBSD) maps.

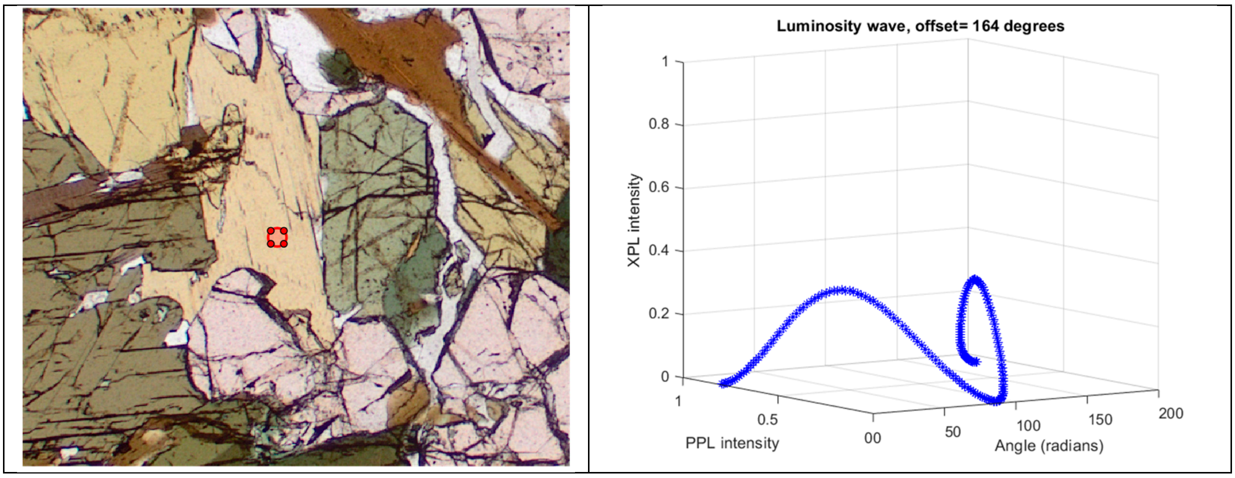


Figure S7. Biotite crystal (not <001>) in KB-67. Left shows a FoV with a region of interest area highlighted in red. Right: the interrogated spectra in 3D showing the relationship between PPL (Y-axis) and XPL (Z-axis) along the stage relative orientation (X-axis). Both estimations were done separately using f(x) and show an 'arc' in the YZ plane projection.

Given that every mineral has a specific and expanding (at orientations away from optic axis) 3D sinusoidal pattern, the LED visible light beam interactions with the light path could be simulated for source-camera ray tracing analysis (such as in videogames that vastly use Fresnel equations). Jones calculus was previously adapted by Gunter [6] to calculate the reflective rotation of incident light on dielectric and metallic minerals during internal and external reflection. Here, we could use it to simulate transmitted light elliptical polarization and interference (E- and O-waves) after crossing the second propagation interface (air and epoxy substrates). The colours perceived by the camera digital process can be artificially generated within our preliminary Michel-Levy chart MatLab script, following [7] GitHub page. It might then be possible using these colours to back-engineer the crystal orientation, e.g., [8]. A single monochromatic (16-bit) camera might be helpful for steadier acquisition conditions and to avoid the need to rescale between different modalities mimicking spectrophotometry targeting refractive index estimations.

Similar extinction curves have been provided by [9] and we have explored the technique proposed by [10] to produce RGB crystal orientation maps but the computational cost

(~75 px/sec) require migrating our implementation to a supercomputer and it excluded from this work.

5. Future Work Recommendations

To improve the user experience with the VS200, we require both more adequate computing power and local network connectivity. The existing upgrades such as using a 'hotel' loader for unsupervised tray exchange or installing the RL PPL/XPL modality in the light path remain as future testing directions. Certainly, in our laboratory, the VS200 scans at such a high rate that it is no longer cost-effective to scan one single tray (6 slides) at a time and keep the users waiting for two hours without interaction. Note that VS200 can also be controlled remotely via TeamViewer software connection with the online PC, therefore, leaving the tray exchange task to a laboratory collaborator. Therefore, future agencies/universities willing to use VS200 require the multi tray loader (MTL) 'hotel' that uses the SCARA robotic arm for loading and batch scanning up to 35 trays (up to 210 slides).

Regarding computing power, general usage of microscopy software requires making 'builds' (hardware assemblages) for maximising the use of single-thread frequency (e.g.: 5 GHz) and the SSD transfer speed. Our experience is that the RAM memory is of subordinate importance in the presence of other data acquisition and post-processing bottlenecks. RAM is required when multi-tasking or using unoptimized software routines that require loading a full image or spectra before saving into the disk drives. Therefore, a few rules of thumb can be followed; (1) purchasing workstations with the primary and secondary storages using SSD Gen4 drives with NVMe open standard, and (2) dedicating one PC for one microscope dataset to avoid filling the disks with multiple software and logon users.

We note that the controlling PC (HP Z2 workstation) has not permitted using the full potential of VS200. The main issue is the data transfer rate of the main saving drive (secondary HDD). Therefore, we further require swapping the main drive to a new SSD drive for performance gains pertaining to the utilization of the GUI during typical simultaneous acquisition and stitching at high magnifications (>20X). Currently, the 10X z-stacking (for virtual rotation) produces RAM memory overloads (at > 5 extracted images) using the 'Combine frames' tool. Such overloading causes the 'Task scheduler' to rarely show correct completion time estimates. An easier workaround is using 'Convert virtual slide images' for batch converting a group of *.vsi projects into individual file collections (also *.vsi) at the desired saving target resolution, and then, individually opening them for frame combining.

With respect to RL, future upgrades will be required to equip the VS200 with either plane polarized RL (at various angles) and/or with XPL. Our initial experiments with processing of images from a traditional petrographic microscope showed the benefits from acquiring multi-pol TL and RL PPL (e.g., evidencing ilmenite twinning in 18-RBE-006h) but limited utility from including RL XPL. In anisotropic dielectric minerals (silicates), RL PPL shows bireflectance (boosted with oil immersion) variations that are akin to pleochroism (mineral absorption tones), whereas RL XPL shows anisotropy colours from two interfering beam internal/external reflections (top and bottom of slide) akin to interference colours. This is a result of the poor reflectivity of these minerals. Given that we require boosting the exposure time about 100 times for visualizing RL XPL colours, the resulting image is as 'volumetrically' informative TL XPL. Therefore, RL XPL is indicative mainly for metallic minerals that provide a single shallow internal reflection.

The image analysis pipeline can also be further improved. The scripts of the semi-automated pipeline have shown that adequate computing power is required for running data-hungry (image annotations) machine learning models. For challenging minerals, the annotations are simpler to do when working with a traditional microscope as the expert has more degrees of freedom in adapting the light path than in a slide scanner and use MatLab to interrogate pixel spectra with sinusoidal variations invisible to human perception and its colouring perceptual bias (see section 4). Therefore, the work does not discard but rather encourages conventional petrography on critical minerals for creating accurate training sets.

Currently, the research group has a large collection of *.groovy scripts connected to the QuPath graphical user interface with IntelliJ v2021.3.3 IDE. They were adapted to our day-to-day petrographic problems and used for multiple tasks that largely exceed the default QuPath capabilities. They include concatenating channels, copying annotations across projects lists of images within the same project, and z-stack levels, exporting pyramid blocks of the virtual environment image and annotations, and experimenting with 2D 'live' filtering operations that highlight textural features. A larger diversity of computer vision tools benefiting from the image block caching system can already be found in QuPath v0.4.

The next experimentation with QuPath could trial the readily available SLIC algorithm (i.e., geographically informed k-means) for semantic segmentation or pre-segmenting the image into super-pixel objects for later classification. Usually, object classification can use annotations' measured polygon properties (intensity features, geometry, Haralick features). Exporting the objects (contours, label numbers, masks, labelled tiles, etc.) into programming languages is straightforward through the 'script interpreter' option. Currently, we are also testing the QuPath connection with object-based segmentation (OBIAS) software used in remote-sensing geospatial research applications, e.g., Orfeo Toolbox, eCognition. Note, that although these tools can process hyperspectral (multi-channel) images, they are not designed to analyse virtual 'z-series' datasets, generate pre-segmentation of planar crystal faces, or scale up to multi-gigapixel scans. In a thin section, ideal OBIAS (i.e., object-based segmentation with perfectly adjacent non-repeated edges) is feasible to generalize for a specific mineral and selected 'scale' (grain-size) parameter.

Overall, the proposed upgrades in image acquisition, computing, processing, annotation, and algorithm design should cope with the challenges related to upscaling more robust prototypes of image analysis pipelines. Although, pixel-based classification has been straightforward for fine-grained textures (down to the diffraction limit of the objectives) and provide acceptable generalization against extreme grain size differences, it involves nontrivial image registration (and stitching) challenges (see xenolith 18-RBE-006h in main text). Follow up work will explore how to minimize misalignment and produce meaningful pre-segmentation maps partnering correlative microscopy analytical work.

References

1. Bankhead, P.; Loughrey, M. B.; Fernandez, J. A.; Dombrowski, Y.; McArt, D. G.; Dunne, P. D.; McQuaid, S.; Gray, R. T.; Murray, L. J.; Coleman, H. G.; et al. QuPath: Open source software for digital pathology image analysis. *Sci Rep* **2017**, *7* (1), 16878. DOI: 10.1038/s41598-017-17204-5.
2. Ross, D. A.; Lim, J.; Lin, R.-S.; Yang, M.-H. Incremental Learning for Robust Visual Tracking. *International Journal of Computer Vision* **2008**, *77* (1), 125-141. DOI: 10.1007/s11263-007-0075-7.
3. Schroeder, A. B.; Dobson, E. T. A.; Rueden, C. T.; Tomancak, P.; Jug, F.; Eliceiri, K. W. The ImageJ ecosystem: Open-source software for image visualization, processing, and analysis. *Protein Science* **2021**, *30* (1), 234-249. DOI: https://doi.org/10.1002/pro.3993 (acccessed 2022/11/06).
4. Cupitt, J.; Martinez, K. VIPS: An imaging processing system for large images. *Proceedings of SPIE - The International Society for Optical Engineering* **1996**, *1663*. DOI: 10.1117/12.233043.
5. Otsu, N. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics* **1979**, *9* (1), 62-66. DOI: 10.1109/TSMC.1979.4310076.
6. Gunter, M. E. Refractometry by Total Reflection. Virginia Polytechnic Institute and State University, 1987.
7. Sørensen, B. E. A revised Michel-Lévy interference colour chart based on first-principles calculations. *European Journal of Mineralogy* **2013**, *25* (1), 5-10. DOI: 10.1127/0935-1221/2013/0025-2252.
8. Heilbronner, R. P.; Pauli, C. Integrated spatial and orientation analysis of quartz c-axes by computer-aided microscopy. *Journal of Structural Geology* **1993**, *15* (3), 369-382. DOI: https://doi.org/10.1016/0191-8141(93)90133-U.
9. Zhang, Y.; Zhong, H.-R.; Zhang, X.; Gao, S.-C.; Zhang, D. Orthogonal microscopy image acquisition analysis technique for rock sections in polarizer angle domain. *Journal of Structural Geology* **2020**, *140*, 104174. DOI: https://doi.org/10.1016/j.jsg.2020.104174.
10. Axer, M.; Graessel, D.; Kleiner, M.; Dammers, J.; Dickscheid, T.; Reckfort, J.; Huetz, T.; Eiben, B.; Pietrzyk, U.; Zilles, K.; et al. High-Resolution Fiber Tract Reconstruction in the Human Brain by Means of Three-Dimensional Polarized Light Imaging. *Frontiers in Neuroinformatics* **2011**, *5*, Original Research. DOI: 10.3389/fninf.2011.00034.