Supplementary Section 5. Problem class, ground truth annotations formats and reported metrics

To perform benchmarking, ground truth annotations associated to the images of a BIAFLOWS project should be encoded in a format that is specific to its problem class. Obviously, BIA workflows are also expected to output results in the same format. Currently 9 problem classes are supported and their respective annotation formats and associated benchmark metrics are described below. For currently existing problem classes, most annotations are stored as mask images as TIFF / OME-TIFF images.

Note: each problem class has a long name (explicit) and short name, for instance Object Segmentation (ObjSeg). The same hold for metrics, for instance DICE (DC).

See section **Benchmarking Metrics** for metrics description.

Problem class: Object Segmentation (ObjSeg)

Task: Delineate objects or isolated regions

Object Encoding: 2D/3D label masks with foreground > 0 (one unique ID per object), background = 0

Reported metrics: DICE (DC), AVERAGE_HAUSDORFF_DISTANCE (AHD), computed by VISCERAL executable (archived here), Fraction overlap (FOVL), Mean Average Precision computed by Data Science Bowl 2018 Python code.

Problem class: Spot / object counting (SptCnt)

Task: Estimate the number of objects

Object Encoding: 2D/3D binary masks, exactly 1 spot/object per non null pixel

Reported metrics: RELATIVE_ERROR_COUNT (REC), computed by custom Python code.

Problem class: Spot / object detection (ObjDet)

Task: Detect objects in an image (e.g. nucleus)

Object Encoding: 2D/3D binary masks, exactly 1 object per non null pixel.

Reported metrics: CONFUSION_MATRIX (TP, FN, FP), F1_SCORE (F1), PRECISION (PR), RECALL (RE), Distance RMSE (RMSE), computed by Particle Tracking Challenge metric Java code (particle matching only) (archived here in bin/DetectionPerformance.jar).

Problem class: Pixel/Voxel Classification (PixCla)

Task: Estimate pixels class

Object Encoding: 2D/3D class masks, gray level encodes pixel/voxel class, background = 0

Reported metrics: F1_SCORE (F1), ACCURACY (ACC), PRECISION (PR), RECALL (RE), computed by custom Python code.

Problem class: Filament Tree Tracing (TreTrc)

Task: Estimate the medial axis of a single filament tree

Object Encoding: SWC file

Reported metrics:
UNMATCHED_VOXEL_RATE (UVR), computed by custom Python code.
NetMets metrics: Geometric False Negative rate (FNR), Geometric False Positive rate (FPR) computed by NetMets Python code.

Metrics parameters:
UVR: GATING_DIST (default = 5, maximum distance between skeleton voxels in reference and prediction skeletons to be considered as matched).
NetMets: skeleton pixel_sampling (default = 3), sigma (= GATING_DIST), subdiv (set to 4).

Problem class: Filament Networks Tracing (LooTrc)

Task: Estimate the medial axis of one or several network(s) of filaments

Object Encoding: 2D/3D skeleton binary masks with skeleton pixels > 0, background = 0

Reported metrics:
UNMATCHED_VOXEL_RATE (UVR), computed by custom Python code.
NetMets metrics: Geometric False Negative rate (FNR), Geometric False Positive rate (FPR) computed by NetMets Python code.

Metrics parameters:
UVR: GATING_DIST (default = 5, maximum distance between skeleton voxels in reference and prediction skeletons to be considered as matched).
NetMets: skeleton pixel_sampling (default = 3), sigma (= GATING_DIST), subdiv (set to 4).

Problem class: Landmark Detection (LndDet)

Task: Estimate the position of specific feature points

Object Encoding: 2D/3D class masks, exactly 1 landmark per non null pixel, gray level encodes landmark class (1 to N, N is the number of landmarks).

Reported metrics: Number of reference / predicted landmarks (NREF, NPRED), Mean distance from predicted landmarks to closest reference landmarks with same class (MRE). All metrics computed by custom Python code.

Problem class: Particle Tracking (PrtTrk)

Task: Estimate the tracks of object centroid (no division)

Object Encoding: 2D/3D label masks, exactly 1 particle per non null pixel, gray level encodes particle track ID.

Reported metrics: Normalized pairing score alpha (NPSA), Full normalized pairing score beta (FNPSB), Number of reference tracks (NRT), Number of candidate tracks (NCT), Jaccard Similarity Tracks (JST), Number of paired tracks (NPT), Number of missed tracks (NMT), Number of spurious tracks (NST), Number of reference detections (NRD), Number of candidate detections (NCD), Jaccard similarity detections (JSD), Number of paired detections (NPD), number of missed detections (NMD), Number of spurious detections (NSD). All computed from Particle Tracking Challenge metric Java code (archived here in bin/TrackingPerformance.jar).

Metrics parameters: GATING_DIST (default = 5, maximum distance between particle detections in reference / prediction tracks to be considered as matched).

Problem class: Object Tracking (ObjTrk)

Task: Estimate object tracks and contours (with possible divisions)

Encoding: 2D/3D TIFF label masks, gray level encodes object ID + division text file (see Cell Tracking Challenge format)

Reported metrics: Segmentation measure (SEG), Tracking measure (TRA). All computed from Cell Tracking Challenge metric command-line executables (archived here in bin/SEGMeasure and bin/TRAMeasure).