What is this project’s target

- Determine the precise coordinates of the 25 intersection points forming the grid within the central large square of the counting chamber, as depicted in the microscope photograph.
- Determine the nanometer-per-pixel ratio by comparing the precise pixel coordinates of a triangular configuration of three grid corners with their corresponding known dimensions in nanometers.
- Utilizing the precise coordinates of the 25 grid intersection points and the mask image delineating the segmented yeast areas, the enumeration of yeast cells within each of the 16 small squares was performed.

Chamber used in this project: Neubauer improved glass cell counting chamber silver-plated (bright line) MARIENFELD Germany 0610030

About Neubauer Chamber

Neubauer-improved

The Neubauer-improved counting chamber’s standard depth is 0.1 mm. The grid consists of 3 x 3 large squares with areas of 1 mm² each. The large square in the center is subdivided into 5 x 5 group squares with edges of 0.2 mm length each. These group squares are again subdivided into sixteen small squares of an area of each 0.05 mm x 0.05 mm = 0.0025 mm².

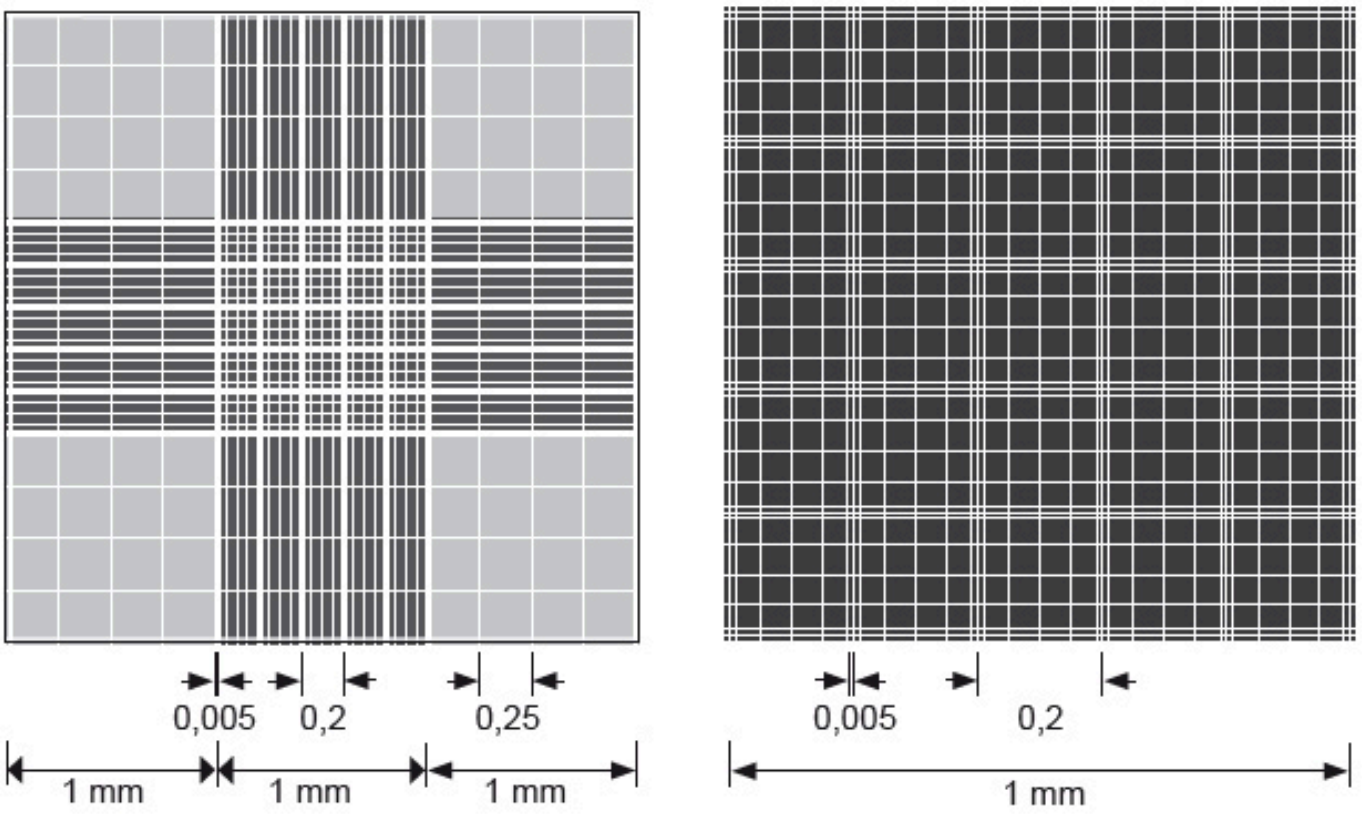
The lines limiting the large squares and the group squares are threefold with the central line as the actual dimension lines. The inner and outer auxiliary lines facilitate counting. They assist determining whether cells near or on the border lines are to be counted as within the area or omitted as outside of the counting area.

As the counting chamber comes with squares of different sizes it can be used for counting different types of cells. E.g. leucocytes are counted in the 4 large squares at the corners of the grid and for counting erythrocytes at least 5 group squares are normally used.

Bright line:

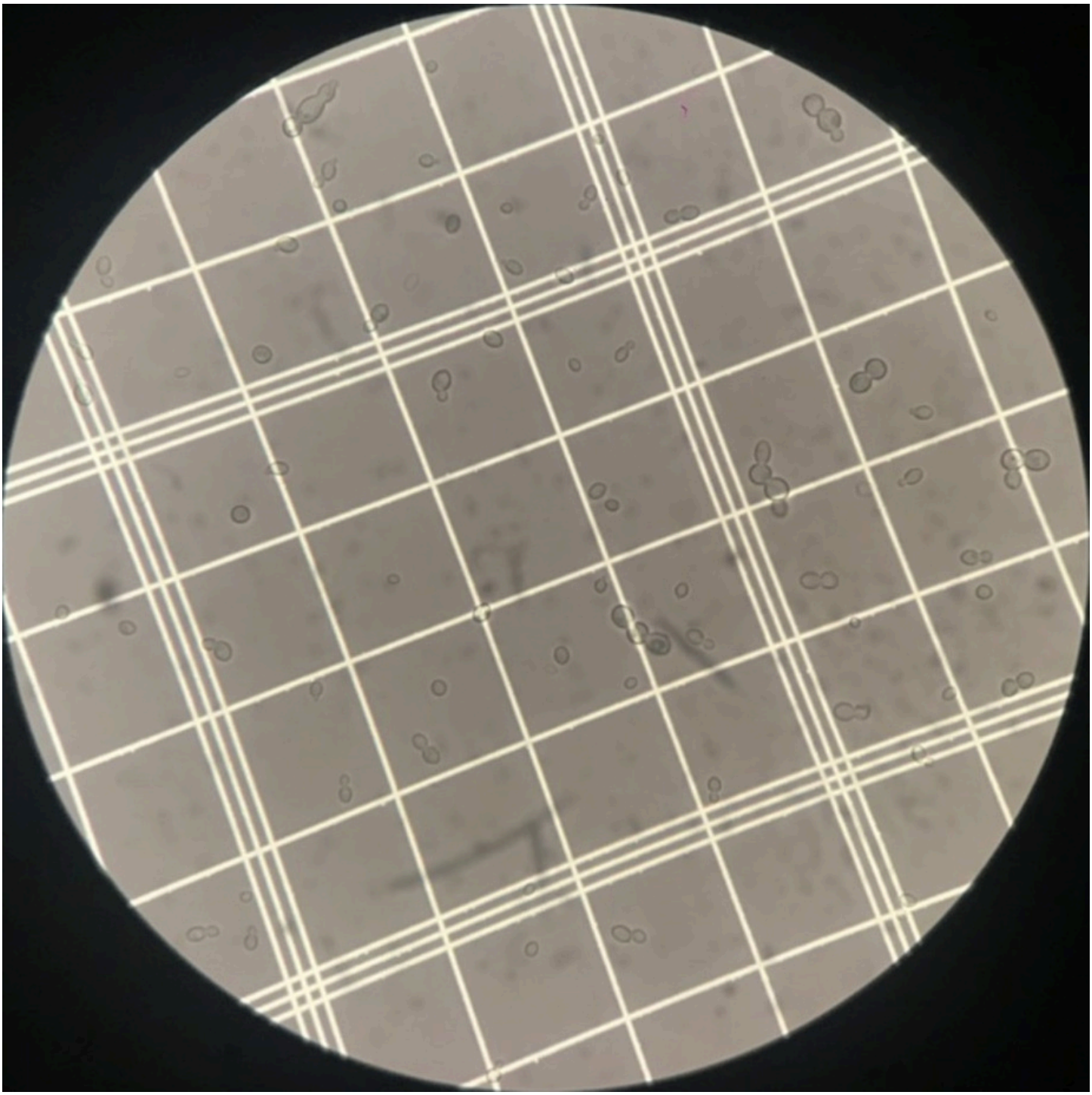
Counting chambers with bright lines have counting grids which are structured into a very thin, transparent metal coating. The bright lines contrast well with the dark metallic background and this facilitates evaluating cell suspensions.

Neubauer-improved with bright lines

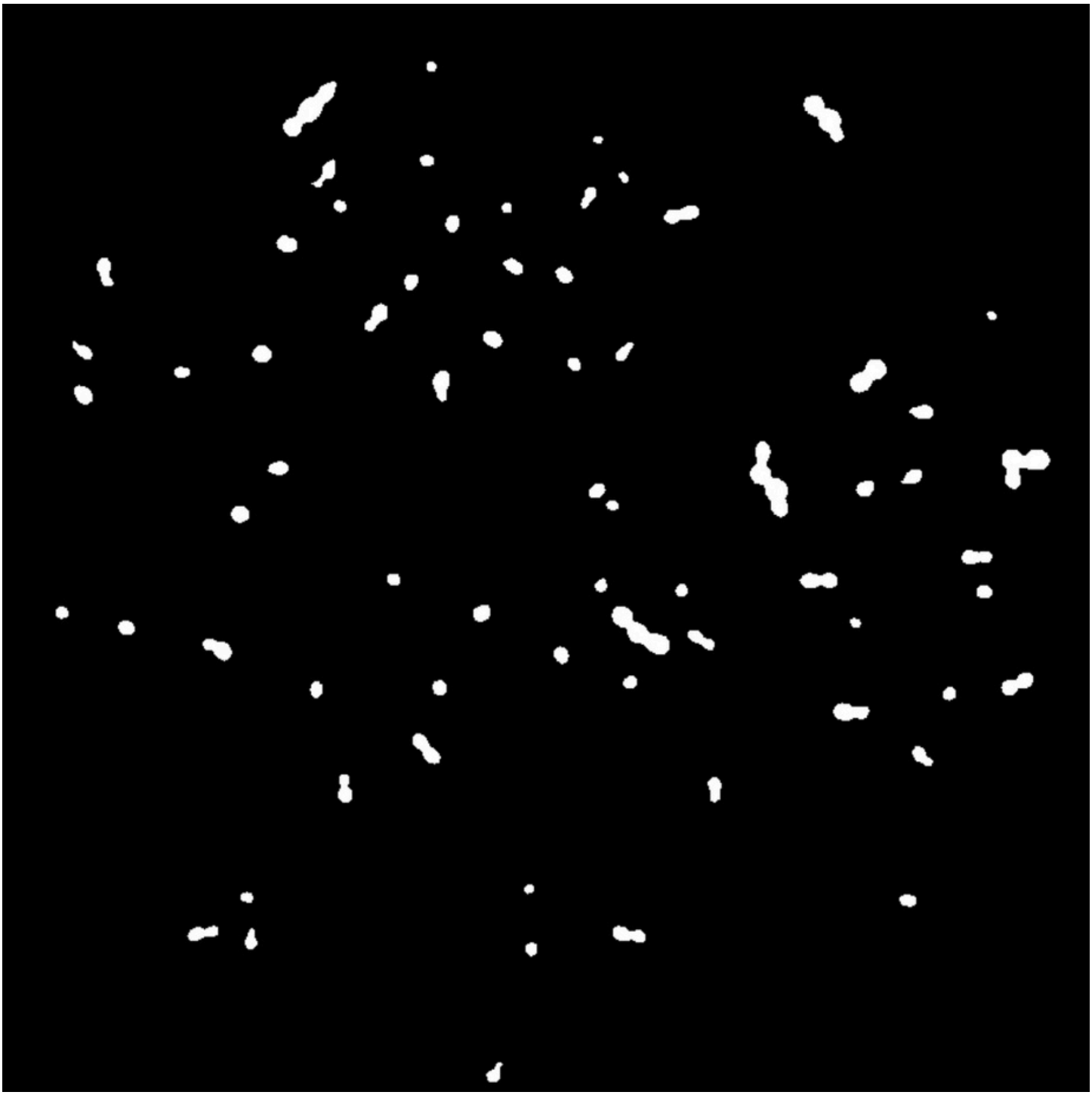


INPUT IMAGE: both at 1024x1024 resolution

origin image



mask image



OUTPUTS:

The precise coordinates of the 25 intersection points forming the grid within the central large square of the counting chamber

```
((123, 430), (237, 388), (362, 340), (485, 295), (595, 254), (166, 541), (278, 498), (403, 450), (528, 405)
```

The number of yeast cells within each of the 16 small squares

```
(3, 3, 5, 2, 4, 5, 1, 1, 2, 2, 2, 3, 1, 4, 1, 1)
```

The nanometer-per-pixel ratio

```
nanometer/pixel 377.38267392590916
```

REFERENCES

https://www.marienfeld-superior.com/counting-chambers-2817.html

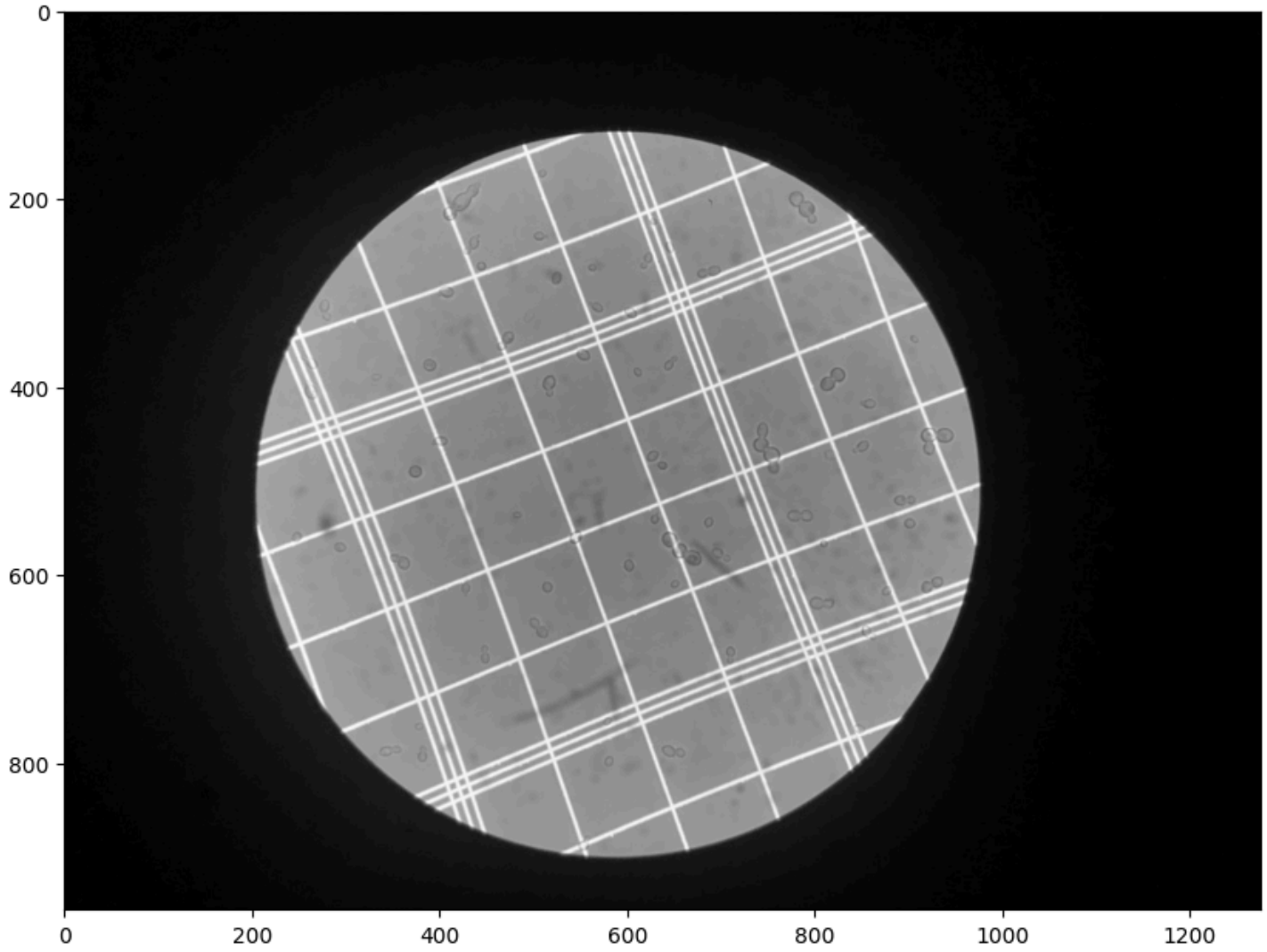
https://www.marienfeld-superior.com/counting-grids.html

VERSION 1

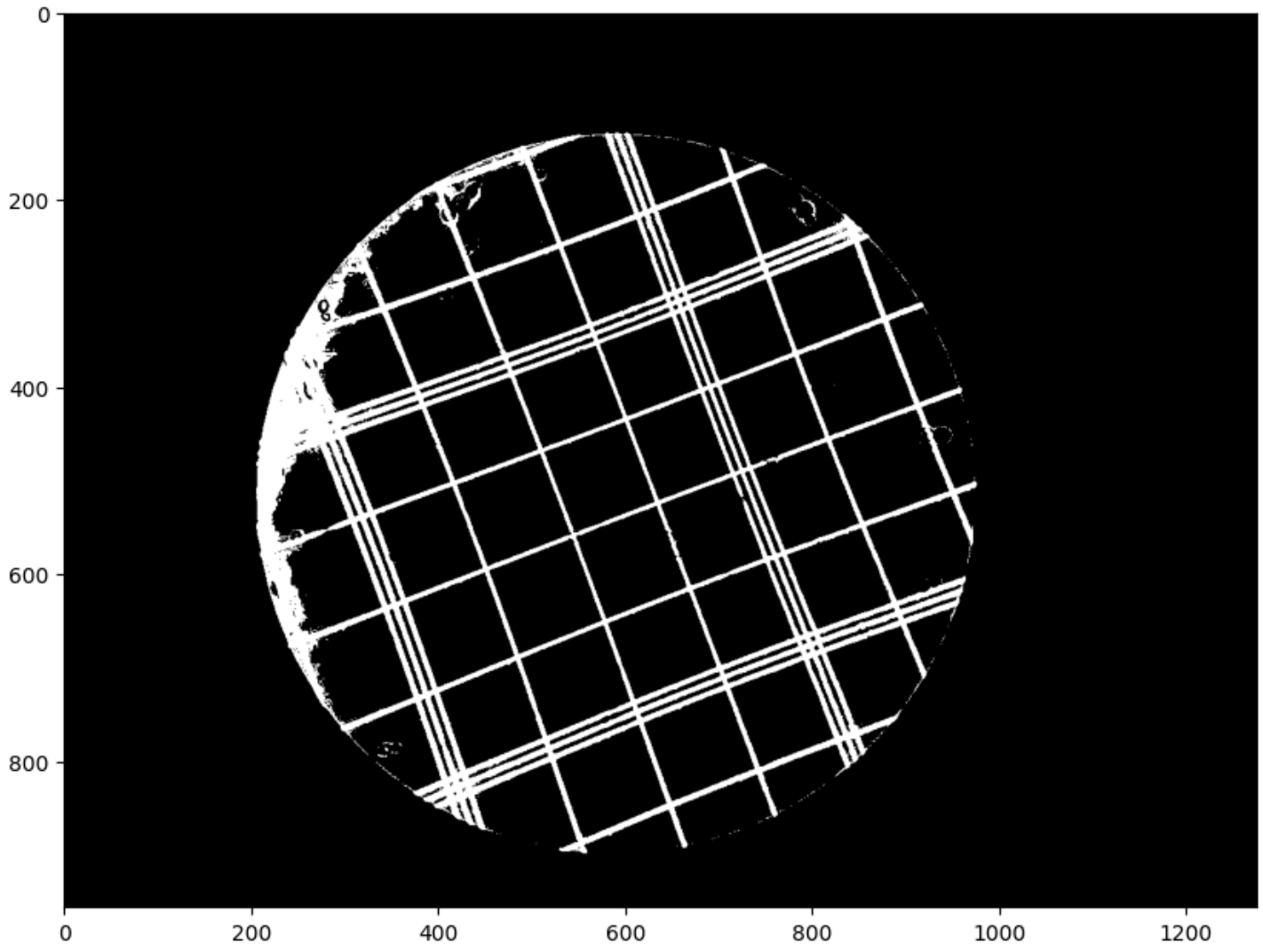
Determine the precise coordinates of the 25 intersection points forming the grid within the central large square of the counting chamber, as depicted in the microscope photograph.

Find suitable threshold and turn input image from gray scale to binary image

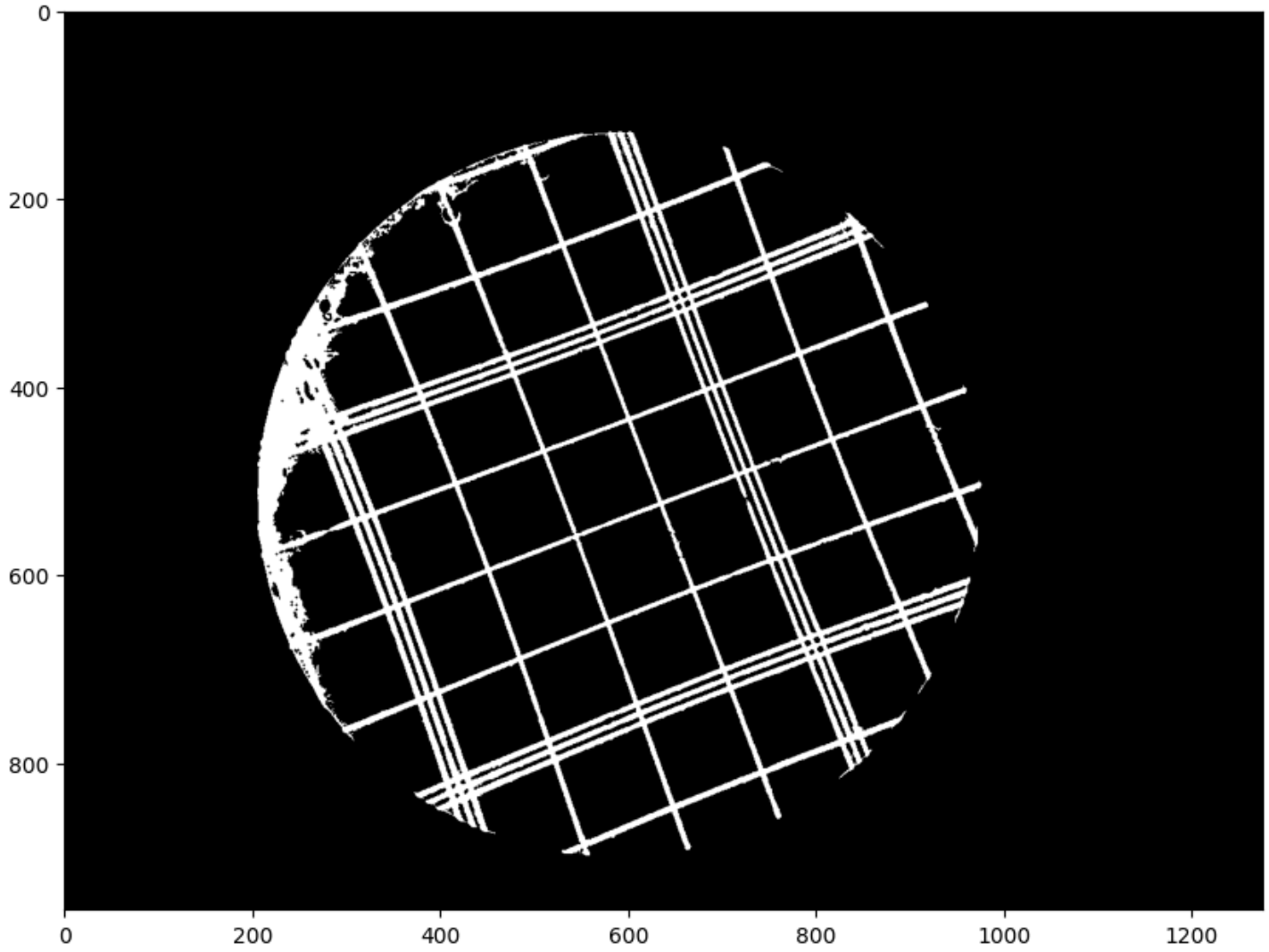
Gray image



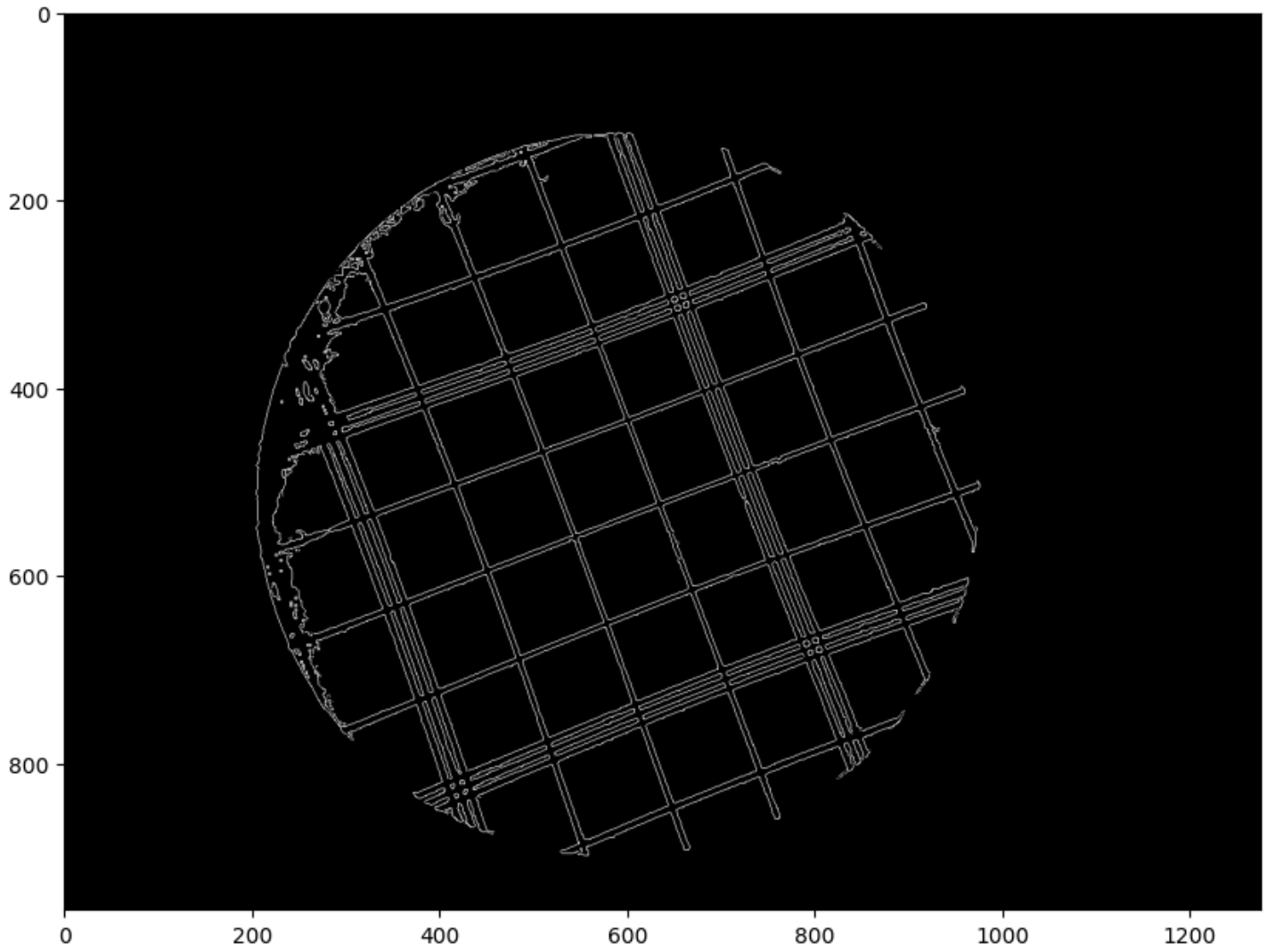
Binary image



Clear small object in the image:

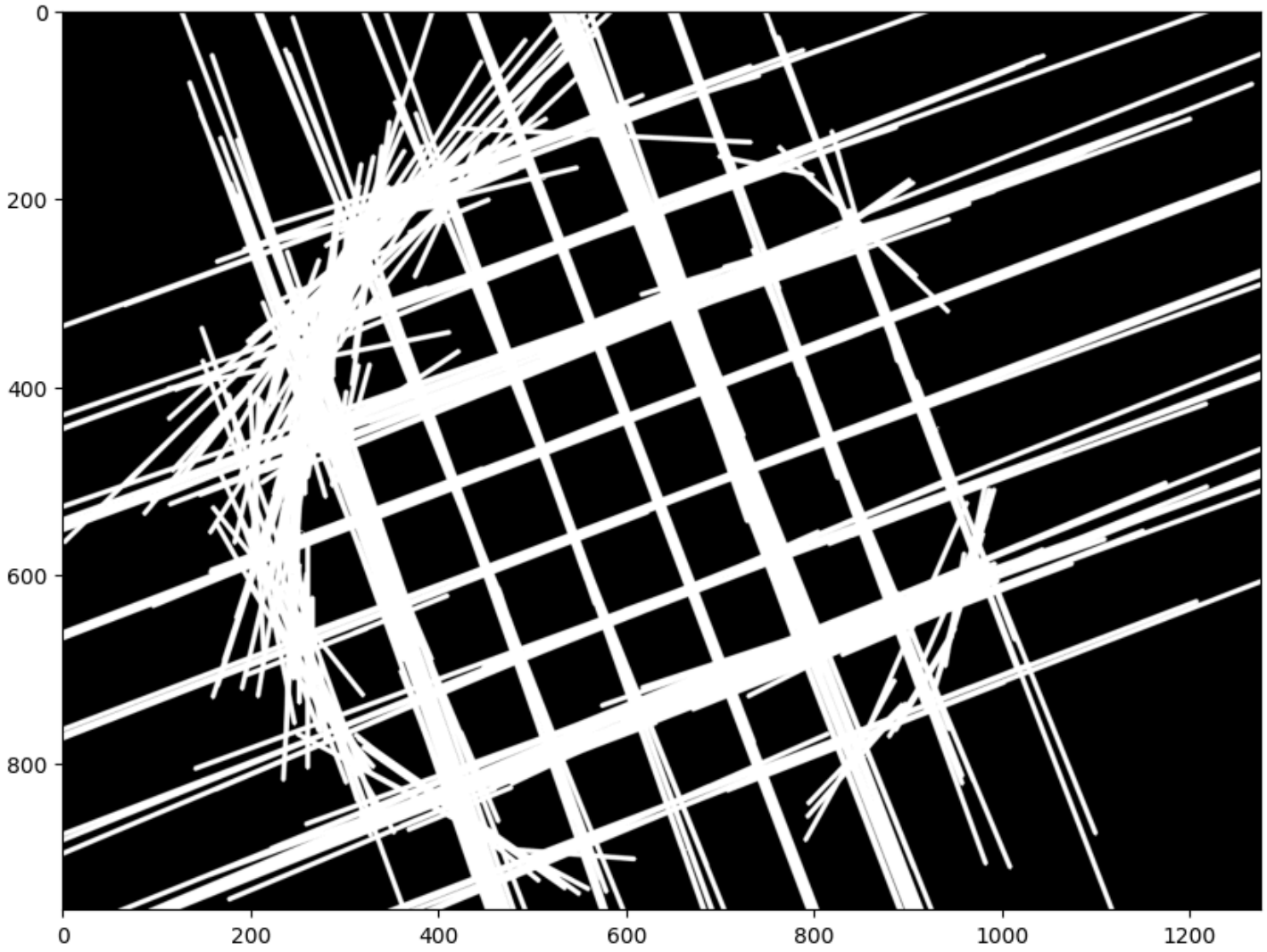


Using Canny algorithm for finding edges:



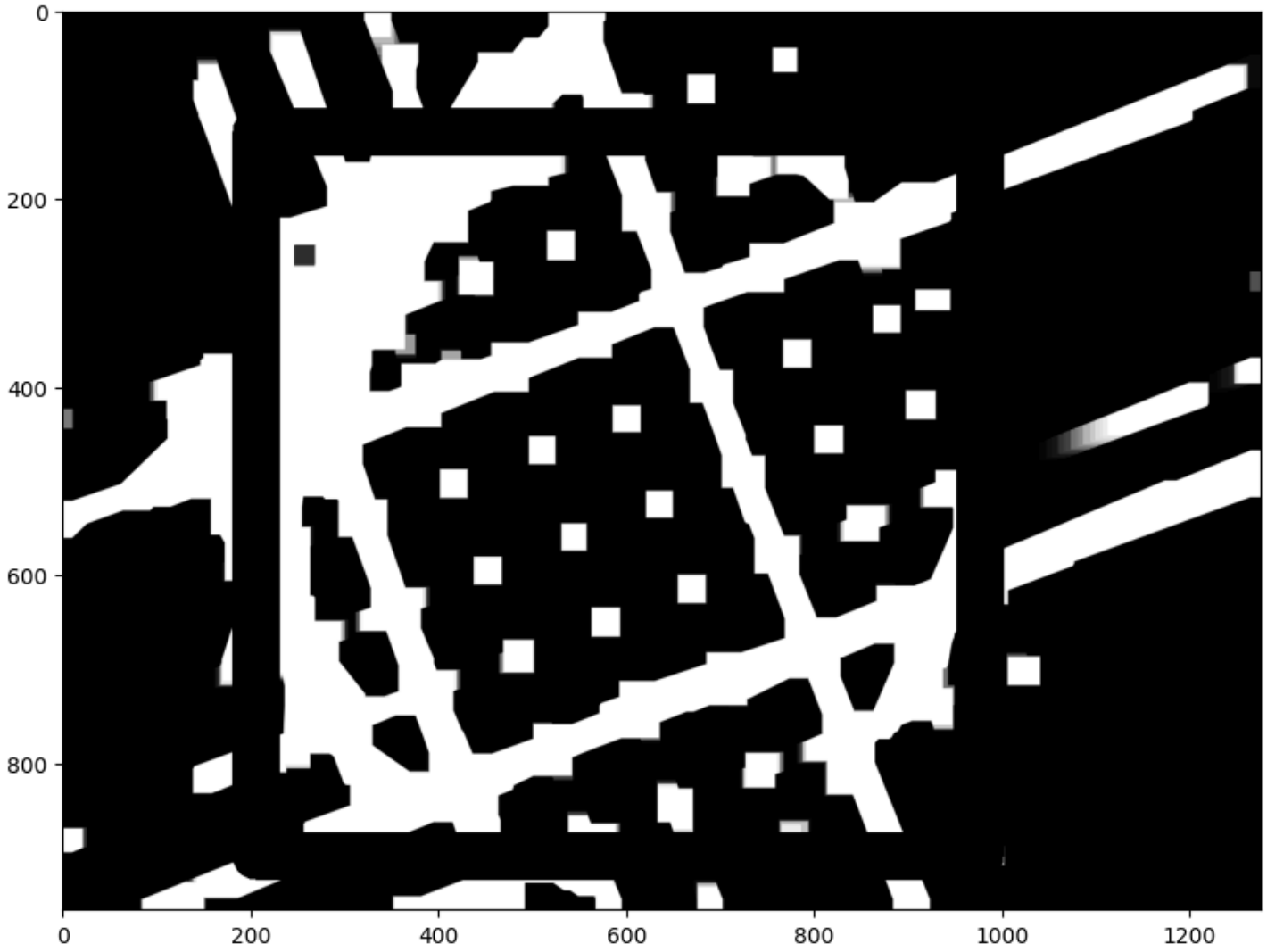
Lines were detected using the Probabilistic Hough Transform applied to the Canny edge-detected image, in conjunction with iterative dilation and erosion operations, to delineate and enhance the region surrounding the central large square. The detected lines were subsequently overlaid in white:

Probabilistic Hough Transform:

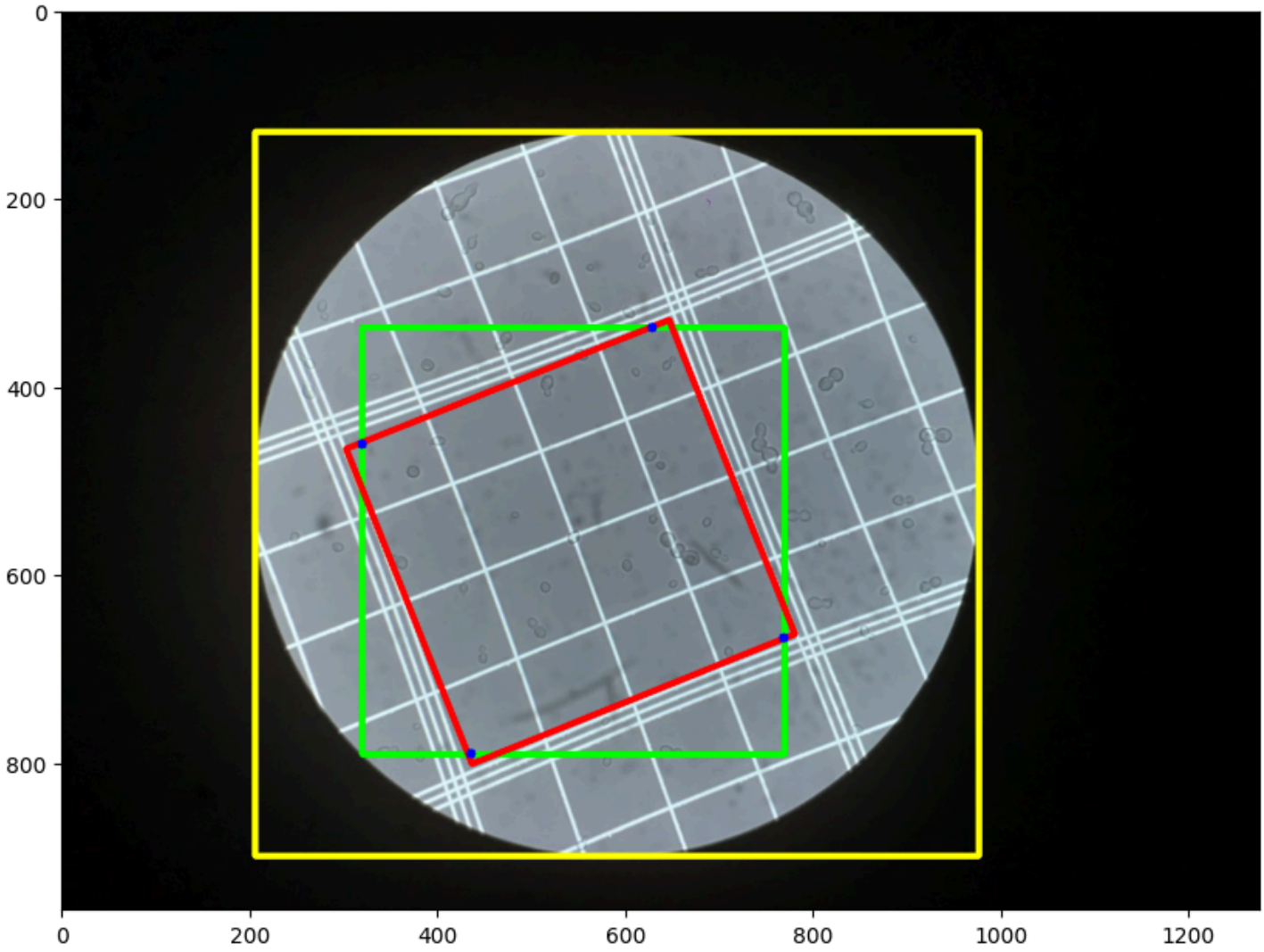


After iterative dilation and erosion operations:

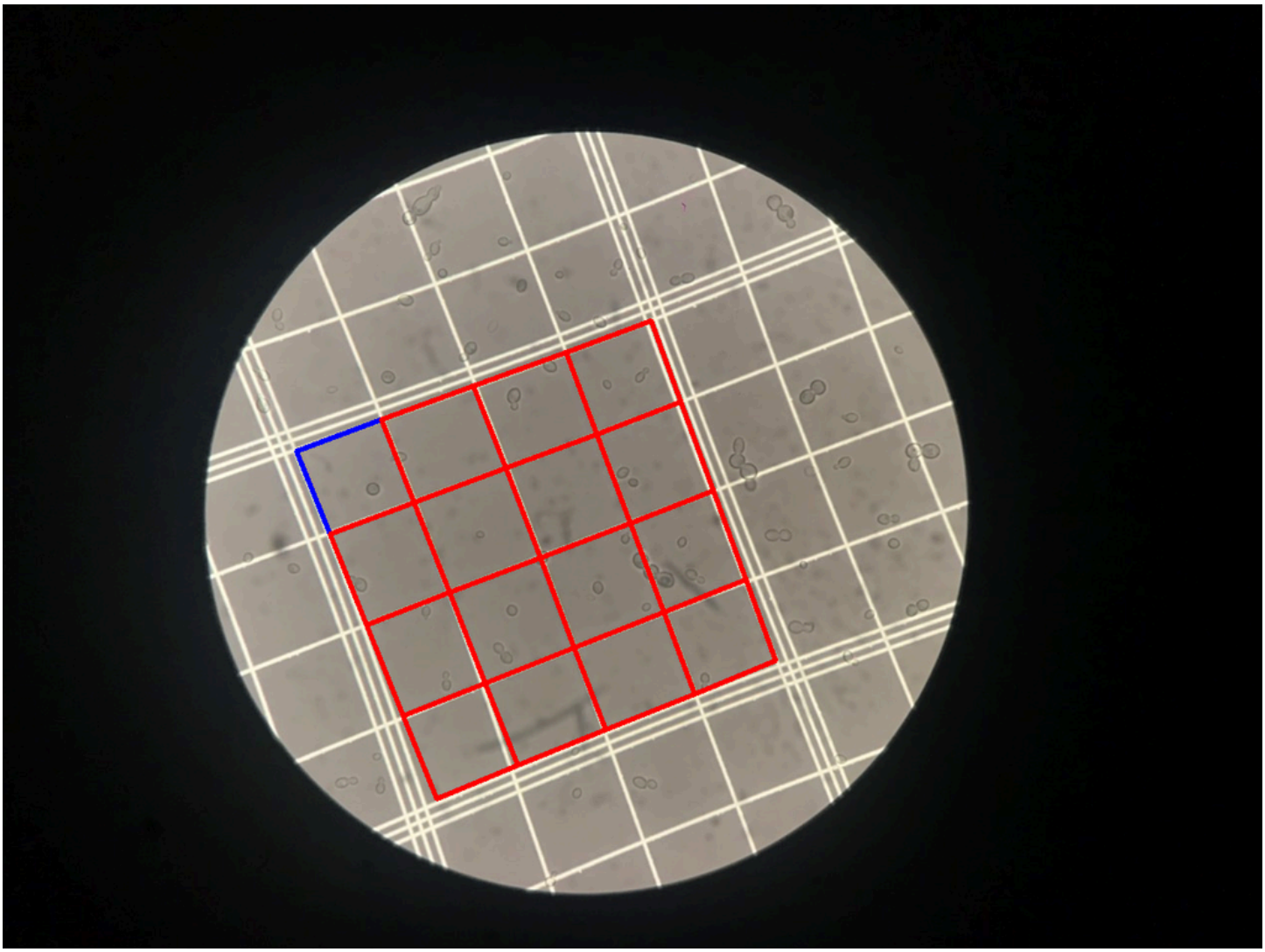
Draw a big black canvas for more stable output:



The contours of the large square were identified, and its minimum bounding rectangle was subsequently determined:



The coordinates of the 25 intersection points were determined through a mathematical ratio, while iteratively refining the coordinates of the large square's corners to minimize the bright line area.



Determine the nanometer-per-pixel ratio by comparing the precise pixel coordinates of a triangular configuration of three grid corners with their corresponding known dimensions in nanometers.

$$\text{Area of Triangle} = \frac{1}{2}\left\|\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}\right\|$$

OR

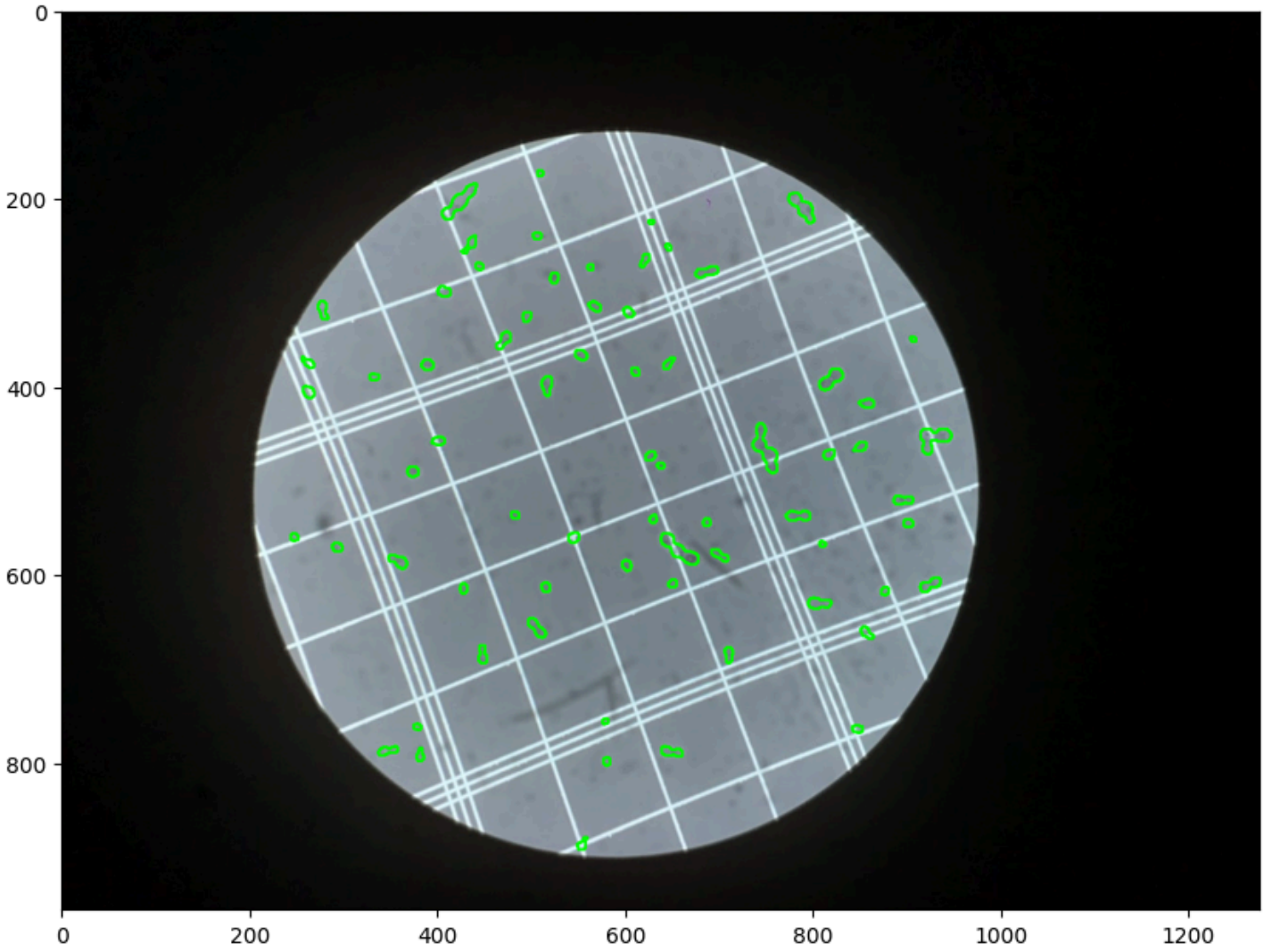
$$\text{Area of Triangle} = (1/2)\,[x_1\,(y_2 - y_3) + x_2\,(y_3 - y_1) + x_3\,(y_1 - y_2)]$$

Utilizing the precise coordinates of the 25 grid intersection points and the mask image delineating the segmented yeast areas, the enumeration of yeast cells within each of the 16 small squares was performed.

Given the inherent spatial relationship between the masked yeast area and its border, determining the containment of yeast within a specific counting square was found to be more computationally efficient by utilizing the border information.

For accurate enumeration within each counting square, the following strict boundary conditions were applied:

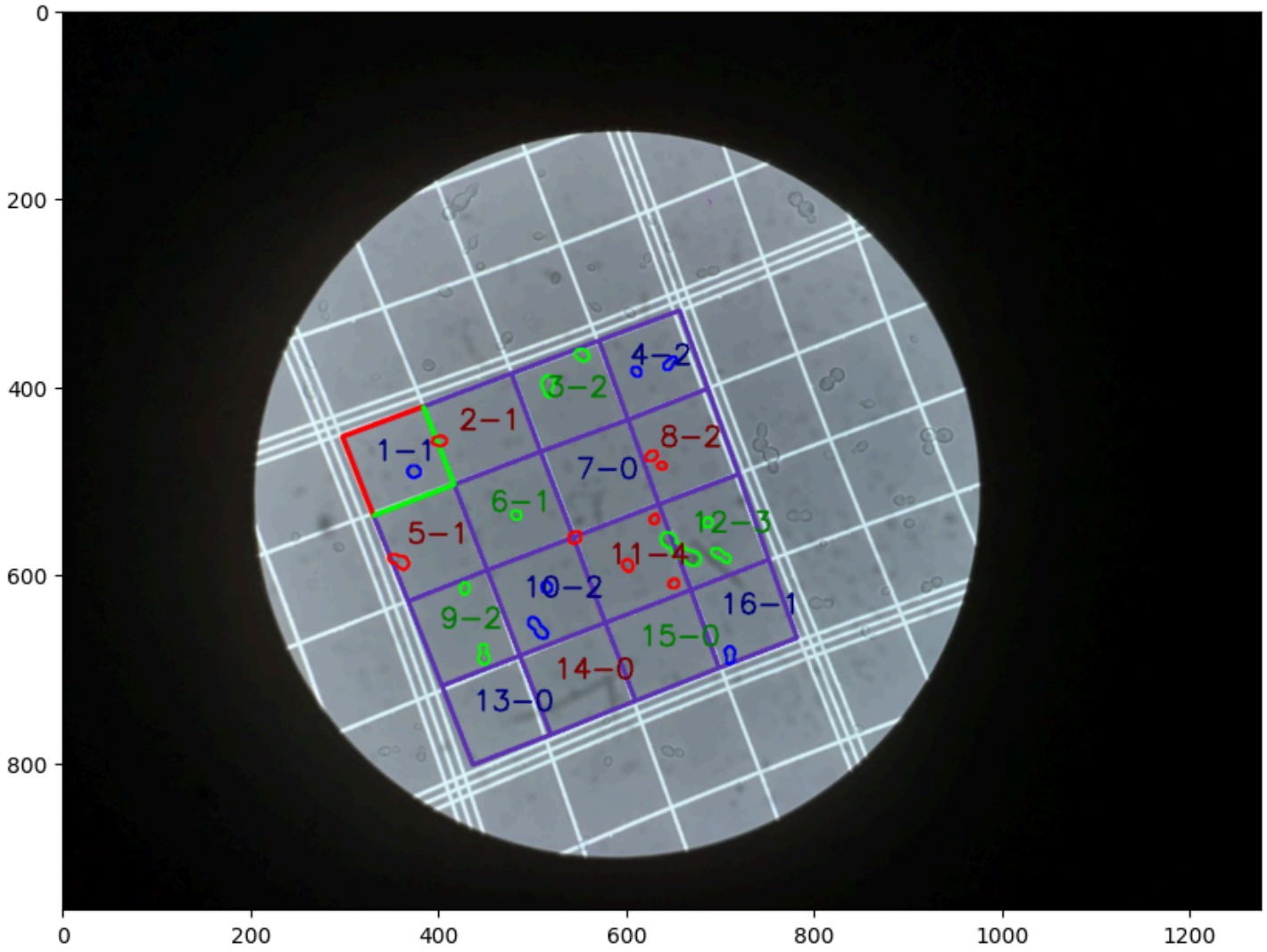
- **A yeast cell was excluded if its boundary *crossed or touched* the left or top perimeter of the counting square.**
- **Conversely, a yeast cell was included if its boundary *crossed or touched* the bottom or right perimeter of the counting square.**



OUTPUT IMAGES SHOWING THE FINAL RESULTS

In the visualization, the red lines demarcate the top and left boundaries of the counting square, while the green lines indicate its bottom and right boundaries.

Within each counting square, the text labels and the borders of the contained yeast cells were rendered in an identical color.



DISCUSSION

Here's a more formal and structured way to present the information about your Version 1 algorithm, suitable for a scientific paper. I've broken it into logical sections and refined the language.

Performance and Robustness of the Algorithmic Approach (Version 1)

The initially developed algorithmic approach (Version 1) was designed to prioritize **precision and robust performance**. Its total processing time for each input image across all processes is approximately **10-15 seconds**. This runtime was significantly reduced to **2-3.5 seconds per image** upon conversion of the entire codebase to Cython.

The algorithm demonstrated notable **robustness** during extensive testing, maintaining an acceptable output quality across approximately **850 input images**, confirming its reliability under varied conditions.

Limitations and Challenges of the Algorithmic Approach

Despite its strengths, the algorithmic approach exhibits several key limitations:

- **Vulnerability to Noise:** A primary weakness is its susceptibility to input image noise, which can potentially lead to algorithmic failure. Furthermore, the detection of such faulty outputs is challenging, complicating quality control.
- **Power Efficiency:** The algorithm currently demonstrates poor power efficiency, which poses a significant hurdle for meeting enterprise-level requirements.

REFERENCES

https://docs.opencv.org/3.4/db/df6/tutorial_erosion_dilatation.html

https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html

https://docs.opencv.org/3.4/da/d5c/tutorial_canny_detector.html

https://docs.opencv.org/3.4/dd/d49/tutorial_py_contour_features.html

https://www.geeksforgeeks.org/area-of-triangle-using-determinants/

https://cython.org/#about

VERSION 2

In this version, the algorithm previously employed for determining the precise coordinates of the 25 grid intersection points (as utilized in Version 1) was superseded by the output of a custom-trained object detection model.

ABOUT OBJECT DETECTION MODEL USED IN THIS PROJECT

RF-DETR: SOTA Real-Time Object Detection Model

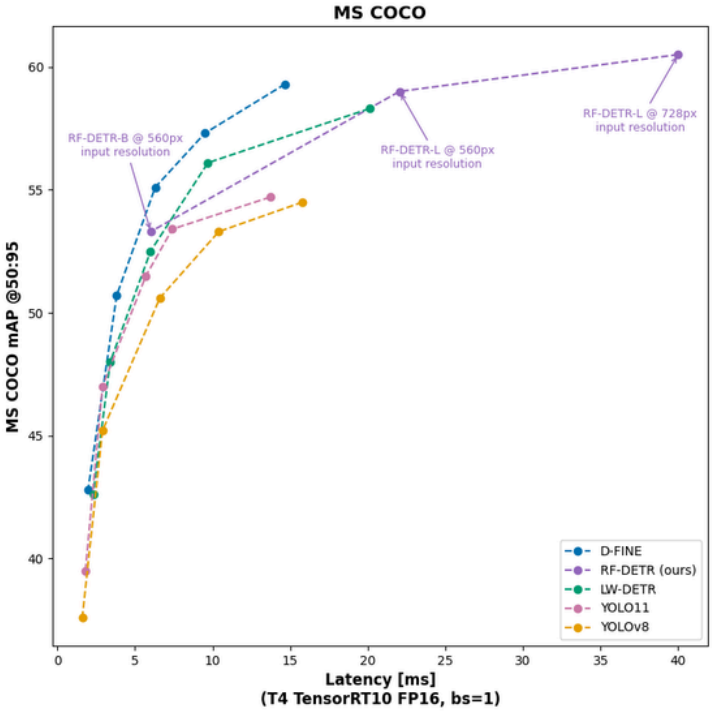
RF-DETR is a real-time, transformer-based object detection model architecture developed by Roboflow and released under the Apache 2.0 license.

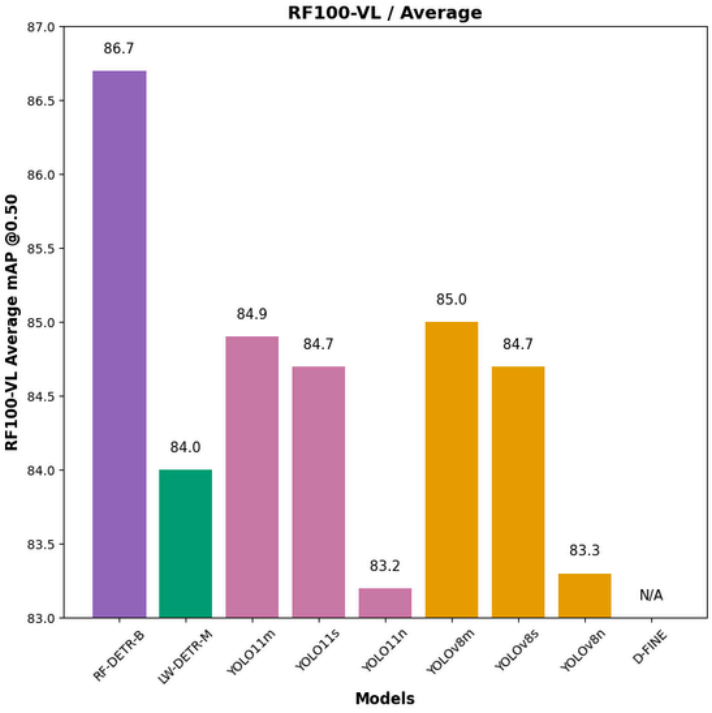
RF-DETR is the first real-time model to exceed 60 AP on the Microsoft COCO benchmark alongside competitive performance at base sizes. It also achieves state-of-the-art performance on RF100-VL, an object detection benchmark that measures model domain adaptability to real world problems. RF-DETR is comparable speed to current real-time objection models.

RF-DETR is small enough to run on the edge using [Inference], making it an ideal model for deployments that need both strong accuracy and real-time performance.

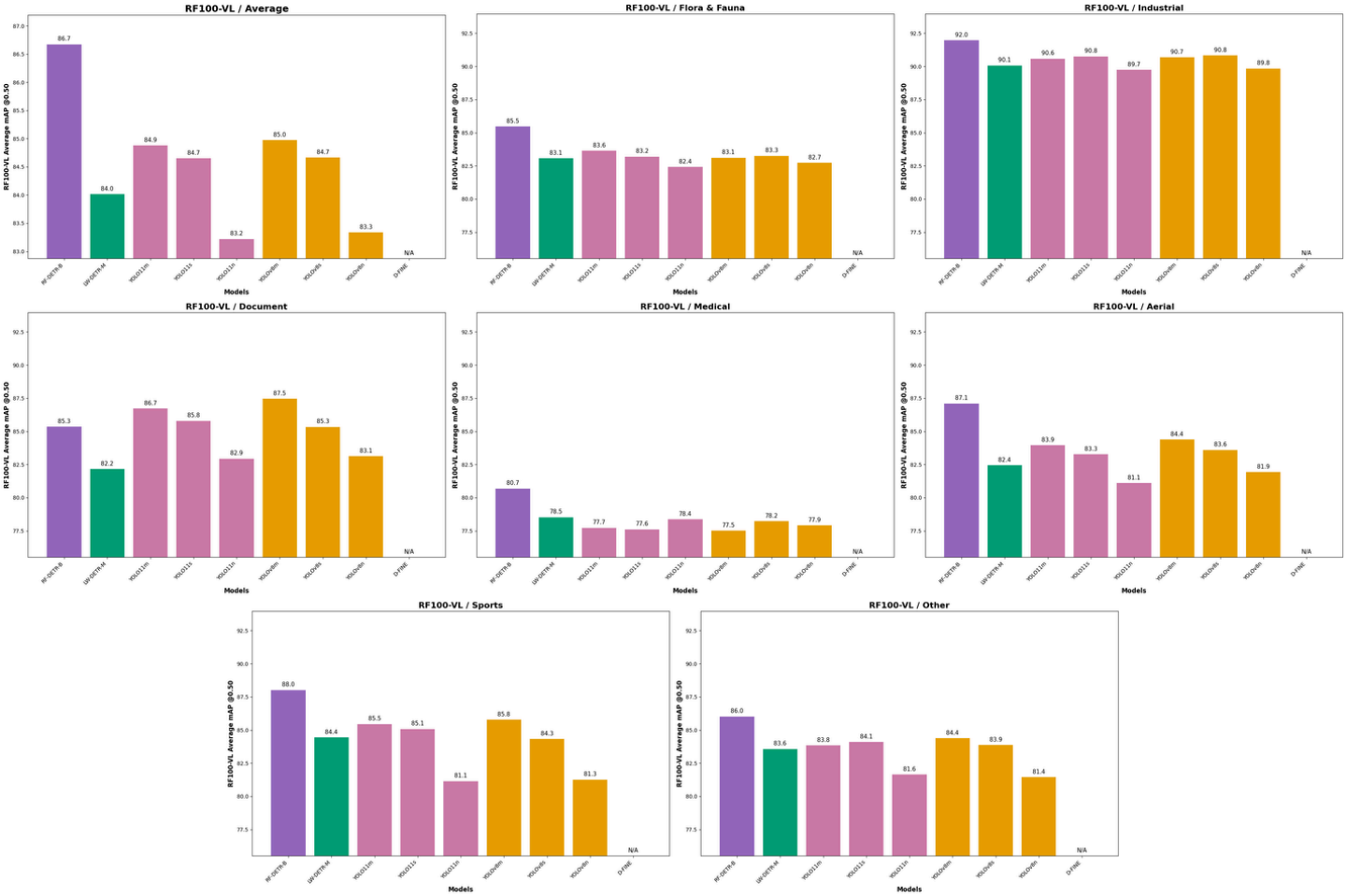
Results

Authors validated the performance of RF-DETR on both Microsoft COCO and the RF100-VL benchmarks.





RF100-VL benchmark results



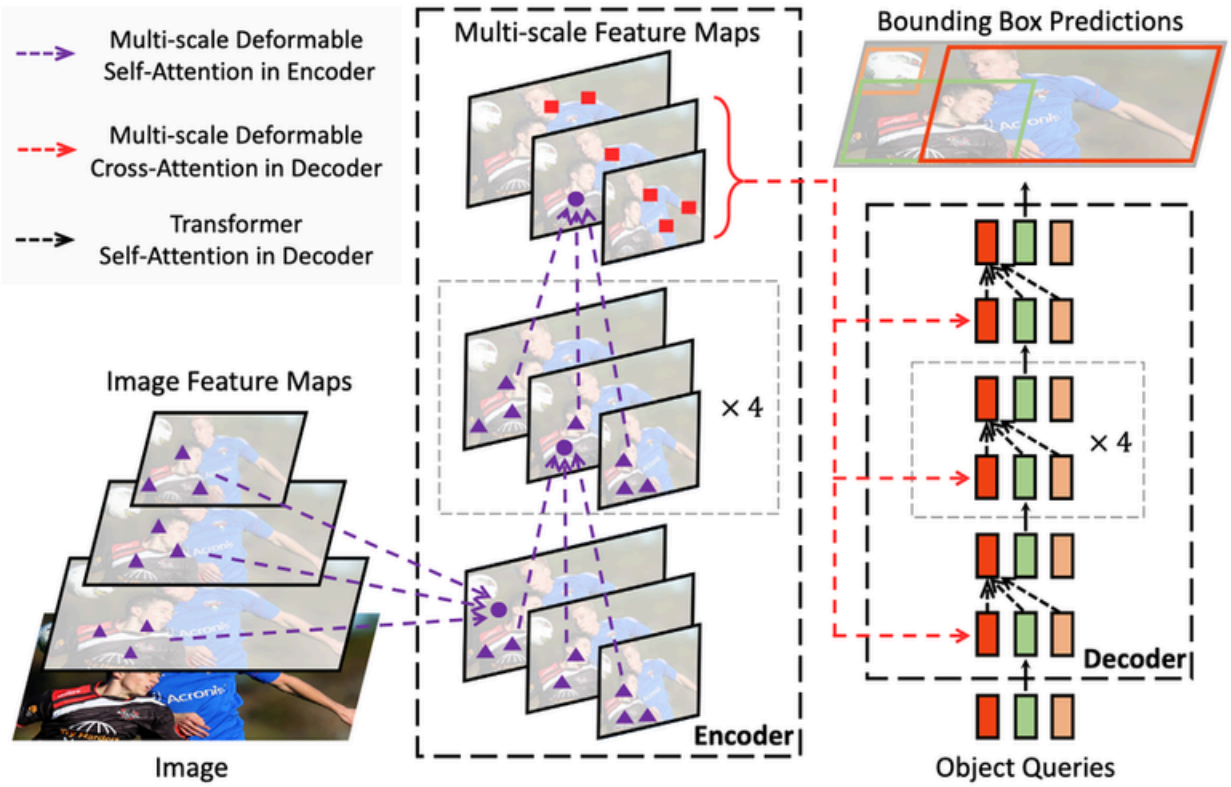
Model	params (M)	mAP$^{COCO\ val}$ @0.50:0.95	mAP$^{RF100\text{-}VL}$ Average @0.50	mAP$^{RF100\text{-}VL}$ Average @0.50:95	Total Latency T4 bs=1 (ms)
D-FINE-M	19.3	55.1	N/A	N/A	6.3
LW-DETR-M	28.2	52.5	84.0	57.5	6.0
YOLO11m	20.0	51.5	84.9	59.7	5.7
YOLOv8m	28.9	50.6	85.0	59.8	6.3
RF-DETR-B	29.0	53.3	86.7	60.3	6.0

RF100-VL benchmark notes

- The "Total Latency" reported here is measured on a T4 GPU using TensorRT10 FP16 (ms/img) and was introduced by LW-DETR. Unlike transformer-based models, YOLO models perform Non-Maximum Suppression (NMS) after generating predictions to refine bounding box candidates. While NMS boosts accuracy, it also slightly reduces speed due to the additional computation required, which varies with the number of objects in an image. Notably, many YOLO benchmarks include NMS in accuracy measurements but exclude it from speed metrics. By contrast, our benchmarking—following LW-DETR's approach—factors in NMS latency to provide a uniform measure of the total time needed to obtain a final result across all models on the same hardware.
- D-FINE's fine-tuning capability is currently unavailable, making its domain adaptability performance inaccessible. The authors caution that "if your categories are very simple, it might lead to overfitting and suboptimal performance." Furthermore, several open issues (#108, #146, #169, #214) currently prevent successful fine-tuning.

RF-DETR Architecture Overview

RF-DETR uses an architecture based on the foundations set out in the Deformable DETR paper. Whereas Deformable DETR uses a multi-scale self-attention mechanism, we extract image feature maps from a single-scale backbone.



CUSTOM TRAINED RF-DETR MODEL

The dataset is created by the output of the algorithm in version 1

Origin images (1024x1024 resolution): 666 images

Preprocessing:

- Resize: stretch to 640x640 resolution

Augmentations:

- Outputs per training example: 3
- Flip: Horizontal, Vertical
- 90 degrees Rotate: Clockwise, Counter-Clockwise, Upside Down

DATAET DETAILS:

- TRAIN SET: 1361 images
- VALID SET: 134 images
- TEST set: 68 images

EPOCHS=20

Averaged stats:

class_error: 0.00 loss: 1.3560 (1.3855) loss_ce: 0.1992 (0.2024) loss_bbox: 0.0184 (0.0199)

loss_giou: 0.0964 (0.0992) loss_ce_0: 0.2246 (0.2304) loss_bbox_0: 0.0216 (0.0233)

loss_giou_0: 0.1076 (0.1091) loss_ce_1: 0.1973 (0.2028) loss_bbox_1: 0.0181 (0.0199)

loss_giou_1: 0.0957 (0.0994) loss_ce_enc: 0.2471 (0.2481) loss_bbox_enc: 0.0206 (0.0217)

loss_giou_enc: 0.1042 (0.1092) loss_ce_unscaled: 0.1992 (0.2024) class_error_unscaled: 0.0000 (0.0000)
loss_bbox_unscaled: 0.0037 (0.0040) loss_giou_unscaled: 0.0482 (0.0496)

cardinality_error_unscaled: 123.0000 (123.9706)

loss_ce_0_unscaled: 0.2246 (0.2304) loss_bbox_0_unscaled: 0.0043 (0.0047)

loss_giou_0_unscaled: 0.0538 (0.0546) cardinality_error_0_unscaled: 127.2500 (129.8382)

loss_ce_1_unscaled: 0.1973 (0.2028) loss_bbox_1_unscaled: 0.0036 (0.0040)

loss_giou_1_unscaled: 0.0478 (0.0497) cardinality_error_1_unscaled: 120.0000 (124.5956)
loss_ce_enc_unscaled: 0.2471 (0.2481) loss_bbox_enc_unscaled: 0.0041 (0.0043)

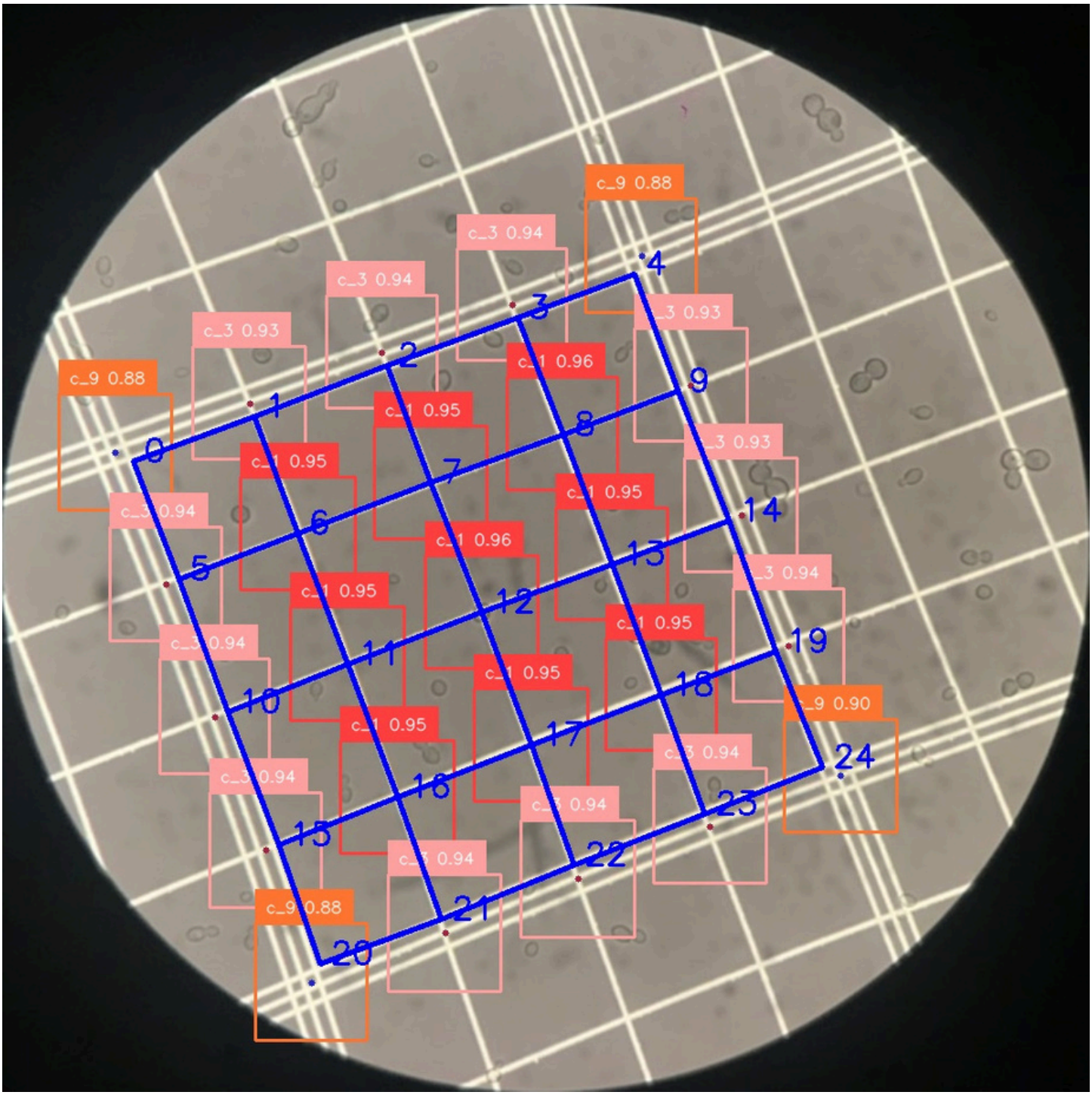
loss_giou_enc_unscaled: 0.0521 (0.0546) cardinality_error_enc_unscaled: 107.7500 (110.9118)

```
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.922
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.996
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.996
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.922
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.139
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.895
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.948
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.948
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.000
Grad accum steps:  4
Total batch size:  16
```

The methodology for enumerating yeast cells within each small square and

determining the nanometer-per-pixel ratio remained consistent between Version 1 and Version 2.

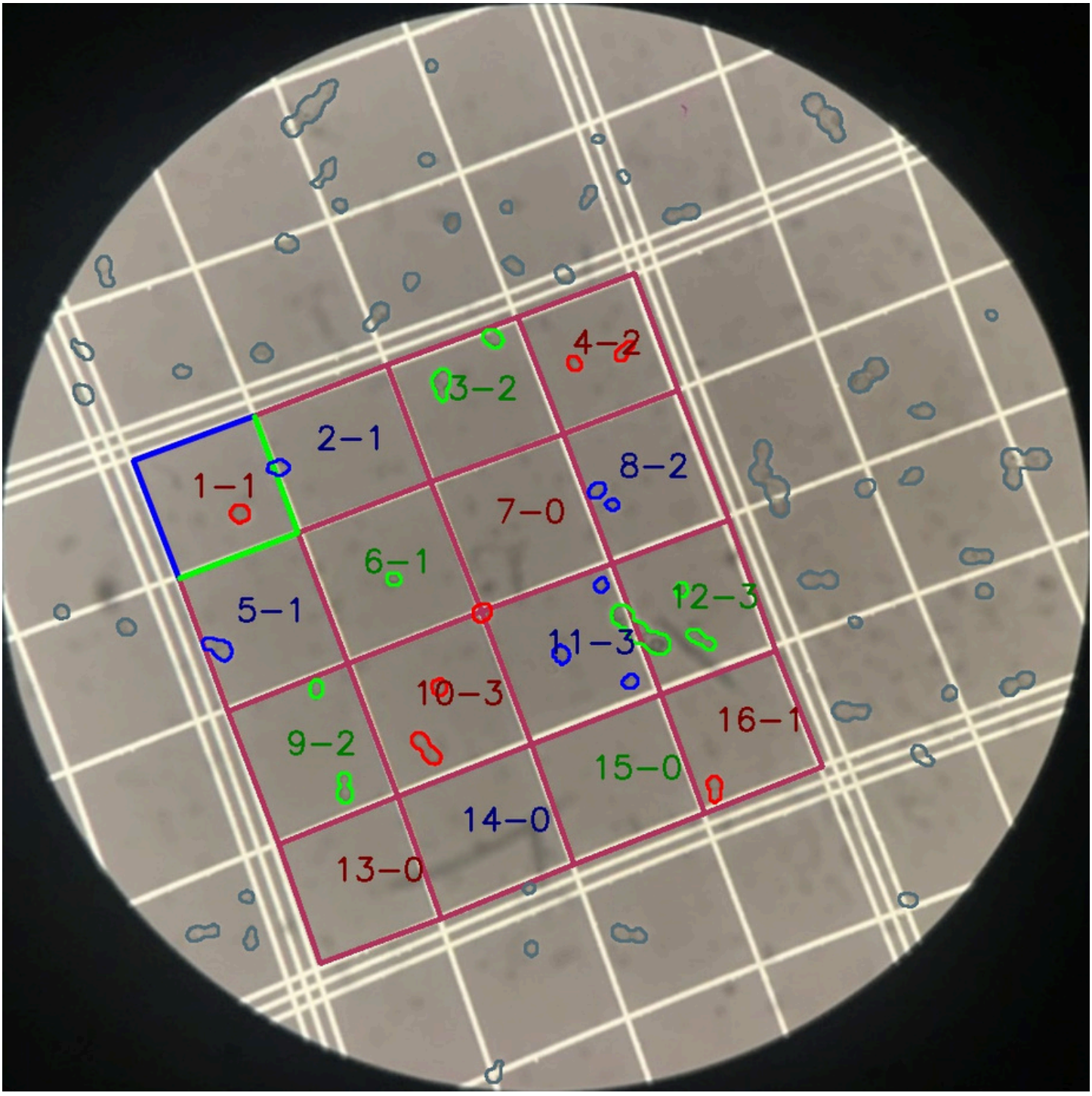
IMAGE SHOWING MODEL’S OUTPUT



OUTPUT IMAGES SHOWING THE FINAL RESULTS

In the visualization, the blue lines demarcate the top and left boundaries of the counting square, while the green lines indicate its bottom and right boundaries.

Within each counting square, the text labels and the borders of the contained yeast cells were rendered in an identical color.



DISCUSSION

Performance and Efficiency of the AI Model

The developed AI model demonstrates enhanced processing efficiency. The model's inference speed is approximately **0.07 seconds when utilizing a GPU T4**. When integrated into the

complete image analysis pipeline, the total processing time per image is approximately **0.5 seconds**.

A significant advantage of this AI-driven approach is its ability to **obviate the need for complex, manually engineered algorithms** traditionally used to determine the precise coordinates of the 25 grid intersection points. The AI model autonomously learns these positional relationships, exhibiting considerable **robustness to varying levels of noise present in input images**, a characteristic that often complicates conventional algorithmic solutions.

Considerations Regarding Precision and Future Enhancements

It is critical to acknowledge that the training dataset for the AI model was generated using the output from the Version 1 algorithm. Consequently, the **precision of the AI model in localizing these intersection points is inherently constrained by the accuracy of its foundational data source**. This implies that the AI model cannot surpass the precision achieved by the Version 1 algorithm.

To address this inherent limitation and achieve superior precision, future work will focus on the **meticulous curation of a new, higher-fidelity dataset**. This improved dataset is expected to provide a more accurate ground truth, thereby enabling the AI model to learn and ultimately improve its positional precision beyond the current capabilities.

Cost-Effectiveness

Despite the aforementioned precision considerations, the AI model offers a substantial operational advantage: it is **significantly faster than the Version 1 algorithm** while yielding comparable precision. This heightened efficiency translates directly into a more **cost-effective solution** for high-throughput image processing and analysis applications. The source code can be converted to Cython for much faster running time.

REFERENCES

https://github.com/roboflow/rf-detr

dataset link: https://app.roboflow.com/neubauerchamber/my-first-project-rn6zt/3/images