Vision Foundation Models for an embodiment and environment agnostic scene representation for robotic manipulation

Kevin Riou[1], Kevin Subrin[1] and Patrick Le Callet[1,2]

***Abstract*— Traditional Imitation Learning (IL) approaches often rely on teleoperation to collect training data, which ensures consistency between training and deployment action and observation spaces. However, teleoperation slows data acquisition, distorts expert behavior and data can be affected by the lack of teleoperation skills. To overcome these limitations, IL training on human demonstrations requires visual representations that are agnostic to both embodiment and environment. Recent advancements in Vision Foundation Models, such as Grounded-Segment-Anything (Grounded-SAM), offer a solution by extracting meaningful scene information while filtering out irrelevant details without manual annotation. In this work, we collected 50 human video demonstrations of a manipulation task from the RLBench benchmark. We evaluated Grounded-SAM's ability to automatically annotate objects of interest and proposed a 3D visual representation using depth maps. This representation was used to train a diffusion policy, which successfully generalized to simulated robot deployment in RLBench, despite being trained exclusively on real-world human demonstrations. Our results demonstrate that efficient training can be achieved with just 50 demonstrations and half-an-hour training time.**

I. Introduction

Robotic manipulation learning is essential for equipping robots with complex skills without extensive programming effort. Two common training paradigms are Reinforcement Learning (RL) and Imitation Learning (IL). RL needs many interactions with the environment, which isn't always practical in real-world settings, and designing rewards can be more labor-intensive than programming the task directly. IL, particularly Behavior Cloning, offers a simpler alternative by training a policy from expert demonstration data without requiring interaction with the environment.

Most behavior cloning approaches collect their datasets by recording teleoperated demonstrations [13]. This is a practical scenario for the policy training, since the observation and action spaces are the same for the expert and the learner. However, this is not ideal for real-life scenarios for several reasons. Firstly, Mandlekar et al. [12] showed that the lack of skill of the expert in teleoperation can negatively impact the performance of the learner. Secondly, a policy trained on a dataset specific to one robot might not generalize well to other robots. The teleoperation process is also intrusive and time-consuming, especially for those unfamiliar with the technology, limiting real-world adoption.

Several studies have addressed the human-to-robot imitation learning (IL) problem, mostly by focusing on affordances. For instance, Bahl et al. [1] detect the first hand-object contact in the human demonstrations, and treat the corresponding hand position as an affordance, that a robot should also reach to perform the same task. They train a model to predict such affordance for the first image of the demonstrations, when the human is not visible in the video yet, to prevent the model from being biased an human operators. However, this restricts their approach to specific camera angles where the human is not visible initially and to tasks that can be resolved with one hand-object interaction only. Training a model from image further bias it to the background environment visible in the demonstrations.

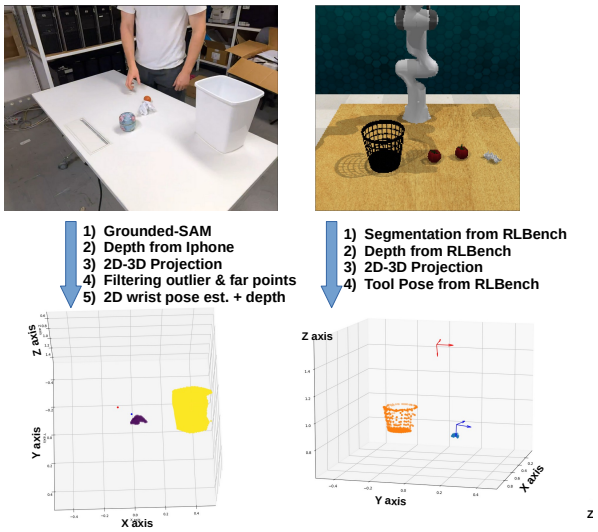

Fig. 1: Our visual representation uses open-vocabulary object detection and segmentation (Grounded-SAM) to represent a scene, focusing only on objects of interest and the hand/tool position, regardless of the operator or the environment.

Two main limitations are therefore hindering the development of human-to-robot IL. First, the **lack of visual representations that are agnostic to the embodiment and to background environment, which would allow to train a policy on human demonstrations from a given background environment, and deploy it on a robot in new environments**. Second, the lack of public benchmarks that can provide both human demonstrations of manipulation tasks along with a publicly available simulation featuring the same tasks.

In this work, we selected one task from the publicly available simulated benchmark RLBench [6], and we collected a dataset of human demonstrations for this task. We collected

[1]Nantes Université, Ecole Centrale Nantes, CNRS, LS2N, UMR 6004, Nantes, France. {kevin.riou, kevin.subrin, patrick.lecallet}@univ-nantes.fr
[2]Institut Universitaire de France (IUF)

50 demonstrations of the task ”put rubbish in bin”, recorded using an iPhone 14 pro.

We leveraged recent advancements in **open-vocabulary object detection [9] and segmentation [8], [16]** to build a scene representation focused on relevant objects, filtering out background and operator embodiment. Specifically, Grounded-SAM [9] was used to segment target objects based on their textual descriptions. These segmentations were then projected onto a 3D point cloud using iPhone depth maps, as shown in Fig. 1.

Using our data, we evaluated 2 things. 1) The ability of the Grounded-SAM to find the right objects, with different prompting strategies. 2) The ability of a diffusion policy equipped with our representation to deploy on a simulated robot while being trained on real human demonstrations. Our representation allowed to achieve a promising 20% success rate from those **50 demonstrations only**, and in a **less than 30 minutes of training**.

Overall, this work showcases the **potential of Vision Foundation Models to extract meaningfull information from the scene, enabling 0-shot transfert to new environments or new embodiments** and paves the way for the development of new benchmarks, visual representations, and learning paradigms around these problems.

II. Dataset and annotation strategy

A. Content

50 demonstrations of the task ”put rubbish in bin” were collected using a moving iPhone 14 pro (carried by an external operator), providing RGB images and depth maps of the scene from various viewpoints. In the RLBench task, the robot is required to pick up a piece of rubbish from the table and place it in a trash bin. The rubbish is a small crumpled piece of paper, and is always accompanied by two distractors, which are other objects that the robot should not interact with. In the demonstrations that we collected, we included various distractors, but also several distinct trash-bins and pieces of crumpled paper as trashes. The set of objects present in the scene in the human demonstrations is shown in Fig. 2.

The intuition behind using a moving camera is to provide data for training viewpoint-agnostic deep-learning policies. If all data are recorded from a fixed viewpoint, the trained policy will be biased toward that perspective. In contrast, using a moving camera captures images from different viewpoints, forcing the policy to generalize across a variety of perspectives.

B. Action annotations

An IL dataset is composed of pairs of observations and corresponding actions. The first step in the annotation process is to extract actions from the human demonstrations. On the robot side, the actions should correspond to the position and orientation that the gripper should reach, from a given state of the scene. Additionally, the action encompasses whether the gripper should be opened or closed after reaching the target position. Therefore, the actions from the human

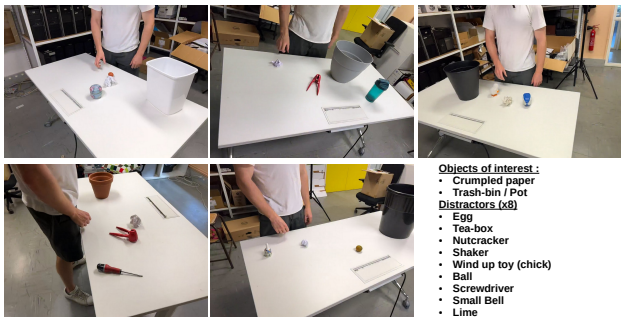

Fig. 2: Visualization of the objects present in the scene in the human demonstrations.

demonstrations should similarly correspond to the position and orientation that the human hand reaches next from a given state of the scene.

We therefore annotated the hand poses in the collected human demonstrations using the Keypoint-Fusion [10] model, that was trained to extract the 3D position of 21 keypoints of the human hand from RGB images and depth maps. Since the hand pose estimation model provided quite noisy predictions on our data, we defined the tool position as the average of the two furthest keypoints of all fingers to mitigate prediction errors. However, defining the orientation of the hand would require to rely on individual keypoints to define a frame in the hand. Since the hand pose estimation used is too noisy for that purpose, the tool’s orientation was kept orthogonal to the table, which is reasonable for the task considered in this study, since objects are always grasped from the top.

At that point, we have extracted dense sequences of hand poses followed by the human operators during the demonstrations. However, training a robot control policy to mimic human trajectories is suboptimal due to the morphological differences between humans and robots. Nevertheless, both humans and robots share similar action primitives, such as ’reaching a grasping position’ and ’reaching a releasing position.’ To address this, we annotated the start and end timestamps of these two primitives in the human demonstrations. Additionnally, as depicted on Fig. 3, when sending the trash to the bin, the operator was systematically passing through a highest point before moving down to the releasing point, in order to avoid a collision with the bin. We annotated the trajectory from the preceding grasp to the highest point as an ”avoid collision” primitive. In section III, these primitives will allow us to train policies that predict these embodiment-agnostic primitives, rather than dense, embodiment-specific action sequences. Finally, the gripper is initially annotated as ”open” at the start of each demonstration. It is then marked as ”closed” once the ”reach and grasp” primitive is completed. Similarly, the gripper remains annotated as ”closed” until the ”reach and release” primitive is completed.

C. Observation annotations

The second step in the annotation process is to extract a visual representation from the observations of the human demonstrations. We annotated the objects of interest in the

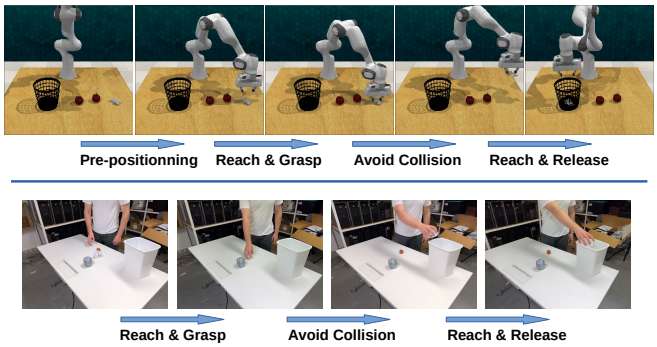

Fig. 3: Comparing robot primitives obtained from the RL-Bench demontration generator and human primitives annotated from the human demonstrations.

scene using the Grounded-SAM framework [16]. It combines two models. The Grounded-Dino [9], an open-vocabulary object detection model, allows to detect objects of interest in the images by specifying their names with an input prompt. The Segment-Anything (SAM) model [8] further segments the objects detected by the Grounded-Dino model. We only segmented the first image of the video, and subsequently tracked the segmented objects using the Cutie tracker [2]. Initially, we prompted Grounded-SAM with the names of the target objects, "crumpled paper" and "trashbin,". We evaluated the model's detection performance using precision, which measures how many of the detected objects are actually of interest, and recall, which measures the number of the actual objects of interest that were correctly detected. We focused the evaluation of Grounded-Dino on detection performance, measured by precision and recall, since segmentation was nearly perfect when the correct objects were detected.

However, as shown in Table I, the initial naive prompt resulted in numerous false positives for "crumpled paper," with objects like a white wind-up toy being misclassified, leading to low precision. Additionally, the trashbin was occasionally missed, also causing poor recall. To improve detection, we enhanced the prompt by including the names of all objects in the scene and subsequently filtering the results by name. This allowed Grounded-SAM to correctly dissociate similar objects, such as the white wind-up toy and the "crumpled paper", under different labels. We also added "pot" to the prompt to increase the likelihood of detecting the trashbin. This refinement improved detection accuracy, with 76% of the demonstrations having all target objects correctly detected without the need for human annotation (Table I). For the remaining 24%, we used the interactive segmentation feature of SAM [8].

The pixels corresponding to the objects of interest in the scene were then projected to the 3D space using the depth maps and known intrinsic parameters of the iPhone 14 Pro camera. This process generates a point cloud representation of the scene, containing only points from objects of interest. Each point is a 4D vector with its 3D position in the camera coordinate system and a scalar indicating the object's

TABLE I: Grounded-SAM performances on the first images of our 50 demonstrations.

	Trashbin		**Crumpled Paper**		**Succ. Rate**
	Recall	Precision	Recall	Precision	
Naive Prompt	0.84	0.95	0.9	0.79	52
Enhanced Prompt	0.92	1.0	0.9	0.92	**76**

category. This representation filters out the background, the operator, and distractors. The tool's position (human hand) is also provided to the policy to help locate the operator in the scene.

Note that an observation and its corresponding action must be defined within the same coordinate system. However, in our case, the coordinate system —which is the camera's frame— is moving during demonstration acquisition. As a result, an observation at time t and its corresponding action, for e.g., the hand pose at time t+10, will not initially be in the same coordinate system. By using the camera's odometry, we can recover the transformation between the action frame and the observation frame, allowing us to project them into the same coordinate system.

III. Primitive based behavior cloning

1) Behavior cloning in fixed horizon settings: We formalize our dataset as a set of demonstration trajectories $D = \{\tau_i\}_{i=1}^{N}$, where each trajectory τ_i is defined as a sequence of observation-action pairs $\tau = \{(o_t, a_t)\}_{t=1}^{T_i}$. Training a policy π using behavior cloning on a fixed action horizon of 1 time step, and equipped with a visual representation function ϕ is equivalent to solving the optimization problem defined by Equation 1.

$$\theta^* = \arg\min_{\theta} \sum_i \sum_t l\left(\pi\left(\phi(o_t); \theta\right), a_t\right). \tag{1}$$

In Equation 1, θ represents the learnable parameters of the deep learning policy and l is the loss function that seeks to minimize the difference between the predicted action $\pi\left(\phi(o_t); \theta\right)$ and the ground truth action a_t. Here a_t can be fully defined as the tuple

$$a_t = (x_{tcp}^{t+1}, y_{tcp}^{t+1}, z_{tcp}^{t+1}, Grip.State^{t+1}), \tag{2}$$

where $(x_{tcp}^{t+1}, y_{tcp}^{t+1}, z_{tcp}^{t+1})$ is the 3D position of the robot's tool center point (TCP) at timestep t+1. As mentioned earlier, the orientation of the gripper will be fixed, orthogonal to the table for the considered pick-and-place task. $Grip.State^{t+1}$ is the opening state of the robot's gripper (open/closed).

The policy can be trained to predict not just the next action, but the next "h" actions, enhancing its planning capabilities. Chi et al. [2] extended this by training a diffusion policy to predict the next "h" actions but only executing the first "a" actions, balancing long-term planning with reliable short-term execution. After a hyperparameter search, they found that "h=16" and "a=8" worked best for tasks using a transformer-based policy trained on teleoperated demonstrations.

2) Behavior cloning in primitives based settings: In the case of primitives based actions [5], [7], the trajectories can be reformulated as $\tau' = \{(o_t, p(t))\}_{t=1}^{T}$., where $p(t) = a_{min(k>t, \forall k \in \{k_1, k_M\})}$ represents the 3D position and opening state of the gripper at the end of the ongoing primitive that is being performed in the scene at time t. $\{k_1, k_M\}$ is the set of timesteps that correspond to the end of the M primitives. Equation 3 defines the primitive-based optimization problem.

$$\theta^* = \arg\min_{\theta} \sum_i \sum_t l\left(\pi\left(\phi(o_t); \theta\right), p(t)\right). \quad (3)$$

3) Two solutions for collision avoidance: In section II-B, we detailed three annotated primitives from human demonstrations: 1) 'reaching a grasping position', 2) 'reaching a releasing position', and 3) 'reaching a point to avoid a collision'. The third primitive can be treated either as a separate action or as part of the reaching actions, where the goal is to avoid collisions while reaching grasp/release points. We implemented two solutions: 1) predicting the 'avoid collision' primitive independently (1-step solution), and 2) training the policy to predict collision-avoidance points along with the grasp/release points of primitives 1 and 2 (2-step solution). For the second solution, if there is no collision to avoid, the point is set midway between the start and end of the reaching and grasp/release primitive.

4) Implementation details and evaluation metrics: The demonstrations were split 80% for training and 20% for validation. We trained the transformer-based diffusion policy from Chi et al. [3], with the configuration they used for the "low dim push-t task", using either our 4D point cloud ('4D Pt.Cl.') or Value-Implicit Pre-training ('VIP') [11], which has shown superior performances for robotic manipulation compared to prior pre-trained visual representations [14], [4], [15]. All policies are trained for 3000 epochs. Every 1000 epochs, they are evaluated on the training and validation sets using Euclidean distance between predicted and ground-truth tool positions (Train/Val Pos. Err), assessing performance on human data first. Note that the errors arising from "avoid collision" keypoints are reported separately (Av. Pos. Err). Afterwards, the policy undergoes 50 simulated rollouts on a robot in RLBench, with front and overhead cameras, and the corresponding success rates (Suc. Rate Front/Overhead) are calculated. We report the best success rate and corresponding position errors among all evaluations.

IV. Human to robot performances

In Table II, we observe that fixed-horizon action prediction ("Dense") results in significantly lower position accuracy during training, compared to Keypoint-based actions ("Two-Step Keypoints"), leading to a 0% success rate during deployment. This may be due to high uncertainty in hand pose estimation, possibly exceeding the distance between successive hand positions.

Regarding the Two-Step Keypoints, while the VIP representation achieves similar tool position accuracy to our 4D Pt.Cl. representation during training, it fails to generalize to simulated robot deployments. In contrast, the 4D Pt.Cl. representation achieves a 20% success rate. Their is a significant difference in terms of missing parts and noisy points in the point-clouds obtained using the iPhone sensors and those obtained directly from RLBench, which is probably leading to this relatively low success rate. It would be interesting to deploy the model on a real robot equipped with the iPhone sensors to validate this hypothesis.

		Train Pos. Err. (mm)	Train Av. Pos. Err. (mm)	Val Pos. Err. (mm)	Val Av. Pos. Err. (mm)	Suc.Rate Front (%)	Suc.Rate Overhead (%)
Two-Step Keypoints	VIP	8.4	10.5	283.0	263.0	0	0
	4D Pt.Cl.	9.0	10.0	128.0	167.0	**20**	0
Dense	VIP	27.2	22.4	154.3	154.3	0	0
	4D Pt.Cl., h=1, a=1	28.4	-	54.0	-	0	0
	4D Pt.Cl., h=16, a=8	261.0	-	328.1	-	0	0

TABLE II: Results of policy training on human demonstrations and deployment on simulated robot.

Table III demonstrates that using two-step primitives significantly improves deployment performance. In contrast, treating "avoid collision" as an independent primitive often led the policy to get stuck around the "avoid collision" keypoint, instead of switching to the "reach and release" primitive. Additionally, the choice of human pose estimation method is crucial, improving the success rate by 4% compared to simply projecting 2D poses with depth maps. Adding a pre-positioning primitive before each grasp further enhances deployment success. However, all models fail to generalize to the overhead view, which contains strong self-occlusions with the robot.

Kpts type	Pre-Grasp Positions	Pose est. method	Train Pos. Err. (mm)	Train Av. Pos. Err. (mm)	Val Pos. Err. (mm)	Val Av. Pos. Err. (mm)	Suc. Rate Front (%)	Suc. Rate Overhead (%)
1-step	No	2D + depth	11.8	13.6	188.1	221.4	0	0
2-step	No	2D + depth	13.8	18.0	128.3	171.0	14	0
2-step	**Yes**	2D + depth	22.2	30.7	126.0	165.4	16	0
2-step	**Yes**	**RGB-D model**	9.0	10.0	128.0	167.0	**20**	0

TABLE III: Ablations with the 4D Pt.Cl. representation

V. Conclusion

The Grounded-SAM model successfully detected all objects of interest in 76% of the 50 demonstrations in our dataset. While this performance is insufficient for direct deployment, it could allow to fine-tune a lighter segmentation model in a few-shot manner [19], [17], [18]. The 4D Pt.Cl. representation used as input for a diffusion policy achieved a 20% success rate when trained on real-world human videos and deployed on a simulated robot, despite significant embodiment, environment and sensor shifts. It would be valuable to explore the impact of incorporating these successful examples into the training data to provide the model with target domain samples, and assess the effects on deployment performance. Data from alternative viewpoints—a fixed camera and an egocentric perspective—were also collected during the 50 demonstrations. Future work should also explore these viewpoints. Lastly, a crucial area for future research is evaluating the model's ability to predict tool orientations.

REFERENCES

[1] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023.

[2] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. Putting the object back into video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3151–3161, 2024.

[3] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[4] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[5] Stephen James and Andrew J Davison. Q-attention: Enabling efficient learning for vision-based robotic manipulation. *IEEE Robotics and Automation Letters*, 7(2):1612–1619, 2022.

[6] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[7] Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew J Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13739–13748, 2022.

[8] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[9] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[10] Xingyu Liu, Pengfei Ren, Yuanyuan Gao, Jingyu Wang, Haifeng Sun, Qi Qi, Zirui Zhuang, and Jianxin Liao. Keypoint fusion for rgb-d based 3d hand pose estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 3756–3764, 2024.

[11] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[12] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[13] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[14] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[16] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.

[17] Kevin Riou, Jingwen Zhu, Suiyi Ling, Mathis Piquet, Vincent Truffault, and Patrick Le Callet. Few-shot object detection in real life: case study on auto-harvest. In *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6. IEEE, 2020.

[18] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *proceedings of the IEEE/CVF international conference on computer vision*, pages 9197–9206, 2019.

[19] Xin Wang, Thomas Huang, Joseph Gonzalez, Trevor Darrell, and Fisher Yu. Frustratingly simple few-shot object detection. In *International Conference on Machine Learning*, pages 9919–9928. PMLR, 2020.