# AN ANALYSIS OF MULTIMODAL LARGE LANGUAGE MODELS FOR OBJECT LOCALIZATION IN EARTH OBSERVATION IMAGERY

**Darryl Hannan, John Cooper, Dylan White, Henry Kvinge, Timothy Doster, & Yijing Watkins**
Pacific Northwest National Laboratory
Seattle, WA, USA
{darryl.hannan,john.cooper,dylan.white,
henry.kvinge,timothy.doster,yijing.watkins}@pnnl.gov

## ABSTRACT

Multimodal large language models (MLLMs) have altered the landscape of computer vision, obtaining impressive results across a wide range of tasks, especially in zero-shot settings. Unfortunately, their strong performance does not always transfer to out-of-distribution domains, such as earth observation (EO) imagery. Prior work has demonstrated that MLLMs excel at some earth observation tasks, such as image captioning and scene understanding, while failing at tasks that require more fine-grained spatial reasoning, such as object localization. However, MLLMs are advancing rapidly and insights quickly become out-dated. In this work, we analyze more recent MLLMs that have been explicitly trained to include fine-grained spatial reasoning capabilities, benchmarking them on EO object localization tasks. We demonstrate that these models are performant in certain settings, making them well suited for zero-shot or limited data scenarios. We then directly compare their performance to a few-shot Faster RCNN, quantifying the amount of data that is needed to surpass MLLM performance in various settings. We hope that this work will prove valuable as others evaluate whether or not it is worth employing an MLLM for a given EO task and that it will encourage further research in utilizing these models in the overhead domain.

## 1 INTRODUCTION

A primary drawback of deep learning architectures is their reliance on large annotated datasets. This can be especially challenging when training a model on an earth observation (EO) task, where annotated data is costly to acquire and large, publicly available datasets are uncommon. Multimodal large language models (MLLMs) offer a potential solution to this problem, as they are strong, general-purpose models that achieve impressive performance across a variety of tasks in zero-shot settings (Dubey et al., 2024; Deitke et al., 2024; Team, 2025; Wang et al., 2024). However, this performance does not always transfer to EO tasks due to the domain gap between overhead imagery and the web images that are used for most MLLM training pipelines (Zhang & Wang, 2024; Hu et al., 2023). While some MLLMs have been fine-tuned specifically on EO data (Kuckreja et al., 2024; Irvin et al., 2024), the generalizability of these models often suffers, making their performance underwhelming on EO tasks that are out-of-distribution relative to the fine-tuning data.

Zhang & Wang (2024) sought to understand the extent to which the robust general knowledge of MLLMs could be transferred to the EO domain. They found found that for high-level tasks, such as image captioning or scene classification, MLLMs tended to perform reasonably well. However, when tasks required more fine-grained spatial reasoning, such as object counting or localization, MLLMs did not perform well. This result aligns with work outside of the EO domain, where it has been demonstrated that MLLMs struggle with these tasks on non-overhead imagery as well (Kaduri et al., 2024). This suggests that MLLMs are capable of leveraging their vast general knowledge and applying it to EO images, but that it is the general deficiency of struggling with fine-grained visual reasoning that produces results such as those observed in Zhang & Wang (2024).


Figure 1: Sample outputs using various MLLMs (green dots=ground truth and red Xs=predictions) across three tasks: building detection (left), animal detection (middle), and plane detection (right).

More recent MLLMs integrate explicit localization information directly into their training pipeline (Deitke et al., 2024; Team, 2025; Wang et al., 2024). These are relatively new capabilities that allow models to output localization coordinates. **We are the first work to benchmark these new models on the EO domain**, directly assessing whether prior results regarding the poor localization capabilities of MLLMs still hold. We evaluate the Molmo family of models (Deitke et al., 2024), the new Qwen 2.5-VL architecture (Team, 2025), and the recent Llama 3.2 model family (Dubey et al., 2024), across three different object localization tasks: plane detection, building detection, and animal detection (cropped examples of these tasks can be seen in Figure 1). We select Molmo and Qwen 2.5-VL because both explicitly include localization capabilities in their training pipeline, and we select Llama 3.2 as an experimental control because it was released contemporaneously with the other architectures but does not advertise localization capabilities. We directly compare the MLLMs' localization performance to a standard object detection architecture, Faster RCNN Ren et al. (2015), trained on various amounts of training data. While *this is not a fully fair comparison due to the MLLMs operating in a zero-shot setting*, it allows us to **precisely quantify how much data is needed for a typical object detector to beat an MLLM on EO tasks**, allowing future researchers to make more informed decisions about utilizing these models. Along with these quantitative results, **we also provide a qualitative evaluation of each models results via visualizations and a detailed discussion of various failure scenarios**.

## 2 Experiments

**Datasets** We evaluate the MLLMs on a variety of remote sensing object localization tasks. The first dataset that we consider is the RarePlanes dataset Shermeyer et al. (2021). We make this dataset a 1 class plane detection task, assessing each MLLM's ability to detect high-level object categories. The second dataset that we consider is the Aerial Animal Population (AAP) dataset Eikelboom et al. (2019), which consists of a set of high resolution images taken from a helicopter with small animals in the scene, belonging to one of three classes: elephant, giraffe, and zebra, assessesing each model's ability to detect and classify small objects. The last dataset that we consider is the xBD dataset Gupta et al. (2019) because its images are at a relatively high GSD, assessing the models ability to localize information in large scenes, while again challenging them to detect small objects. A more detailed description of these datasets, and our pre-processing, is available in the supplementary.

**Metrics** Molmo only outputs centerpoints Deitke et al. (2024), therefore we evaluate the models in a centerpoint regime. We implement a center mAP score, where object association is based upon pixel distance. We set this pixel distance based upon the size of the target(s) in each dataset. To evaluate Qwen 2.5-VL, which outputs bounding boxes, and Faster RCNN Ren et al. (2015), we convert the bounding boxes to centerpoints by considering the center of the box as the prediction.

**RarePlanes Results** The second column of Table 1 contains the results for each MLLM on the RarePlanes plane detection task. This is the easiest task out of the three due to the size and distinct shape of the target objects. We find that Molmo 72B performs the best out of all of the MLLMs. Note that both Llama models obtain a mAP of 0 due to the fact that all of its outputs are invalid. Even after exploring a variety of prompting strategies, it was difficult to get the model to output anything

| Model | RarePlanes mAP@30pix | AAP mAP@30pix | xBD mAP@15pix |
|---|---|---|---|
| Molmo 7B O | 62.62 | **30.26** | 2.97 |
| Molmo 72B | **72.12** | 29.82 | **4.22** |
| Qwen 2.5-VL 7B | 46.62 | 30.01 | 0.49 |
| Qwen 2.5-VL 72B | 50.03 | 12.09 | 0.50 |
| Llama 3.2 11B | 0.00 | 0.00 | 0.00 |
| Llama 3.2 90B | 0.00 | 0.00 | 0.00 |

Table 1: Object detection results for various MLLMs across three different datasets.

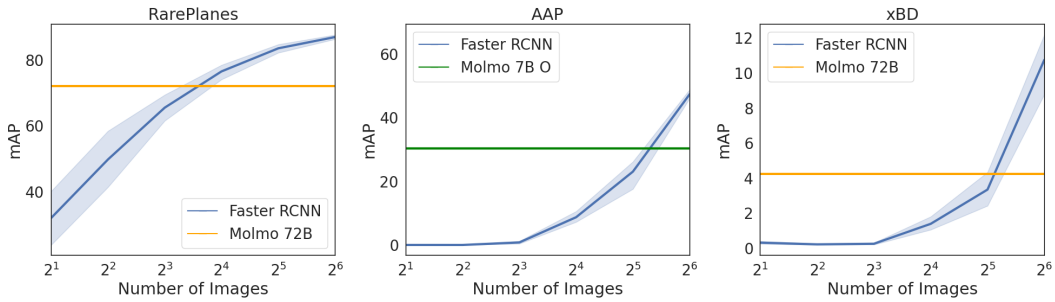


Figure 2: Few-shot Faster RCNN (Ren et al., 2015) performance with varying amounts of training images (blue lines) vs. top-performing MLLM's performance (alt-color lines) for each task.

reasonable, with it sometimes even generating bounding boxes outside the boundaries of the image. This suggests that Llama was never exposed to this type of data during training and, despite getting strong performance across many MLLM benchmarks, cannot easily generalize to localization tasks. Qwen 2.5-VL is the newest of the models that we evaluated and it performs the best for its size, beating Molmo 7B by close to 10 AP. However, even Qwen 72B falls short of Molmo 72B. The substantial jump in performance from Molmo 7B to 72B, compared to the relatively small performance jump from Qwen 7B to 72B, suggests that the performance of one model size within a model family is not indicative of the performance of another model size. Therefore, it is worth exploring all available sizes for a given family before drawing any conclusions about their capabilities.

Next, we compare the most performant MLLM, Molmo 72B, to a Faster RCNN, trained on randomly sampled subsets of data. Figure 2 (left) illustrates the performance observed as more data is added to the Faster RCNN's training dataset, while the horizontal line represents the performance of Molmo 72B. We can see that the performance of the Faster RCNN intersects with Molmo between 8 and 16 examples, indicating that this is the number of images needed to beat the zero-shot performance of Molmo on this task. This suggests that when a task is relatively simplistic and even just a few labeled examples are available, it may not be worth leveraging an MLLM. However, the performance that Molmo achieves on this task is impressive for a zero-shot model and, in data scarce settings, it is certainly still worth considering. Various visualizations are available in the supplementary, illustrating impressive successes as well as failures, including producing false positives on shadows, missing a plane when it is close to another, and missing planes that are partially obscured.

**Aerial Animal Population Results** The third column of Table 1 contains the results for the AAP animal detection task. This task is more challenging than RarePlanes due to the objects of interest being smaller and it being a multiclass problem. Due to this difficulty, each of the models perform worse relative to RarePlanes. LLama once again is incapable of localizing any of the objects in the scene. Surprisingly, Molmo 7B O outperforms all other models on this task, including Molmo 72B. This again suggests that rather than the size of the model predicting performance, there is likely a complex interaction between the original training data and the number of parameters in the model that determines its performance for a given task. For most models, 'elephant' has the highest AP, likely due to its size and the fact that it is less likely to be confused with the other targets.

Figure 2 (middle) contains results for Faster RCNN on this task. Like the MLLMs, Faster RCNN struggles with the task compared to RarePlanes. Notably, its performance does not intersect Molmo

7B until just over 32 examples are included during training. While acquiring 32 labeled images may be trivial for some tasks, this task is a great illustration of a scenario in which this might be challenging. Namely, the targets are relatively small, making them easy to miss for a human annotator, and, when tiled, there are a large number of images without targets in them. While some scenarios may require a high level of accuracy, thus necessitating this labeling, other scenarios may not require it, and in those scenarios, MLLMs are attractive. Even when high accuracy is desired, the MLLM could still be leveraged as an initial model for filtering empty scenes, then the labels could be refined and fed into a traditional detector. Figures illustrating the performance of Qwen on this dataset, along with detailed descriptions of various success and failure cases, are available in the supplementary. In many images, Qwen performs quite well, correctly identifying all or most of the animals. However, there are catastrophic failure scenarios. For instance, in some cases, Qwen only places a single box on a group of animals, substantially hurting average precision.

**xBD Results** The fourth column of Table 1 contains the results of each MLLM on the xBD building detection task. None of the models perform particularly well at the task. The Molmo family of models is the only one that outputs some valid predictions. Examples are included in the supplementary illustrating the challenging nature of this task. When buildings are sufficiently large, the model is able to accurately detect them. However, there are many small buildings in the image that are difficult to discern without zooming into the image, making this a challenge for all models, and even humans without zooming in and carefully scanning each image. The Faster RCNN performance is visible in Figure 2 (right). Even a trained detector does not perform well when up to 128 examples are included in the training set; this is another testament to the task's difficulty.

**Failure Scenarios and Limitations** While each of the MLLMs perform differently across each of the tasks, they share some common failure scenarios (outside of Llama 3.2 which fails to generate valid coordinates for almost every example). Across all tasks, models tend to produce more false negatives than false positives. Objects are more likely to be missed if they are very small, obscured, or close to other objects. One explanation for some of these failure cases is the fact that none of the models place points with extreme precision, likely leading the model to think that it placed a point on an object when in fact it was meant for the adjacent one. Another reason that might explain this propensity towards false negatives is that unlike traditional detection architectures, there is no notion of an object score; the model itself is deciding which regions in the scene most strongly correspond to the desired target category. Perhaps if a score was available, or we were able to lower the model's threshold for determining whether something was an object, some of these false negatives would be avoided. In some cases, false negatives result from too many objects being contained in the image. We notice that all of the models can only scale to a certain number of objects, likely due to the distribution of object counts in the training data. Beyond this number, the model stops generating predictions, even with sufficient output tokens. Generally, false positives tend to align with reasonable distractors. For instance, a large boulder or tree that looks like an elephant may be mistaken for one, or one target is missed within a tightly packed group of targets. However, there is an exception to this observation where the model will generate a long string of false positives (see Figure 1 in the supplementary), resulting in severe failure. We hope that future work may investigate these hallucinations further, better understanding and mitigating them.

## 3 Conclusion

In this work, we analyzed a suite of recent MLLMs across a variety of EO object localization tasks. We found that in certain scenarios, where objects are relatively distinct and are of sufficient size, that these models are capable of obtaining strong results. We then compared their performance to a standard object detector, precisely quantifying the amount of training data needed to match their performance, providing a starting point for others to determine whether these models might be useful to their use cases. Explicit object localization capabilities are relatively new to MLLMs, offering exciting possibilities as they are further developed. As MLLMs are applied to new tasks, such as computer control (OpenAI, 2024; Anthropic, 2024), fine-grained spatial reasoning capabilities will become more important to the companies that train these models. We believe that there will be an increased interest in localization moving forward and that the EO community will be able to benefit as new models are released. We hope that this work will get others interested in these models and encourage further work focused on leveraging and understanding them.

## ACKNOWLEDGMENTS

The research described herein was funded by the Generative AI for Science, Energy, and Security Science & Technology Investment under the Laboratory Directed Research and Development Program at Pacific Northwest National Laboratory (PNNL), a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy. This work was also supported by the Center for AI.

## REFERENCES

Anthropic. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. https://www.anthropic.com/news/3-5-models-and-computer-use, 2024.

Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art multimodal models. *arXiv preprint arXiv:2409.17146*, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jasper AJ Eikelboom, Johan Wind, Eline van de Ven, Lekishon M Kenana, Bradley Schroder, Henrik J de Knegt, Frank van Langevelde, and Herbert HT Prins. Improving the precision and accuracy of animal population estimates with aerial image object detection. *Methods in Ecology and Evolution*, 10(11):1875–1887, 2019.

Ritwik Gupta, Bryce Goodman, Nirav Patel, Ricky Hosfelt, Sandra Sajeev, Eric Heim, Jigar Doshi, Keane Lucas, Howie Choset, and Matthew Gaston. Creating xbd: A dataset for assessing building damage from satellite imagery. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 10–17, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Yuan Hu, Jianlong Yuan, Congcong Wen, Xiaonan Lu, and Xiang Li. Rsgpt: A remote sensing vision language model and benchmark. *arXiv preprint arXiv:2307.15266*, 2023.

Jeremy Andrew Irvin, Emily Ruoyu Liu, Joyce Chuyi Chen, Ines Dormoy, Jinyoung Kim, Samar Khanna, Zhuo Zheng, and Stefano Ermon. Teochat: A large vision-language assistant for temporal earth observation data. In *ICLR*, 2024.

Omri Kaduri, Shai Bagon, and Tali Dekel. What's in the image? a deep-dive into the vision of vision language models, 2024. URL https://arxiv.org/abs/2411.17491.

Kartik Kuckreja, Muhammad Sohail Danish, Muzammal Naseer, Abhijit Das, Salman Khan, and Fahad Shahbaz Khan. Geochat: Grounded large vision-language model for remote sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27831–27840, 2024.

OpenAI. Operator system card. https://cdn.openai.com/operator_system_card.pdf, 2024.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

Jacob Shermeyer, Thomas Hossler, Adam Van Etten, Daniel Hogan, Ryan Lewis, and Daeil Kim. Rareplanes: Synthetic data takes flight. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 207–217, 2021.

Qwen Team. Qwen2.5-vl, January 2025. URL https://qwenlm.github.io/blog/qwen2.5-vl/.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

Chenhui Zhang and Sherrie Wang. Good at captioning, bad at counting: Benchmarking gpt-4v on earth observation data. *arXiv preprint arXiv: 2401.17600*, 2024.

# A Experimental Details

**MLLM Details** We used the HuggingFace implementations for each of the LLMs that we considered (Wolf et al., 2020) and opted to use the instruct variants of each model. For Molmo, we used the standard release of Molmo 7B O and a 4-bit quantized version of Molmo 72B. For Qwen 2.5-VL 7B, we used the public code release and we used the 4-bit quantized version of Qwen 2.5-VL 72B. For Llama 3.2 11B, we used the release by Meta and we used a 4-bit quantized version of Llama 3.2 90B. All MLLMs were run on a single Nvidia H100 GPU. We set the max new tokens able to be generated to 800 tokens. We used the default settings for Llama 3.2 and Qwen 2.5-VL and for Molmo we specified a greedy sampling strategy for token generation. The prompts that we used for each model are available in Table 2.

| Model | Dataset | Prompt |
|---|---|---|
| Molmo 7B O | RP | "Where are the {category}?" |
| Molmo 7B O | AAP | "Point to {category}. Please say 'This isn't in the image.' if it is not in the image." |
| Molmo 7B O | xBD | "Place a point on each {category} in the image." |
| Molmo 72B | RP | "Place a point on each {category} in the image." |
| Molmo 72B | AAP | "Look for {category} in the image and show me where they are." |
| Molmo 72B | xBD | "Place a point on each {category} in the image." |
| Qwen 2.5-VL | All | "Detect all {category} in the image." |
| Llama 3.2 | All | "Detect all {category} in the image. Output the coordinates in the form: [x1, x2, y1, y2]." |

Table 2: The prompts that we used for each model to extract localization information across each task. We initially tried to match the prompts that were provided in the original papers and, in some cases, did our own additional tuning. {category} is the class of the target we are locating.

**Few-shot Details** To construct few-shot splits for each of our datasets, we randomly sampled $K$ images from the dataset, where $K$ is the number of shots that we are considering. We created 10 seeds for each K-shot split, each consisting of different images and a varying number of annotations. For RarePlanes, we trained our Faster RCNN using Detectron2 (Wu et al., 2019) for RarePlanes. The Faster RCNN has a ResNet-50 (He et al., 2016) backbone and we used the ResNet 50 weights provided by Detectron2 to initialize our model. We used the standard configuration provided by Detectron2, modifying the images per batch to 2, the base learning rate to 0.0025, and the ROI head batch size to 64. We trained each model on an Nvidia A100 GPU for 1000 iterations, as we find this sufficient for convergence. For AAP and xBD, we used MMDetection Chen et al. (2019) to train the Faster RCNNs. We used the standard COCO configuration for MMDetection and the available ResNet 50 pre-trained weights to initialize the model.

**Data Details** For both the Aerial Animal Population dataset and xBd, we use the pre-processed data from Zhang & Wang (2024). xBD is a satellite image dataset originally constructed for change detection. We convert it into an object detection dataset by only considering the first image in each example and asking the models to identify the buildings in the image. The RarePlanes (Shermeyer et al., 2021) scenes were tiled into 1333x800 tiles with a 200 pixel overlap. We also tile the test portion of the Aerial Animal Population dataset (Eikelboom et al., 2019) into 900x700 tiles with 200 overlap, based upon the original paper Eikelboom et al. (2019). Note that the training data was already tiled into roughly this size. For both of these datasets, the metrics that we present evaluates on individual tiles, rather than reporting results at the scene level.

# B Additional Figures

Figure 3: Image from xBD dataset with Molmo 72B labels (ground truth represented by green dots and predictions represented by red Xs). This illustrates the common failure scenario that is discussed in the main text, where models will sometimes generate a sequence of many detections in a line. We are uncertain what results in this behavior but we notice it more with small models.

Figure 4: Illustration of an example from RarePlanes using Molmo 72B (ground truth represented by green dots and predictions represented by red Xs). Here, the model successfully detects most aircraft in the scene, despite the terminal providing many distractors. It even detects one of the two aircraft that appear at the edge of the image, suggesting that it is able to detect parts of planes.

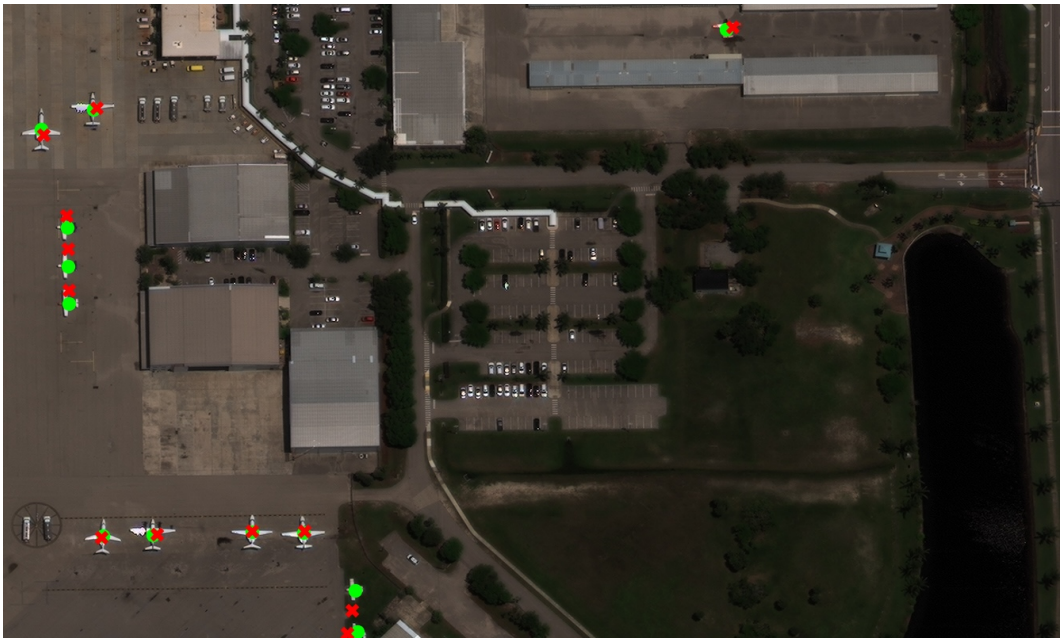

Figure 5: Illustration of an example from RarePlanes using Molmo 72B (ground truth represented by green dots and predictions represented by red Xs). Here, the model successfully detects all aircraft in the image, despite variations in size and orientation.

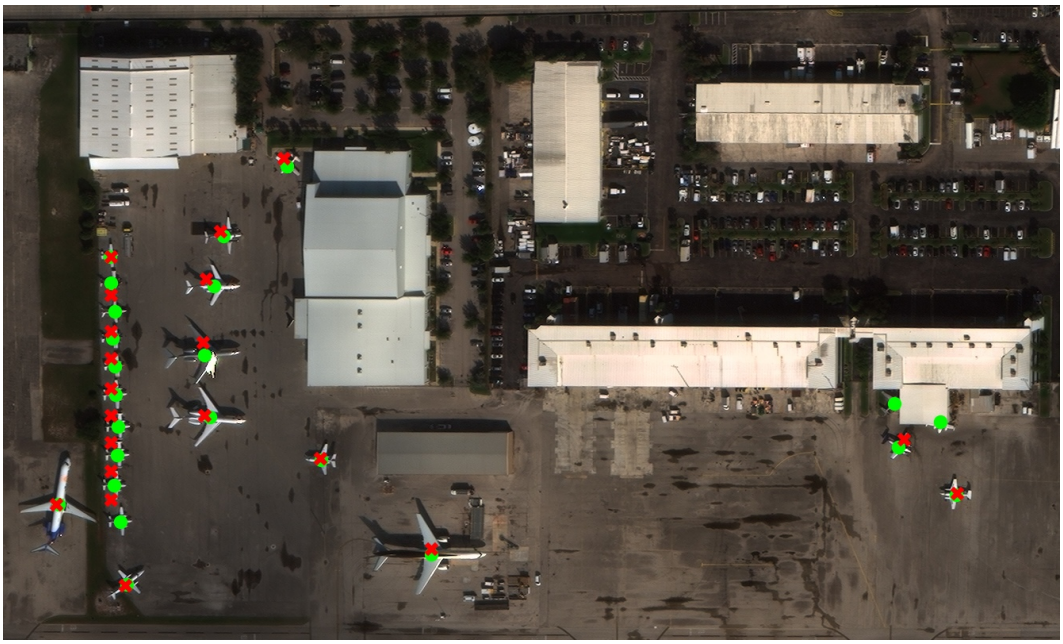

Figure 6: Illustration of an example from RarePlanes using Molmo 72B (ground truth represented by green dots and predictions represented by red Xs). The model successfully predicts most planes in the image, despite the large number of targets and potential distractors. It misses a plane that is in close quarters to other planes and it misses two planes that are partially obscured.

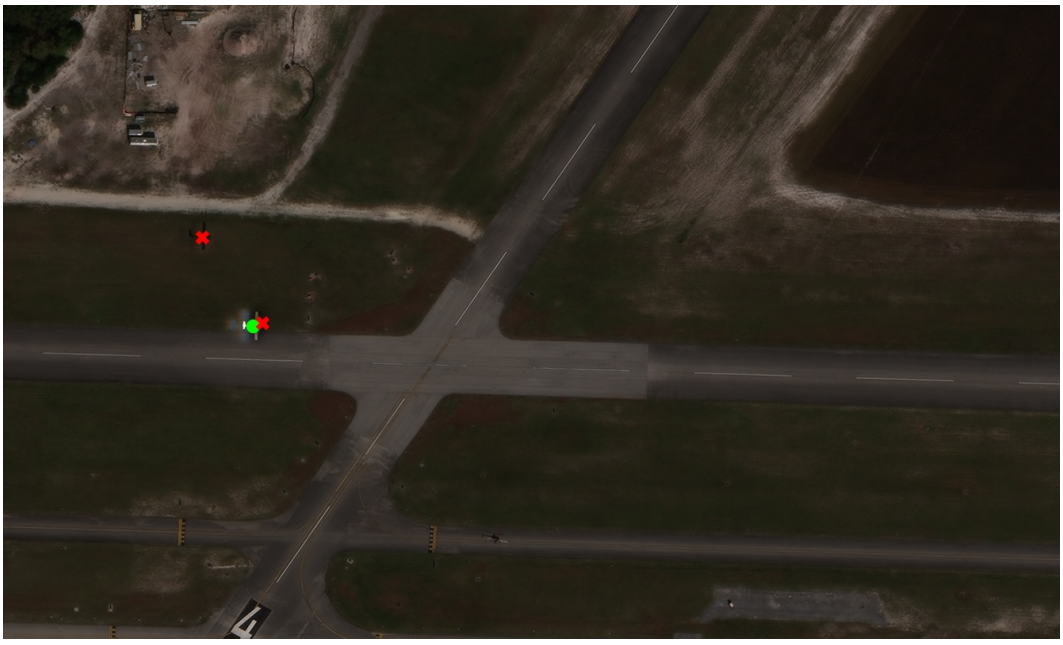

Figure 7: Illustration of a failure case on RarePlanes using Molmo 72B (ground truth represented by green dots and predictions represented by red Xs). An example of a failure where the model detects the plane's shadow as an additional plane. We noticed models making this mistake in other datasets as well.

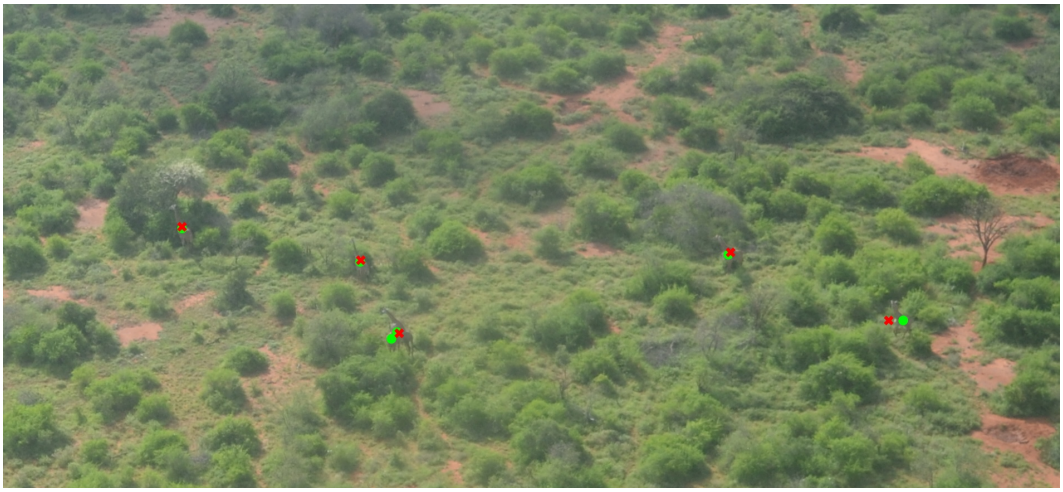

Figure 8: Zoomed in illustration of a success case on the Animal Population dataset using Qwen 2.5-VL 7B (ground truth represented by green dots and predictions represented by red Xs). Note how difficult it is to distinguish the giraffes from the background, nevertheless, Qwen successfully locates each one.

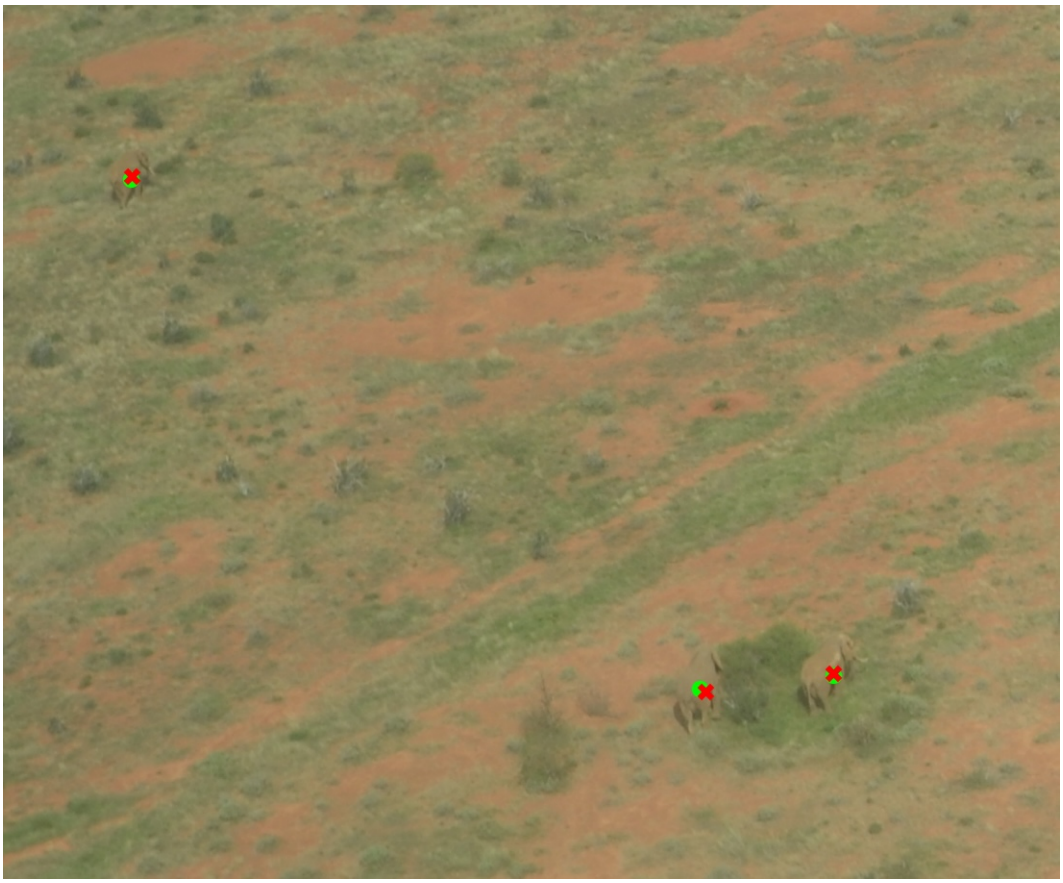

Figure 9: Zoomed in illustration of a success case on the Animal Population dataset using Qwen 2.5-VL 7B (ground truth represented by green dots and predictions represented by red Xs). Qwen successfully locates the elephants in the image. While they initially appear distinct, Figure 10 contains the zoomed out version of the image, where the elephants are no longer clear and are actually quite small relative to the scene.

Figure 10: Full image from the Animal Population dataset with Qwen 2.5-VL 7B labels (ground truth represented by green dots and predictions represented by red Xs).

Figure 11: Zoomed in image from the Animal Population dataset with Qwen 2.5-VL 7B labels (ground truth represented by green dots and predictions represented by red Xs). This illustrates a common failure case where Qwen places a point on a group of animals, rather than individually identifying each one. This substantially hurts the AP of the model.

Figure 12: Zoomed in image from the Animal Population dataset with Qwen 2.5-VL 7B labels (ground truth represented by green dots and predictions represented by red Xs). This is a less severe failure case, more aligned with the types of mistakes one might expect to see, where one individual in a group of animals is missed.

Figure 13: Image from xBD dataset with Molmo 72B labels (ground truth represented by green dots and predictions represented by red Xs). This is an example of a scenario where the model was relatively successful. It definitely possesses the required knowledge to detect buildings from an overhead angle. However, it misses certain buildings, especially when they are quite small, with some of them being fully subsumed by the small dots that we placed on the image.

Figure 14: Image from xBD dataset with Molmo 72B labels (ground truth represented by green dots and predictions represented by red Xs). Another scene in which Molmo was relatively successful. In this case, it again detected many of the homes, which are already relatively small, mostly missing smaller buildings which appear to be sheds or garages.

Figure 15: Image from xBD dataset with Molmo 72B labels (ground truth represented by green dots and predictions represented by red Xs). An exmaple of a failure scenario, where there are too many houses in the image for molmo to detect. It is worth noting that it does get many of the buildings. However, we found that even with expanding the generated token limit, Molmo falls apart past a certain number of objects in a given image, likely due to the distribution of object counts that it was trained on.

Figure 16: Image from xBD dataset with Molmo 72B labels (ground truth represented by green dots and predictions represented by red Xs). Another example of a failure scenario, where in this case the model incorrectly segments a larger building into multiple smaller buildings.