Keep Trying for The AutoBio Challenge

Chengying Tu
Yuanpei College
Peking University
2000017802@stu.pku.edu.cn

Xiangcan Xu
School of EECS
Peking University
2100012996@stu.pku.edu.cn

Abstract

In biological experiments, an auxiliary monitoring system can help timely discover errors, find the cause of experimental failure, and improve the success rate of the experiment, by monitoring the experimental operation through the camera. The tasks of the auxiliary monitoring system can be roughly divided into three parts: transparent object segmentation, liquid tracking, and collaboration. In the laboratory scenario, the many transparent instruments and liquids, as well as the occlusion caused by experiments, *etc*., make these tasks extremely difficult. Due to the limited time and the difficulty of the challenge, we mainly try to solve the first task and the second task. Here we show that the existing model Trans2Seg can be improved to solve the first task well but a variety of training datasets are needed to improve the segmentation performance for different shooting perspectives. And for the second task, we successfully implement container detection as subtask1 and try to do liquid classification in the container as subtask2. Our attempts illustrate the difficulties of these tasks, and we try to propose some possible directions for improvement.

1 Introduction

Biological experiments tend to have a low success rate. A variety of complicated manual operations have brought great trouble to biological experiments, such as marking the test tube with a marker pen. Any careless operation may affect the experimental results and lead to the failure of the experiment. Therefore, we hope to implement an auxiliary monitoring system to monitor the operation of experimental personnel through the camera, help timely discover errors, find the cause of experimental failure, and improve the success rate of the experiment.

To develop the function of this auxiliary monitoring system step by step, we first try to implement three functions: transparent object segmentation, liquid tracking, and collaboration(including human-object interaction and human-human collaboration).

For the transparent object segmentation task, we first review the existing transparent object segmentation algorithms and try to apply them to our task dataset. We find that the Trans2Seg [9] model performed well, so further optimize the model structure to improve the effect. For the liquid tracking task, we consider it an MOT(Multiple Object Tracking) problem and try to divide it into two subtasks: get all detection boxes of the experimental container and get the category of solution in the container. For the collaboration task, we also try to divide it into some subtasks: detect experimenters and objects, then human-object interaction and human-human collaboration, and then use QA models to generate answers to relevant questions.

Due to limited time and computing resources, our project mainly focuses on solving task1 and task2. But we have some thinking about each task and try to propose possible solutions for each task. For task1, we demonstrate the performance of Segformer and Trans2Seg on the task1 dataset and make model improvements for the better-performing Trans2Seg model. Since the images of the dataset are taken from a few videos, the content of the training/test splits is highly consistent. To evaluate the performance of the model on the unseen dataset, we test the model on the task2 dataset and analyze it

accordingly. For task2, we mainly try and improve the container detection for subtask1. We train a YOLO [4] model under MMDetection [2] framework, In the subsequent improvement, we add the pre-training of Trans10K dataset and refer to the idea of DeepSort [5] and ByteTrack [10]. For the failures and the limitations of the model, we try to propose some possible directions for improvement.

2 Related Work

2.1 Transparent Object Segmentation

Transparent object segmentation is a new subfield of semantic segmentation. Most of the previous research focused on the transparent object in the wild. In 2020, a large-scale dataset Trans10k for transparent objects segmentation and a model TransLab trained on the dataset was proposed [7]. Then, the transformer was used to build a more powerful model Trans2Seg [9]. And it got the SOTA on the Trans10k dataset.

The two main models we tried in this project are Segformer [8] and Trans2Seg [9].

Segformer [8] is a semantic segmentation framework based on the transformer. Compared with other transformer structures, Segformer uses lightweight multilayer perception (MLP) decoders. Such a structural design also brings powerful performance to Segformer.

Trans2Seg [9] is a semantic segmentation framework of transparent objects based on the transformer, which got SOTA on the Trans10k dataset. And Trans10k [7] is the first large-scale transparent object segmentation dataset. The transparent objects in Trans10k are extremely challenging due to the high diversity in scale, viewpoint, and occlusion. Trans10k helps the development of semantic segmentation of transparent objects.

2.2 Multiple Object Tracking Based on SORT Framework

In 2016, a simple model SORT [1] was proposed. Compared with previous methods, this model is not only real-time but also accurate. Then, DeepSORT used a feature extraction network [5] to improve the matching. To better process the information of low confidence detection frames, ByteTrack [10] proposed some improvement methods to the matching based on the previous SORT framework.

We mainly tried SORT [1] and ByteTrack [10] in the project.

SORT [1] is an MOT model. It uses a detector to detect all the detection boxes in the current frame. Then it uses the Kalman filter [6] to predict the position of tracks in the next frame and uses the Hungarian algorithm to match the previous tracks and current detection boxes. The proposal of SORT [1] brings a real-time and efficient MOT model.

ByteTrack [10] inherits from a classic structure of SORT [1]. It uses second matching to dig out the helpful information from detection boxes whose confidence is lower than the threshold. This simple but effective idea makes the model more powerful and robust.

3 Tasks and Datasets

3.1 Instance Segmentation

Task1 is a semantic segmentation task. In this task, we need to train a model to segment 11 types of transparent objects. An example is shown in Figure 1 to illustrate what problem task1 tries to solve. As we can see, the objects that we need to segment are transparent containers or transparent liquids. In the case of a container, whether or not it contains liquid affects the appearance of the container. Moreover, in the laboratory scene, the objects that need to be segmented take up a rather small proportion of the picture, and some objects may even be blocked by human hands sometimes. All of these make it difficult to segment objects in this scene.

The dataset consists of 15 videos, with a total of 3235 images. Table 1 shows the data distribution of dataset1.

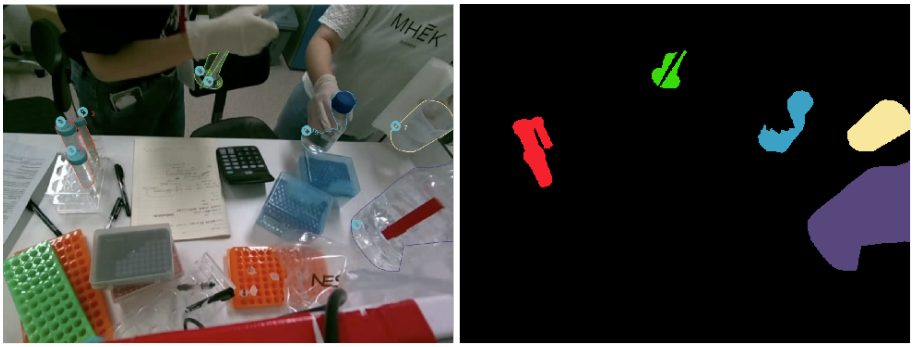


Figure 1: An example that shows what problem task1 tries to solve.

Table 1: **Data distribution of dataset1**

	Object num	Image num
waste_box	1694	722
tube	6353	1452
solution_P1	302	298
PCR_tube	2389	635
vial	899	803
water_bottle	445	441
measuring_flask	955	773
erlenmeyer_flask	3507	1409
culture_plate	4083	687
wash_bottle	77	45
beaker	35	35

3.2 Liquid Tracking

Task2 is an MOT(Multiple Object Tracking) task. In this task, we need to track a specified liquid in a video and output its position(s). All liquids are contained in transparent containers. So actually we need to track all the containers containing the liquid. Examples are shown in Figure 2 to illustrate what problem task2 tries to solve. The liquid can be colored (example1 in Figure 2) or transparent (example2 in Figure 2). The same liquid can be packed in different containers at the same time and can be transferred from one container to another.

(a) Example1 (b) Example2

Figure 2: Examples that show what problem task2 tries to solve.

The dataset consists of 15 videos, with a total of 2661 images. There are 12 container categories and 5 liquid categories (including the 'none' liquid) in the dataset. The dataset2 contains all the container categories in dataset1 except the category wash_bottle and breaker. So we use this to get our dataset. Specifically, we use dataset2 as the test set of dataset1 and use dataset1 as the test set of dataset2 in subtask1.

Table 2 shows the liquid distribution of dataset2.

Table 2: **Liquid distribution of dataset2**

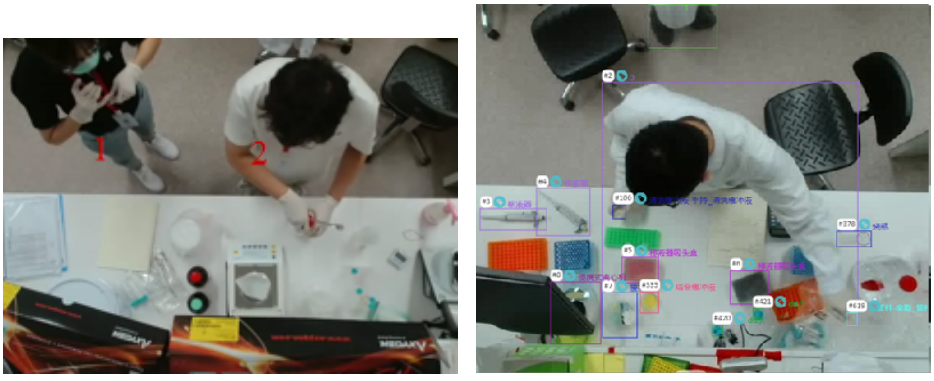
	Object num	Image num
none	8736	2300
bacteria	1570	174
solution_P1	4216	528
dd_water	1584	634
LB_solution	5443	1302

Table 3: **Data distribution of dataset3**

	Object num	Image num	HOI categories
solution_P1	439	390	4
pipette	449	449	5
stopwatch	390	390	4
tube	8349	1290	6
vial	275	275	5
measuring_flask	128	128	2
agarose	334	293	3
electronic_scale	732	732	0
spoon	633	633	4
D_sorbitol	656	375	5
LB_solution	96	96	4

3.3 Collaboration

Task3 is a comprehensive task, generally divided into two parts. First, we need to get all interactions between people and containers (called 'HOI') in each frame (shown in Figure 3 (b)). In this step, because of the continuity of action, we must consider the connection between video frames. Second, we will receive a sentence with blanks. For example, "() is helping () to () with ()". By using all of the HOIs in a continuous video segment, we need to get human-human collaboration (shown in Figure 3 (a) and fill in the blanks of the sentence, like "(Person A) is helping (Person B) to (add agarose) into (LB solution)".



(a) Human-human collaboration (b) Human-object interaction

Figure 3: Examples that show what problem task3 tries to solve.

A tip to task3 is the total number of interactions is limited (including interactions between people). So we can do multiple-choice questions instead of filling in blanks.

The dataset consists of 9 videos, with a total of 1465 images. There are 12 container categories and 43 HOI categories in the dataset. Each container can have up to 6 HOIs. Table 3 shows the data distribution of dataset3.

Table 4: **IoU % for Task Instance Segmentation in Task1 Dataset**

	Segformer	Trans2Seg -resnet50	Trans2Seg -resnet101	Trans2Seg_large -resnet101
Avg	46.45	92.35	91.58	92.85
waste_box	55.09	96.14	96.29	96.25
tube	53.09	88.77	89.00	89.14
solution_P1	6.09	87.97	88.03	87.83
PCR_tube	13.53	80.74	81.42	81.46
vial	67.91	95.82	95.93	95.99
water_bottle	76.18	97.94	98.04	98.04
measuring_flask	52.45	95.05	95.20	95.21
erlenmeyer_flask	76.76	96.15	96.29	96.32
culture_plate	64.44	93.93	93.91	93.83
wash_bottle	10.52	88.21	82.81	89.22
beaker	34.9	95.17	90.49	95.09

4 Experiments and Analysis

4.1 Instance Segmentation

4.1.1 Overview

In this task, we have mainly tried two models that performed well in transparent object segmentation: Segformer [8] and Trans2Seg [9]. The experimental results are shown in Table 4. We calculate mIoU (Mean Intersection over Union) for each object category and compute the average over all the categories as the evaluation metric. The experimental results show that Segformer performs not well in this task, which may be due to the simplicity of the model. In contrast, Trans2Seg, a transformer-based segmentation method, performs quite well in this task. So our subsequent attempts are centered around Trans2Seg. We have tried to change the backbone of Trans2Seg from resnet50 to resnet101 and further enrich the model structure.

4.1.2 Analysis of Experimental Results in Task1 Dataset

From the results, we can see that the hardest object to segment is PCR_tube. Despite PCR_tube's huge amount of data, it is hard to segment it from the background due to the perspective limitations and excessive similarity to the background (an example is shown in Figure 4).

(a) Image (b) Mask

Figure 4: Ground-truth of a failure case: PCR_tube close to the experimenter's left hand is subtle and indistinguishable, which makes it difficult to segment (from Task1 Dataset). Our model trained on Task1 Dataset fails to segment it from the background.

By changing the backbone of Trans2Seg from resnet50 to resnet101, we can see that the segmentation effect of PCR_tube is improved by about 0.7%. However, it becomes significantly worse in the wash_bottle and beaker categories, which also results in a decrease in the average effect. We then continue to enrich the model structure, building on resnet101, deepening the model, and increasing

Table 5: **IoU % for Task Instance Segmentation in Task2 Dataset**

	Segformer	**Trans2Seg -resnet50**	**Trans2Seg -resnet101**	**Trans2Seg_large -resnet101**
Avg	29.36	42.00	42.05	41.67
waste_box	50.87	55.60	56.89	55.11
tube	52.54	57.30	57.16	58.35
solution_P1	0.10	0.00	0.00	0.00
PCR_tube	2.59	5.08	3.17	6.71
vial	54.85	81.07	81.43	77.61
water_bottle	65.88	86.06	87.17	84.83
measuring_flask	30.2	85.54	85.38	85.20
erlenmeyer_flask	7.17	7.42	7.23	7.20
culture_plate	0.00	0.00	0.00	0.00

the embedding dim and hidden dim. It turns out that the average performance improves by about 0.5%. The Trans2Seg pipeline may have reached a bottleneck in its performance on this task, and another pipeline may be needed to further improve the performance. As for task1 dataset, we select Trans2Seg_large-resnet101 as our best model for task1.

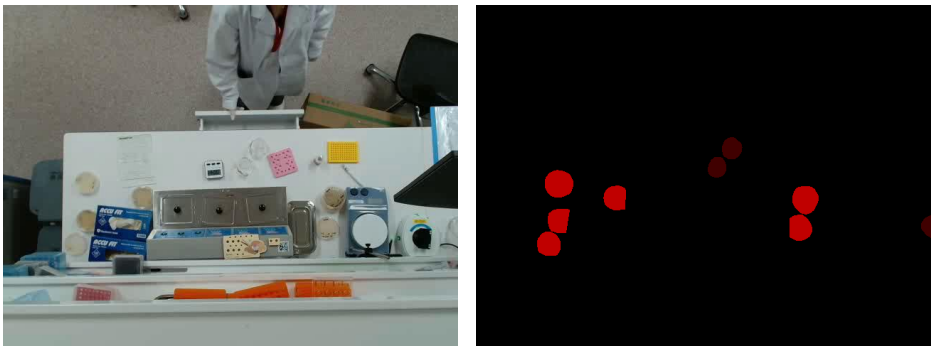
4.1.3 Analysis of Testing in Task2 Dataset

To test whether the model learns to segment objects, we further test the model using the data from task2. Task2's dataset lacks the wash_bottle and beaker categories, so we only test the remaining 9 categories. The results are shown in Table 5. From the results, we can see that for the three Trans2Seg models, the one with the best performance in task1 dataset performs worst in task2 dataset, and overall their performances are close.

Besides, we can see all the models perform terribly in segmenting solution_P1 and culture_plate categories. Digging deeper into the dataset, we have found that culture_plate tends to be stacked together in task2 dataset (an example is shown in Figure 5), while it tends to be scattered in task1 dataset (an example is shown in Figure 6). This could explain why the model is hard to segment culture_plate in task2 dataset. Different shooting angles will result in different shapes of the objects and the way they are placed will also affect the performance of the model. So rich training dataset is an important guarantee of model robustness. For solution_P1, it is different from other categories. This category marks the solution rather than the container. So the shape is often expressed as the shape of the container the experimenter is working with, while the color of the solution itself is transparent. As a result, the shape of the category changes very much and the color characteristics are not obvious. Plus the low overlap between the task1 dataset and the task2 dataset in this category, the model performs poorly in the task2 dataset. Therefore, richer datasets with more shooting angles may be necessary if the model relies solely on visual information to segment objects.

(a) Image (b) Mask

Figure 5: A ground-truth example of culture_plate from Task2 Dataset.



(a) Image (b) Mask

Figure 6: A ground-truth example of culture_plate from Task1 Dataset.

4.2 Liquid Tracking

4.2.1 Overview

In this task, we divided it into two subtasks.

In current attempts, we have got some results in subtask1. But the result is not very good, so we pay more attention to the feasibility of our ideas. By analyzing the difference between the basic model and the pre-trained model, we have come to the fact that our model is only sensitive to a certain scale. Then we try a simple multi-scale test method to verify this fact. At last, we try to use the structure of ByteTrack [10] to solve the problem of the previous model.

For subtask2, We didn't get meaningful results. Due to time constraints, we abandoned subtask2. So the following discussion is mainly about subtask1.

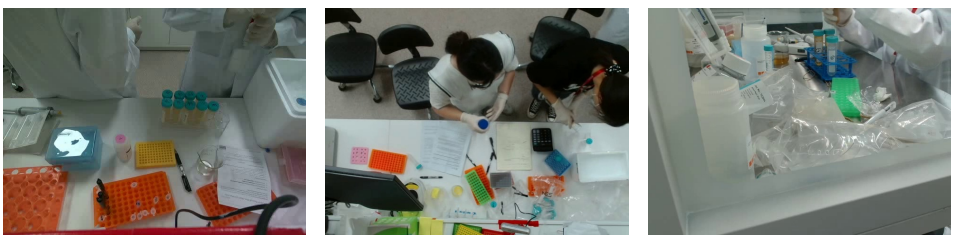
4.2.2 Metric

We use COCOeval API to evaluate our detection methods. This is the standard metric of the COCO dataset [3].

4.2.3 Difficulties in the task

Compared with the general MOT task, this task has two special difficulties.

- The task needs to track the liquid instead of the solid. Liquid tracking is a more challenging task due to little related work.
- The dataset has lots of complex situations, which brings us two problems:
  - There are three types of Angles of View in the dataset: front view, end view, and vertical view. In the vertical view, the object looks very small and only shows partial features. But in the front view and the end view, the object looks much bigger and has richer features. Figure 7 shows this problem
  - The occlusion between multiple containers is common and important. Sometimes the container behind another one also needs to be detected because the front container is transparent. See figure 8 for a clear understanding.



(a) Front view (b) Vertical view (c) End view

Figure 7: **Examples of different perspectives.**

(a) Sample

Figure 8: **Sample shows the occlusion.** The object in the blue box is completely covered by the object in the red box, but we can still see it .

4.2.4 Algorithm flow

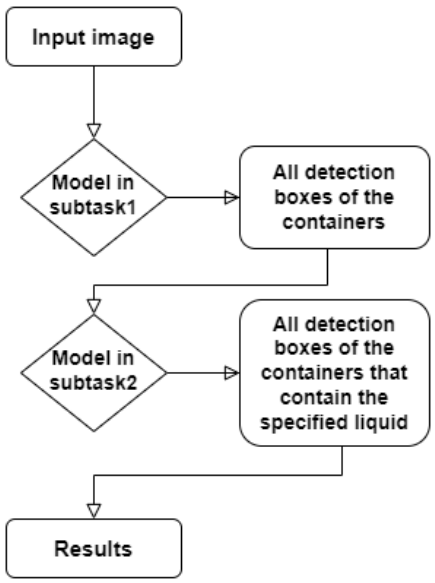


Figure 9: **The algorithm flow.**

Due to the complexity of this task, we try to divide it into two subtasks:

- Subtask1 is to train a model to get all detection boxes of the experimental container from an image. Here we don't care about the types of containers. Because of the high similarity between subtask1 and task1(Instance Segmentation), we predict the model will work.
- Subtask2 is to train a model that receives a detection box to get the category of solution in the container. By analyzing the dataset, we found that the dataset has few liquid categories (only 5). And different categories have some distinctive features, such as colors. So we also predict the model will work.

The flowchart is presented in figure 9.

4.2.5 Dataset for subtask1

We found the Liquid Tracking dataset(dataset2) contains all the container categories in the Instance Segmentation dataset(dataset1) except two categories with the least number. we use this to get our dataset in subtask1. Specifically, we use dataset2 as the training set and dataset1 as the test set.

The train set consists of 15 videos, with a total of 2661 images. And the test set consists of 15 videos, with a total of 3235 images.

Table 6: **The mAP of different methods.** IoU=0.05:0.95 and maxDets = 100

	Area			
Method	All	Small	Medium	Large
Basic	0.229	0.166	0.405	0.159
Pretrained	0.202	0.117	0.402	0.272
Multi-scale(5)	0.249	0.180	0.421	0.457
Multi-scale(7)	0.259	0.203	0.409	0.462

Table 7: **The mAR of different methods.** IoU=0.05:0.95 and maxDets = 100

	Area			
Method	All	Small	Medium	Large
Basic	0.301	0.196	0.558	0.311
Pretrained	0.250	0.152	0.476	0.587
Multi-scale(5)	0.364	0.274	0.572	0.703
Multi-scale(7)	0.399	0.319	0.584	0.685

4.2.6 Subtask1

Our simple and basic idea is to ignore the connection between video frames and directly train a YOLO [4] model under MMDetection [2] framework on our dataset.

We trained 273 epochs with a 416 scale YOLOv3 model. The result shows in table 6 and table 7.

The result is not good. We think there are two main problems that influence the result:

- The model didn't learn the feature of the transparent object. Thus the changing background may influence the detection results.
- The model is only sensitive to a few scales. Actually, The result of medium objects is much better than others. But in our dataset, there are rich and important objects of different scales. So the object scale influences the result a lot.

To solve problem1, We try to use the Trans10k training set for pre-training. The table 6 and the table 7 shows the result. Actually, we can see that the performance of large objects is improving, while the performance of small objects and medium objects is decreasing.

That's an interesting fact. We thought that the main reason might be the difference scales of the datasets.

The difference in the scale between the two datasets is huge and obvious. In Trans10k, the images are usually large(millions of pixels), and the objects usually occupy 20% to 50% of the image. But in our dataset, the size of the image is 640*480. And the objects are usually small. Figure 10 shows the contrast.

We think the different scales influence the performance of our YOLO model. Due to lack of time, we did not retrain the model. So the following experiments are based on this pre-trained model. Thus, in the later experiment, we will pay more attention to the feasibility of the method rather than the improvement of the performance.

In the previous analysis, we realized that for some reason, our model is sensitive to objects at a certain scale. So from the perspective of verifying this conjecture and improving performance, we decided to try to scale the test image to different scales, and then integrate all the results as the final result. (some NMS methods are used) This is also our method to solve problem2. The specific results are presented in table 6 and table 7.

We see that not only the performance has improved, but also the improvement is relatively large. And, an interesting fact here is that the AR metric of each object scale has increased by a similar range. This further proves our conjecture.

This result also guides our next step. It shows that the small objects are the key to the model's improvement. So, in the next step, we will focus on improving the model's ability to detect small objects. Unfortunately, we don't have time for the next step.

(a) Our dataset

(b) Trans10k

Figure 10: **The different scales in the two datasets.** In our dataset, small and medium objects account for a large proportion. But in Trans10k, Large objects account for more.

Table 8: **The mAP of different methods.** IoU=0.05:0.95 and maxDets = 20

	Area			
Method	All	Small	Medium	Large
Multi-scale(7)	0.258	0.202	0.408	0.462
Multi-scale(7)-filtered	0.231	0.170	0.384	0.453
ByteTrack	0.240	0.184	0.386	0.386

All the previous experiments ignore the connection between the video frames. So in the next step, we try to use some MOT methods to improve our model's performance.

There is a problem in the previous model: The model does not make good use of the information from the low-confidence detection box. If the confidence of a box is lower than the threshold, the model will filter it directly. However, sometimes the confidence of the detection boxes will be lower than the threshold due to the influence of occlusion or illumination variations and other factors. Therefore, we need a method to rediscover such detection boxes.

To solve the problem, we want to use an algorithm to dig out the correct detection box of low confidence. Then, we find that the ByteTrack is what we need.

ByteTrack [10] inherits from a classic structure of SORT [1]. That is, using Kalman filter [6] to predict the position of tracks in the next frame, and using the Hungarian algorithm to match the previous tracks and current detection boxes. We think this structure may help us solve the problem.

The result shows in table 8 and table 9. We found it is worse than the previous model. We think it is the default COCO metric that causes the problem. We don't have time to explore the details. Here is only our judgment basis.

For the detection boxes with a confidence lower than 0.3, we will directly discard them during display. In the model of ByteTrack, we will also filter out most of the detection boxes with a confidence lower than 0.3. However, in the default COCO metric, these detection boxes may affect the final results. So

Table 9: **The mAR of different methods.** IoU=0.05:0.95 and maxDets = 20

	Area			
Method	All	Small	Medium	Large
Multi-scale(7)	0.381	0.302	0.563	0.660
Multi-scale(7)-filtered(0.3)	0.285	0.209	0.459	0.587
ByteTrack	0.314	0.239	0.484	0.609

here we filter the detection frame with a confidence lower than 0.3 and then retest the multi-scale (7) model. The final results show in table 8 and table 9.

Comparing the results, we do find that using the ByteTrack structure is helpful to improve the performance of our model.

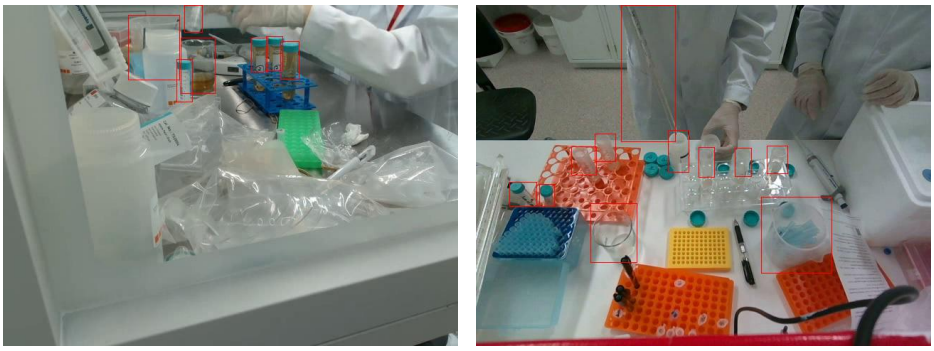
At the end of this subtask, we are trying to suggest some directions that can be tried in the future:

- We think the containers in hand and containers near hands are most important because liquid transfer only occurs in these containers. So we can use some models that can detect human feature points to help us identify these important containers.
- We can get how the liquid is transferred by dealing with the relationship between people and containers, and between containers.

We think these ideas try to focus on the critical areas in an image or a video. So these ideas may be the key to further improvement. However, these are too far away from us. If there is a chance in the future, we may continue to explore it.

4.2.7 Subtask2

We didn't have time to deal with this subtask, so it was abandoned. We found the Trans2Seg model works well in segmenting the liquid. So this may be a direction.



(a) Example1 (b) Example2

Figure 11: Examples of the results.

5 Conclusion

In this project, we surveyed the methods related to the three tasks and explored solving the tasks based on the existing work.

For the Instance Segmentation task, due to our lack of experience, we chose to improve on the off-the-shelf model that is proven to perform well in segmenting transparent objects in the wild. The improved Trans2Seg model gets considerable results. We know that even such a large model cannot perform perfectly on this task in the lab scenario, which is enough to show the difficulty of the task. Due to the small number of videos in the dataset and the large amount of repetitive information between the images, we have doubts about the richness of the dataset and the learning of the model. Therefore, the model trained on task1 dataset is tested on task2 dataset, and we found that the limited dataset and insufficient shooting perspective might lead to poor robustness of the model. It may be feasible to enrich the dataset with richer shooting angles or to train the model for some specified shooting views.

For the Liquid Tracking task, we have done some basic works of subtask1. We carried out three stages of work in turn. First, we tried to design a basic model. Second, we tried to use some simple methods to improve the performance of the basic model. Third, we tried to use the MOT method to improve the performance of the model.

We also got some results. Although these results are not so good, we can get the next improvement direction through the analysis of these results. For example, our model performs well for medium and large objects. So in the next step, we will pay more attention to the detection of small objects.

At the end of the task, we predicted that the key points of the human body and the interaction between the human and the container would be the key to the next step of improvement.

Author Contribution

The specific ideas are derived from our discussion and exchange. In this course project, Chengying did the experiments of Trans2Seg and analysis for Instance Segmentation, and wrote the main body of this report. Xiangcan did the experiments and analysis for Liquid Tracking and reviewed the related work.

References

[1] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2, 10

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 2, 9

[3] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 7

[4] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2, 9

[5] Balaji Veeramani, John W Raymond, and Pritam Chanda. Deepsort: deep convolutional networks for sorting haploid maize seeds. *BMC bioinformatics*, 19(9):1–9, 2018. 2

[6] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995. 2, 10

[7] Enze Xie, Wenjia Wang, Wenhai Wang, Mingyu Ding, Chunhua Shen, and Ping Luo. Segmenting transparent objects in the wild. In *European conference on computer vision*, pages 696–711. Springer, 2020. 2

[8] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021. 2, 5

[9] Enze Xie, Wenjia Wang, Wenhai Wang, Peize Sun, Hang Xu, Ding Liang, and Ping Luo. Segmenting transparent object in the wild with transformer. *arXiv preprint arXiv:2101.08461*, 2021. 1, 2, 5

[10] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. In *European Conference on Computer Vision*, pages 1–21. Springer, 2022. 2, 7, 10