ZoomEarth: Active Perception for Ultra-High-Resolution Geospatial Vision-Language Tasks

Supplementary Material

6. LRS-GRO Dataset

6.1. Data Generation Details

The dataset construction process involves three distinct settings: **object-level**, **region-level**, and **global-level** VQA with their corresponding CoT annotations. For each setting, both the input to the annotation model (GPT-4o) and the expected output format are defined explicitly.

6.2. Region-level and Object-level VQA.

For non-global cases, human annotators first manually select ROIs in the image by drawing bounding boxes. Each ROI is further categorized as either:

- **Region**: a spatially extended area containing multiple objects or structures, where the cropped image is taken directly from the bounding box.
- **Object**: a single target object of interest, where the bounding box is expanded before cropping to provide additional surrounding context.

For each selected ROI, the following information is passed to GPT-4o:

1. A downsampled version of the full image (for global context).
2. A high-resolution cropped image derived from the bounding box (region: direct crop; object: expanded crop).
3. The human-provided label describing the object's or region's position and type.
4. The bounding box coordinates.

Then GPT-4o will generate Question-Answer pairs based on prompts shown in Appendix 6.2.

Dataset Annotation Prompt of Object Questions

You are an expert dataset annotator for an *Object-level Visual Question Answering (VQA)* dataset.

You will be provided with:

1. A downsampled full image showing the global context.
2. A high-resolution cropped image centered on the target object (ROI).
3. A bounding box for the object: `{item.get('bbox')}`.
4. A short human-provided label describing the object's **global position and type**: `{item.get('label')}`.

Your tasks:

1. **Refine the label** into a clear and specific English name for the object, preserving its spatial description. Examples:
   - "top-most sports field" → "top-most tennis court"
   - "central building" → "central white office building"
2. **Generate several high-quality, image-dependent question–answer pairs** about this object:
   - Use the refined label **explicitly** (no pronouns like "this object" or "it").
   - Be **visually grounded** — cannot be answered by common sense alone.
   - Require looking at the provided images to answer.
   - Avoid trivial or universal facts (*e.g.*, airplane has wings, grass is green).

Object-level Question Categories:

1. **Object category refinement** — Make the label more specific based on visual cues. Examples:
   - "Is the top-most sports field a tennis court or a basketball court?"
   - "Is the left-side bridge designed for vehicles or pedestrians?"
2. **Object color / pattern** — Non-obvious or detailed color info. Examples:
   - "What color is the roof of the central building?"
   - "Is the top-right ship mainly white or blue?"
   - "Does the top-most tennis court have a green or red surface?"
3. **Object shape / structure** — Shape or structural details. Examples:
   - "What is the shape of the top-most building's roof?"
   - "Is the left-side bridge straight or curved?"
   - "Is the bottom-right ship narrow or wide?"
4. **Object function / usage** — Purpose or role inferred from context. Examples:
   - "What is the function of the central rectangular building?"
   - "Is the left-side bridge used for vehicles or trains?"

5. **Object state / motion / activity** — Current condition or movement. Examples:
   - "Is the right-most vehicle moving or parked?"
   - "Is the top-left airplane taking off or landing?"
   - "Is the central crane operating or idle?"
6. **Object material / surface** — Visible material cues. Examples:
   - "Is the left-side bridge made of metal or concrete?"
   - "Does the roof of the central building appear metallic or tiled?"
7. **Object relative position / context** — Spatial relations to nearby elements. Examples:
   - "What is located beneath the top-most bridge?"
   - "Is there water below the bottom bridge?"
   - "What is on the right side of the central building?"

Guidelines:

- Use the refined label directly in all questions (no pronouns).
- Ensure each question is visually discriminative — answerable only by observing the images.
- Keep answers concise: "yes", "no", "concrete", "curved", "green", "train", "parked", "water".
- Each question must include:
  - `"category"` — one of the seven categories above.
  - `"higher_level"` — one of:
    * `"perception"` — visual recognition (color, shape, count)
    * `"localization"` — spatial position or relative location
    * `"attribute"` — appearance or material properties
    * `"function"` — role or purpose
    * `"reasoning"` — inferred or dynamic states

Output format:

```
{
  "label": "top-most tennis court",
  "qa_pairs": [
    {
      "question": "Is the top-most
          sports field a tennis court
          or a basketball court?",
      "answer": "tennis court",
      "category": "Object category
          refinement",
      "higher_level": "attribute"
    },
    {
      "question": "Does the top-most
          tennis court have a green or
          red surface?",
      "answer": "green",
      "category": "Object color /
          pattern",
      "higher_level": "perception"
    },
    {
      "question": "Is the top-most
          tennis court surrounded by
          fences?",
      "answer": "yes",
      "category": "Object shape /
          structure",
      "higher_level": "reasoning"
    },
    {
      "question": "What is located
          next to the top-most tennis
          court?",
      "answer": "parking lot",
      "category": "Object relative
          position / context",
      "higher_level": "localization"
    }
  ]
}
```

Dataset Annotation Prompt of Region Questions

You are an expert dataset annotator for a Visual Question Answering (VQA) dataset.
You will be given:

1. A downsampled full image.
2. A high-resolution cropped image of the region of interest (ROI).
3. A bounding box representing the ROI: `{item.get('bbox')}`.
4. A short human-provided description of the ROI: `{item.get('label')}`.

Your task:

- Refine the label to make it a **precise and natural English name** for this region (*e.g.*, "central bridge area", "left-most parking lot", "top-most construction site").
- Generate several **diverse question–answer pairs** about the ROI, following the categories below.

Question categories (choose those applicable to the region):

1. **Counting** — Ask about the **number** of visible

objects. Examples:
- "How many vehicles are on the bridge?"
- "How many ships are docked near the pier?"

2. **Object existence** — Ask if certain objects are **present** in the ROI. Examples:
   - "Is there a ship passing under the bridge?"
   - "Are there any cars in the parking lot?"
   - "Is any airplane on the runway?"
3. **Region status** — Ask about **activity, usage, or condition** of the region. Examples:
   - "Is the bridge busy or empty?"
   - "Is the construction site still active?"
   - "Are there ships currently docking at the port?"
   - "Is the road under construction or in use?"
4. **Object category** — Ask about the **types of main objects** found in the ROI. Examples:
   - "What types of vehicles are in the parking lot?"
   - "What kind of boats are docked at the pier?"
5. **Region function** — Ask about the **purpose or role** of the region. Examples:
   - "What is this area mainly used for?"
   - "What is the function of this rectangular region?"
6. **Other visual features** — Ask about **appearance, color, or shape** of the region or its objects. Examples:
   - "Are most buildings in this area red-roofed?"
   - "What is the overall shape of this region?"
   - "Is the area circular or rectangular?"

Output requirements:
- Only ask **reasonable** questions that can be answered directly from the provided images.
- Provide concise answers (one word or short phrase). Examples: "yes", "no", "asphalt", "empty", "circular", "urban", "in use".
- Each question must include:
  - `"category"`: one of the six above.
  - `"higher_level"`: one of these abstract reasoning levels:
    * `"perception"` — direct visual recognition
    * `"localization"` — position or spatial relation
    * `"attribute"` — appearance or measurable quality
    * `"function"` — purpose or role
    * `"reasoning"` — requires inference or contextual understanding

Output format:


```
{
  "label": "top-most bridge area",
  "qa_pairs": [
    {
      "question": "How many vehicles
          are on the bridge?",
      "answer": "3",
      "category": "Counting",
      "higher_level": "perception"
    },
    {
      "question": "Is the bridge
          currently in use?",
      "answer": "yes",
      "category": "Region status",
      "higher_level": "reasoning"
    },
    {
      "question": "Are there ships
          passing under the bridge?",
      "answer": "no",
      "category": "Object existence",
      "higher_level": "perception"
    },
    {
      "question": "What type of
          vehicles are visible on the
          bridge?",
      "answer": "cars",
      "category": "Object category",
      "higher_level": "attribute"
    },
    {
      "question": "What is the main
          function of this bridge area
          ?",
      "answer": "transportation",
      "category": "Region function",
      "higher_level": "function"
    }
  ]
}
```


6.3. Global-level VQA

For global-level questions, the full high-resolution satellite image is directly provided to GPT-4o, together with a task-specific prompt (see Appendix 6.3). The model is instructed to generate scene-level, visually grounded question–answer pairs that require holistic understanding of the image, such as scene type, counting of large-scale objects, or seasonal inference.

Dataset Annotation Prompt of Global Questions

You are an expert dataset annotator for a Visual Question Answering (VQA) task focusing on *global-level understanding* of high-resolution remote sensing images.
You will be provided with:

1. A full high-resolution image covering the entire scene.

Your task: Generate several **high-quality, globally grounded question–answer pairs** in English about the image.

Requirements for each question:

- **Scene-level**, not object-level.
- **Visually grounded** — answerable purely by looking at the image.
- **Specific and unique** — avoid vague or overly general questions.
- Generate only if the image clearly supports it; otherwise, output an empty list.

Possible question types (use only when appropriate):

- **Counting** — *e.g.*, "How many airplanes are visible in the image?"
- **Urban–Rural** — *e.g.*, "Does this image mainly depict an urban or rural area?"
- **Scene Type** — *e.g.*, "What is the main type of area shown — airport, residential, or farmland?"
- **Season** — *e.g.*, "What season does the scene appear to be?"

Higher-level reasoning categories:

- `perception`
- `localization`
- `attribute`
- `function`
- `reasoning`

Output format (strict JSON):

```
{
  "qa_pairs": [
    {
      "question": "How many airplanes
          are visible in the image?",
      "answer": "3",
      "category": "Counting",
      "higher_level": "perception",
      "justification": "Airplanes are
          distinct and countable
          across the visible runways."
    },
    {
      "question": "What is the main
          type of area shown in this
          image airport, residential,
          or farmland?",
      "answer": "airport",
      "category": "Scene Type",
      "higher_level": "reasoning",
      "justification": "The image
          shows large runways and
          parked airplanes typical of
          an airport."
    }
  ]
}
```

6.4. SFT Dataset Annotation

In order to train the model to master both the step-by-step reasoning process and the standardized tool invocation format for answering questions, we employed GPT-4o to generate annotations using a two-stage reasoning–cropping–reasoning CoT paradigm, as defined in the prompt provided in Appendix 6.4.

Reasoning-based remote sensing VQA Annotation Prompt

You are an intelligent remote sensing analyst. Your task is to generate reasoning-based annotations for Visual Question Answering (VQA) using satellite imagery.

I will provide:

- A global satellite image (downsampled for efficiency)
- The bounding box $[x_{\min}, y_{\min}, x_{\max}, y_{\max}]$ of the **reference object** mentioned in the question
- A natural language question referring to the image
- The ground truth answer

Important:

- The reference bounding box corresponds to the **referent object** in the question (*e.g.*, if the question asks "What is the structure parallel and closest to the bridge?", the reference bbox is for the bridge).
- The final target answer is derived **by reasoning relative to this referent object**.
- The global image should only be used for context and to explain how one would locate the referent region.
- The final answer must be derived **by analyzing the cropped region corresponding to the refer-**

ent and its surroundings.

When writing `<stage_1_reasoning>`, follow these rules:

- At the very start, always begin with: *"This question is asking about <short intent>, therefore I need to crop the image to examine the surroundings of the mentioned target."*
- **Localization Strategy**: Describe the approximate location of the referent object in natural language (e.g., "the bottom-most long bridge across the river"). Do not output exact coordinates.
- **Reasoning Result**: First output exactly: *"I need to pay attention to the reference object at"* Then output the bounding box in JSON format on the next line:

```
[
    {"bbox_2d": [x_min, y_min, x_max
        , y_max], "label": "<short
        description of the referent
        object>"}
]
```

No additional explanation in this section.

Output must strictly follow this structure:

```
<global> - Provide a brief but
    informative description of the
    global satellite image (e.g., main
     structures, spatial layout). </
    global>

<stage_1_reasoning>
Question Intent: Identify the type of
    question being asked (e.g., object
     category, count, color, spatial
    relation, etc.), and determine
    what visual information is needed
    to answer it.

Localization Strategy: Parse the
    question to identify the referent
    object (e.g., bridge, river,
    building cluster). Translate the
    description into a visual query
    and locate it in the global image
    using semantic cues (shape, size,
    color, spatial arrangement).
    Summarize the approximate location
     of the referent in natural
    language.

Reasoning Result:
I need to pay attention to the
    reference object at
[JSON bounding box]
</stage_1_reasoning>

<stage_2_reasoning>
Given the cropped region of the
    referent object, explain how to
    reason about the final target
    answer. Specify what visual
    features or spatial relations
    should be observed. Clearly
    connect the reasoning steps from
    the referent to the final answer.
</stage_2_reasoning>
```

Constraints:

- Do not reveal the final answer in `<stage_1_reasoning>`.
- The `<global>` description must be neutral and avoid giving away the answer.
- The `<stage_2_reasoning>` must directly connect the referent to the final target.

Input:

```
Question: {result["question"]}
Ground Truth Answer: {result["
    ground_truth"]}
Reference Bounding box: {[int(x /
    scale) for x in hbox]}
```

6.5. Data Refinement

Using GPT-4o, we initially generated over 40,000 question–answer pairs. However, several issues were observed during quality inspection. For binary (True/False) questions, the model exhibited a tendency to generate questions whose correct answer was "Yes," while rarely producing negative cases. For multiple-choice questions, certain answer options were disproportionately favored, for example, "cement" in material-related questions or "summer" in season-related questions. To address these biases, we manually removed redundant or overly similar questions and supplemented the dataset with additional questions to balance the distribution of answer options. Furthermore, some model-generated outputs contained factual or logical errors, which were corrected through manual revision. The refinement process was carried out by six annotators, each contributing over ten hours of work. Compared to fully manual annotation of tens of thousands of questions, our proposed semi-automatic data annotation pipeline significantly reduced the annotation workload while maintaining both label accuracy and distributional balance.

Configuration	VQA	
	SFT	GRPO
Training component	Full	Full
Learning rate	3e-5	1e-7
Warmup step	500	50
Weight decay	0.01	0.01
Batch size	4	32
α	-	200
β	-	0.05
γ	-	0.04
Optimizer	AdamW	
Dataset	LRS-GRO/sft	LRS-GRO/rl
Training epoch	1	1

Table 8. Experimental configurations and hyperparameters for SFT and GRPO training. In the GRPO objective, α controls the Region-Guided reward, β scales the pattern reward in the overall reward formulation, and γ regulates the KL-divergence penalty.

6.6. Dataset Visualization

As shown in Figs 8, 9, 10 and 11, we visualize a subset of QA pairs from the LRS-GRO dataset, covering multiple representative geographic landscapes such as airports, factories, ports, bridges, and rural areas. The LRS-GRO dataset provides abundant question–answer pairs and precise bounding box annotations for the global, regional, and object levels.

6.7. Comparison with LRS-VQA Dataset

As illustrated in Figs 12, 13, 14 and 15, the LRS-VQA dataset innovatively introduced a GPT-based automated annotation pipeline, offering a novel approach to data labeling. However, due to noticeable hallucinations when GPT processes high-resolution imagery, as illustrated in the figure, incorrect labels may provide the model with misleading rewards during training, thereby hindering convergence. Following the annotation paradigm of LRS-VQA, we further refined and proposed the LRS-GRO dataset. Through meticulous manual annotation and verification, we provide a high-resolution RS image dataset with precise labels.

7. Training Details

Table 8 summarizes the experimental configurations and hyperparameters used for the ZoomEarth under the SFT and GRPO training settings.

For a detailed implementation of GRPO, the objective function is defined as follows:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}\Big[q \sim P_{\text{sft}}(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(O|q)\Big] \\ \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\left[\hat{A}^*_{i,t} - \gamma \mathbb{D}_{\text{KL}}[\pi_\theta \parallel \pi_{\theta_{\text{ref}}}]\right] \quad (3)$$

where $P_{\text{sft}}(Q)$ denotes the distribution of queries sampled from the supervised fine-tuning dataset, and $\pi_\theta(O|q)$ represents the current policy parameterized by θ, which generates output sequences conditioned on the query q. $\hat{A}^*_{i,j}$ denotes the clipped advantage reweighted by importance sampling, and $\mathbb{D}_{\text{KL}}[\pi_\theta \parallel \pi_{\theta_{\text{ref}}}]$ represents an unbiased estimator of the KL divergence. G denotes the number of samples per group, and $|o_i|$ indicates the length of each trajectory. Specifically, γ is set to 0.04, and G is set to 4.

$$\mathbb{D}_{\text{KL}}[\pi_\theta || \pi_{\theta_{\text{ref}}}] = \frac{\pi_{\theta_{\text{ref}}}(o_{i,t}|q, o_{i,<t})}{\pi_\theta(o_{i,t}|q, o_{i,<t})} - \log \frac{\pi_{\theta_{\text{ref}}}(o_{i,t}|q, o_{i,<t})}{\pi_\theta(o_{i,t}|q, o_{i,<t})} - 1 \quad (4)$$

$$\hat{A}^*_{i,j} = \min\Bigg[\frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})}\hat{A}_{i,j}, \\ \text{clip}\left(\frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})}, 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,j}\Bigg] \quad (5)$$

$$\hat{A}_{i,j} = \frac{r_i - \text{mean}(r)}{\text{std}(r)} \quad (6)$$

The detailed reward components are defined as follows:

$$r_{IoU} = \begin{cases} IOU, & \text{if match} \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

$$r_{R-G} = \text{sigmoid}\left(\frac{\alpha}{distance + \epsilon}\right) \quad (8)$$

$$r_{answer} = \begin{cases} 1, & \text{if } simularity > 0.8 \\ simularity, & \text{otherwise} \end{cases} \quad (9)$$

$$r_{pattern} = \begin{cases} 1, & \text{if match the patten} \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

$$r = r_{IoU} + r_{R-G} + r_{answer} + \beta r_{pattern} \quad (11)$$

Specifically, we set ϵ to 0.2. The parameter β is used to constrain the model outputs to adhere to the predefined format, thereby preventing training collapse. Our experiments show that when $\beta < 0.05$, e.g., $\beta = 0.01$, it fails to take effect and may lead to potential training collapse. Under the premise of stable output formatting, more than 99% of the r_{pattern} rewards are equal to 1, and thus increasing β has no impact on the overall reward after normalization. Therefore, we set $\beta = 0.05$ in our final configuration.

Finally, the model updates its parameters by performing gradient ascent on the GRPO objective:

$$\theta \leftarrow \theta + \eta \nabla_\theta \mathcal{J}_{\text{GRPO}}(\theta) \quad (12)$$

where η denotes the learning rate, which is set to 1×10^{-7} in our experiments.

To equip the model with the ability to perform chain-of-thought reasoning in this specific scenario, we applied instruction tuning during both the SFT and RL stages, and used the same instruction prompts during inference to elicit the reasoning capability acquired during training. Specifically, our instruction is as follows:

Instruction Used in Training and Evaluation

You are an intelligent remote sensing analyst. Given a natural language question about a satellite image, generate a structured reasoning answer as follows:

1. <think> ... </think>
   - Provide a neutral one-sentence description of the whole image scene.
   - Cropping task:
     "This question is asking about <short intent>, therefore I need to crop the image to examine the surroundings of the mentioned target."
   - Non-cropping task:
     "This question is asking about <short intent>, therefore I need to analyze the entire image without cropping."
   - Include:
     - **Question Intent**: describe the type of question (object category, spatial relation, count, etc.) and the visual information needed.
     - **Localization Strategy**:
       * Cropping: approximate referent object location in natural language (no coordinates).
       * Non-cropping: strategy to detect all relevant objects across the entire image.
     - **Reasoning Result**:
       * **Cropping**: output exactly one JSON-formatted bbox for the referent, for example:

```
[{"bbox_2d": [x_min, y_min,
   x_max, y_max],
  "label": "<short description
    >"}]
```

       * **Non-cropping**: summarize how detected objects will be used to produce the count or answer.
2. <think> ... </think> (only when the cropped image is provided)
   - Explain step-by-step how to reason from the referent (or detected objects) to the final answer.
3. <answer> ... </answer>
   - Provide your final answer as a single word or short phrase.

Rules:

- Always return exactly one <answer> block.
- For tasks that need cropping:
  - Provide the bounding box of the object of interest in the first <think> block.
  - After the cropped image is given, generate another <think> block to derive the answer.
  - Also include a bounding box in the <stage_2_reasoning> block when required.
- If unsure about localization, make a best reasonable guess — never state uncertainty.

8. Inference Details

8.1. Visualization of the Reasoning Process

As illustrated in the Figs 16, 17, 18 and 19, the reasoning process of ZoomEarth is structured into several sequential stages to ensure both global comprehension and precise localization.

1. **Global Description**: The process begins with a comprehensive global analysis of the input image. This stage aims to establish an overall semantic understanding of the scene, enabling the model to capture contextual relationships and spatial configurations before focusing on specific regions.
2. **Question Intent Identification**: Following global perception, the model analyzes the user's query to determine its underlying intent. This step isolates the core informational demand, ensuring that subsequent reasoning is aligned with the question's focus.
3. **ROI Localization Strategy**: Based on the global understanding and the identified question intent, the model performs targeted localization reasoning. This involves determining the spatial regions most relevant to the query and representing them via bounding boxes.
4. **Tool Calling**: Once ROIs are identified, the model invokes appropriate tools to process the localized areas. This stage is referred to as *secondary perception*, as the model re-examines the image after cropping and zooming the identified regions. The objective is to increase resolution and focus on fine-grained visual details that may be lost in the global view.
5. **Stage 2 Reasoning**: The refined, high-resolution inputs derived from the localized regions are then used for a second stage of reasoning. This stage integrates global context with localized detail to produce the final answer, ensuring accuracy and relevance to the original query.

This multi-stage process is designed to balance broad contextual awareness with precise visual focus. By first establishing a global understanding and then iteratively narrowing attention to relevant regions, the ZoomEarth model mitigates the risk of missing context while maximizing the accuracy of localized inference.

8.2. Comparison with Other Models

Many methods have been proposed to address high-resolution image processing, as shown in Fig. 7. Techniques such as dynamic resolution and visual token pruning have become mainstream. Dynamic resolution methods first pad the image to an integer multiple of small patches and then split it into a sequence of small patches. This allows the model to handle high-resolution images but does not reduce the number of visual tokens, so a large number of tokens still need to be processed. Visual token pruning removes tokens with low information content or redundant information according to manually defined rules, reducing the number of visual tokens and enabling the model to handle high-resolution images. However, this approach relies on hand-crafted rules and has limited generalizability.

Our proposed method first feeds a downsampled low-resolution image into the model. The model then identifies ROIs and calls the cropping-zooming tool to obtain detailed information for these areas. This reduces the total number of visual tokens while preserving fine-grained details in the ROIs.

As illustrated in the Figs 20, 21 and 22, for questions at the region-level or object-level, general VLMs often produce incorrect answers due to limited visual resolution or hallucinations. Although VLMR[3] is capable of performing secondary perception, its training methodology is not well-suited for high-resolution RS imagery, rendering it ineffective in this context. In contrast, our proposed ZoomEarth model can accurately localize the region of interest and perform reasoning based on the secondary perception of that region, thereby producing correct answers.

8.3. Impact of RL on the Reasoning

We observe that the model trained with SFT tends to exhibit degraded performance after invoking the cropping tool, whereas the model further optimized with RL demonstrates a significant performance improvement when using the same tool. The SFT-trained model fails to effectively leverage the cropping–zooming tool for active perception, primarily due to two reasons: (1) insufficient localization ability, which leads to cropping incorrect regions and consequently causes misinterpretation (see Fig. 23); and (2) inability to perform correct reasoning based on the cropped images, resulting in attention to irrelevant content (see Fig. 24 and Fig. 25).

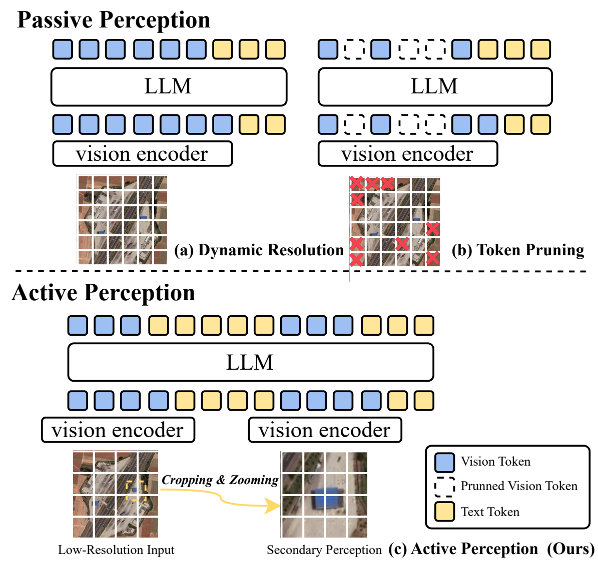

Figure 7. Detailed comparison between passive perception and our proposed active perception method.

8.4. Evaluation of APO IoU

During inference, the model only outputs the region of interest or target, while the answer is often located in the adjacent area outside the ROIs. Therefore, we expand the predicted bounding box to a size of 512 and crop it from the original image. Accordingly, the APO IoU is computed after enlarging both the ground-truth and predicted bounding boxes to a fixed size of 512.

9. Downstream Tasks

9.1. Downstream Instructions

ZoomEarth can autonomously invoke external tools to perform downstream tasks such as cloud removal, denoising, segmentation, and image editing without additional training. By simply modifying the instruction, the model can be endowed with the ability to utilize these tools effectively. Specifically, the instructions we used in downstream tasks are:

Instruction Used for Downstream tasks

You are an intelligent remote sensing analyst. Given a natural language question about a satellite image, generate a structured reasoning answer as follows:

1. `<think> ...  </think>`
    - Provide a neutral one-sentence description of the whole image scene.
    - Cropping task: "This question is asking about <short intent>, therefore I need to crop the image to examine the surroundings of the men-

tioned target."
- Non-cropping task: "This question is asking about <short intent>, therefore I need to analyze the entire image without cropping."
- Include:
  - **Question Intent**: describe the type of question (object category, spatial relation, count, etc.) and needed visual information.
  - **Localization Strategy**:
    - Cropping: approximate referent object location in natural language (no coordinates).
    - Non-cropping: strategy to detect all relevant objects.
  - **Reasoning Result**:
    - **Cropping**: output exactly one JSON-formatted bbox for the referent:

```
[{"bbox_2d": [x_min, y_min,
    x_max, y_max],
  "label": "<short description
    >"}]
```

    - **Non-cropping**: summarize how detected objects will be used to produce the count.

2. <tool_call> ... </tool_call>
- After generating the bounding box of the ROI, you must call the tools below.
- Follow these formatting rules strictly:
  - You must output exactly one <tool_call> block.
  - The content inside must be a valid JSON object in the following format:

```
<tool_call>
{
    "name": "tool_name",
    "arguments": {
        "arg1": arg1
    }
}
</tool_call>
```

  - Available tools:
    - **Cloud removal:**

```
"name": "cloud_removal",
"arguments": {
    "bbox_2d": [x_min, y_min,
        x_max, y_max]
}
```

    - **Segmentation:**

```
"name": "segmentation",
"arguments": {
    "bbox_2d": [x_min, y_min,
        x_max, y_max],
    "objects": [obj1, obj2]
}
```

    - **Denoise:**

```
"name": "denoise",
"arguments": {
    "bbox_2d": [x_min, y_min,
        x_max, y_max]
}
```

    - **Image editing:**

```
"name": "image_editing",
"arguments": {
    "bbox_2d": [x_min, y_min,
        x_max, y_max],
    "description": "Remove the
        tree"
}
```

  - Do not include any reasoning or comments outside the <tool_call> block.
  - If localization is uncertain, make a best-guess bounding box.
  - Ensure that the JSON inside <tool_call> is valid and machine-readable.

3. Example:

```
<tool_call>
{
    "name": "cloud_removal",
    "arguments": {
        "bbox_2d": [120.5, 230.1, 6
            40.3, 880.2]
    }
}
</tool_call>
```

9.2. Downstream Examples

Below are examples of ZoomEarth's response to downstream task's instructions.

Cloud removal

Prompt: Show me the bottom-most blue-roofed building. And then remove the cloud over it using tools. + downstream instruction

```
<think>
The global satellite image shows a
    large industrial facility
```

```
    surrounded by forested areas and
    some urban infrastructure. The
    layout includes multiple
    rectangular buildings, open
    storage yards, and adjacent access
     roads and rail lines. There are
    also some construction zones and
    partially visible structures,
    along with surrounding natural
    land cover patterns.


Question Intent: This question is
    asking about identifying the
    position of a specific blue-roofed
     building at the bottom-most part
    of the image, therefore I need to
    crop the image to examine the
    surroundings of the mentioned
    target.

Localization Strategy: The referent
    object is described as a blue-
    roofed building located at the
    bottom-most part of the scene. In
    the global image, these blue-
    roofed buildings appear as dark-
    colored rectangular structures
    with light-colored roofs,
    positioned towards the lower end
    of the industrial complex, near
    the edge of forested areas and
    road networks.

Reasoning Result: First, I locate the
    bottom-most blue-roofed building
    by identifying the lowest
    rectangular structure with a
    lighter roof within the industrial
     complex. Its position is at the
    southernmost part of the scene,
    directly adjacent to forested
    terrain, and slightly above a
    cluster of smaller buildings. Then
    , I crop the image to focus solely
     on this building to perform
    further operations.

</think>
    <tool_call>
    {
        "name": "cloud_removal",
        "arguments": {
            "bbox_2d": [379.5, 400.5,
                393.0, 411.5]
        }
    </tool_call>
```

Segmentation

Prompt: Show me the bottom-most blue-roofed building. And then segment it using tools. + downstream instruction

```
<think>
The global satellite image shows a
    large industrial facility with
    multiple buildings, open storage
    areas, and adjacent forested
    regions. The layout includes long
    rectangular structures arranged in
     a linear fashion, some connected
    by overpasses or elevated roads,
    and several smaller buildings
    scattered around the complex.


Question Intent: This question is
    asking about identifying a
    specific blue-roofed building at
    the bottommost position in the
    facility, therefore I need to crop
     the image to examine the
    surroundings of the mentioned
    target.

Localization Strategy: The referent
    object is the bottom-most blue-
    roofed building. In the global
    image, this corresponds to a small
     rectangular structure with a
    bright blue roof located toward
    the lower right corner of the
    facility, near the edge of the
    wooded area.

Reasoning Result:
I need to pay attention to the bottom-
    right section of the global image
    where a small, distinct blue-
    roofed structure can be seen. This
     is positioned at the lower end of
     the main facility area, above
    open storage lots, and in close
    proximity to the forested terrain.
     The blue roof distinguishes it
    from other nearby buildings or
    structures.

</think>
```

```
<tool_call>
{
    "name": "segmentation",
    "arguments": {
        "bbox_2d": [340.5, 395.5, 398.
            0, 461.0],
        "objects": ["bottom-most blue-
            roofed building"]
    }
}
</tool_call>
```

Denoising

Prompt: Show me the bottom-most blue-roofed building. And then denoise the area around it using tools. + downstream instruction

````
<think>
The global satellite image shows a
    large industrial facility with
    multiple warehouse-like buildings,
     large storage yards, and elevated
     conveyor systems. The site is
    surrounded by hilly terrain with
    open fields and some scattered
    clusters of buildings. Road
    networks and bridges connect
    different parts of the complex,
    indicating active transportation
    links.


Question Intent: This question is
    asking about identifying the
    bottom-most blue-roofed building
    after locating the referent,
    therefore I need to crop the image
     to examine the surroundings of
    the mentioned target.

Localization Strategy: The referent
    object is described as the bottom-
    most blue-roofed building. In the
    global image, blue-roofed
    structures stand out due to their
    bright coloration against the
    lighter-toned roofs and
    surrounding terrain. The lowest
    building among them would be the
    southernmost one in the facility
    layout, which is distinctly
    located at the bottom of the image
    . I can locate it by scanning
    along the southern edge of the
    facility and identifying the
    lowest blue-roofed structure.

Reasoning Result:
I need to pay attention to the
    referent object at
```json
[
    {"bbox_2d": [347.5, 398.5, 398.0,
        449.0], "label": "bottom-most
        blue-roofed building"}
]
```
</think>
<tool_call>
{
    "name": "denoise",
    "arguments": {
        "bbox_2d": [347.5, 398.5, 398.
            0, 449.0]
    }
}
</think>
````

Image editing

Prompt: Show me the bottom-most blue-roofed building. And then paint its roof red using tools. + downstream instruction

```
<think> The global satellite image
    shows a large industrial facility
    with multiple rectangular
    buildings arranged in a grid-like
    pattern across several plots.
    There are large paved areas, some
    with clearings, and surrounding
    the facility are open land and
    vegetation. Roads connect
    different sections of the complex
    within and around the periphery.

Question Intent: This question is
    asking about identifying and then
    visually changing the roof color
    of a specific blue-roofed building
    , therefore I need to crop the
    image to examine the surroundings
    of the mentioned target.

Localization Strategy: The referent
    object is described by its roof
    color - specifically blue. In the
    global image, blue roofs can be
```

````
    identified among the large
    rectangular buildings occupying
    the lower right section of the
    facility. These buildings have
    distinctively colored roofs,
    making them stand out from other
    structures due to their hue. The
    blue-roofed building in question
    is located toward the bottom-right
     corner of the facility, near the
    edge where the paved area meets
    the vegetation.

Reasoning Result:
I need to pay attention to the
    reference object at
```json
[
    {"bbox_2d": [357.5, 409.0, 399.5,
        446.0], "label": "bottom-most
        blue-roofed building"}
]
```
</think>
<tool_call>
{
    "name": "image_editing",
    "arguments": {
        "bbox_2d": [357.5, 409.0, 399.
            5, 446.0],
        "description": "Paint the roof
             of the bottom-most blue-
            roofed building red using
            appropriate tools."
    }
}
</think>
````

1107

Figure 8. The visualization of examples from LRS-GRO dataset.

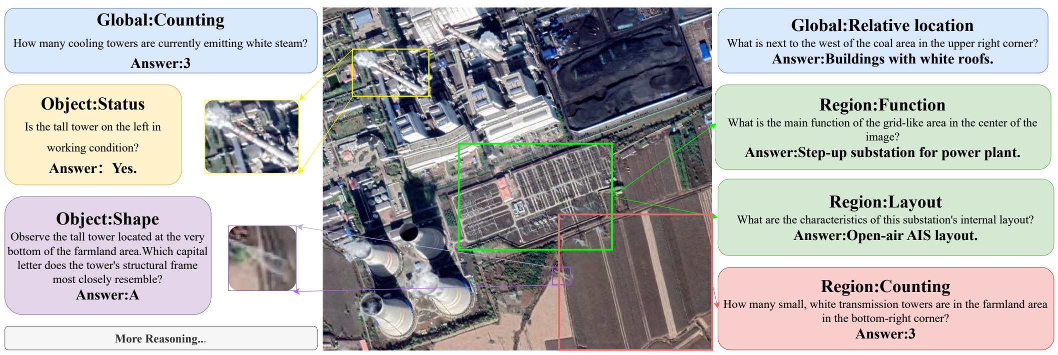

Figure 9. The visualization of examples from LRS-GRO dataset.

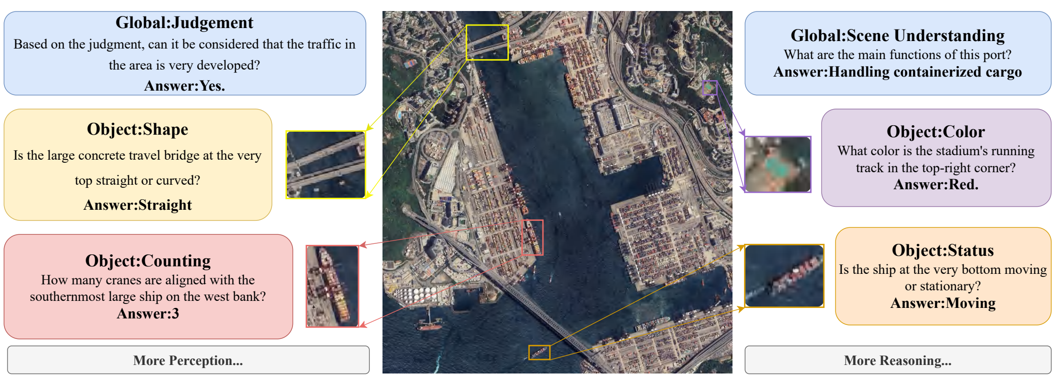

Figure 10. The visualization of examples from LRS-GRO dataset.

Figure 11. The visualization of examples from LRS-GRO dataset.

Figure 12. Comparison Between LRS-GRO and LRSVQA.

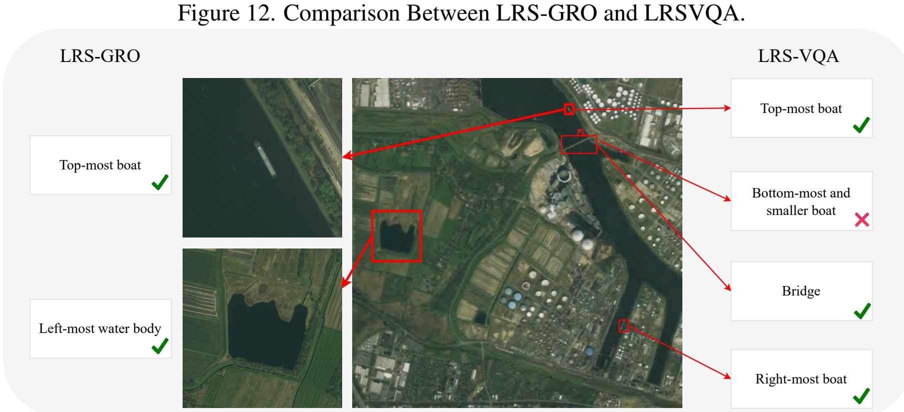

Figure 13. Comparison between LRS-GRO and LRSVQA.

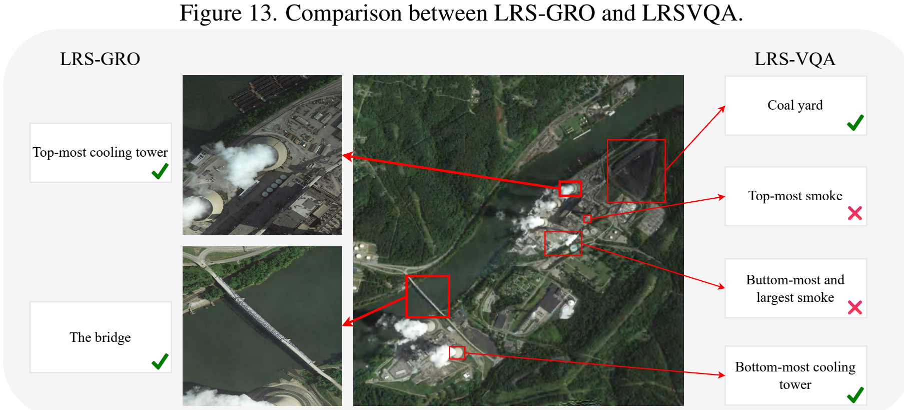

Figure 14. Comparison between LRS-GRO and LRSVQA.

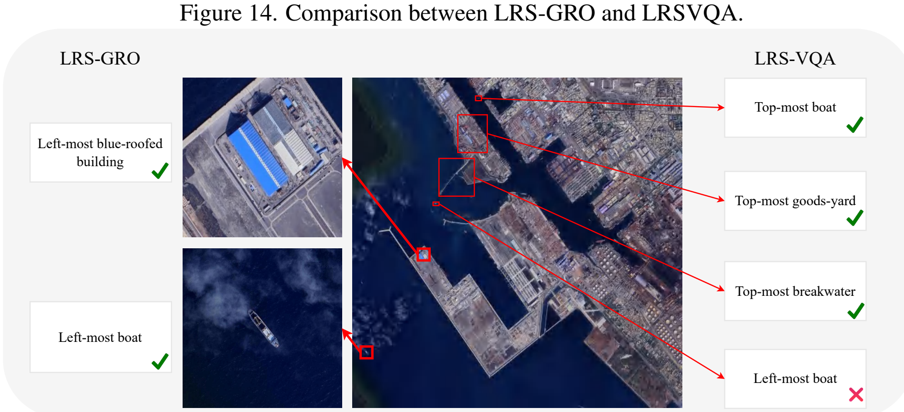

Figure 15. Comparison between LRS-GRO and LRSVQA.

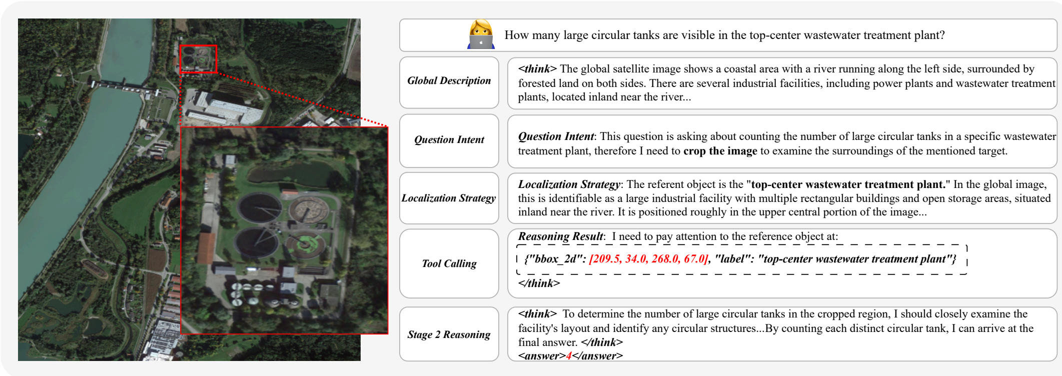

Figure 16. Reasoning process visualization of ZoomEarth.

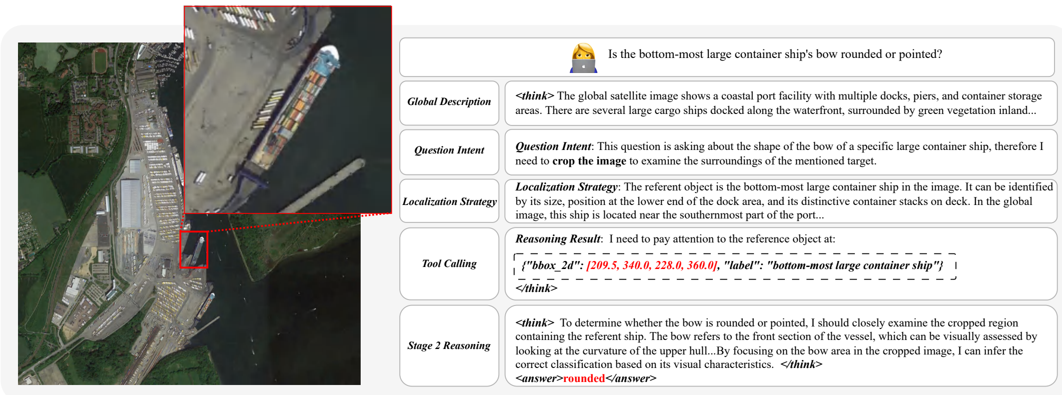

Figure 17. Reasoning process visualization of ZoomEarth.

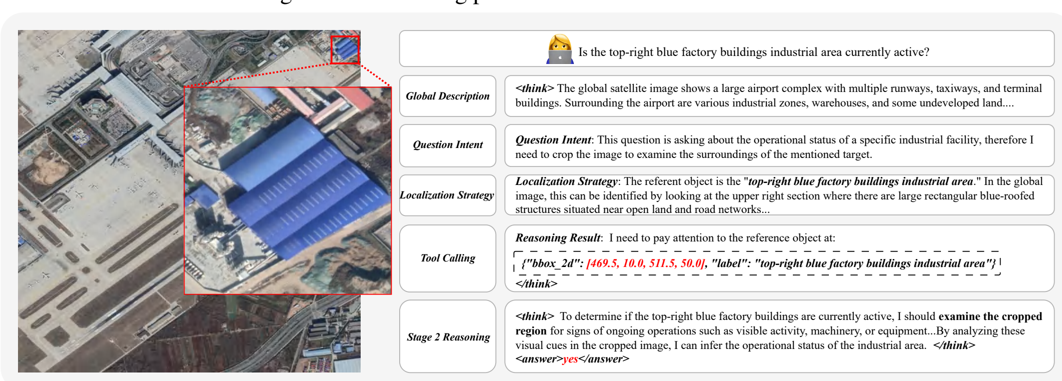

Figure 18. Reasoning process visualization of ZoomEarth.

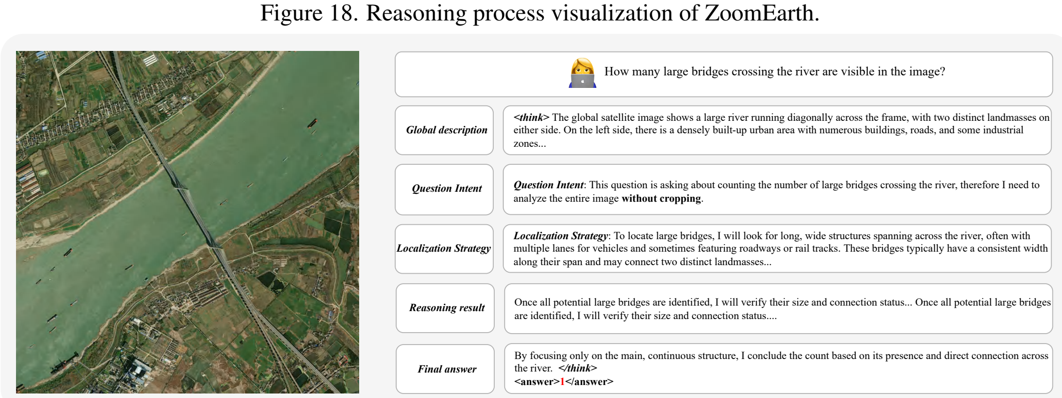

Figure 19. Reasoning process visualization of ZoomEarth.

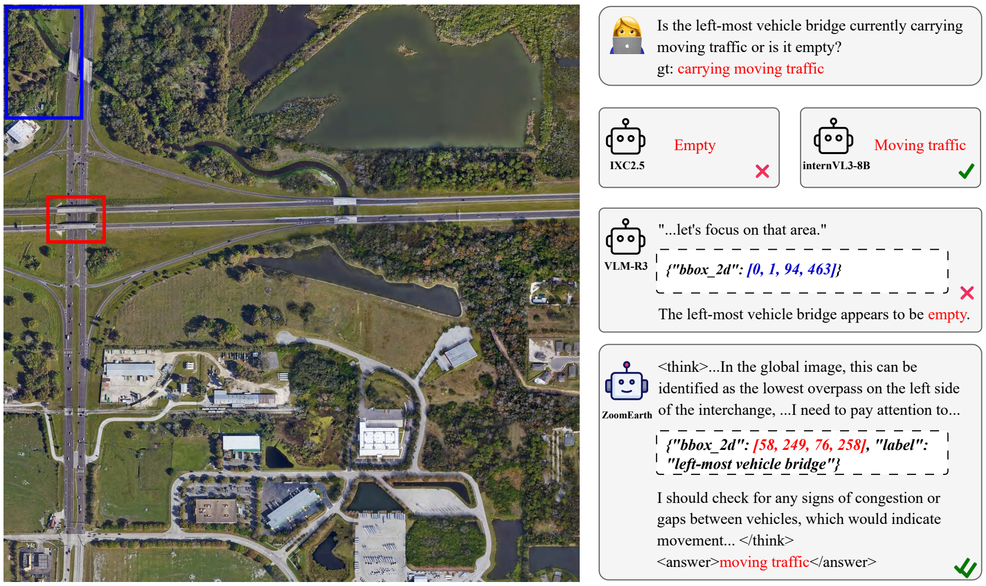

Figure 20. The comparison of the answer between different models.

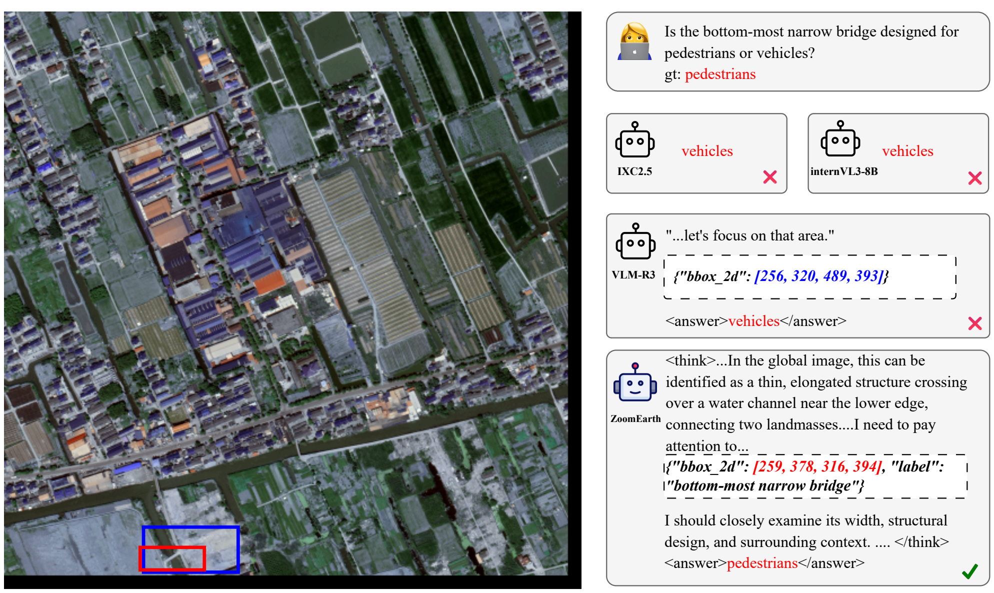

Figure 21. The comparison of the answer between different models.

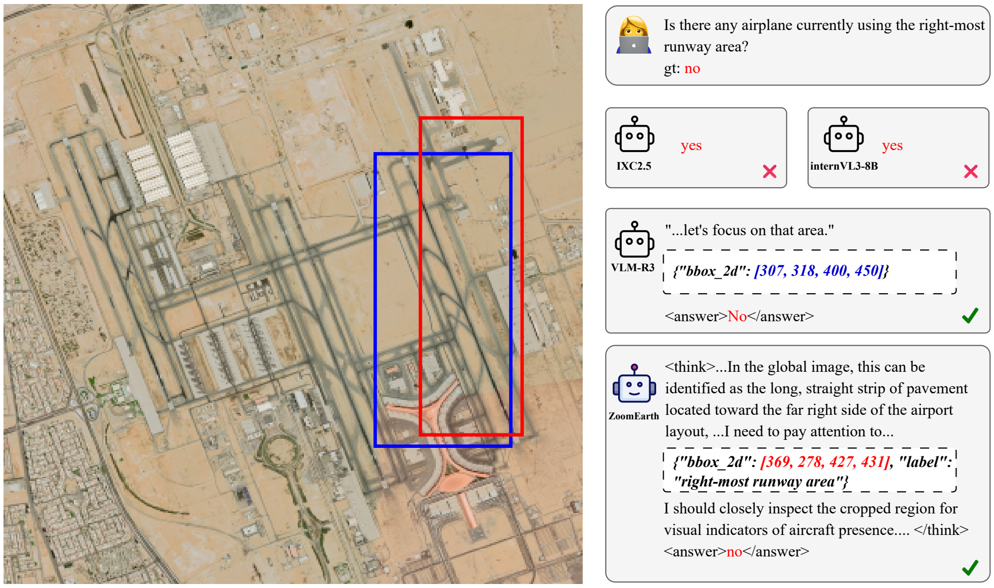

Figure 22. The comparison of the answer between different models.

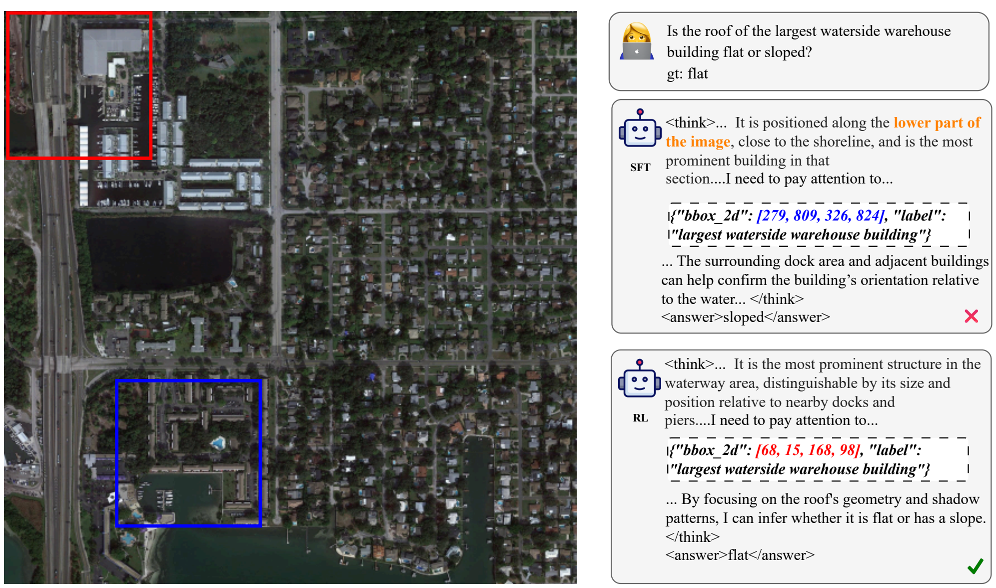

Figure 23. Comparison of reasoning results before and after RL. The text marked in orange indicates incorrect reasoning processes.

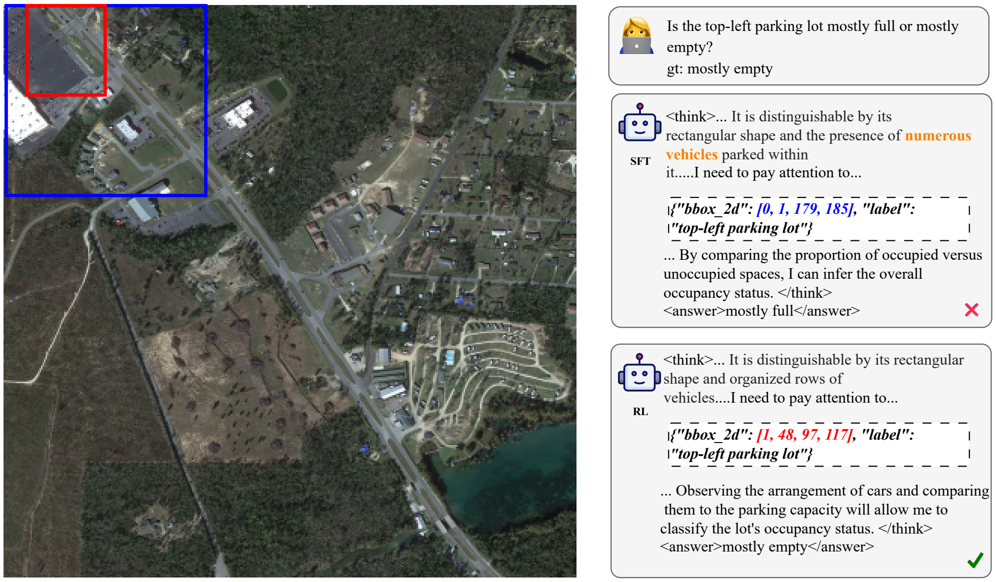

Figure 24. Comparison of reasoning results before and after RL. The text marked in orange indicates incorrect reasoning processes.

Figure 25. Comparison of reasoning results before and after RL. The text marked in orange indicates incorrect reasoning processes.