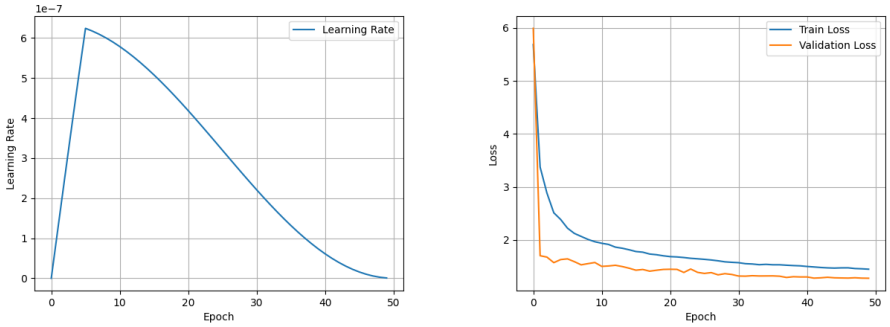
A Training details

In our method, we chose the ViT-L/16-based model of SAM (Kirillov et al., 2023) to attempt to balance the speed-accuracy trade-off. We observed that this model gave high-quality segmentation masks while being $\sim 2\times$ smaller compared to ViT-H/16 in terms of the number of parameters. On the other hand, for the CLIPSeg (Lüddecke and Ecker, 2021) model, we used the ViT-B/16 based model, with $reduce\_dim = 64$. Throughout our training process, we freeze the weights and biases of both foundational models, SAM and CLIPSeg.

Throughout the training phase, our text prompt remains "cables," with the aim of obtaining instance segmentation for *all* the cables within the image. During training, we conduct augmentations applied to the input images. These augmentations encompass random grayscale conversion, color jitter, and patch-based and global Gaussian blurring and noise. In terms of computing, our model is trained using 2 NVIDIA A5000 GPUs, each equipped with 32 GB of memory. All testing is carried out on a single NVIDIA RTX 2080 GPU with 16 GB of memory.

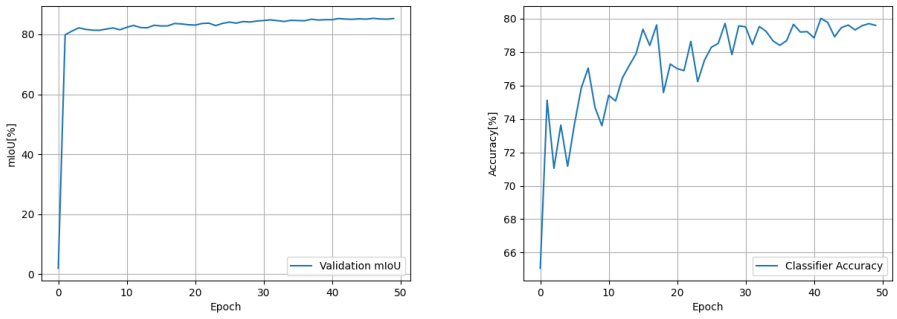
Our training procedure employs a learning rate warm-up spanning 5 epochs, followed by a cosine decay. The peak learning rate is set to $lr = 0.0008$, in line with recommendations from Kirillov et al. (2023). We employ the default *AdamW* optimizer from Paszke et al. (2019) with a weight decay of 0.01. The convergence graphs and learning rate profile of all the models can be seen in Figure 7. Figure 8 displays the binary classification accuracy and mIoU computed on the validation dataset during training. No smoothing was applied in creating the plots.

In the training process, we apply augmentations such as blurring, color jitter and random grayscale, to enhance generalizability across various DLO colors.



(a) Learning rate profile (b) Convergence graphs

Figure 7: Learning rate profile and convergence graphs



(a) mIoU[%] (b) Binary classifier accuracy[%]

Figure 8: Binary classification accuracy and mIoU

A brief summary of all the *hyperparameters* can be seen the Table 4

Hyperparameter	Value
Number of epochs	50
Max learning rate	8×10^{-4}
Learning rate warmup	5 (epochs)
Optimizer	*AdamW*
Optimizer weight decay	0.01
Batch size	1
Attention dropout	0.5
Number of prompts per batch (N)	11
Number of points per prompt (N_p)	3
Number of attention heads (for all models)	8
Embedding dimension	256
SAM model type	*ViT-L/16* (frozen)
CLIPSeg model type	*ViT-B/16* (frozen)
Focal loss weight	20
DICE loss weight	1
Positive label weight (binary cross-entropy)	3
Classifier MLP activation	*ReLU*
Prompt encoder MLP activation	*GELU*
Train dataset size	20038
Validation dataset size	3220
Test dataset size	3233
Image size	(1920×1080)
Total number of parameters (including foundation models)	466M
Trainable parameters	3.3M

Table 4: Hyperparameters

B Generated dataset

The dataset we presented contains 20038 train images, 3220 validation images, and 3233 test images. Each image we provide is accompanied by its corresponding semantic mask and binary submasks for each DLO in the image. The images are located in `{train,test,val}/RGB`, and named as `train/RGB/00000_0001.png`, `train/RGB/00001_0001.png`, and so on. In the `{train, test, val}/Masks` folder, we have a sub-folder containing the binary submasks for each corresponding RGB image. For example, `train/Masks/00000` contains all the submasks corresponding to `train/RGB/00000_0001.png`. Additionally, the semantic mask for `train/RGB/00000_0001.png` is called `train/Masks/00000_mask.png`. The folder structure can be seen in Figure 9.

Each image and its corresponding masks are 1920×1080 in resolution. The number of cables, their thickness, color, and bending profile are randomly generated for each image. There are 4 possible colors for the cables - *cyan, white, yellow, black*. The number of cables in each image is randomly sampled from 1 to 10. A sample image from the dataset, along with its corresponding submasks and semantic mask, are portrayed in Figure 10.

C Ablation study

The structure of our system is defined by two principal components: the *prompt encoder network* and the *classifier network*, each distinguished by their unique functions and designs. The *prompt encoder network* incorporates a self-attention layer, a sampler attention (cross-attention) layer, a filtering multi-layer perceptron (MLP), and a linear layer for prompt labeling, all critical to its functionality and in their minimal form.

In contrast, the *classifier network*'s architecture is more complex and requires detailed exploration through ablation studies. Our modular approach allows for the independent evaluation of both the classifier and prompt encoder networks. We experimented with multiple configurations of the

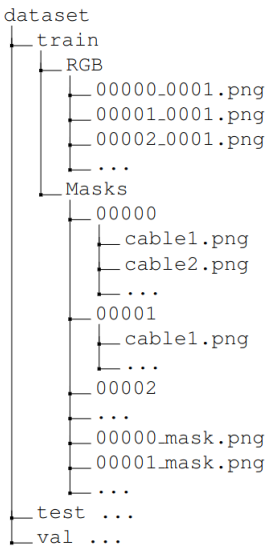


Figure 9: Dataset directory tree

Figure 10: Dataset Example

classifier network to determine the best setup based on classification accuracy on a test dataset. This section will outline the various configurations tested for the classifier network and the resulting labeling accuracies, with a detailed summary provided in Tab. 5.

The *classifier network* is tasked with processing two particular kinds of information: duplication of instance masks and detecting incorrect or low-quality instance masks. These instance masks must undergo a self-attention operation to extract relative information before classification. Initially, we attempted to route these mask tokens through a self-attention layer followed by an MLP classifier. This approach, however, did not converge, as indicated in **A1** in Tab. 5.

Inspired by DETR Carion et al. (2020) and further developed in implementations such as MaskFormer Cheng et al. (2021), we examined the application of learnable token embeddings for classification. Adopting DETR's framework, we initialized N trainable *classifier tokens*. We passed them through a self-attention layer, a cross-attention layer for merging with the mask tokens, and an MLP for classification, achieving a binary classification accuracy of 76.37%, documented in **A2** in Tab. 5. A slight modification in this setup, specifically the reordering of the self and cross-attention layers, resulted in an improved accuracy of 81.18%, as documented in **A3** in Tab. 5.

Deformable-DETR Zhu et al. (2020) introduced the concept of utilizing localized object queries instead of randomly initialized ones. In **A4**, we substituted the randomly initialized tokens with point prompt embedding directly selected by the prompt encoder without applying any intermediate transformations, leading to an accuracy of 80.26%. Furthermore, by applying an MLP transformation to these queries before their integration into the cross-attention layer, as demonstrated in **A5**, we achieved our highest accuracy of 84.83%. Consequently, **A5** was selected for all further experiments, showcasing its effectiveness in classification accuracy.

Config	Queries	Attention Order	Binary Accuracy[%]
A1	Mask Tokens	Only SA	NA(diverged)
A2	Trainable Tokens	SA-CA	76.37
A3	Trainable Tokens	CA-SA	81.18
A4	Point Prompt Tokens	CA-SA	80.26
A5	Point Prompt Tokens (MLP)	CA-SA	**84.83**

Table 5: Ablations of the classifier network (values in bold signify best performance). SA and CA stand for self-attention and cross-attention respectively. The Keys and Values in the CA are always the mask tokens

D Limitations of Foundation Models

The original SAM framework offered three variants based on ViT-B, ViT-L, and ViT-H. We opted the ViT-L model to achieve a balance between computational speed and model performance. Nevertheless, incorporating the ViT-H variant could further enhance our model's performance.

CLIPSeg, designed to effectively manage point prompt embedding, sometimes fails to generate precise heatmap embeddings for specific text prompts, as evidenced in Fig. 11. This issue occasionally impacts the model's performance and its ability to generalize across different scenarios. Looking ahead, we intend to explore the inherent limitations of the backbone models on our setup.

In addition, in the absence of ground truth masks, our model's performance is contingent on the accuracy of the classifier network. In Appendix C, we note that the classifier network currently achieves a peak binary accuracy of 84.83%. Future iterations of our model or similar research endeavors will need to focus on enhancing the classifier network to improve the overall model performance..

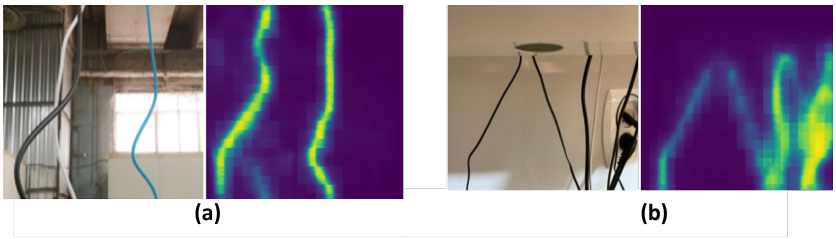


Figure 11: CLIPSeg generated heatmaps when using the prompt "cables." Example (a) shows unsuccessful detection of the white cable by CLIPSeg. Example (b) demonstrates a successful detection of all the cables in the frame.

E More qualitative results

Figures 12 and 13 show more instance segmentation masks generated by ISCUTE, on datasets from RT-DLO and mBEST as well as on our generated test dataset, respectively. All the images are examples of masks generated **without oracle**.

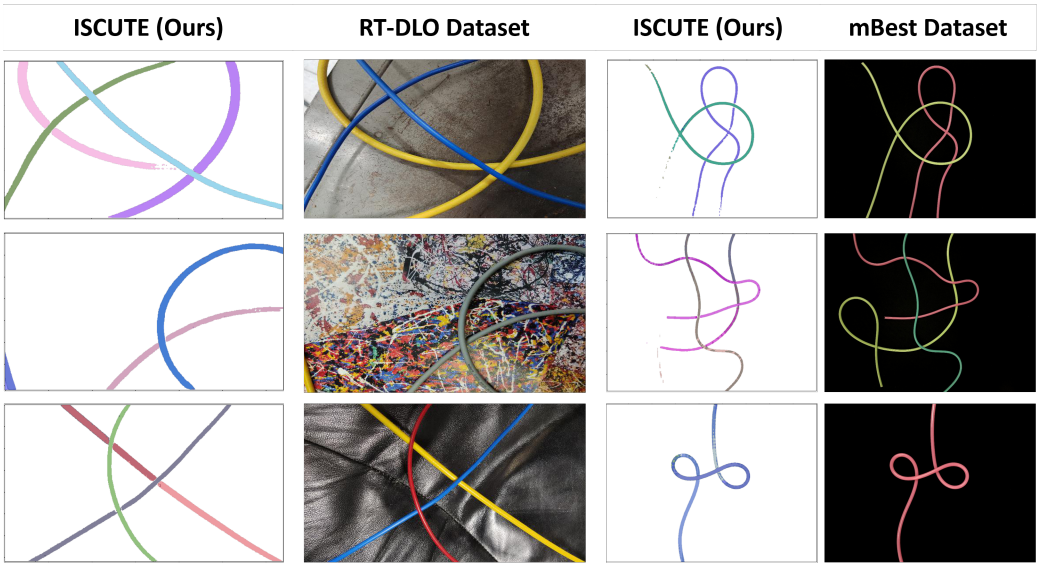


Figure 12: Qualitative results on RT-DLO and mBEST images

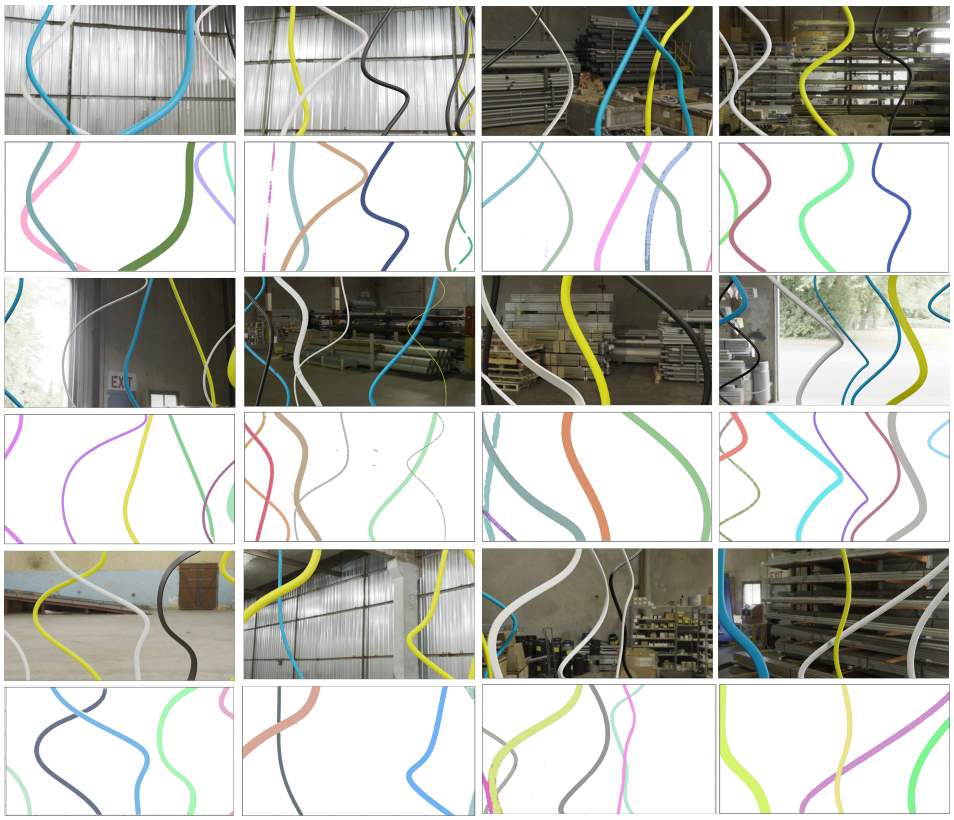


Figure 13: Qualitative results on our test images