Utilisation of Enhanced Thresholding for Non-Opaque Mineral Segmentation in Optical Image Analysis

Andrei Poliakov and Eugene Donskoi

Supplementary material: algorithm details

To further assist the reader, the overall algorithm for enhanced threshold-based identification of non-opaque minerals has been subdivided into 3 separate steps: sub-algorithms numbered according to their order within the overall workflow shown in Figure S1. To improve algorithm readability, comments are typically provided only for the steps not discussed in substantial detail in the main paper; please refer to the paper itself for more explanations.

The parameters required by sub-algorithms can be adjusted by the operator when working with the software in interactive manual mode as described in [21] in the main paper reference list. Once satisfied, they can be recorded in an analysis profile (for the concept of "profile" see [21]) to be used in subsequent automated analysis over multiple images. All kernel-based transformations use 5-pixel "cross" kernel. The algorithms may contain empirically derived values that are not specifically commented. Efficiency-related checks allowing early interruption of the erosion cycle in Algorithm 3 have been removed to improve readability.

Algorithm S1 (block A in Figure S1): Image view enhancement

Inputs: original RGB (16-bit per channel) image *i_orig*

Outputs: enhanced RGB image *i_enhanced* of the same dimensions

Parameters (user-defined or read from profile): *thr_low*, *thr_high* (low and high threshold brackets for the reflectivity range of interest)

For each pixel of the output image calculate brightness in each RGB channel according to formula

*br_enhanced = (br_orig – thr_low) * (65535 / (thr_high – thr_low))*

If *br_enhanced* is outside 0-65535 range

Clip the value to 0 or 65535 accordingly.

Algorithm S2 (blocks B and C in Figure S1): Produce map of non-opaque mineral

Inputs: enhanced RGB image *i_enhanced*

Outputs: non-opaque mineral binary map *n_o_map*

Parameters (user-defined or read from profile): thresholds for initial non-opaque mineral identification (low and high in all 3 channels), *scrap1* ("Extra scrapping", here and further in Figure S2) and *erosion1* ("Erosion") values for cleanup, *close1* ("Close"), *fill* ("Fill after thresholding")

Threshold *i_enhanced* to produce *temp_map* of non-opaque mineral, which may include under-identifications and noise readings from epoxy. In manual mode this is performed via an interactive form available by clicking "Threshold bulk".

If *close1 > 0*

Perform *Close* transformation to the given value over *temp_map.*

If *fill*

Perform *Fill* transformation over *temp_map.*

Perform Mineral5 *cleanup()* operation over *temp_map* supplying *scrap1* and *erosion1* as parameters with no other parameters specified. This will essentially perform *Open* to *erosion1* value and *Scrap* to the highest of *scrap1* and Mineral5 auto-cleanup ("Auto-cleanup scrap size") values. Send the resulting map to *n_o_map*.

Algorithm S3 (blocks D and E in Figure S1): Improve map of non-opaque mineral by "reduction to border"

Inputs: enhanced RGB image *i_enhanced,* map of all opaque objects *opaque_map,* existing non-opaque mineral binary map *n_o_map*

Outputs: improved non-opaque mineral binary map *reduced_map*

Parameters (user-defined or read from profile): thresholds for non-opaque mineral border identification (low and high in all 3 channels), *scrap2* ("Don't erode by size", here and further in Figure S2) and *erosion2* ("Don't erode by erosion") values for preservation of small objects, *erosion* ("Erosion limit"), *close2* ("Close"), *restore* ("Restore particles")

Threshold *i_enhanced* to produce *border_map* of non-opaque mineral borders. In manual mode this is performed via an interactive form available by clicking "Threshold borders".

If *restore > 0*

Perform *Dilate* transformation to the given value to both *n_o_map* and *opaque_map*. Find the intersection of dilated maps (*And*), add that intersection (*Or*) to both *n_o_map* and *border_map*. That will ensure no erosion where non-opaque material is in contact with another mineral.

Create a copy of *n_o_map* for further processing (*temp_map*) and a blank *reduced_map*.

If *erosion2* > 0

Perform *Open* transformation to the given value over *temp_map.*

If *scrap2* > 0

Perform *Scrap* to the given value over *temp_map.*

Find the difference (*Xor*) between *temp_map* and *n_o_map,* and keep it as *preserve_map*.

For each object (non-opaque particle or grain) in *temp_map*

Fill it, measure its *min_feret* and *perimeter.*

Calculate *erosion_limit* = min (*min_feret* / 4, *erosion*) to prevent substantial loss of matter for smaller objects.

Create map of this particle alone *p_map.*

Find the intersection (*And*) between *p_map* and *border_map*. This is the border currently inside the particle. Measure its *area*. For 1-pixel border exactly on the particle boundary that would roughly correspond to particle *perimeter*; the actual value may be different both ways depending on border thickness and breaks.

If *area* * 10 < *perimeter*

erosion_limit = 1 as there is obviously not enough border for anything reliable

step = 0

While *step* < *erosion_limit* (the actual erosion occurs within this cycle!!!)

Perform single *Erosion* of *p_map.*

If *step* > 0

Partially restore *p_map* with *keep_map* (*Or*)

Determine the part of *border_map* now outside *p_map* (*And, Xor*). *Dilate* it once and find intersection (*And*) with also dilated copy of *p_map*: this is *keep_map* to be restored at the next iteration, the area where the erosion has already reached the border and we don't want it to go further.

step = *step* + 1

Add (*Or*) the resulting *p_map* to *reduced_map*.

If *close2 > 0*

Perform *Close* transformation to the given value over *reduced_map.*

Combine *reduced_map* with *preserve_map* (*Or*).