Received: 4 August 2021 | Revised: 19 October 2021 | Accepted: 5 November 2021

DOI: 10.1002/rob.22049

RESEARCH ARTICLE

WILEY

TagLab: AI-assisted annotation for the fast and accurate semantic segmentation of coral reef orthoimages

Gaia Pavoni[1] | Massimiliano Corsini[1] | Federico Ponchio[1] | Alessandro Muntoni[1] | Clinton Edwards[2] | Nicole Pedersen[2] | Stuart Sandin[2] | Paolo Cignoni[1]

[1]ISTI-CNR, Pisa, Italy

[2]Scripps Institution of Oceanography, UCSD, La Jolla, California, USA

Correspondence
Massimiliano Corsini, ISTI-CNR, Pisa, Italy.
Email: massimiliano.corsini@isti.cnr.it

Funding information
PNRA18 00263-B2, National Antarctic Research Program; Ministero dell'Istruzione, dell'Universitàe della Ricerca

Abstract

Semantic segmentation is a widespread image analysis task; in some applications, it requires such high accuracy that it still has to be done manually, taking a long time. Deep learning-based approaches can significantly reduce such times, but current automated solutions may produce results below expert standards. We propose agLab, an interactive tool for the rapid labelling and analysis of orthoimages that speeds up semantic segmentation. TagLab follows a human-centered artificial intelligence approach that, by integrating multiple degrees of automation, empowers human capabilities. We evaluated TagLab's efficiency in annotation time and accuracy through a user study based on a highly challenging task: the semantic segmentation of coral communities in marine ecology. In the assisted labelling of corals, TagLab increased the annotation speed by approximately 90% for nonexpert annotators while preserving the labelling accuracy. Furthermore, human–machine interaction has improved the accuracy of fully automatic predictions by about 7% on average and by 14% when the model generalizes poorly. Considering the experience done through the user study, TagLab has been improved, and preliminary investigations suggest a further significant reduction in annotation times.

KEYWORDS
artificial intelligence, computer vision

1 | INTRODUCTION

Orthoimages are an essential source of information in many fields, from landscape ecology to computational archaeology and industrial applications. Recently, their use and the data extracted from them have increased, as facilitated by image-based surveys conducted by autonomous vehicles such as drones or UAVs, and through expanding access to GIS analysis solutions. Large area imaging condenses vast amounts of information and represents a mass of data that can quickly become challenging to manage. Machine learning technologies have great potential to expedite image analysis. Deep learning (DL), and convolutional neural networks (CNNs) in particular, have demonstrated performance capabilities that are comparable with humans in a variety of image analysis tasks, such as classification, detection, and segmentation. However, the optimization of CNN models through a supervised learning approach requires a huge amount of annotated data. Once the

data set has been prepared, current GPU performance allow for the rapid optimization of automatic recognition models. While fully automated solutions offer dramatic reductions in human effort, their accuracy is still lower than human experts can achieve for complex scenarios.

Such general considerations are applicable to the field of underwater monitoring. Large-area imaging is an increasingly common solution in the study of subtidal environments. Coral is a framework-building species, and its growth is directly responsible for creating and maintaining coral reef habitats. Spatio-temporal analyses of seabed orthoimages increase the understanding of demographic patterns and the spatial dynamics of coral reef communities. Images are annotated (i.e., corals are outlined for the fine-scale colony mapping) either with standard general-purpose photo editing software or special purpose marine image annotation software (a short review is given in Section 2.2). However, the application of artificial intelligence (AI)-based assisted tools remains marginal, as the required pixel-wise tracing accuracy is only achievable manually. Manual tracing is very time-consuming, as each square meter of imagery demands up to an hour of human effort. As large volumes of unprocessed imagery data are already available and new imagery is continually being created, such human-driven data extraction efforts limit the rate of productivity in the analytical process. Reef-building corals are an incredibly diverse group of organisms, consisting of around 850 species (Hoeksema & Cairns, 2019), and thus offer a unique set of challenges to the field of automatic image processing. Coral biology introduces challenges to the process of automated segmentation, due to the complexity and asymmetry of many coral growth forms and the considerable morphological variability within and among species. Importantly, as corals are relatively slow-growing species (linear extension rates can be less than 1 cm/yr), the level of precision required to accurately document changes and colony evolution is exceptionally high. Poor visibility and floating particles damage image clarity in underwater data. These factors complicate the design of fully automatic semantic segmentation models for coral taxa, a task in which human experience remains central and not replaceable.

We introduce TagLab (https://taglab.isti.cnr.it), an open-source, AI-powered, interactive image segmentation software, for the pixel-wise accurate, scale-aware labelling and analysis of orthoimages. TagLab implements a human-centric pipeline that has been proven to speed up the annotation work, retaining the accuracy of the manual approach and ensuring experts keep control of the annotation process. The annotation pipeline comprises three steps: (1) an AI-assisted/manual labelling, in which intelligent tools based on CNNs speed up the annotation from scratch; (2) a learning pipeline to create, test, and use custom recognition models; (3) an editing/validation final step, in which the expert can improve automatic predictions. In terms of data analysis, TagLab integrates ad hoc image-processing and image-analysis tools, supports georeferencing, and interoperates with GIS software. The software includes the following original resources:

- an AI-based flexible annotation workflow, which unlike similar software allows the per-pixel editing of predictions. More details are given in Sections 2.2 and 3,
- an *Edit Border* tool, which facilitates the manual editing of complex boundaries (Section 3.1.2),
- the support of multichannel images, which enables multimodal coregistered data to be handled. For example, the loading of digital elevation models (DEMs) allows users to approximate the 3D surface area of coral colonies (see Section 4),
- a *Multitemporal comparison* tool, which automatically tracks the temporal evolution between segmented regions and allows for interactive visual inspections of extracted data (see Section 4).

This paper reports a thorough evaluation of the improvements brought by TagLab both in terms of *annotation time* (efficiency) and *accuracy*. This evaluation was carried out through a comprehensive user study conducted with the Scripps Institution of Oceanography (UCSD). Semiautomatic and automatic tools demonstrated their efficiency in speeding-up up coral reef large-image analysis. In addition, the interactive editing of automatic predictions also proved essential for achieving the high accuracy levels required in ecological studies.

TagLab is available on GitHub (https://github.com/cnr-isti-vclab/TagLab). Reducing the time required for postprocessing of coral reef imagery enables researchers to process increasingly large volumes of data, thus facilitating a greater capacity to understand and predict future changes to coral reef ecosystems. We closely collaborate with several marine research laboratories, continuously updating the software (see Section 6.4) and developing novel automated/assisted strategies to support digital underwater monitoring (Figure 1).

2 | RELATED WORK

The widespread use of supervised deep-learning solutions has recently led to the development of several software applications and algorithms that speed up the preparation of training data sets. In terms of the semantic segmentation task, many of these applications exploit *weak supervision*. Object areas are rapidly marked using points, scribbles, bounding boxes, or polygons. Starting from this partial information, an algorithm then generates segmentation masks. This section provides a brief description of the most popular weakly supervised annotation methodologies, followed by an overview of the current tools developed for marine species annotation.

2.1 | Weakly supervised methods

Drawing a bounding box is a quick and intuitive task. Khoreva et al. (2017) use bounding boxes to extract an initial proposal of the object mask. The region outside the box is marked as the background, while an algorithm (Pont-Tuset & Van Gool, 2015) evaluates the inside area. A recursive training frame-work based on a CNN achieves the final prediction. Deep Grabcut (Xu et al., 2017) uses the bounding box as a soft constraint, designing a CNN which takes as its input an image concatenated with a "distance map" defined as starting from the object bounding box.

One of the first interactive segmentation methods (Boykov & Jolly, 2001), (Boykov and Jolly, 2001), involves the tracing of the object's background and foreground scribbles. The segmentation task

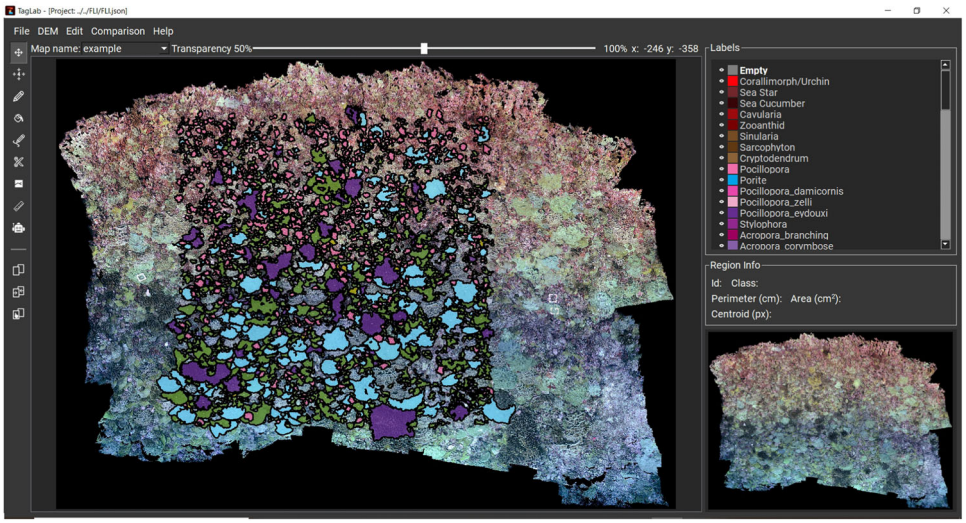

FIGURE 1 TagLab's main user interface splits into three main components: the central *Working View*, the *Toolbar* on the left, and a right area containing three panels: the *Labels*, the *Region info*, and the *Map Viewer*. The Working View covers the central part of the interface and visualizes the orthoimage with overlaid semantic annotations (colored polygons)

is then formulated and solved as a graph cut problem. Other classic solutions, such as GrabCut (Rother et al., 2004), are based on energy minimization. Geodesic Star (Gulshan et al., 2010) exploits a weighted geodesic distance based on pixel statistics to obtain segmented regions starting from scribbles. ScribbleSup (Lin et al., 2016) uses a graph-based model in conjunction with a fully convolutional network (FCN) (Long et al., 2015) to propagate the scribble information to entire segmentation regions. The graph is built on a superpixel subdivision of the input image.

Many interactive methods output semantic regions starting from point clicks. Xu et al. (2016), integrate positive clicks in the foreground with negative clicks in the background in a learning scheme. Euclidean distance transforms the clicked points into two separate maps, which are later concatenated with the input image to feed a FCN. A graph cut optimization (Rother et al., 2004) applied on the FCN output leads to the final segmentation. Le et al. (2018), transform user clicks into an interaction map by expanding Gaussians centred on them; which then feed an FCN network (Long et al., 2015) to output a rough predicted mask. Finally, a standard geodesic path solver (Cohen, 2006), applied on the boundaries map refines the segmentation. A recent solution (Forte et al., 2020) builds upon a U-Net (Ronneberger et al., 2015) architecture reaches an exceptional accuracy of between 95% and 99% of mean intersection-over-union (mIoU), a measure of overlap between labelled regions, by using an elevated number of user's clicks (around 20). A solution between bounding boxes and point-clicks is represented by clicking the object's extremes (top, left, bottom, and right). The efficiency of using extremes in terms of the bounding boxes has been demonstrated by Papadopoulos et al. (2017). They report a median time for annotating an object of 34.5 s, 25.5 s for drawing the bounding box, and 9.0 s for confirming annotation's correctness. Picking extreme points is five times faster then drawing bounding boxes and requires only 7 s on average, thanks to its small cognitive workload. Thus, the experimental results of Papadopoulos et al. show that the performance of automatic recognition models (Fast R-CNN, Girshick, 2015, for object detection and the DeepLab for object segmentation) trained by using them is higher. This means that, in general, humans provide tighter bounding boxes around the objects using this paradigm. Maninis et al. (2018), propose Deep Extreme Cut (DEXTER), a CNN for the interactive agnostic segmentation based on the extremes point paradigm. DEXTER follows technical solutions used in DeepLabV3+ to achieve high-res results.

Objects without holes can be precisely annotated using enclosing polygons, but as drawing a polygon typically requires many clicks (30–40), this is a high time-consuming labelling method. Polygon-RNN (Castrejón et al., 2017) speed up polygon tracing using a recursive neural network (RNN). As soon as the user starts clicking points, the Polygon-RNN network processes the placed clicks and automatically predicts the next ones. This process has been found to speed up the generation of segmentations by a factor of 4.7 when tested on the Cityscapes data set (Cordts et al., 2016). RNN++ (Acuna et al., 2018) then follows Polygon-RNN, in which the features are extracted using a modified version of the ResNet-50 to increase their resolution. An *Evaluator Network* then estimates the accuracy of predicted polygon via Reinforcement Learning. Finally, the output is refined using a Graph Neural Network (GNN).

2.2 | Annotation and segmentation solutions for marine organisms

In this section we review the algorithms and software tools developed for the annotation and segmentation of marine organisms. The web platform CoralNet is a widely known AI-based solutions for creating manual and assisted point-based annotations (Beijbom

et al., 2015). Images are annotated directly in the web browser, and when a sufficient number of data have been annotated, CoralNet trains a classifier and helps label the remaining images. Squidle+ (Friedman, 2017) is a cloud-based platform for annotating and georeferencing underwater visual data. This is extremely versatile in handling image, video, and orthomosaics (as a collection of tiles). TagLab and Squidle+ follow different approaches. First, Squidle+ handles point-based annotation while TagLab labels regions. Generally, point-based information is not sufficient for identifying the demographic drivers of change in coral communities (Edmunds & Riegl, 2020). Second, the AI-assisted part of Squidle+ implements an active learning approach; the interactive system asks the user additional inputs to improve its classification performance. TagLab offers a nonrigid working pipeline, offering assistive tools for the direct editing of automatic predictions. In terms of interactive tracing, DeepSegment (Andrew, 2018) adopts an image segmentation approach based on GrabCut (Rother et al., 2004) and superpixels (Achanta et al., 2010). Parameters must be tuned manually for each colony to achieve high accuracy. DeepSegment segments the entire image in small subregions. The user must add sematics separately to each one, which is a time-consuming process. CoralSeg is another recent algorithm that exploits superpixels in a hierarchical way to expand the sparse labelling, thus obtaining a coherent semantic segmentation (Alonso et al., 2019). This algorithm has been successfully applied to repeatable surveys of benthic communities (Yuval et al., 2021). CoralMe (Blanchet, 2016) adapts the Geodesic star convexity algorithm (Gulshan et al., 2010) to corals segmentation. This algorithm takes an internal and an external sketched curve as input and returns the colony's accurate boundary outlining. The two initial curves must already be close to the contours to be effective, making the process accurate but not fast. Biigle (Langenkämper et al., 2017), is a web-based image and video annotation software that allows collaboration between users. It integrates an instance segmentation CNN, the Mask R-CNN (He et al., 2017); and like TagLab, the fine-tuning of this network follows a human-in-the-loop approach, as detailed in (Zurowietz et al., 2018). The main differences between this approach and our method is that the fine-tuning of the Mask R-CNN is achieved by accepting or discarding automatically generated proposals (yes/no paradigm) while TagLab allows for the rapid creation of a data set for the fine-tuning of DeepLab V3+ from scratch (due to assisted annotation) or by editing the obtained predictions and reusing them for the training. The complete workflow of TagLab is described in the next section.

3 | TAGLAB: A HUMAN-CENTRIC AI APPROACH

Scientific applications usually involve specific image data containing uncommon objects and complex recognition tasks, which require deep field knowledge and a high cognitive effort. Uncommon objects are usually underrepresented in machine learning benchmark datasets (which contain mostly everyday objects), thus affecting the potential of current CNN recognition models. In addition, the automation of complex recognition tasks following a supervised approach demands a massive amount of highly targeted training data. Automatic labelling techniques can then fail to reach the accuracy levels achieved by experts (over 90%), and AI human-centric technologies that empower (rather than replace) human abilities are usually more successful than fully automated solutions. TagLab follows this principle by proposing the working pipeline illustrated in Figure 2. First, TagLab speeds up the manual annotation through a combination of AI-assisted tracing algorithms and specialized tools, thus creating suitable training data sets (Step 1). Next, the user is guided to a fully automatic custom semantic segmentation model optimization (Step 2). The process starts with the custom data set

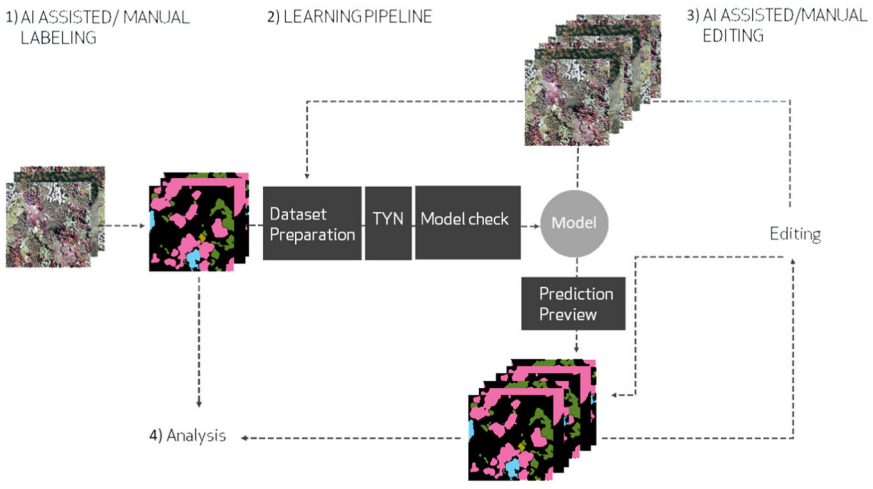

FIGURE 2 TagLab's annotation pipeline consists of three steps. (1) The assisted annotation. (2) The learning pipeline, which guides users to optimize a custom semantic segmentation model. (3) The AI-assisted manual editing, where humans re-enter the annotation loop by correcting the automatic results using specialized tools. Additionally, TagLab integrates data analysis functionalities (4) accessible from different stages of the annotation process

preparation; then the user sets the learning hyperparameters using the train-your-network (TYN) feature and launches the model optimization. Once the optimization ends, the user evaluates the learning metrics (such as the confusion matrix and the mIoU), visualizes predictions on the test tiles, and decides whether to save the model. This model can then be used to infer predictions on new unlabelled orthoimages. After the automatic classification, the human expert can re-enter the annotation loop (Step 3) and correct the prediction errors with the editing tools, as in Step 1. Finally, TagLab offers several options for analysing the annotated images (Step 4). In addition, TagLab supports the import of color-coded images, allowing for the refinement of annotations/predictions as inferred outside of TagLab. TagLab has been implemented in Python using the Pytorch framework for neural networks, and the PyQt (Python version of the Qt framework) library handles the GUI. All the GUI components have been implemented as Qt custom widgets to increase modularity and adaptability. Image processing resources are mainly based on scikit-image. GIS-related functionalities are based on GDAL and GEOPY. TagLab runs on Linux, Windows, and MacOS. The main requirements are Python 3.6 or 3.7 and the NVIDIA CUDA Toolkit, versions 9.2, 10.1, or 10.2. To complete the entire learning pipeline, TagLab requires at least 6GB of RAM, and preferably 8 GB. TagLab is an open-source project released under the GPL V3 license. We next examine the design choices behind each step of Figure 2.

3.1 | AI assisted/manual labelling

TagLab handles the pixel-wise assisted/manual labelling of large orthoimages, eventually containing thou- sands of labelled regions. All of the interactive tools work at the orthoimage full resolution, and segmented regions are approximated and stored as polygons with subpixel accuracy.

3.1.1 | AI boundaries tracing

AI-based interactive annotation tools have two major advantages over standard image processing algorithms for interactive segmentation. First, CNNs are content-aware, and thus knowing what an object is, for example, in coral outlining, allows them to distinguish between the internal and external regions of a colony. Second, no additional parameters need to be specified (as in DeepSegment). TagLab integrates two interactive segmentation CNNs that are both fine-tuned to work on corals shapes: the *4-clicks* and *positive/negative clicks* tools. Below we only detail the *4-clicks* tool, as the second interactive CNN was introduced after the user study (see Section 6.4).

The *4-clicks* tool implements a custom version of the DEXTER (Maninis et al., 2018), which exploits the *extreme points* paradigm, as described in Section 2. This CNN was originally trained using two datasets, PASCAL VOC 2012 and the Semantic Boundaries Data set, specifically for semantic contour prediction. These data sets mainly contain everyday objects, so the original CNN tends to trace regular profiles. The version implemented in TagLab has been optimized to predict complex, jagged natural shapes after learning from a data set of 15,000 manually segmented coral instances. The Deep Extreme Cut network takes as inputs 4-channel data, the RGB image object, and a heat map created by centring four Gaussians on the extreme points, as indicated by the user. To produce the heat maps, we extracted the extreme points from each segmentation then simulated the uncertainty induced by human annotation (it is hard for an annotator, however accurate, to exactly pick the extreme points with pixel precision), by adding a random displacement in a range of 10 pixels around each extreme point. All of the network parameters were unfrozen during the fine-tuning. To avoid any forgetting effect, we set a learning rate ten times lower than the first training. The augmentation included both colour and geometry. The optimized model achieved an accuracy of 0.967 and a mIoU of 0.853. In Taglab, the *4-clicks* tool activates a cross cursor which helps the user place the coral extreme points. The CNN outlines a pixel-level mask, which TagLab converts into a dense polygonal line approximating the colony's boundaries at sub-pixel accuracy.

3.1.2 | Advanced editing tools

The *Edit Border* and the *Refinement* are advanced editing options in TagLab. The *Edit Border* tool enables the manual pixel-level editing of polygon outlines without the need for select-and-drag edit points or for specifying a size for the drawing tool. The user sketches an arbitrary number of curves crossing the polygon boundary. TagLab snaps them to the initial polygon removing the leftover parts, as shown in Figure 3. The *Edit Border* algorithm performs pixel-level morphological operations on the binary mask (black background) rasterized with a subpixel-accuracy from the selected polygonal label (Figure 4). The algorithm works as follows:

1. Save and remove the internal holes of the current mask *M* (if present).
2. Draw editing curves using white pixels.
3. Fill the black foreground regions with a hole-filling method.
4. Redraw the curve using black pixels.
5. Use a connected labelling algorithm to obtain the different regions.
6. Keep only the region with the largest area.
7. Readd the original internal holes.

Inner polygons are created by drawing closed curves that are then subtracted from the label. Existing internal holes are treated as a separated mask and readded to the outer mask after confirming the operation during the editing. After each editing or hole creation, the corresponding demographic statistics are updated. The *Edit Border* uses fast image processing operations on local binary masks working instantaneously. This solution has proven to be more efficient in pixel-wise editing than click-based interactive refinements solutions,

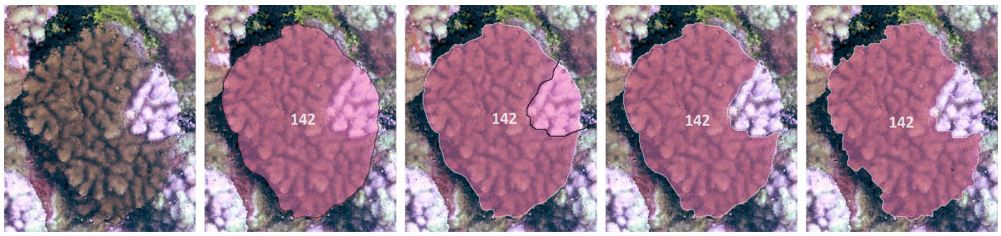

FIGURE 3 From left to right: A Pocillopora colony, the associated labelled polygon, the edit curve, the edited polygon, and the automatically refined polygon. The editing curves snap to the mask allowing pixel-level editing operations. The automatic refinement uses a variant of the graph-cut algorithm to improve polygon adherence to the coral boundaries

FIGURE 4 The algorithm of the *Edit Border* tool explained. From left to right: (1) The binary mask and the input edit curves (drawn in blue) snapped to the mask. (2) Edit curves are first drawn in white. (3) Filling of the subtended regions. (4) Edit curves are redrawn in black. (5) The largest connected component becomes the final mask. Note that the black and white lines are thin to enable pixel-level precision during the mask processing. We thicken the lines to improve the clarity of the figure

particularly when using a graphics tablet, given the high complexity of coral morphologies.

The *Refinement* automatically improves the segmentation accuracy with the constraint that the refined segments must be close to the originals (Figure 3). This tool implements a custom version of the graph-cut segmentation algorithm (Boykov & Jolly, 2001). In the graph-cut based approaches, separation curves are determined by a boundary term (usually related to the RGB image gradient) and by a regional term. In TagLab, the regional term is computed by exploiting the color histograms of the foreground H_f, and of the background H_b: image colors are quantized using 4 bits per component and the histograms are normalized according to the area values. These normalized histograms, are used to define two per pixel functions $H_f(x, y)$ and $H_b(x, y)$, where the color of the pixel is looked up in the histograms. These functions can be combined to roughly approximate the probability that a pixel belongs to the foreground or the background. To force the new boundary to remain close to the original we compute the signed distance to the original boundary $D(x, y)$ and the overall contribution is weighted by two parameters a and b; $a(H_f(x, y) - H_b(x, y)) + bD(x, y)$. The default values are $a = 0.1$ and $b = 0.05$. The boundary term is computed from the RGB values of two neighboring pixels: $\exp(-\|RGB(x_1, y_1) - RGB(x_2, y_2)\|^2)$. To speed up the computation, both the distance transform and the graph-cut algorithm are calculated only in the original contour neighbourhood. If a polygon is manually edited before activating the refinement operation, the refinement acts only in the edited parts.

3.1.3 | Basic creation/editing tools

The basic creation and editing tools are based on simple image processing operations. Manual labelling occurs via the *Freehand* drawing tool, which creates a new polygon for each drawn closed curve. The closed curve can be drawn as different overlapping segments rather than continuously. Curves can intersect but will always produce a unique closed polygon without discontinuities or holes. Thus, there is no need to match the start and end of the line precisely, and any additional segments, inside or outside the polygon, are automatically removed. *Cut Segmentation* splits a polygon into several independent polygons. This tool works exactly like the *Edit Border* tool except that the subtended regions are separated, not removed, from the starting polygon. The Crack tool (Figure 5) interactively creates empty cracks inside a polygon. It implements a simple colour-based flood fill algorithm, and the user can use a slider to threshold unwanted pixels. TagLab performs simple region-based morphological operations on the segmented regions, such as dilation, erosion, and hole filling and boolean operations, and subtracts or merges overlapped labels. Divide labels avoid counting pixels belonging to overlapped regions twice, which can invalidate spatial analyses.

3.2 | Learning pipeline

The high specificity of scientific data requires the creation of ad hoc classifiers tuned to custom data. TagLab, therefore, applies specific

solutions for preparing training datasets, by decomposing large orthoimages (much larger than the typical input size of CNNs) and performing the fully automatic semantic segmentation on them. Once the labelling of one or more orthoimages has been concluded, users can create a data set by opening the Export New Training Data set window, selecting the area of interest, and choosing a strategy for partitioning the map into training, validation, and test subareas. This choice can follow different criteria depending on the class distribution: uniform-vertical, uniform-horizontal, random, or ecologically inspired-partition (see Pavoni, Corsini, et al., 2020). The rationale is to prefer a partitioning strategy that distributes the semantic regions as evenly as possible in the three subareas. The extension of these subareas is chosen according to the usual data partition in supervised DL applications. The sub-areas are then sliced (scan order) into overlapping squared tiles. The data set preparation algorithm allows for oversampling of the minority classes to improve the class balance. The oversampling follows the approach described in Pavoni, Corsini, et al. (2020). Tiles size consider both the input size of the Deeplab V3+ (Chen et al., 2018) architecture and the overlap required to aggregate multiple predictions, thus avoiding tiling artifacts. In some applications, such as ecological monitoring, the object scale is a discriminant feature for classification. To combine data from multiple orthoimages uniformly and independently by the pixel sizes of each map, while avoiding introducing scale inconsistency, users can set an intermediate target pixel size, and tiles are then exported scaled to this value.

The *Train Your Network* function runs the DeepLab V3+ model optimization (Figure 6) The learning pipeline interface is specifically designed to guide scientists without a computer science background through the model optimization process. Thus, we tested several hyperparameter configurations on different datasets, and for the TagLab default hyperparameter settings, we chose the configuration that mainly outputs stable models, mitigating overfitting, and forgetting. At the end of the optimization, the training results are visualized through the *Training Results* window (Figure 7). Some simple metrics quantify the classification performance, such as the accuracy, the mIoU, the normalized confusion matrix, and the loss training and validation graphs. Finally, the *Training Results* window allows for the side-by-side inspection of test tiles and the corresponding ground truth labels and predictions before saving the classifier. The *Fully automatic segmentation* tool loads one of the classifiers and infers predictions on the active orthoimage. As for the interactive tools, the fully automatic segmentation works on subimages at a full resolution. TagLab calculates multiple predictions on a sliding window and combines them to avoid inconsistencies between windows.

3.3 | Assisted/manual editing

Automatic predictions may not meet the standards of applications involving high pixel accuracy. To improve accuracy through human supervision (Step 3), TagLab visualizes semantic regions as

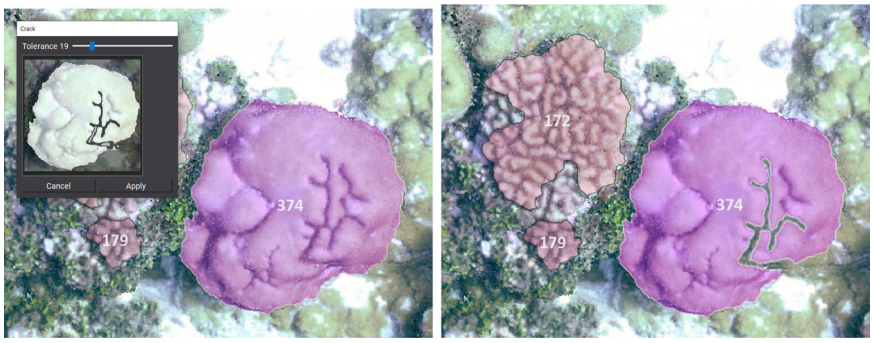

FIGURE 5 Crack tool. Picking a point inside an inner crack generates an inner hole. A manually tunable threshold allows to filter out unwanted pixels

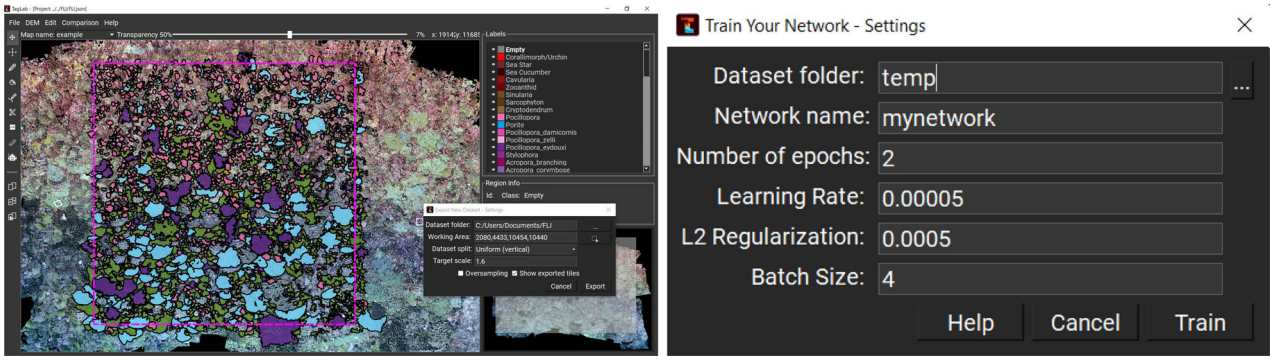

FIGURE 6 The export new training data set window (left); the train your network window (right)

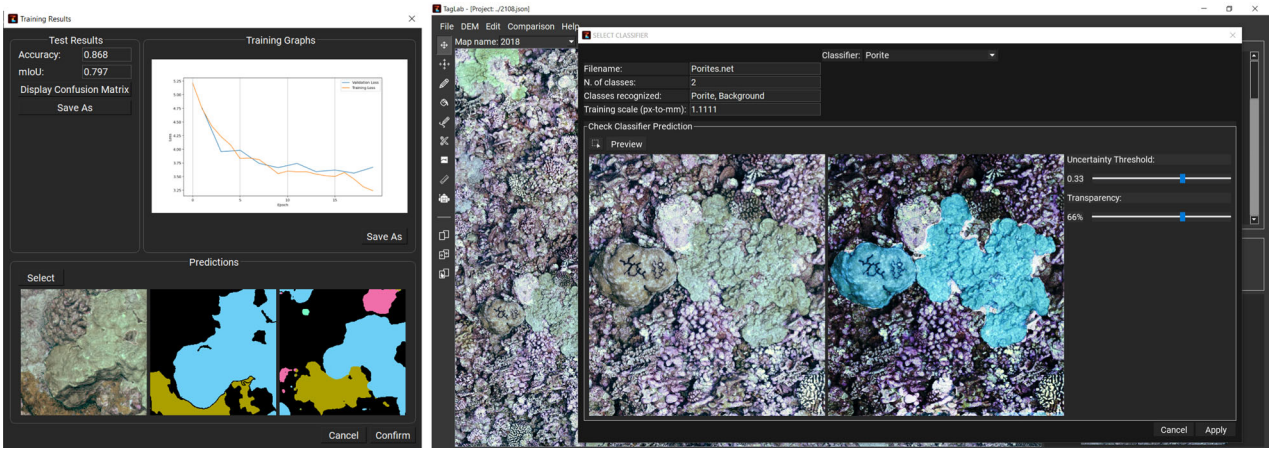

FIGURE 7 The *Training Results* window (left) displays information regarding the training and the quality of predictions. The *Fully Automatic Classification* tool (right) opens a results preview on new data

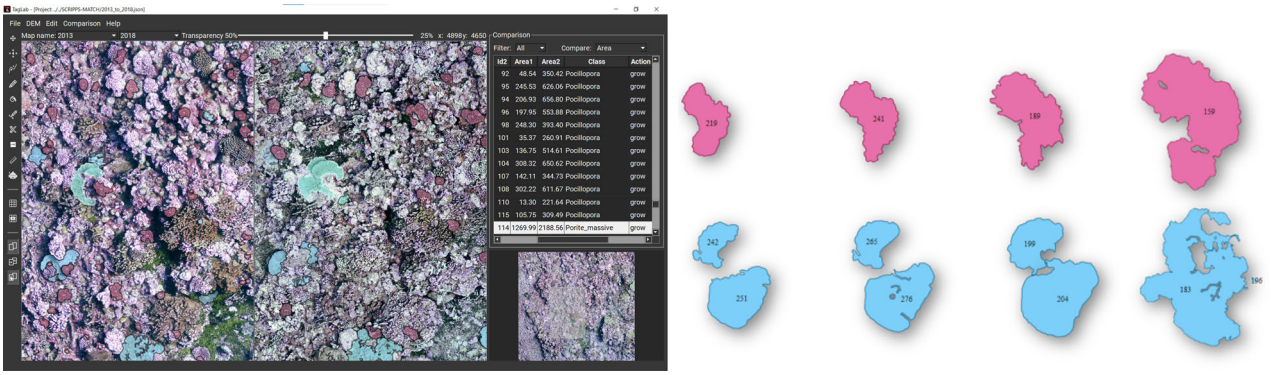

FIGURE 8 On the left we show the synchronized navigation of two coregistered orthoimages acquired at 5 years of distance. The highlighted row of the table, which contains all of the automatic matches found between colonies, shows the growth of the selected Porite's colony; the colony is highlighted in both views. On the right we show the evolution of a Pocillopora and a Porite colony exported from TagLab. This type of information enables the tracking of genetically unique individuals (genetic tracking)

transparent polygons with unique IDs, thus enabling rapid regions checks and further editing at the pixel level using the assisted/manual labelling resources described in Section 3.1.1.

4 | RESOURCES FOR AUTOMATIC ANALYSIS

In addition to the advanced labelling functionalities, TagLab offers several image analysis options involving higher-level information, such as DEMs (widely used in spatial/landscape analysis) or the automatic detection of change in multitemporal surveys. All information related to the positioning and extent of semantic regions (such areas, centroids, perimeters) is directly accessible to the user for computing ecological statistics.

DEM processing. The DEM can be imported together with coregistered orthoimages and used to refine labels while considering depth jumps or approximate the 3D surface areas of labelled regions. Annotated polygons can then be used as a clipping mask on the depth raster, to obtain a set of semantically segmented raster objects. When loading the depth information, TagLab supports the export of the RGB-D training data set, thus enabling the training of multi-modal networks.

Temporal evolution of coral colonies. In TagLab, multiple annotated orthoimages can be loaded into the same project, and these semantic maps (orthoimages plus labels) are navigated in pairs through two synchronized views (Figure 8) thus enabling changes to be visually inspected. In addition, when orthoimages are coregistered (coregistrations are usually obtained by placing underwater markers), TagLab automatically tracks morphological changes between segmented regions by analysing polygons overlaps. Other image-matching solutions are unreliable in underwater scenarios, due to the nonrigid deformations on seafloors; everything moves and changes underwater. After the computation, matches are displayed in an interactive

table synchronized with the two comparison views. The user can interact with the table, receiving visual feedback. By selecting a row, the views centre the corresponding coral colonies or, vice versa, selecting a coral in a view highlights the corresponding rows of the table. The table also contains an automatically assigned tag that summarizes the region's morphological change (growth, erode, born, died, split, and fuse). To our knowledge, this last simple but effective visualization feature is not available in any other marine image analysis tool. The users can interactively correct eventual mismatches. As each orthoimage may contain thousands of colonies, this tool greatly simplifies the temporal evolution analysis of benthos. Although the user study in this paper does not include this analysis tool, the functionality has already been adopted in a recent publication (Sandin et al., 2020).

5 | USER STUDY

This study is aimed at assessing the performance of the TagLab assisted annotation pipeline (Step 1 in Figure 2) and of the automatic labelling plus editing pipeline (Step 2 and 3 in Figure 2) by evaluating both the *annotation time* and the label *accuracy*. Six ecologists from the Scripps Institution of Oceanography were involved as annotators, completing the study in February 2020. All the materials of this user study (orthoimages, ground truth labels, label maps) are collected in the Supporting Information Material.

5.1 | Materials

The 10 orthoimages used as the training data set for the model optimization and the four orthoimages labelled during the user study were obtained from the 100 Island Challenge project (http://100islandchallenge.org), headed by the Scripps Institution of Oceanography, UC San Diego. The protocol for orthoimage creation is detailed in Kodera et al. (2020). To summarise, plots were imaged using a Nikon D7000 camera, which captured highly overlapping images per plot to create a single contiguous 3D model of each plot using the structure-from-motion software Agisoft Metashape (Agisoft, 2019). The dense cloud was then imported into the custom visualization platform *Viscore* (Petrovic et al., 2014) to create orthoprojections. Finally, scale bars and ground control points were deployed in the field to provide scaling and orientation of the 3D model relative to the ocean surface, which is required for subsequent orthorectification. The annotated training data set and the ground truth labelled maps for the user study were created following the consolidated Photoshop-based annotation pipeline used at the Scripps Institution of Oceanography. The 10 orthoimages used for network optimization portray 10 × 10 m of coral reefs, and more details are given in Pavoni, Corsini, et al. (2020). The four photogrammetric surveys for the user study were conducted in 2013 at the Millennium Atoll (*MAL*), Vostok Island (*VM01* refers to orthoimage 1 of the Vostok Island and *VM03* to orthoimage 3), and Malden Island (*SMA*) in the Southern Line Islands. The geographic locations having varying water conditions. Each orthoimage measures around 3 × 3 m and the average pixel size is around 1 mm. Figure 9 displays the *MAL* orthoimage and the associated ground truth labels. The colours associated with each class are black for the Background, light blue for *Porites*, green for plating *Montipora*, olive green for encrusting *Montipora* and pink for *Pocillopora*. The five selected taxa represent a gradient of morphological complexity, and the coverage and distribution frequencies differ in each orthoimage.

5.2 | Methods

Ground-truth annotations were generated through mutual agreement by the two ecologists who lead the processing and analysis of

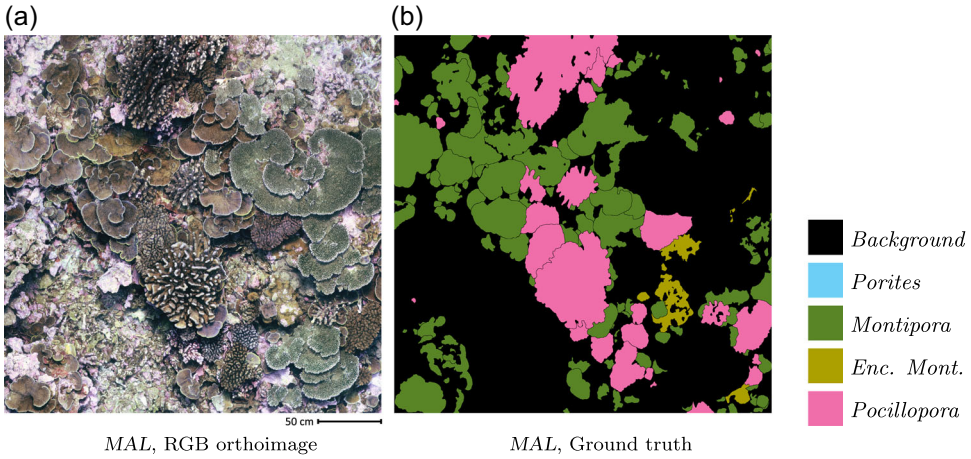

FIGURE 9 *MAL* orthoimage (left), the associated ground truth label map (right), and the colour code used. Label colours are black for the Background, light blue for *Porites*, green for plating *Montipora*, olive green for encrusting *Montipora* and pink for *Pocillopora*. Note that, in this orthoimage, no *Porites* is present

the image-based data products at Scripps. Remaining ecologists formed the two subgroups of "annotation beginners" (indicated with *U1* and *U2* in the following) and "annotation experts" (indicated with *U3* and *U4* in the following). Beginners had ample experience of coral taxonomy and ecology but minimal experience with the Photoshop-based labelling workflow, while the experts were already comfortable with manual annotation using Photoshop. Both experts and beginners had no previous experience with TagLab. Before embarking on the user study, both groups received the same written instructions about using TagLab and practiced alone for about one hour. Each user performed each task on the four orthoimages in a randomly assigned order, to prevent systematic bias and to avoid, for example, the same image always occurring in the same task. We logged all the user's operations to evaluate the interactions with each specific tool and estimated the annotation time. The annotation tasks assigned to annotators are:

Task 1: Label the orthoimage following the Scripps' Photoshop pipeline and report the annotation times.

Task 2: Label the orthoimage by exploiting only the manual unassisted drawing tools of TagLab (the *Freehand*, *Edit Border*, *Cut*, and *Refinement* tools). Task 2 aimed at testing the basic, standard TagLab drawing functionality compared to the Adobe Photoshop approach (Task 1).

Task 3: Label the orthoimage by using the assisted agnostic segmentation tool. Editing and refinement options are allowed. The goal of Task 3 is to assess the improvements offered by the 4-click based segmentation tool, considering both the time reductions in labelling (comparing the total time required to complete Task 3 relative to Tasks 1 and 2) and the labelling accuracy (of Task 3 in terms of the ground truth label maps created by the head ecologists).

Task 4: Run the fully automatic classification and correct any outliers. This test evaluates if editing the automatic prediction is more convenient than the assisted labelling. Again, we evaluate both time and accuracy.

Table 1 reports the combinations between images, tasks, and users that occurred during the study. The automatic classifier has been trained using ten annotated orthoimages; the details of the training are reported in Pavoni, Corsini, et al. (2020). Data set tiles were exported according to the default settings of the Export New Training Data set function. By exploiting the Train Your Network feature, we fine-tuned the DeepLab V3+ for 80 epochs, using a learning rate of 0.00005, a weight decay of 0.0005, and a batch size of 64. The model minimized a Focal Tversky loss function (Abraham & Khan, 2019) using the Hyperbolic Adam Optimizer (Ma & Yarats, 2019); scores on the test data set were 0.90 of accuracy and 0.84 of mIoU.

TABLE 1 Orthoimage used for the different combinations of user and task

	U1	*U2*	*U3*	*U4*
Task 1	*MAL*	*VM03*	*VM01*	*SMA*
Task 2	*SMA*	*MAL*	*VM03*	*VM01*
Task 3	*VM01*	*SMA*	*MAL*	*VM03*
Task 4	*VM03*	*VM01*	*SMA*	*MAL*

Note: Each column represents a single user. *U1* and *U2* are annotation beginners, while *U3* and *U4* experts.

We assessed the labelling quality of each task by calculating the accuracy and the mIoU of each label map compared to the ground truth. Additionally, we evaluated the per-class user agreement through Cohen's kappa coefficient (Schoening et al., 2016), while at the end of Section 6.1 we applied a voting scheme with the two purposes of first, visualising the per-pixel agreement among different annotators and second, assessing the reliability of the user study, as the votes were derived from the labels produced in different tasks. Finally, we calculate TagLab's efficiency gain by estimating the per-pixel contour tracing speedup relative to Task 1. As the orthoimages can contain from a few to thousands of corals with different shape complexity, the per-pixel tracing speed is a reliable and robust measure.

6 | RESULTS AND DISCUSSION

As a general consideration, the high quality and the consistency of results confirmed the reliability of the user study and the absence of systematic errors. In addition, the overall accuracy of each task relative to the ground truth was high and consistent among the users. Figure 10 gives the four label maps produced by the different annotators using different tasks associated with the *MAL* orthoimage.

6.1 | Accuracy and user agreement

Figure 11 gives the accuracy values calculated for each task, map, user, and class. Table 2 summarizes the average accuracy values. *VM01* and *VM03*, the two orthoimages containing all five classes and showing a denser coverage according to the *average accuracy per orthoimages (a)*, have a higher proportion of incorrectly classified pixels. The *average accuracy per task* is similar although slightly lower in Task 2; the values for Taglab's manual segmentation without assisted tools are lower than when using Photoshop. However, all users are more experienced in Photoshop freehand drawing and editing and less accustomed to using TagLab. In Task 3 and 4, accuracy was comparable with that of Task 1 on average, thus clearly demonstrating that the use of TagLab with assisted instruments enables large reductions in annotation times without any impact on the accuracy of results. The *average accuracy per user* across the four different tasks shows that the annotators generally performed at the same skill level. User *U3* was slightly less accurate than the others. The *average accuracy per class* highlights that coral classes are not equally detectable. Encrusting *Montipora* is correctly classified in less

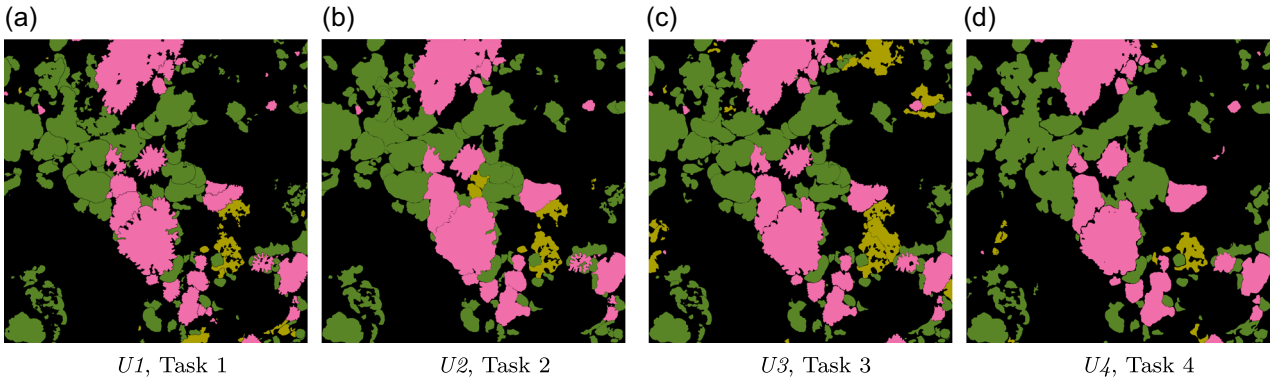

FIGURE 10 Labels for the orthoimage *MAL*. *MAL* contains four of the five classes under investigation and displays a medium level of benthic coral coverage. User *U3* (in Task 3) (c) classified a significant quantity of encrusting *Montipora* (olive green) not reported in the other labels. User *U1* (in Task 1) (a) created much more complex regions, rich of internal contours and ramifications, especially when compared with User *U4* (Task 4) (d)

Site	Photoshop Manual Tracing (Task 1)	TagLab Manual Tracing (Task 2)	TagLab Assisted Segmentation (Task 3)	TagLab Fully Auto Classification	TagLab Editing of Predictions (Task 4)
MAL	User *U1*	User *U2*	User *U3*		User *U4*
	Accuracy: 0.958; mIoU: 0.920	Accuracy: 0.953; mIoU: 0.912	Accuracy: 0.929; mIoU: 0.886	Accuracy: 0.888; mIoU: 0.809	Accuracy: 0.941; mIoU: 0.890
	IoU: 0.937 / IoU: 0.900 / IoU: 0.648 / IoU: 0.905	IoU: 0.935 / IoU: 0.869 / IoU: 0.603 / IoU: 0.915	IoU: 0.893 / IoU: 0.884 / IoU: 0.228 / IoU: 0.921	IoU: 0.839 / IoU: 0.751 / IoU: 0.278 / IoU: 0.820	IoU: 0.914 / IoU: 0.865 / IoU: 0.359 / IoU: 0.883
SMA	User *U4*	User *U1*	User *U2*		User *U3*
	Accuracy: 0.995; mIoU: 0.990	Accuracy: 0.994; mIoU: 0.989	Accuracy: 0.994; mIoU: 0.988	Accuracy: 0.987; mIoU: 0.975	Accuracy: 0.994; mIoU: 0.988
	IoU: 0.994 / IoU: 0.956 / IoU: 0.923	IoU: 0.994 / IoU: 0.948 / IoU: 0.925	IoU: 0.993 / IoU: 0.945 / IoU: 0.926	IoU: 0.986 / IoU: 0.884 / IoU: 0.854	IoU: 0.993 / IoU: 0.945 / IoU: 0.913
VM01	User *U3*	User *U4*	User *U1*		User *U2*
	Accuracy: 0.938; mIoU: 0.883	Accuracy: 0.943; mIoU: 0.894	Accuracy: 0.925; mIoU: 0.861	Accuracy: 0.848; mIoU: 0.732	Accuracy: 0.917; mIoU: 0.847
	IoU: 0.884 / IoU: 0.894 / IoU: 0.904 / IoU: 0.150 / IoU: 0.781	IoU: 0.892 / IoU: 0.899 / IoU: 0.911 / IoU: 0.347 / IoU: 0.828	IoU: 0.855 / IoU: 0.866 / IoU: 0.883 / IoU: 0.139 / IoU: 0.805	IoU: 0.756 / IoU: 0.585 / IoU: 0.817 / IoU: 0.000 / IoU: 0.497	IoU: 0.846 / IoU: 0.841 / IoU: 0.881 / IoU: 0.372 / IoU: 0.700
VM03	User *U2*	User *U3*	User *U4*		User *U1*
	Accuracy: 0.923; mIoU: 0.858	Accuracy: 0.804; mIoU: 0.678	Accuracy: 0.910; mIoU: 0.841	Accuracy: 0.806; mIoU: 0.696	Accuracy: 0.915; mIoU: 0.845
	IoU: 0.860 / IoU: 0.902 / IoU: 0.858 / IoU: 0.706 / IoU: 0.767	IoU: 0.828 / IoU: 0.517 / IoU: 0.632 / IoU: 0.347 / IoU: 0.693	IoU: 0.844 / IoU: 0.926 / IoU: 0.844 / IoU: 0.453 / IoU: 0.791	IoU: 0.692 / IoU: 0.786 / IoU: 0.728 / IoU: 0.365 / IoU: 0.241	IoU: 0.850 / IoU: 0.892 / IoU: 0.857 / IoU: 0.585 / IoU: 0.741

FIGURE 11 The average accuracy and mIoU values and the IoU per class (highlighted with the associated colours) for each experiment. Orthoimages *MAL* and *SMA* have only 4 and 3 coral classes, respectively. The grey-coloured column contains the accuracy results of the fully automatic semantic segmentation

than half of pixels. *Porites*, plating *Montipora*, and *Pocillopora* per-pixel classification show comparable accuracy. The slightly lower accuracy of *Pocillopora* segmentation is likely due to the difficultly of correctly tracing or predicting the jagged perimeters of *Pocillopora* colonies.

The difficulty of correctly identifying encrusting *Montipora* by the human observers is also evident in the agreement values calculated using Cohen's Kappa (κ). These values mostly range from poor (below 0.4) to mediocre (values between 0.4 and 0.6) (see Figure 7 in Supporting Information materials). The remaining classes are labelled with excellent agreement and have Cohen's Kappa values of over 0.8. The semantic segmentation model was trained on a data set labelled by different users from the same laboratory. The uncertainty introduced by discordant human annotations explains the network's poor performance in distinguishing encrusting *Montipora*. However, the average of agreement values (reported in the order of Background, Porites, plating Montipora, encrusting Montipora, and Pocillopora), between Task 1 and Task 2 (0.89, 0.62, 0.62, 0.36, and 0.88) and between Task 1 and Task 4 (0.88, 0.70, 0.66, 0.38, and 0.87), suggests that automatic classification slightly improves the user agreement. Comparing the mIoU difference in the fully automatic annotation with Task 4 quantifies the human effort in refining the classified polygons. The manual editing tools resulted in an accuracy gain of 0.9% (see Figure 11) in the *MAL* label, 1.9% in the *SMA* label, 11.5% in the *VM01* label, and finally 14.9% in the *VM03* label. These improvements demonstrate the advantages of the human-in-the-loop approach when model performance suffers from generalization issues.

TABLE 2 Results in terms of average accuracy and mIoU

(a) Average mIoU per orthoimage	
Orthoimage	***Avg.* mIoU**
MAL	0.90
SMA	0.98
VM01	0.87
VM03	0.80
(b) Average mIoU for each task	
Task	***Avg.* mIoU**
*Task*1	0.91
*Task*2	0.86
*Task*3	0.89
*Task*4	0.89
(c) Average mIoU for each user	
User	***Avg.* mIoU**
U1	0.90
U2	0.90
U3	0.85
U4	0.90
(d) Average mIoU for each class	
Class	***Avg.* mIoU**
Background	0.90
Porites	0.87
Montipora p.	0.85
Montipora enc.	0.41
Pocillopora	0.83

Note: Overall, annotating VM01 and VM03 was more complex, regardless of the task or operator (see panel a). The values in panel b confirm that Taglab ensures a segmentation accuracy comparable with Photoshop. Panel c demonstrates that all users were accurate and produced excellent results relative to the ground truth. According to panel d, encrusting Montipora is harder to identify.

TABLE 3 Evaluation of labelling obtained by voting

Orthoimage	Accuracy	mIoU	Pixels with vote ≥2
MAL	0.97	0.93	97.38%
SMA	0.99	0.99	99.69%
VM01	0.96	0.91	96.20%
VM03	0.94	0.88	94.17%

Note: The class for each pixel corresponds to the most assigned.

The analysis of the *vote maps* reveals the classification reliability and agreement among users. A vote map counts how many times the same label has been assigned by the four annotators for each pixel, and takes the maximum of these votes. Table 3 illustrates that for each orthoimage, between 94% and 98% of pixels have been classified as belonging to the same class by at least three users. *Agreement labels*, that is, the labelling obtained by voting, are built by considering the label that receives the maximum number of votes for each pixel. Figure 12 displays the vote map and the corresponding agreement labels for the *MAL* orthoimage (see the Supporting Information material for the other orthoimages). When comparison per-pixel voting with the ground truth, we find excellent accuracy, and greater than that obtained by single users (see Figure 11 and Table 2). The per-pixel voting maps are derived from different tasks, demonstrating that the annotators produced highly accurate labelling (close to the ground truth) *independently* of the tasks and the tools used. This also demonstrates that the design of the user study does not suffer from any bias. The others voting maps are included in the Supporting Information material. The voting maps of *VM01* and *VM03* confirm that the annotators disagree on entire regions of the encrusting Montipora.

6.2 | Work time analysis and TagLab efficiency

Figure 13 shows each user's registered time in performing each task. The overall time demonstrates that there is no advantage of using TagLab without the automatic or assisted tools when compared to the Photoshop workflow. However, this perceived lack of advantage has several caveats, including the users' unfamiliarity with TagLab. The log files analysis reveals some periods of inactivity (see, e.g., user *U4* in Figure 15). Even if users documented the breaks during the annotation, there are some discrepancies between the reported time spent working and the log files report. However, since such differences are not great (about 10% on average with a maximum of 20% for *U2*), and we have only the declared time for the Photoshop labelling (Task 1), we used the self-declared working times in our evaluations. As demonstrated in Figure 10, some users were more likely to separate colonies, while others were more likely to include several distinct colonies in the same polygon. This decision is typically based on whether a given patch of coral is represented by contiguous live tissue (Bak & Meesters, 1998), a difficult cognitive task even for those with considerable expertise. This effect is clear in the results, as some users produced more accurate label maps but required more time to do so. Figure 14 shows contour tracing per-pixel speed. The introduction of the extreme click paradigm achieves a gain of about 42.6%, while the speed gain obtained by correcting the automatic classification is only 12.1%. Novice users *U1* and *U2*, who were less accustomed to using Photoshop, had significantly greater gains: about 88.2% for assisted segmentation and 27.5% for automatic classification plus corrections. The relatively smaller time savings for experienced users may result from their familiarity with using some Photoshop tools, such as the eraser tool, which are not implemented in TagLab. The smaller overall time savings with the fully automatic segmentation is largely due to the elevated annotation times of user *U1* in *VM03*. The *VM03* map is the most complex orthoimage in terms of both human and automatic recognition. This is confirmed both by the data reported in terms of accuracy (mIoU) in Table 2 and by the agreement between users. The average accuracy of users in classifying *VM03* was 0.80, a significantly lower value than reported in the

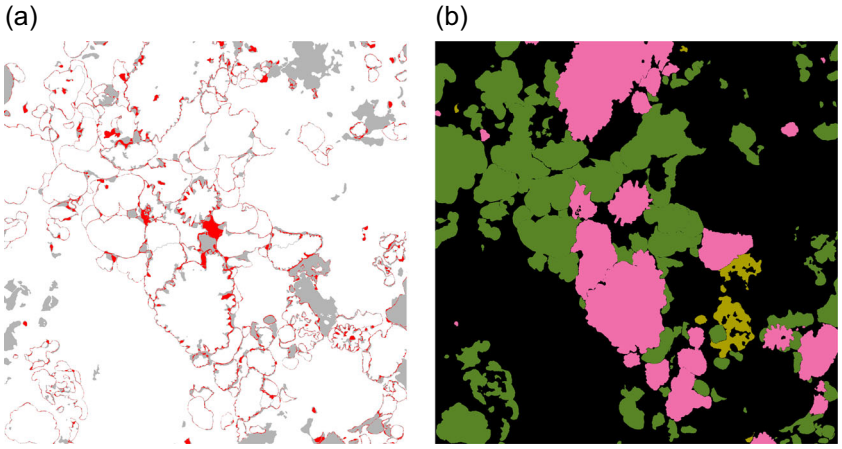

FIGURE 12 Vote map and agreement label for the *MAL* orthoimage. White corresponds to four agreed votes, light grey to three votes, red to two votes, and dark red to one. Almost all the pixels are white or light grey, highlighting the high agreement between users. The pixel labelling produced according to the maximum votes is very close to the ground truth (see also Table 3)

Site	Photoshop Manual Tracing	Taglab Manual Tracing	Taglab Assisted Segmentation	Taglab Editing of Predictions	
MAL	6h 10m	4h 30m	4h 15m	2h 35m	User *U1*
STA	0h 30m	1h 15m	0h 15m	0h 30m	User *U2*
VM01	3h 0m	8h 20m	5h 8m	3h 0m	User *U3*
VM03	8h 0m	6h 45m	5h 15m	8h 45m	User *U4*
Mean Speed:	18h 10m	20h 50m	14h 53m	14h 50m	

FIGURE 13 The table shows the self-reported times each user spent on each task for each map. The final row shows the total time spent by each user on each study. These times are used in the speed calculations, although the log files' analysis reveals several periods of inactivity (lasting up to tens of minutes)

Site	Photoshop Manual Tracing	Taglab Manual Tracing	Taglab Assisted Segmentation	Taglab Editing of Predictions	
MAL	4.97	5.01	6.55	8.16	px / sec
STA	7.52	3.04	13.52	7.16	px / sec
VM01	8.41	4.92	7.31	7.77	px / sec
VM03	5.89	6.23	8.72	6.11	px / sec
Mean Speed:	6.7	4.8	9.03	7.3	px / sec
Speed Gain:		-28.50%	+42.60%	+12.10%	

	Speed Gain Task 3 (%)	Speed Gain Task 4 (%)
User *U1*	47.1	22.9
User *U2*	129.5	31.9
User *U3*	-22.1	-14.9
User *U4*	16	8.5

FIGURE 14 Stated times do not provide a measure of how fast each user produces segmentations, as each drew a different total length of outlines. The per-pixel speed provides a better understanding of the actual improvement introduced by TagLab. The (average) speed gain is evident in Task 3 (42.6%) and less in Task 4 (12.1%). Annotation beginners, *U1* and *U2*, benefitted from assisted annotation far more than the experts. This larger performance improvement for the beginners likely means that their limited Photoshop pipeline experience has made them more adaptable in using TagLab

other maps. The automatic segmentation accuracy of *VM03* was also the lowest of the four maps, (see Figure 11), particularly for the class *Pocillopora*. This is probably because *VM03* contains *Acropora*, a class of corals that are morphologically similar to *Pocillopora* but do not appear in the training data. Note that in the automatic classification, the IoU values per class were (0.69, 0.78, 0.72, 0.36, and 0.24) but were much higher after human correction (0.85, 0.89, 0.85, 0.585, 0.74). The difference between these quantities (0.15, 0.10, 0.12, 0.22, and 0.50) indicated that user *U1* required additional time to modify classes, including 50% of the *Pocillopora* pixels. This reduces the advantages of using TagLab in Task 4. However, the user's ability to manually edit predictions directly led to an improvement in the mIoU from 0.69 to 0.84 for *VM03*, again highlighting the advantages of the human-in-the-loop approach.

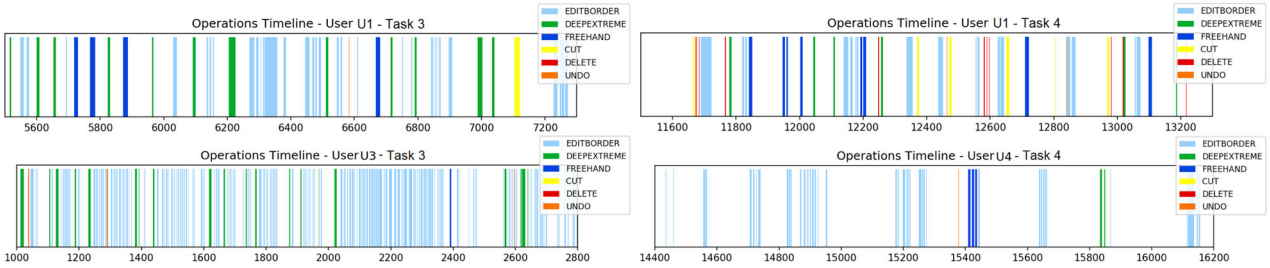

FIGURE 15 Excerpt of the editing operations performed by different users in different annotation tasks. Note that each user has a very different annotation style. Timelines more detailed are available in the Supporting Information material

6.3 | Tools' analysis and limitations

An in-depth analysis of the log files reveals that the cognitive workload for assigning four points is moderate, even when coral shapes are complex. The mean time for this assignment ranges from 4 to 8 s, depending on the user. This is in line with the previous study of Papadopoulos (Papadopoulos et al., 2017), who reported an average time of 7 s to indicate an object's extremes. The processing time varies with the corals' size but is around 3–4 s on the PCs used in this study (each equipped with a GeForce GTX 1080Ti). Our implementation of the *4-clicks* tool does not support the editing of polygons' boundaries, and according to the logged time (Figure 15), manual editing greatly impact the overall efficiency. Figure 15 gives a synopsis of the users' editing operations during different tasks. A row represents 30 min of editing, with the starting time corresponding to a period of intense activity. Different users exhibit different behaviour. User *U1* navigated the orthoimage and evaluated the required manual work for some time before editing, while *U3* performed small continuous adjustments. The mean time required to create a polygon using the *Freehand* tool gradually decreases as users become more familiar with it. For example, *U4* required 23.6 8s on average to create polygons in Task 2, 5.6 s in Task 3, and 4.61 s in Task 4. This trend is confirmed by all users' time analyses, suggesting that users rapidly improve their familiarity with TagLab workflows.

During the user study, we realized that some tools had implementation deficits that made their use impractical. For example, the *Freehand* drawing tool used to delete unclosed curves made it necessary to draw the complete segmentation again. Also, the *Edit Border* tool did not allow the creation and modification of internal holes, which had to be done using the other tools.

6.4 | Overcoming limitations: The new TagLab release

In April 2021, we released a new version of TagLab that solves the problems related to the manual tracing tools (*Freehand* drawing and *Edit Border*), highlighted in Section 6.3, and includes a new AI-based interactive solution that experts can use both for the creation and the editing of complex-shaped labels. Since we received positive feedback for these improvements, we decided to conduct a preliminary experiment to assess their efficiency gain.

The new assisted tracing solution, the *positive/negative clicks* tool, implements a custom version of the CNN presented in Sofiiuk and Ilia Petrov (2021). The *positive/negative clicks* tool enables object segmentation from a sketch by placing a few inner (positive) or outer (negative) points. A single positive point placed in the object's centre is generally sufficient for achieving complete colony outlining. This tool completes the automation of the annotation pipeline, adding an assisted interactive tool also for Step 3 (see Figures 2, 16, and 17).

We evaluated the performance of the new release by repurposing the assisted segmentation task at only two users, the ones who had segmented the two medium-difficulty orthomosaics, *MAL* and *VM01*, in Task 3. We asked them to repeat Task 3 using TagLab with the improved manual tracing tool (Task A) and TagLab with the improved manual tracing plus the new interactive CNN (Task B). After more than 1 year from the first experiment, we can assume there is no repetition bias in Task 3; moreover, these annotators have not done any further labelling work using TagLab. The users demonstrated slightly better performance in Task A, achieving an average annotation speed of 8.25 px/s compared to 6.93 px/s for the previous attempt (see Figure 14). They then achieved an average speed of 13.1;5 px/s in Task B. In Task A, both users conducted fewer manual operations (the number of *Edit Border* operations decreased overall from 2400 to 1570). We assumed that they both would increase their annotation speed using the revised *Freehand* drawing tool, so we introduced a compensation factor to highlight only the speed gain introduced by the new CNN. After taking this into account, the annotation speed achieved by the positive/negative clicks tool was 11.05 px/s, with a gain of about 59% over the TagLab version that includes only the *4-clicks* annotation tool. Thus, the current release halves the annotation time of the manual Photoshop annotations (overall speed gain is about 96% on average). The per-pixel classification accuracy achieved by users in both Tasks A and B is reported in Figure 18. The value remains in line with that previously recorded, with a slight improvement in accuracy scored with the new interactive tool.

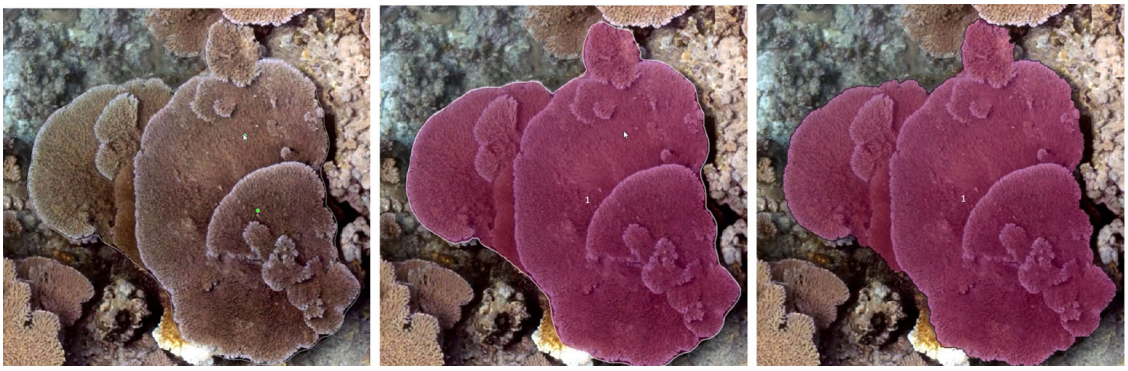

FIGURE 16 Labelling a colony using the *positive/negative clicks* tool. Positive points are drawn in green. Colony outlining placing two positive points (left), the same polygonal shape after the class assignment (center). The use of automatic refinements further improves the segmentation (right)

FIGURE 17 Edit a colony using the *positive/negative clicks* tool. The original form (left). Drawing negative points in red excludes an area. Drawing positive points in green includes an area. Using automatic refinements further improves segmentation (right)

Site	TagLab Assisted Segmentation (Task A)					TagLab Assisted Segmentation (Task B)				
MAL	User *U3*					User *U1*				
	Accuracy: 0.923; mIoU: 0.864					Accuracy: 0.953; mIoU: 0.914				
	IoU: 0.890		IoU: 0.806	IoU: 0.272	IoU: 0.894	IoU: 0.840		IoU: 0.858	IoU: 0.498	IoU: 0.930
VM01	User *U1*					User *U3*				
	Accuracy: 0.892; mIoU: 0.829					Accuracy: 0.909; mIoU: 0.834				
	IoU: 0.832	IoU: 0.858	IoU: 0.841	IoU: 0.058	IoU: 0.724	IoU: 0.828	IoU: 0.859	IoU: 0.855	IoU: 0.334	IoU: 0.721

FIGURE 18 Accuracy of *U1* and *U3* in the new experiments. Task A tests the improved manual drawing tools (*Freehand* drawing and *Edit Border*). Task B tests the new interactive annotation CNN performance (*positive/negative clicks* tool)

7 | CONCLUSIONS AND FUTURE DIRECTIONS

The semantic segmentation of image-based landscapes composed of complex natural shapes demands novel workflows to ensure pixel-level accuracy. Fully automatic models that yield pixel-wise predictions and perfectly generalize on complex data and in complex tasks are beyond the scope of current technologies. Therefore, experts must retain control over the annotation pipeline, which motivates the design of human-centric AI-system solutions that support them and provide a significant speed increase. TagLab fulfills this demand by offering several integrated functionalities that accelerate robust data annotation production.

We tested TagLab's potential on the spatial analysis of coral colonies on reefs, which is a challenging real-world scenario. We found that TagLab successfully sped up the coral colony tracing task, preserving a level of accuracy comparable to that of humans. The results are surprisingly good, if we consider that the performance of TagLab's tools suffered from a lack of previous experience. TagLab's interactive segmentation feature, the *4-clicks* tool, dramatically reduces the human effort in annotating complex object from scratch (Step 1 of Figure 2) without affecting the segmentation accuracy. Our user study indicates that TagLab-assisted segmentation provides an annotation speed gain by about 42% on average (90% for nonexpert annotators). TagLab's interactive annotation efficiency increases by combining the new interactive labelling solution, the *positive/negative clicks* tool, with the *4-clicks*. According to the preliminary results in 6.4, the

assisted annotation with the current Taglab version leads to a total speed gain of +93%, halving the annotation time w.r.t the manual Photoshop-based annotation pipeline.

The Train-Your-Network feature gives scientists access to automatic ad hoc model optimization, which represents a powerful resource for accelerating data extraction from imagery (Step 2 of Figure 2). To enhance the functionality of this feature, in the next releases we will investigate further domain adaptation strategies for improving model generalization. When the automatic classification generalizes properly (as in the case of *MAL*), editing the automatic predictions (Step 3 of Figure 2) almost halves the assisted annotation time and reduces the manual time by two-thirds (for the same level of accuracy). However, the speed-up gain is limited by the great number of manual editing operations when the automatic model poorly generalizes, even if the image processing interactive tool for the boundary adjustment is functional, and particularly with the graphics tablet. The introduction of an instance segmentation network for the coral taxa could further reduce editing times, thus improving Step 3 of the proposed workflow.

TagLab also performs data analysis; some of its functions, such as the *Multitemporal Comparison* tool, proved extremely useful for extracting spatial information from orthoimages (Sandin et al., 2020). However, the current version of TagLab has limitations on the size of the orthoimages handled, which cannot exceed 32000 × 32000 pixels. Therefore, we are evaluating a multiresolution approach for managing larger images. TagLab has already been tested successfully on other application contexts, such as Architectural Heritage (Pavoni, Giuliani, et al., 2020). Here, the fully automatic plus editing annotation strategy (Steps 2 and 3) was found to be extremely efficient.

ACKNOWLEDGMENTS

This study was partially supported by the Italian Minister of University and Research (grant PNRA18_00263-B2, "Ross Sea Benthic Monitoring Program: new non-destructive and machine-learning approaches for the analysis of benthos patterns and dynamics"). We want to thank our annotators Adi Khen, Esmeralda Alcantar, Orion McCarthy, and Mary Liesegang that did an incredible job, and Marco Callieri for his useful suggestions and the time to test the tool. Open access funding provided by Consiglio Nazionale delle Ricerche within the CRUI-CARE Agreement.

ORCID

Gaia Pavoni https://orcid.org/0000-0003-4083-3835
Massimiliano Corsini http://orcid.org/0000-0003-0543-1638
Federico Ponchio https://orcid.org/0000-0002-2974-0577
Alessandro Muntoni https://orcid.org/0000-0001-5502-3582
Clinton Edwards https://orcid.org/0000-0003-4222-0290
Nicole Pedersen https://orcid.org/0000-0003-4332-5561
Stuart Sandin https://orcid.org/0000-0003-1714-4492
Paolo Cignoni http://orcid.org/0000-0002-2686-8567

REFERENCES

Abraham, N., & Khan, N. M. (2019). A novel focal tversky loss function with improved attention u-net for lesion segmentation. In 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019). IEEE (pp. 683–687).

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2010). Slic superpixels.

Acuna, D., Ling, H., Kar, A., & Fidler, S. (2018). Efficient interactive annotation of segmentation datasets with polygon-RNN++. In CVPR.

Agisoft. (2019). Metashape. http://www.agisoft.com/

Alonso, I., Yuval, M., Eyal, G., Treibitz, T., & Murillo, A. C. (2019). Coralseg: Learning coral segmentation from sparse annotations. *Journal of Field Robotics*, *36*(8), 1456–1477.

Andrew K. (2018). DeepSegment. https://andrewking.io/portfolio/deep-segments

Bak, R., & Meesters, E. (1998). Coral population structure: The hidden information of colony size-frequency distributions. *Marine Ecology Progress Series*. *162*, 301–306.

Beijbom, O., Edmunds, P. J., Roelfsema, C., Smith, J., Kline, D. I., Neal, B. P., Dunlap, M. J., Moriarty, V., Fan, T. -Y., Tan, C. -J., Chan, S., Treibitz, T., Gamst, A., Mitchell, B. G., & Kriegman, D. (2015). Towards automated annotation of benthic survey images: Variability of human experts and operational modes of automation. *PLOS One*, *10*(7), 1–22.

Blanchet, J. N. (2016). CoralMe. https://github.com/jnblanchet/CoralMe

Boykov, Y. Y., & Jolly, M. (2001). Interactive graph cuts for optimal boundary region segmentation of objects in N-D images. In Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, (Vol. 1, pp. 105–112).

Castrejón, L., Kundu, K., Urtasun, R., & Fidler, S. (2017). Annotating object instances with a polygon-RNN. In CVPR (pp. 4485–4493).

Chen, L., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. CoRR, abs/1802.02611.

Cohen, L. (2006). Minimal paths and fast marching methods for image analysis. In *Handbook of mathematical models in computer vision*. Springer (pp. 97–111).

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., & Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3213–3223).

Edmunds, P. J., & Riegl, B. (2020). Urgent need for coral demography in a world where corals are disappearing. *Marine Ecology Progress Series*.

Forte, M., Price, B., Cohen, S., Xu, N., & Pitié, F. (2020). Getting to 99% accuracy in interactive segmentation. arXiv preprint arXiv:2003.07932.

Friedman, A. (2017). Squidle. https://squidle.org/content/snippet-about.html

Girshick, R. (2015). Fast R-CNN. In The IEEE International Conference on Computer Vision (ICCV).

Gulshan, V., Rother, C., Criminisi, A., Blake, A., & Zisserman, A. (2010). Geodesic star convexity for interactive image segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. In The IEEE International Conference on Computer Vision (ICCV).

Hoeksema, B., & Cairns, S. (2019). World list of scleractinia. Scleractinia. Accessed through: World Register of Marine Species: http://www.marinespecies.org/aphia.php

Khoreva, A., Benenson, R., Hosang, J., Hein, M., & Schiele, B. (2017). Simple does it: Weakly supervised instance and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Kodera, S. M., Edwards, C., Petrovic, V., Pedersen, N. E., Eynaud, Y., & Sandin, S. A. (2020). Quantifying life history demographics of the

scleractinian coral genus pocillopora at palmyra atoll. *Coral Reefs*, *39*, 1091–1105.

Langenkämper, D., Zurowietz, M., Schoening, T., & Nattkemper, T. W. (2017). Biigle 2.0-browsing and annotating large marine image collections. *Frontiers in Marine Science*, *4*, 83.

Le, H., Mai, L., Price, B., Cohen, S., Jin, H., & Liu, F. (2018). Interactive boundary prediction for object selection. In The European Conference on Computer Vision (ECCV).

Lin, D., Dai, J., Jia, J., He, K., & Sun, J. (2016). Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3159–3167).

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEEconference on Computer Vision and Pattern Recognition (pp. 3431–3440).

Ma, J., & Yarats, D. (2019). Quasi-hyperbolic momentum and adam for deep learning. In International Conference on Learning Representations.

Maninis, K.-K., Caelles, S., Pont-Tuset, J., & Van Gool, L. (2018). Deep extreme cut: From extreme points to object segmentation. In Computer Vision and Pattern Recognition (CVPR).

Papadopoulos, D. P., Uijlings, J. R. R., Keller, F., & Ferrari, V. (2017). Extreme clicking for efficient object annotation. In ICCV (Vol. *2017*, pp. 4940–4949).

Pavoni, G., Corsini, M., Callieri, M., Fiameni, G., Edwards, C., & Cignoni, P. (2020). On improving the training of models for the semantic segmentation of benthic communities from orthographic imagery. *Remote Sensing*, *12*(18).

Pavoni, G., Giuliani, F., Falco, A. D., Corsini, M., Ponchio, F., Callieri, M., & Cignoni, P. (2020). Another brick in the wall: Improving the assisted semantic segmentation of masonry walls. In M. Spagnuolo and F. J. Melero, (eds.), *Eurographics workshop on graphics and cultural heritage*. The Eurographics Association.

Petrovic, V., Vanoni, D., Richter, A., Levy, T., & Kuester, F. (2014). Visualizing high resolution three-dimensional and two-dimensional data of cultural heritage sites. *Mediterranean Archaeology and Archaeometry*, *14*, 93–100.

Pont-Tuset, J., & Van Gool, L. (2015). Boosting object proposals: From pascal to COCO. In The IEEE International Conference on Computer Vision (ICCV).

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention. Springer, 234–241.

Rother, C., Kolmogorov, V., & Blake, A. (2004). Grabcut -interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*.

Sandin, S., Edwards, C., Pedersen, N., Petrovic, V., Pavoni, G., Alcantar, E. A., Chancellor, K. S., Fox, M., Stallings, B., Sullivan, C. J., Rotjan, R., Ponchio, F., & Zgliczynski, B. (2020). Considering the rates of growth in two taxa of coral across pacific islands. *Advances in Marine Biology*, *87*(1), 167–191.

Schoening, T., Osterloff, J., & Nattkemper, T. W. (2016). Recomia-recommendations for marine image annotation: Lessons learned and future directions. *Frontiers in Marine Science*, *3*, 59.

Sofiiuk, K., & IliaPetrov, A. K. (2021). Reviving iterative training with mask guidance for interactive segmentation. arXiv preprint arXiv:2102.06583.

Xu, N., Price, B., Cohen, S., Yang, J., & Huang, T. (2017). Deep grabcut for object selection. arXiv preprint arXiv:1707.00243.

Xu, N., Price, B., Cohen, S., Yang, J., & Huang, T. S. (2016). Deep interactive object selection. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Yuval, M., Alonso, I., Eyal, G., Tchernov, D., Loya, Y., Murillo, A. C., & Treibitz, T. (2021). Repeatable semantic reef-mapping through photogrammetry and label-augmentation. *Remote Sensing*, *13*, 4.

Zurowietz, M., Langenkämper, D., Hosking, B., Ruhl, H. A., & Nattkemper, T. W. (2018). Maia-a machine learning assisted image annotation method for environmental monitoring and exploration. *PLOS One*, *13*(11), 1–18.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher's website.

How to cite this article: Pavoni, G., Corsini, M., Ponchio, F., Muntoni, A., Edwards, C., Pedersen, N., Sandin, S., & Cignoni, P. (2022). TagLab: AI-assisted annotation for the fast and accurate semantic segmentation of coral reef orthoimages. *Journal of Field Robotics*, 39, 246–262. https://doi.org/10.1002/rob.22049