Interactive Multi-Class Tiny-Object Detection (Supplementary Materials)

Chunggi Lee Seonwook Park Heon Song Jeongun Ryu
Sanghoon Kim Haejoon Kim Sérgio Pereira Donggeun Yoo
Lunit Inc.
{cglee, spark, heon.song, rjw0205, seiker, oceanjoon, sergio, dgyoo}@lunit.io

Overview. This supplementary material includes further information about the multi-class tiny-object datasets we evaluate on in the main paper, including class and patch statistics and images. We also provide additional implementation details of the approach described in the main paper. Finally, we share visualizations of annotations acquired from the user-study, and further discuss annotation quality in terms of mAP.

In addition to this document, we provide a demo video of the annotation tool that we implemented for our user study. This annotation tool is capable of providing real-time feedback for interactive multi-class tiny-object detection, as demonstrated in the video.

1. Datasets

Tiny-DOTA We supplement the details regarding the Tiny-DOTA dataset by providing in Tab. 1, information regarding the number of patches and the number of objects of each object class, for the subsets: train, validation, and test. The conditions listed at https://captain-whu.github.io/DOTA/dataset.html stipulate that the annotations of the DOTA dataset (and in extension, the Tiny-DOTA dataset) are available for academic purposes only, with commercial use being prohibited.

LCell The LCell dataset is composed of patches taken from 688 whole-slide images of breast cancer biopsies[1]. The patches were labeled by annotating breast cancer histopathology images for 8 cell classes. The 8 cell classes annotated in our LCell include: lymphoplasma (LC), fibroblast (Fi), macrophage (Ma), nuclear grade 1 (NG1), nuclear grade 2 (NG2), nuclear grade 3 (NG3), necrotic tumor (NTC), and endothelial cell (EC). These classes are chosen by expert pathologists according to cells that are common in breast cancer histology. Tab. 2 states the number of patches, number of slides[2], and the number of cells from each individual cell class. We visualize some cell images along with their ground-truth bounding boxes in Fig. 1. Each bounding box represents a cell nucleus and the color denotes the annotated class of the cell.

[1]Due to the existing agreements regarding the whole-slide image data, we are unable to open-source this dataset, and it is therefore proprietary.

[2]We select fixed-size patches from several slides (whole-slide images), to use in training and validating models for cell detection.

	Train	Val	Test
Num. patches	11198	1692	2823
Num. objs in total	431056	72483	99766
Num. Plane	15161	2775	4236
Num. Bridge	3908	671	673
Num. SV	271252	41994	64612
Num. LV	46956	5412	8159
Num. Ship	75835	18155	17723
Num. ST	12954	2534	3145
Num. SP	3996	686	1096
Num. HELO	994	256	122

Table 1. **Further statistics on Tiny-DOTA.** The abbreviated classes are: *small-vehicle* (SV), *large-vehicle* (LV), *storage-tank* (ST), *swimming-pool* (SP), and *helicopter* (HELO).

	Train	Val	Test
Num. patches	3423	234	821
Num. slides	419	182	87
Num. cell in total	271434	19184	81745
Num. LC	92973	7630	18708
Num. Fi	43838	3128	11935
Num. Ma	6414	365	1801
Num. NG1	15496	1133	6066
Num. NG2	61024	4008	29525
Num. NG3	17222	867	5436
Num. NTC	17168	1052	4938
Num. EC	17299	1001	3336

Table 2. **Further statistics on LCell.** The annnotated classes are: lymphoplasma (LC), fibroblast (Fi), macrophage (Ma), nuclear grade 1 (NG1), nuclear grade 2 (NG2), nuclear grade 3 (NG3), necrotic tumor (NTC), and endothelial cell (EC).

2. Model Implementation Details

Feature Pyramid Network (FPN) Our approach and baseline models are trained with a ResNet-50 based Feature Pyramid Network (FPN [2]) as the CNN feature extractor (backbone) for Faster R-CNN [4] and RetinaNet [3], as mentioned in the *"Experimental Results"* section. In a nutshell, the FPN is a top-down architecture with lateral connections which uses multi-scale features better in detecting objects of various scales. For each stage in the ResNet-based backbone, we forward the outputs through the FPN at the corresponding feature-map scale. The number of output channels at each feature-map scale is 256 and the number of pyramid levels is 5.

Late Fusion (LF). Our CNN-based feature extractor (backbone) for processing user-input heatmaps is a ResNet-18. Unlike in the case of the main task network (where we freeze the ImageNet-pretrained parameters up to the 1st ResNet stage), we do not freeze any layers of the LF module. This is because user input heatmaps are very different in characteristic and number of channels compared to natural RGB images. We also apply an FPN to the LF module, later concatenating features (256 features each) at matching feature-map scales from the main feature extractor. The 512 feature maps ($2 \times 256 = 512$) are passed through a 1×1 convolution layer to produce 256 channels before using as an input to the RPN (in the case of Faster R-CNN), or the classification and box regression subnets (in the case of RetinaNet).

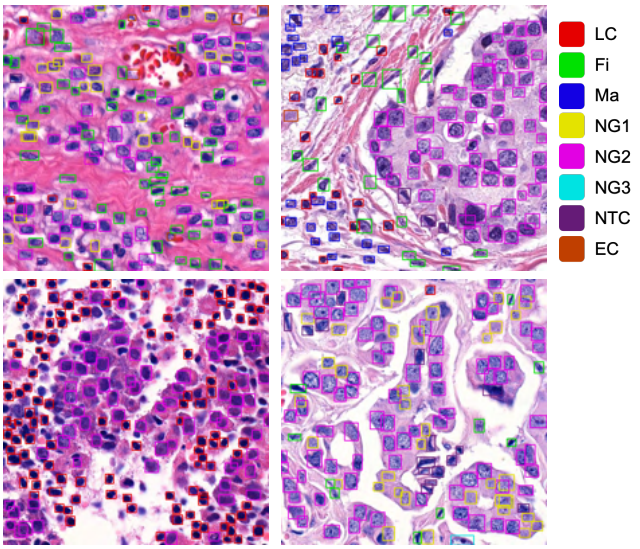


Figure 1. **Example images and annotations of LCell.** Each patch typically contains a large number of objects (cells), which may be challenging to distinguish. The annotation of such images require expert pathologists, who can benefit from an interactive annotation method that aids them in annotating many classes and objects from a few provided clicks. C3Det promises to be such a method.

Training Configuration for Tiny-DOTA. We train Faster R-CNN and RetinaNet on Tiny-DOTA for 24 and 36 epochs, respectively. We start from a learning rate of 0.01 (after a 500-step warmup) and scale it by 0.1x at 16 and 22 epochs (for Faster R-CNN), and 24 and 33 epochs (for RetinaNet), respectively.

Training Configuration for LCell. On LCell, we train Faster R-CNN for 100 epochs. Training starts from a learning rate of 0.01 (after a 500-step warmup) and the is scaled by 0.1x at 30 and 60 epochs.

3. Interactive detection on LCell

The LCell dataset is a particularly challenging dataset to annotate, requiring expertise and a large number of annotations per patch. An effective interactive annotation method can easily reduce the necessary mouse clicks per patch from hundreds (no. of objects times 4) to just a few clicks. We demonstrate that C3Det can achieve this in Fig. 3.

Fig. 3 illustrates 3 example cases from the LCell dataset, each with an increasing number of user inputs (described as clicks). Even with a few given user clicks, object classes that were not explicitly provided in the user input are detected. With more user inputs (up to 6 or 8), we find that the ground-truth can almost be reproduced.

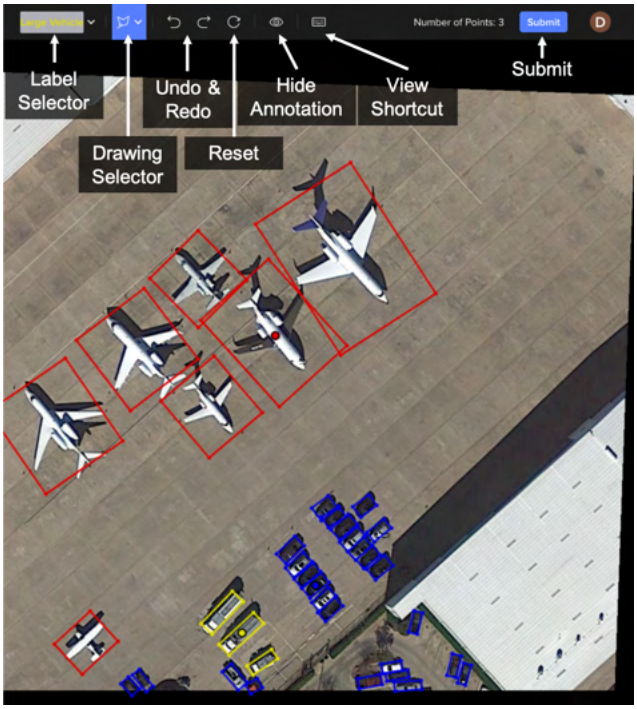


Figure 2. **User Study Frontend.** Our user-study GUI in semi-automatic annotation mode. Annotator inputs are shown as dots, while model predicted bounding boxes are drawn as quadrilaterals.

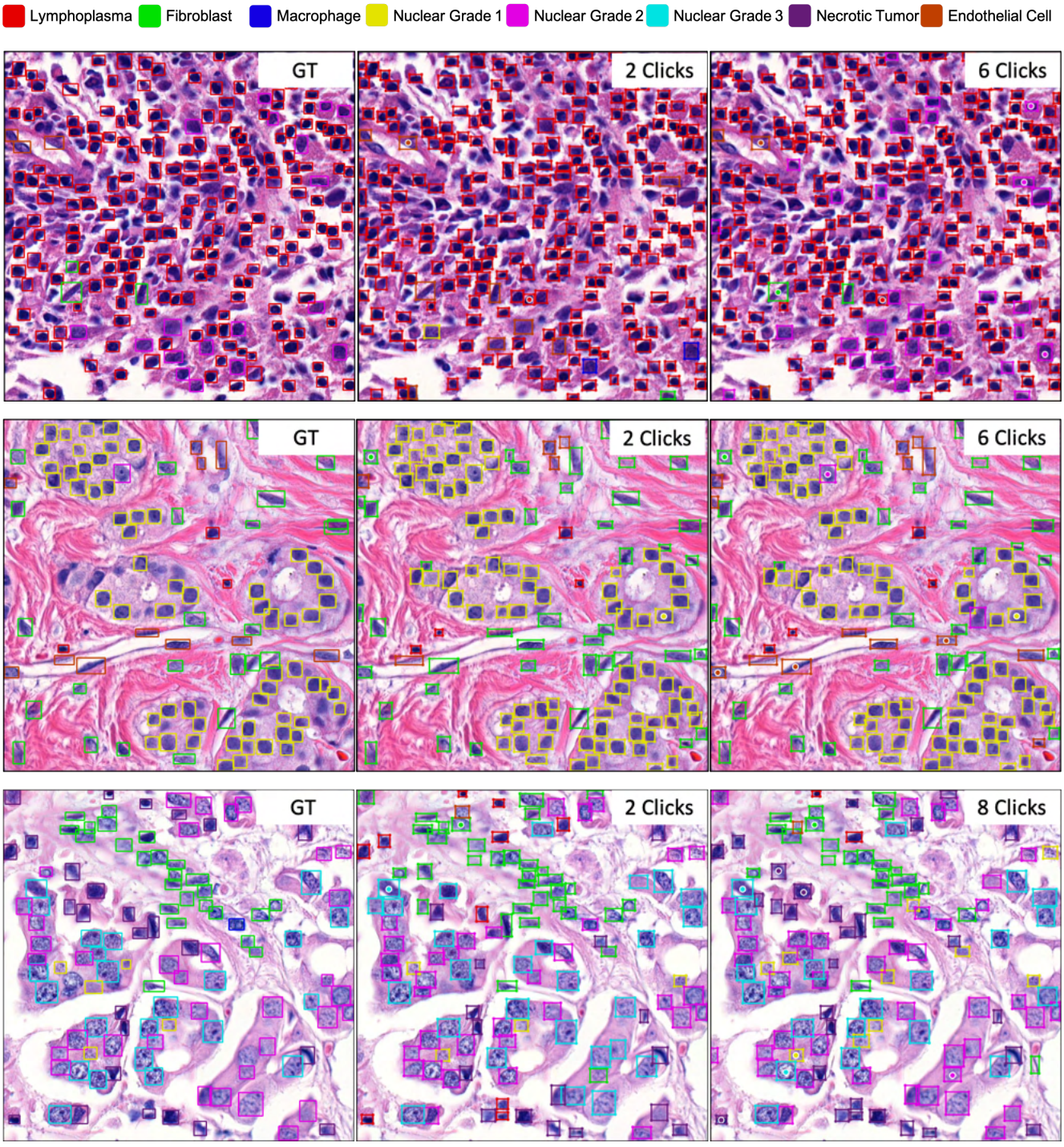


Figure 3. **Visualizing Interactive Detection on LCell.** Example images of LCell with several prediction results by giving user inputs and ground-truth. The boxes and dots represents final annotated objects and user inputs from real annotators.

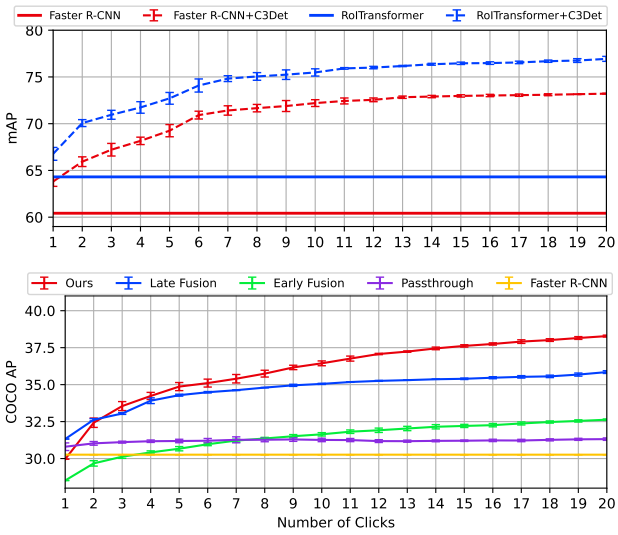


Figure 4. Performance on Tiny-DOTA. **Top:** Both Faster R-CNN and RoI Transformer [1] (solid lines) are improved by adding C3Det (dashed lines). **Bottom:** Our method against Faster R-CNN baselines measured with COCO AP@[.50:.05:.95].

In addition, we find that objects from the same class that are far away from the click position are detected. As argued in the main paper, we believe that C3Det considers local (nearby objects) and global context (far away objects) in relation to the input image and user inputs well, and address the multi-class tiny-object detection setting appropriately.

4. Additional Experimental Results

Comparison to another Tiny Object Baseline (RoI Transformer). Faster R-CNN and RetinaNet are standard baselines for detecting oriented bounding boxes on the DOTA dataset and are adopted in many recent papers. Therefore the results reported in the main paper suggest that our C3Det can apply to variations of these standard networks. Here, we demonstrate that this is indeed the case by selecting RoI Transformer [1], a recent detection method designed to perform well on DOTA. Adding C3Det to the purpose-built RoI Transformer still results in a significant and consistent increase in performance on Tiny-DOTA (see Fig. 4, top). This further demonstrates the value of our proposed approach.

Performance on COCO AP metric. An IoU threshold of 0.5 is the standard procedure for evaluating on the DOTA dataset and thus we follow it in our experiments. We believe, however, that it is also meaningful to compute the COCO AP metric, which takes into account objects of varying scales and therefore show this in Fig. 4 (bottom). In comparison to Fig. 5a in main paper, we find that similar trends can be seen, though the numeric values are decreased due to the difficulty of the tiny objects task at a high IoU threshold.

5. User Study

We present a few more results and visualizations from our User Study. In our user study, we asked participants to annotate images from the Tiny-DOTA dataset, using a fully-manual (*Manual* condition) or semi-automatic (*C3Det + Manual* condition) approach.

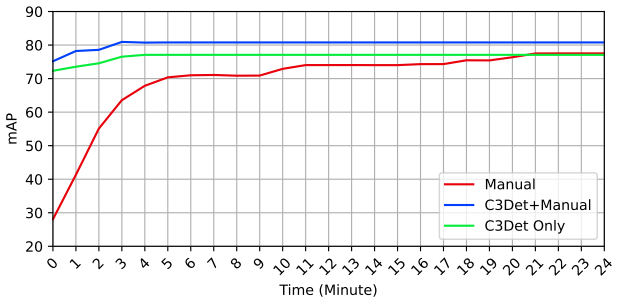
At the beginning of each user study session, we asked for consent from the participant for their participation as well as the storing of their annotation and mouse clicks. As no personally identifiable information was collected in our user study, we do not require approval from an institutional review board (IRB).

Implementation Details. We describe here the server specification and library used for implementing and serving the annotation tool used in our user study.

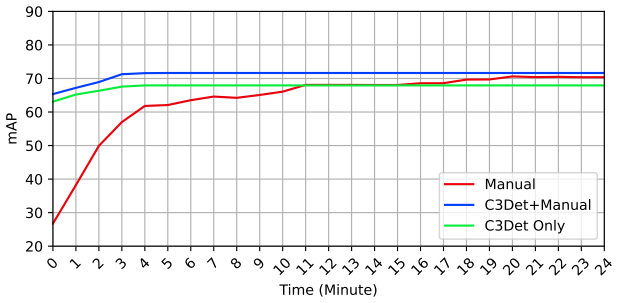
Our user-study GUI (see Fig. 2) is implemented using several libraries such as FastAPI [3] for the back-end and React [4], Redux-Saga [5], and TypeScript [6] for the front-end. The images (to-be-annotated) are drawn on an HTML5 canvas using OpenSeadragon [7] for convenient zooming and padding. Likewise, user-inputs (points) and annotations (bounding boxes) are drawn using basic canvas methods. Model inference via PyTorch takes only a few seconds (on a Titan X (Pascal) GPU) and we further show this real-time capability in a supplementary demo video.

Comparison of annotated images on Tiny-DOTA. Fig. 6 shows example images with ground-truth annotations as well as annotations acquired by our user study conditions: fully-manual (*Manual* condition) or semi-automatic (*C3Det + Manual* condition). Compared to the ground-truth, the *Manual* condition and *C3Det + Manual* condition achieve good quality, with small objects being annotated well. However, there are few difference between them due to confusing object (in terms of object class), misconception of class definition and overly small objects. Fig. 6a and Fig. 6b are compelling example of frustrating objects (broken *plane* and *helicopter*) and misunderstanding of classes (*small-vehicle* confused as *large-vehicle*). On the other hand, Fig. 6c has many *small-vehicle* and *ship* objects in the bottom-right and top-left part of the image, respectively. Although the original DOTA dataset does not have annotations for those very tiny objects (does not exist in the ground-truth), our annotator labeled these as small-vehicle objects (C3Det + Manual). In some manner, our semi-automatic approach may be reducing required effort, and allowing for more rich annotations to be produced.

[3] https://fastapi.tiangolo.com/
[4] https://reactjs.org/
[5] https://redux-saga.js.org
[6] https://www.typescriptlang.org
[7] https://openseadragon.github.io/



(a) IoU threshold for mAP = 0.1



(b) IoU threshold for mAP = 0.5

Figure 5. **Annotation quality (mAP) versus annotation cost (time).** for different annotation schemes in the user study. Lowering the IoU threshold (**top**) for calculating mAP allows more loosely drawn bounding boxes to become valid annotations as well, compared to the results shown in the main paper (**bottom**). In fact, the *C3Det + Manual* condition produces better annotations overall than the *Manual* condition. We believe that this may partly be due to the novice-level expertise of our annotators.

Further Evaluation of Annotation quality (mAP). A typical assessment of the accuracy of bounding boxes is via the calculation of the mAP metric, with true-positives being assessed based on an intersection-over-union (IoU) threshold of 0.5 between a prediction and corresponding ground-truth box. We therefore evaluated the annotation quality yielded by the different conditions in our user study using an IoU threshold of 0.5.

However, our user study participants are novices, and with the added complexity of drawing (often) very small bounding boxes using a computer mouse, we find that the acquired annotations were not always sufficiently covering the tiny objects (in particular, classes such as *small-vehicle* suffered from this issue). We therefore evaluate the mAP of acquired annotations with an IoU threshold of 0.1 and report it in Fig. 5 (top).

In line with our observation, we find that the mAP increases for all annotation conditions. In particular, our *C3Det + Manual* is able to yield better annotations overall than the *Manual* or *C3Det Only* conditions. This eludes to two possibilities: (a) C3Det can guide novice annotators to produce better quality annotations, and (b) allowing manual edits on top of C3Det outputs allows for an even higher final annotation quality.

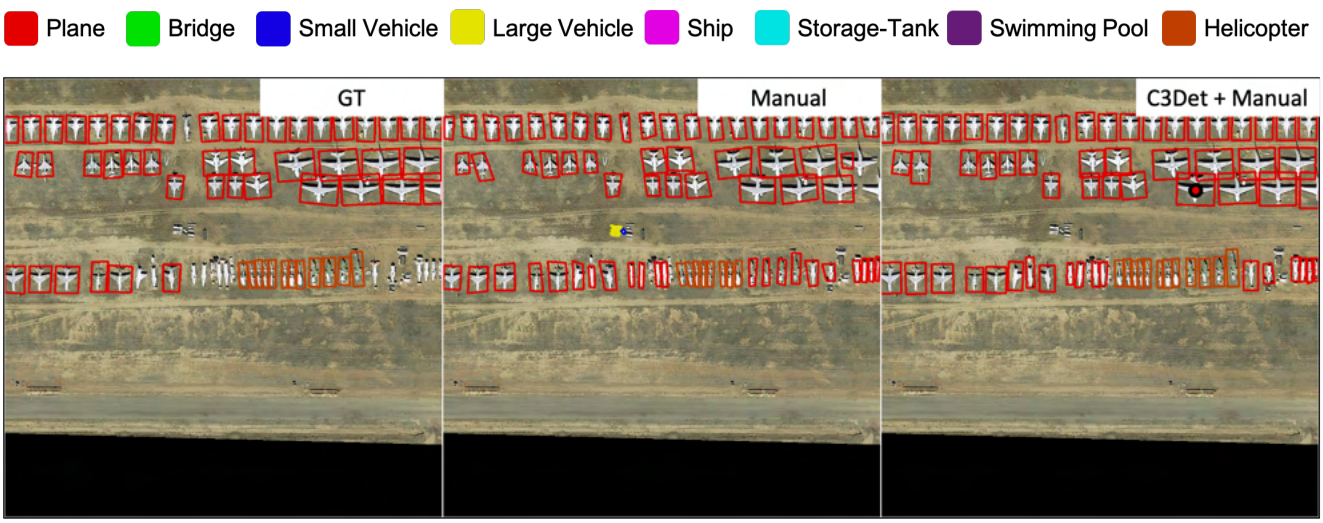
References

[1] Jian Ding, Nan Xue, Yang Long, Gui-Song Xia, and Qikai Lu. Learning RoI Transformer for Detecting Oriented Objects in Aerial Images. In *CVPR*, 2019. 4

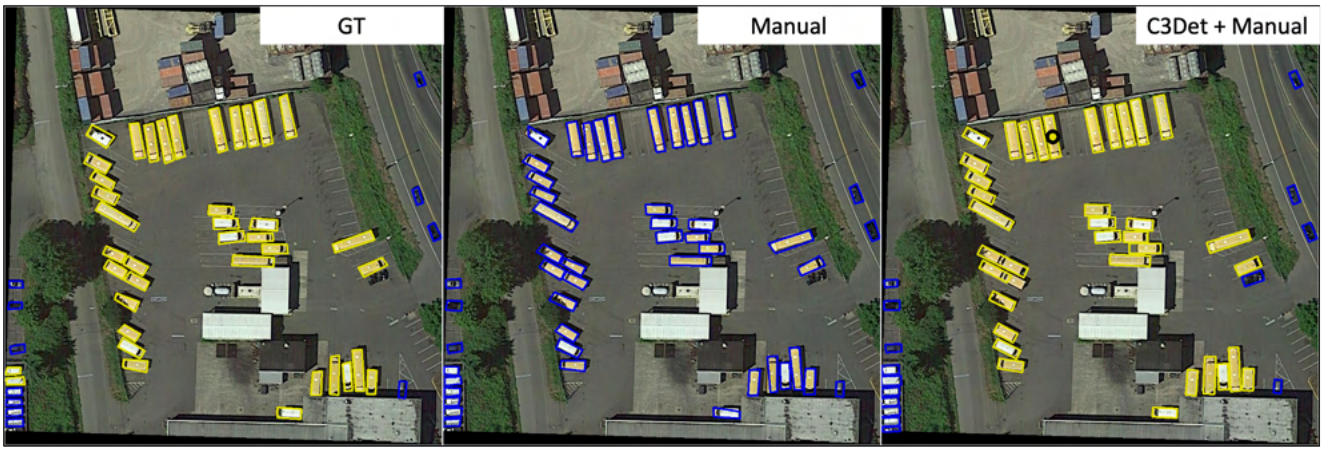
[2] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 2

[3] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 2

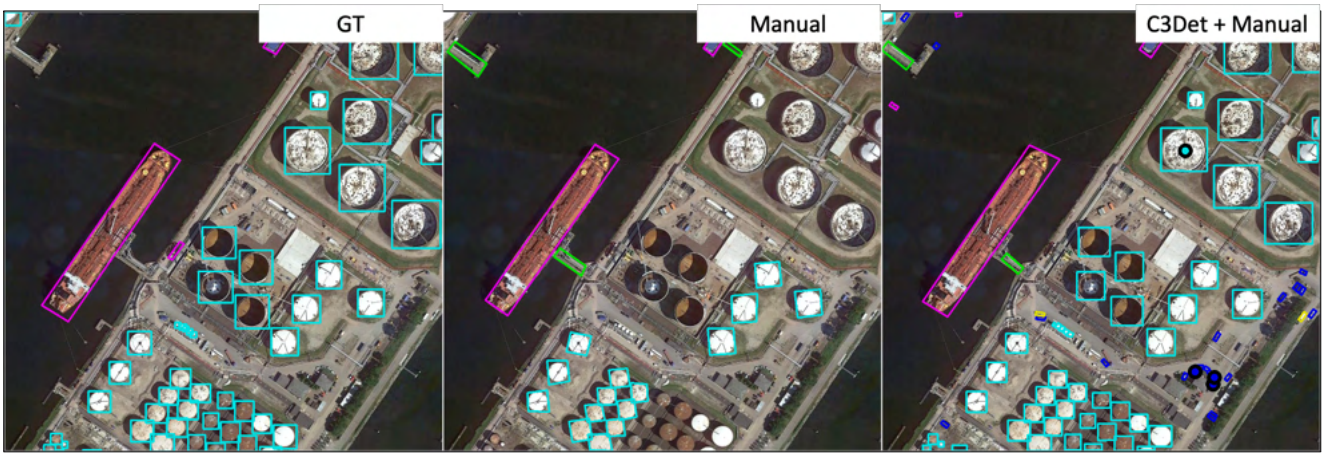
[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *TPAMI*, 39(6):1137–1149, 2016. 2



(a) In some cases, the ground-truth (GT) omitted some objects, which our user study conditions captured.



(b) Our novice annotators can make critical mistakes (mislabeling *large-vehicle* objects as *small-vehicle*), which can be corrected by C3Det.



(c) C3Det can allow for better completion in the case where annotators are unsure about certain objects, or do not sufficiently zoom in to annotate very tiny objects.

Figure 6. **Samples from the User Study.** Example images of Tiny-DOTA with ground-truth, manual, and C3Det + manual with annotations. The boxes and dots represents final annotated objects and user inputs from real annotators. To visualize the user inputs, we draw larger dots compared to the actual User Study GUI.