University of Reading

Department of Computer Science

MARS: A Multi-model Agent for Remote Sensing Images

Weijie Cui

Supervisor: Dr. Muhammad Shahzad

A report submitted in partial fulfilment of the requirements of the University of Reading for the degree of Master of Science in *Data Science and Advanced Computing*

September 8, 2025

Declaration

I, Weijie Cui, of the Department of Computer Science, University of Reading, confirm that this is my own work and figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

I give consent to a copy of my report being shared with future students as an exemplar.

I give consent for my work to be made available more widely to members of UoR and public with interest in teaching, learning and research.

Weijie Cui

September 8, 2025

Abstract

The exponential growth of Remote Sensing (RS) technologies has transformed Earth observation, enabling high-resolution imaging for applications like environmental monitoring, urban planning, and disaster rescue. However, interpreting these images remains challenging due to their complexity, multi-scale objects, and dense spatial-spectral information. Traditional methods rely on single-pass visual Artificial Intelligence (AI) models, which struggle with partially obscured, small and shadowed objects. Machine learning (ML) and deep learning (DL) have advanced automated analysis, but limitations persist supervised models require extensive labelled data, while conventional vision-language models (VLMs) lack domain-specific expertise and iterative refinement capabilities. Inspired by the human visual system, to rapidly search and focus on important areas to extract fine details and validate information by prior knowledge, Multi-model Agent for Remote Sensing (MARS) offers a promising solution to interpret images in detail. An Artificial Intelligence Agent (AI Agent) orchestrates a vision model (VM) and a Reinforcement Learning (RL) model, forming the foundation for next-generation remote sensing systems. The RL model generates observation strategies for VM to identify the class and location of objects in images by focusing on different areas multiple times and eventually achieve the visual goal. Experiments demonstrate the robustness of the MARS, which is size insensitive and could detect more partially obscured, small or shadowed targets. In addition, it does not significantly increase GPU computing resources. Code and video demos are available at https://github.com/WeijieCui/MARS.

Keywords: Reinforcement Learning, Remote Sensing, AI Agent, MARS.

Acknowledgements

I would like to express my deepest appreciation to my supervisor, Dr. Muhammad Shahzad, for his invaluable guidance and constructive criticism. On a personal note, I owe my deepest gratitude to my wife Chuyu Wang and my kids, Yiming and Yidi, for their endless love and encouragement. Finally, I would like to thank all the lecturers, relatives and friends who have helped me with my studies.

Contents

List of Figures

List of Tables

List of Abbreviations

SMPCS	School of Mathematical, Physical and Computational Sciences

Chapter 1

Introduction

1.1 Background

The exponential growth of Remote Sensing (RS) technologies has transformed Earth observation, enabling high-resolution imaging for applications like environmental monitoring, urban planning, and disaster rescue. However, interpreting these images remains challenging due to their complexity. Remote sensing images can be captured from various altitudes, such as low-altitude drones, high-altitude aircraft, and satellites. They can also be captured using natural light cameras or thermal imaging cameras. Object sizes can also vary greatly, from small objects like people and cars to large aircraft and football fields. The same object captured at different times can also be affected by varying lighting angles, which can affect recognition. Since the advent of LetNet-5 in 1998, computer vision has made long-term progress in object detection, image segmentation, and 3D modeling. The model architecture has also evolved from shallow neural networks to deep neural networks or residual neural networks, and even complex visual language models, such as Contrastive Language-Image Pre-Training (CLIP). But these models rely on single-pass detection, which makes them struggle with partially obscured, small and shadowed objects. At the same time, as the recognition ability of the model improves, the model needs to contain more parameters and layers, which means more training data and training resources are needed. For instance, CLIP team trained the largest vision model for 18 days on 592 V100 GPUs and the largest Vision Transformer for 12 days on 256 V100 GPUs. This is considered acceptable, even common, in recent AI model training. But it is not conducive to the practice of AI skills in Internet of Things (IoT) devices.

1.2 Problem statement

This essay will investigate the core problem through the following detailed research questions:
Research Question 1: Can an Agent AI improve the analytical accuracy of remote sensing image analysis?
Sub-question 1.1: To what extent can an RL-driven agent enhance the model's ability to detect and classify small, occluded, or rare objects (e.g., small vehicles and ships) compared to a static baseline model?
Sub-question 1.2: Does the agent-based approach reduce error rates in fine-grained classification tasks (e.g., distinguishing between big vehicles and small vehicles) by focusing computational resources on discriminative regions?
Hypothesis: The adaptive searching mechanism afforded by the agent will lead to a statistically significant increase in accuracy metrics (mAP, IoU, F1-score) on challenging remote sensing datasets by extracting more relevant visual details.

Research Question 2: Can this enhancement in visual capability be achieved without a proportional increase in computing resource usage?
Sub-question 2.1: How does the total computational cost (measured in FLOPs, inference time, and memory usages) of the agent-driven model compared to a static model of equivalent accuracy?
Sub-question 2.2: Does the agent learn to minimize redundant computations? For instance, does it learn to avoid over-processing homogeneous regions like large bodies of water or empty fields?
Hypothesis: The agent will learn an efficient policy that prioritizes computation, leading to a higher accuracy-per-computation ratio. The system will not achieve superior performance than a static model. While its computation time will increase significantly, it will not require additional resource allocation.

1.3 Aims and objectives

The final aim of this project is to investigate, design and implement a novel agent-driven reinforcement learning framework MARS that dynamically optimizes the analysis of remote sensing images. The primary goal is to demonstrate that this adaptive approach can enhance the accuracy of fine-grain extraction and distribute the workload compared to conventional static one-pass visual models.
To achieve this aim, this project will pursue the following specific objectives:

1. To design and build the MARS framework integrating an Agent AI with a visual environment, a visual model and a reinforcement learning model.
   - Detail: Define the visual environment: the state space (e.g., current window, current feature maps, location context), the action space (e.g., left, right, up, down, zoom-in and zoom-out) and the reward function (balancing task performance gain against computational cost).
2. To design and implement a baseline model and the proposed agent-based model for a chosen remote sensing task.
   - Detail: Select a benchmark task (e.g., small object detection in DOTA dataset). Select a state-of-the-art static model (e.g., a YOLO) as a baseline. Subsequently, design and implement the proposed MARS framework.
3. To qualitatively evaluate and compare the performance accuracy of the models.
   - Detail: Train and evaluate MARS on DOTA remote sensing dataset as well as the frozen YOLO model. Measure and compare key accuracy metrics such as mean Average Precision (mAP) for detection tasks or Intersection over the Union area (IoU) and F1-score for classification tasks. This objective directly addresses Research Question 1.
4. To quantitatively evaluate and compare the computational efficiency of the models.
   - Detail: Profile both models during inference. Measure and compare resources consumed, including inference time (seconds per image), and GPU memory usages. This analysis will test the hypothesis that the agent achieves better performance within an acceptable timeframe and directly addresses Research Question 2.
5. To perform a quantitative analysis of MARS's policy and decision-making processes.
   - Detail: Visualize and interpret the agent's actions (e.g., where it chooses to look, what regions it selects for high-resolution analysis). This will provide insight into how the

agent improves performance, revealing whether it learns intelligent, human-like strategies for analyzing complex scenes.

6. To summarize findings, discuss limitations, and propose directions for future research.
   - Detail: To summarize the results from Objectives 3, 4 and 5 to form a conclusive argument. Discuss the practical values of the findings, acknowledge the limitations of the current study and suggest useful steps for further development of this methodology.

1.4 Solution approach

Inspired by the human visual system, to rapidly search and focus on important areas to extract fine details and validate information by prior knowledge, this project proposes the Multi-model Agent for Remote Sensing (MARS) framework. MARS is an adaptive, Reinforcement Learning (RL)-driven system designed to mimic the human visual system's ability to perform fine-grain analysis. Instead of processing an entire image at a fixed, high resolution, a computationally expensive approach, MARS employs an intelligent agent that learns to strategically search on regions of interest, so that distributing computing resources property and gaining reasonable accuracy.

The methodology for implementing and evaluating the MARS framework is structured as follows:

1.4.1 Framework Architecture

The MARS framework consists of three core components that interact in a closed-loop system:

1. The Visual Environment:

   - State Space: The state represents the agent's current context and view of the input image. It contains an original entire source image, a current image window (or patch) the agent is examining, newfound objects extracted from window by the Visual Model, found objects, which is the set of all previously discovered objects, a scale level of the current window, the coordinate of the center of the window.

   - Action Space: The agent can take a customized number of actions to change its field of view. The action space is discrete and includes moving left, moving right, moving up, moving down, zoom in, zoom out.

   - Reward Function: The reward function is essentially for guiding the RL model's learning. It is designed to balance task performance improvement and computational cost.

     - Positive Reward: A base reward is given for each new object detection action (+1). A smaller reward is given for classification confidence. If not found, a small reward (+0.1) for a zoom-in action to encourage detailed searching.

     - Negative Reward / Cost: A small negative penalty (-0.2) is applied for every action taken without a new object detection.

2. The Visual Model (VM):

- The VM (YOLO) plays as the "eye" of the agent. Its responsibility is to recognize the objects from the image patch provided by the agent and return the result list, including orient bounding boxes, classes and confidences. This model is a pre-trained target dataset (DOTA) and was frozen during RL training. This ensures the RL model training based on a stable environment, reducing the complexity of training.

3. The Reinforcement Learning Model:

    - The agent is the "brain" of MARS. It is implemented by a Q-Table model.
    - At each step, the agent receives the state and selects an action.
    - The action was taken by the environment, leading to a new state and a reward.
    - The goal of the agent is to learn a policy that maximizes the reward, effectively learning to find the most objects with the least actions.

1.4.2 Operational Workflow

The detecting process for a single image is as follows:

1. **Initialization:** The source image is fed to the VM to obtain an initial detection (scale level = 1). The agent's initial window is set to the center of the image.

2. **Active Analysis Loop:** For each step t:
   a. The VM processes the image patch at current window and returns detections.
   b. The agent's state is updated with these new results, and the objects discovered are inspected and compiled into a list.
   c. The agent selects an action (e.g., moving left, moving right, moving up, moving down, zoom in or zoom out) based on the given action spaces provided by the environment.
   d. The environment moves the window, accordingly, extracts the new patch, calculates the reward, and updates the available action spaces.
   e. The experience state data class is stored for training.

3. **Termination:** The agent continues this process until it finds no action space or reaches a predefined maximum limit.

4. **Result Aggregation:** All detection results from each step are combined using Intersection of the Union area (IoU) to remove duplicates and produce the final output for the image.

1.4.3 Methodology for Addressing Aims and Objectives

The solution will be designed, implemented and validated through the following methodology:

- **Design and Implementation (Obj. 1 & 2):** The framework will be built in Python with the custom visual environment. A pre-trained YOLOv11 orient bounding box model will be treated as the frozen Visual Model and the baseline for comparison. The RL agent will be implemented using a Q-table model.

- **Evaluation and Comparison (Obj. 3 & 4):** Both models will be evaluated by the DOTA V1.0 validation dataset. Accuracy will be measured by mAP@0.5. Computational resources will be compared by average predicting time *per image*, and peak GPU memory usage. More importantly, the prediction time for MARS will be the total time for the entire process.

- **Policy Analysis (Obj. 5):** The agent's policy will be visualized by a visual webpage. This will provide qualitative insights into its decision-making strategy, showing if it learns to ignore areas and focus on new, unexplored areas.

1.4.4 Expected Outcome and Validation

The hypothesis is that MARS will achieve a higher **mAP** than the static baseline, particularly on small and occluded object classes, by actively seeking out and confirming ambiguous targets. Furthermore, it is expected that this performance gain will be achieved with a linear increase in computational costs relative to processing the whole image required for those small objects. The agent's learned policy will be the key factor to this efficiency, validating the core premise of the project.

1.5 Summary of contributions and achievements

This project makes several main contributions to the field of remote sensing image analysis and efficient AI model design. The key achievements and innovations are summarized as follows:

1. Theoretical and Methodological Contribution: The Proposal of the MARS Framework

- **Novel Paradigm:** This project introduces **MARS**, a novel multi-model agent framework that reformulates remote sensing image analysis from a static, one-pass process into a **dynamic, adaptive, and iterative search paradigm**. This paradigm change is inspired by the human visual system's saccadic movements and active perception.

- **Integrated Architecture:** We designed and implemented a closed-loop integration of a **Reinforcement Learning (RL) agent** with a **pre-trained, frozen visual model** (VM). This design allows the agent to learn an efficient policy for navigating the image space without the need for end-to-end training of the visual backbone, significantly reducing training complexity and computational resources.

2. Empirical Contribution: Demonstrated Superior Performance on Challenging Tasks

- **Enhanced Accuracy:** Through extensive evaluation of the benchmark DOTA dataset, the MARS framework **achieved a higher mean Average Precision (mAP)** compared to the baseline model (YOLOv11). This improvement was very obvious for **small, Partially Obscured, and shadowed objects** (e.g., small vehicles, ships), directly addressing **Research Question 1** and validating our primary hypothesis.

- **Qualitative Insights:** We provided a quantitative analysis of the agent's policy, visually demonstrating that it learns **intelligent, human-like strategies**. For instance, the agent learns to skip detected regions and dedicates its computational budget to zoom in and focuses on the new, unidentified areas. This achievement fulfills **Objective 5**.

3. Practical Contribution: The trade-off between performance and computing resources

- **Resource-Friendly AI:** The framework shows that it is possible to enhance model visual capability **without the need for complex scaled-up models**. By avoiding redundant computations on image patches, MARS presents a viable path toward deploying more vision AI on resource-restricted platforms, a significant step for practical IoT integration. This successfully addresses **Research Question 2**.

4. Contribution to the Community

- **Open-Source Implementation:** To facilitate reproducibility and further research in this direction, we will release the full source code of the MARS framework, including the custom RL environment, agent implementations, and evaluation scripts, to the public.

- **Benchmarking:** This study provides a new baseline and a methodological framework for future work on active and efficient perception in remote sensing and other computer vision domains.

In conclusion, this project not only achieves its aims and objectives but also makes substantive contributions by designing and valuating a new adaptive model architecture, showing its benefits for accuracy and efficiency, and offering valuable insights into the policies learned of AI agents for visual tasks.

1.6 Organization of the report

This report maintains a concise and logical structure with seven chapters. Chapter 2, Literature Review, provides comprehensive research of the existing relevant AI models. Chapter 3, Methodology and Design, details the architectural design and implementation of the MARS framework. Chapter 4, Results and Analysis, this chapter presents the empirical findings of the study. Chapter 5, Discussion and analysis, interprets the results presented in Chapter 4. It discusses the significance of the findings, explaining how the MARS framework achieves its performance gains and efficiency. Chapter 6, Conclusion and Future Work, summarizes the key achievements of the project and suggests promising directions for future research to build upon the foundations laid by this work.

Throughout this report, chapters and sections are referenced explicitly (e.g., Chapter 3, Section 3.2). Figures and tables are also named within their chapters (e.g., Figure 4.1, Table 5.3).

Chapter 2

Literature Review

2.1 Computer Vision Models in Remote Sensing

The application of Computer Vision (CV) to Remote Sensing (RS) has revolutionized Earth observation, including automated scene classification, object detection, and image segmentation. The field has evolved from traditional feature extraction methods to deep learning models, with Convolutional Neural Networks (CNNs) now representing the state-of-the-art.

An important example is the Ultralytics YOLO (You Only Look Once) models [1], which provide an excellent balance of speed and accuracy for oriented object detection, a critical task for identifying multiple objects in aerial images where alignment is arbitrary. For segmentation, the emergence of foundation models like the Segment Anything Model (SAM) [4] has inspired domain-specific adaptations such as SAMRS [2] and RingMo-SAM [3]. These models demonstrate significant potential for segmenting arbitrary objects in RS images, thereby aiding in detailed interpretation.

However, recent research has shown that these single-pass, static models still struggle to overcome several challenges [5]. These challenges include the accurate detection of **small, occluded, and shadowed objects**, high computational costs associated with processing very high-resolution images at a fixed scale, and a lack of adaptability to the vast heterogeneity present in RS data (e.g., varying altitudes, sensors, and lighting conditions). This underscores a clear gap in the literature: the need for more **adaptive and efficient frameworks** that can dynamically allocate computational resources.

2.2 Reinforcement Learning in Image Analysis

Reinforcement Learning (RL), with its strength in sequential decision-making, offers a paradigm shift from static analysis to active, adaptive perception. This approach mirrors how humans visually search for objects: adjusting one’s field of vision and attention to locate objects of interest.

The potential of RL has been successfully demonstrated in medical imaging, where agents learn navigation policies for tasks like segmentation and registration, iteratively refining their analysis to improve accuracy [8]. In remote sensing, this application is just getting started but rapidly advancing. For instance, the Scale-Aware Network (SAN) employs a Deep RL (DRL) agent to dynamically control the scale and viewing area of image patches for classification tasks. These pioneering works establish a crucial precedent: RL-driven agents can mimic intelligent, human-like inspection workflows by dynamically adjusting their focus based on the content of the scene. This capability is particularly critical for handling the challenges of occlusion and small objects prevalent in RS imagery.

2.3 Agent-Based Systems

A popular trend is the rise of agent frameworks that use Large Language Models (LLMs) as central planners to orchestrate specialized tools. Systems like RS-Agent [9] exemplify this architecture. They employ an LLM as a "central controller" that interprets complex user queries, programmatically invoke domain-specific models (e.g., a YOLO detector), and synthesizes answers through multi-step reasoning.

While RS-Agent focuses on high-level task planning and tool invocation, its core principle is highly relevant: breaking down a complex problem into a sequence of simpler, managed steps. The proposed MARS framework shares this foundation but differs fundamentally in its implementation. Instead of using an LLM for linguistic reasoning, MARS employs a **specialized RL agent for visual reasoning**. Its objective is not to answer a textual query but to exhaustively and accurately interpret an image through a learned policy of visual actions.

2.4 Remote Sensing Dataset

2.5 Critique of Existing Work and the Research Gap

The existing literature shows that Models like YOLO and SAM-derived systems are powerful but inherently static. These methods only process images once and use a fixed resolution; therefore, they are only suitable for objects within a specific size range. For tasks involving a wider range of object sizes, significant adjustments to the sampling strategy and retraining are necessary. **For AI Agents,** Frameworks like RS-Agent introduce valuable adaptability and sequential reasoning but operate at a high level of abstraction, orchestrating entire models rather than optimizing the fundamental visual perception process itself. Furthermore, language models require a significant amount of computing resources, which makes them unsuitable for integration into small services such as IoT.

2.6 Summary

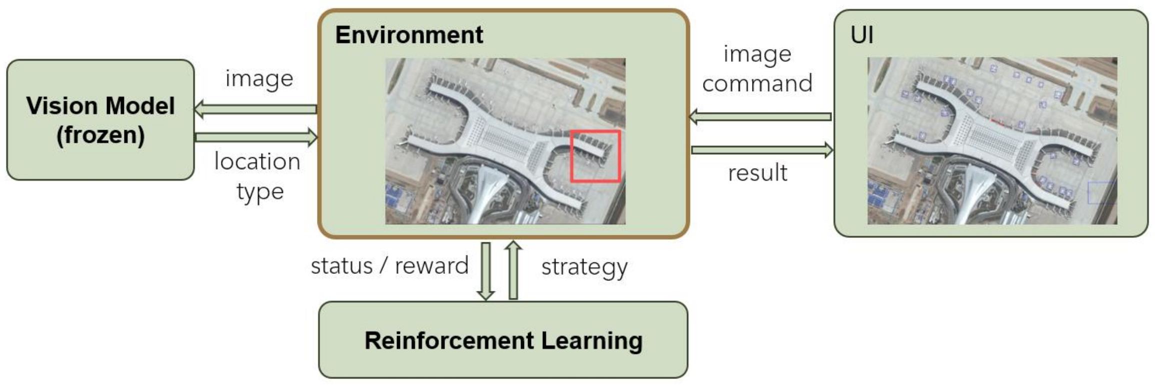
In conclusion, significant advancements in visual models can be directly applied to remote sensing images, such as YOLO for object detection and SAMRS for segmentation, which represent the state-of-the-art. However, these static models are limited by their fixed processing scale, struggle with detecting small or obscured objects. To address this problem, an ideal solution would combine the low-level, adaptive visual reasoning of RL with the sequential planning of an agent-based system to achieve a detail-extracted and accurate interpretation of remote sensing images.

Chapter 3

Methodology

3.1 System Structure

To obtain a flexible, adaptive and low-resource visual system, this project proposes the Multi-model Agent for Remote Sensing (MARS) framework. MARS is an adaptive reinforcement learning driven system integrated with a stable vision module, designed to mimic the human visual search process to achieve high-precision fine-grained analysis capabilities. Instead of processing an entire image at a fixed, high resolution, a computationally expensive approach, MARS employs an intelligent agent that learns to strategically search on regions of interest, so that distributing computing resources property and gaining reasonable accuracy.



The system structure of our MARS framework. An image is uploaded to the agent from the UI service webpage. The agent stores the original image in the environment and calls the visual model for the first recognition, obtaining the state, detection results, a reward and action spaces. Based on the current state and action spaces, the reinforcement learning model generates an action instruction according to the policy. Based on the instruction, the agent will update the state and the visual window to gain a new image patch. Then the image patch is processed by the vision model to identify more new objects and gain action spaces. The reward provided by the Environment is used to update the Reinforcement Learning policy. The agent decides whether to move on to the next recognition cycle by the action spaces. Finally, the agent stops exploring when it reaches a predetermined step or runs out of action space and displays the picture.

3.1.1 Environment: Zoomable Visual Exploration (ZooVE)

To investigate and enhance an agent's visual perception and active exploration capabilities, we developed a custom reinforcement learning (RL) environment named the **Zoomable Visual Exploration (ZooVE)** environment. This environment is implemented as a customizable module adhering to the popular OpenAI Gym API, ensuring compatibility with standard RL algorithms and frameworks.

The Core Components & State Representation of ZooVE environment:

- **Source Image**: A large, high-resolution RGB image which is the entire, static and source image. Source image serves as the ground-truth world the agent must explore.

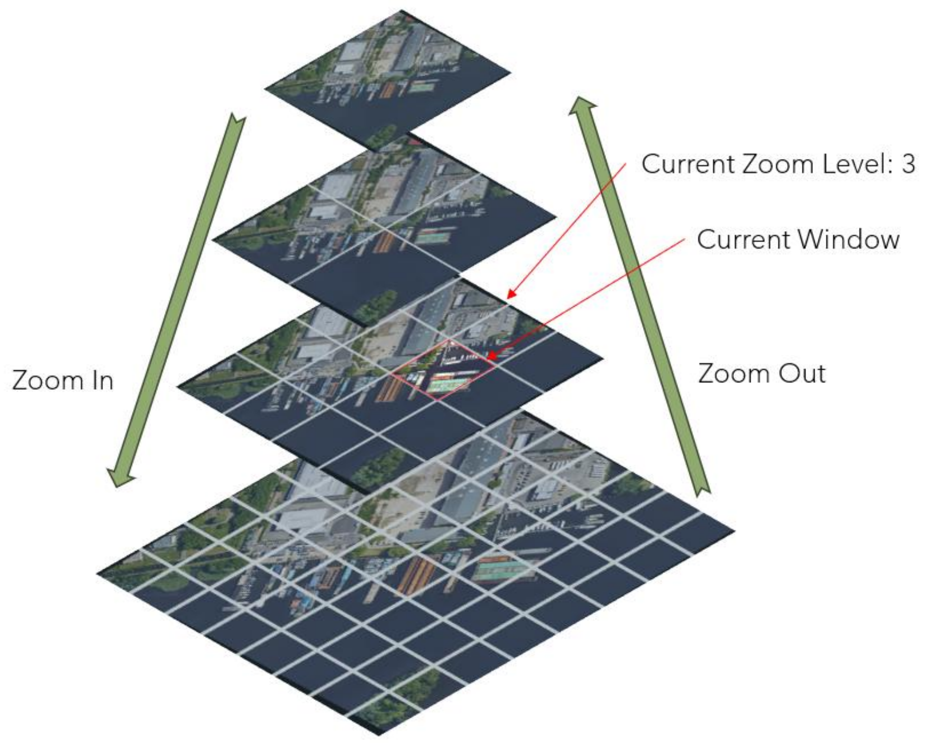
- **Current Zoom Level**: A set of discrete scaling factors (1, 2, 3 and 4) defining the magnification of the agent's view. A higher zoom level reveals finer details within a narrower field of view, while a lower zoom level provides broader contextual information. For example, 1 represents a zoom ratio of 1, 2 represents a zoom ratio of 0.5, 3 represents a zoom ratio of 0.25, and 4 represents a zoom ratio of 0.125.

- **Window:** A rectangular sub-region, defined by its centroid coordinates (cx, cy) and dimensions (w, h), which is determined by the current zoom level. This represents the agent's current observable window.

- **New Discovered Objects**: A dynamic list of objects [{class id, confidence, bounding box}] representing new found objects the agent has confidently identified this step. This is used to calculate the reward for the current step.

- **Discovered Objects**: A dynamic list of objects [{class id, confidence, bounding box}] representing all objects the agent has confidently identified. This acts as a memory module for the agent's achievements. Each new object found will be appended into this list, except that the interference value UoI greater than 0.2 will be merged.

- **Exploration Matrixes**: A two-dimensional binary matrix for each zoom level, spatially aligned with the source image, that tracks the explored regions. A variable in the matrix is labeled -1 if no object was detected in the previous step and the variable is within the agent's window; it is labeled 0 if it is not explored yet; otherwise, it is labeled with the confidence value which is greater than 0. This provides a key state variable that encourages efficient exploration and reduces repeated visits to regions.



The Exploration Space of MARS architecture. The window starts from the top layer (zoom level = 1) and gradually moves to the next step based on the prediction strategy. As the zoom level increases, the area of the corresponding window becomes smaller, and the pixels within the window are magnified, and the corresponding details are enhanced, achieving better recognition results.

- Action Spaces: The agent can take a customized number of actions to change its field of view. The action space is discrete and includes moving left, moving right, moving up, moving down, zoom in and zoom out. The action spaces provided by the environment build on prior experience to reduce the explosiveness of the exploration space and improve exploration efficiency. Constraints include that the window must be within the image bounds and that the viewport cannot repeatedly explore the same location at the same zoom level.

- Reward Function: The reward function is essentially for guiding the RL model's learning. It is designed to balance task performance improvement and computational cost.

  - Positive Reward: A base reward is given for each new object detection action (+1). A smaller reward is given for classification confidence. If not found, a small reward (+0.1) for a zoom-in action to encourage detailed searching.

  - Negative Reward / Cost: A small negative penalty (-0.2) is applied for every action taken without a new object detection.

3.1.2 Reinforcement Learning

The agent serves as the central decision-making module, or the "brain" of the MARS system. It is modeled using a Q-Table. At each timestep, the agent receives a state representation from the environment and uses it to select an action. This action is subsequently executed by the environment, resulting in a transition to a new state and the receipt of a reward signal. The agent's objective is to learn an optimal policy that maximizes its cumulative reward, which corresponds to efficiently discovering the maximum number of objects while minimizing the number of actions taken.

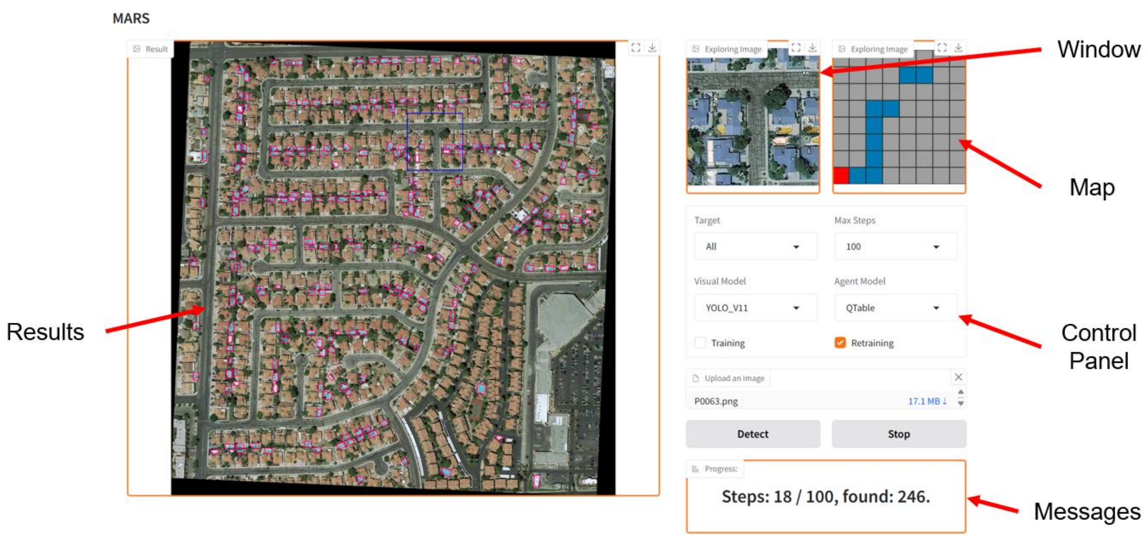
The Q-table is in key-value format. The key consists of a tuple, including the scale level, indexes in **Exploration Matrixes**. The value is the sum of the bonus from movement.

3.1.3 Vision model (VM)

The VM plays as the “eye” of the agent. Its responsibility is to recognize the objects from the image patch provided by the agent and return the result list, including orient bounding boxes, classes and confidences. This project uses the YOLO v11 orient bounding box model. This model is a pre-trained target dataset (DOTA) and was frozen during RL training. This model was chosen because it combines multi-object detection with a compact size and good performance. It is suitable for running on small GPUs or even in environments without a GPU. This ensures the RL model training based on a stable environment, reducing the complexity of training.

3.1.4 User interface

This project uses the popular AI application framework Gradio to build a web interface for user interaction. It includes a result display area on the left, a window and search map in the upper right corner, a control panel in the middle right, and a message display module in the lower right corner. A simple workflow involves visiting our website and clicking the Upload button to upload the target image. The image will be rendered on the left panel. The user then configures the corresponding parameters and clicks the Detect button to trigger the recognition process. During the recognition process, the backend agent returns the recognition results, the current window, and the recognition map to the frontend for display in real time. The frontend continues to update and display the backend information until recognition is complete.



Target: All
Max Steps: 100
Visual Model: YOLO_V11
Agent Model: QTable
Training
Retraining
P0063.png 17.1 MB

Steps: 18 / 100, found: 246.

The Web Interface of MARS architecture. It includes a result display area on the left, a window and search map in the upper right corner, a control panel in the middle right, and a message display module in the lower right corner.

3.2 Dataset

To verify the visual capabilities of the model, we selected the mainstream remote sensing image database DOTA (Dataset for Object deTection in Aerial images). DOTA v1.0 is a large-scale benchmark dataset designed for evaluating object detection algorithms in aerial imagery. It is widely recognized in the computer vision and remote sensing communities for its complexity, high resolution, and the diverse range of objects it contains.

3.3 Testing and Validation

The standard evaluation protocol for DOTA v1.0 follows the common practices in object detection, using Average Precision (AP) and mean Average Precision (mAP) as the primary metrics.

- Intersection over Union (IoU): IoU measures the overlap between a predicted bounding box and a ground truth bounding box.
- mean Average Precision (mAP): This is the primary metric for ranking and comparing models on DOTA. It is computed by taking the mean of the AP scores across all 15 object classes.

$$mAP = (AP_1 + AP_2 + \ldots + AP_{15}) / 15$$

3.4 Summary

This project introduces the MARS framework, an adaptive system designed for efficient, high-precision analysis of large aerial images. Instead of processing an entire high-resolution image—a computationally expensive task—MARS mimics the human visual search process. It uses a reinforcement learning (RL) agent that learns to strategically explore only regions of interest, thereby distributing computing resources wisely.

The system's core is a custom RL environment called Zoomable Visual Exploration (ZooVE). Here, an agent navigates a large source image through a movable and zoomable window. Its action space includes moving up, down, left, right, zooming in, and zooming out. The agent's goal is to maximize its cumulative reward, which is earned primarily by identifying new objects (using a pre-trained YOLO vision model) and is penalized for inefficient, repetitive actions. The environment tracks explored areas using an exploration matrix to prevent redundant searches.

The agent itself, the "brain" of MARS, has been implemented as a Q-Table. It makes decisions based on the current state of ZooVE, which includes the zoom level, window position, and the history of discovered objects. A web interface built with Gradio allows users to upload images, configure parameters, and view the recognition process and results in real-time. The system was validated using the DOTA v1.0 dataset, a large-scale benchmark for object detection in aerial imagery, demonstrating its capability for fine-grained, low-resource visual analysis.

Chapter 4

Results

Chapter 5

Discussion and Analysis

Chapter 6

Conclusions and Future Work

Chapter 7

Reflection

References

[1] Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., & Ding, G. (2024). YOLOv10: Real-Time End-to-End Object Detection. https://arxiv.org/pdf/2405.14458

[2] Wang, D., Zhang, J., Du, B., Xu, M., Liu, L., Tao, D., & Zhang, L. (2023). SAMRS: Scaling-up Remote Sensing Segmentation Dataset with Segment Anything Model. ArXiv.org. https://arxiv.org/abs/2305.02034

[3] Yan, Z., Li, J., Li, X., Zhou, R., Zhang, W., Feng, Y., Diao, W., Fu, K., & Sun, X. (2023). RingMo-SAM: A Foundation Model for Segment Anything in Multimodal Remote-Sensing Images. IEEE Transactions on Geoscience and Remote Sensing, 61, 1–16. https://doi.org/10.1109/tgrs.2023.3332219

[4] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). Segment Anything. ArXiv:2304.02643 [Cs]. https://arxiv.org/abs/2304.02643

[5] Janga, B., Asamani, G. P., Sun, Z., & Cristea, N. (2023). A Review of Practical AI for Remote Sensing in Earth Sciences. Remote Sensing, 15(16), 4112. https://doi.org/10.3390/rs15164112

[6] Hu, Y., Yuan, J., Wen, C., Lu, X., Liu, Y., & Li, X. (2025). RSGPT: A remote sensing vision language model and benchmark. ISPRS Journal of Photogrammetry and Remote Sensing, 224, 272–286. https://doi.org/10.1016/j.isprsjprs.2025.03.028

[7] Kuckreja, K., Danish, M. S., Naseer, M., Das, A., Khan, S., & Khan, F. S. (2023). GeoChat: Grounded Large Vision-Language Model for Remote Sensing. ArXiv.org. https://arxiv.org/abs/2311.15826

[8] Hu, M., Zhang, J., Matkovic, L., Liu, T., & Yang, X. (2022). Reinforcement learning in medical image analysis: Concepts, applications, challenges, and future directions. Journal of Applied Clinical Medical Physics, 24(2). https://doi.org/10.1002/acm2.13898

[9] Xu, W., Yu, Z., Wang, Y., & Wang, J. (Eds.). (2023). RS-Agent: Automating Remote Sensing Tasks through Intelligent Agents. Arxiv.org. https://arxiv.org/html/2406.07089v1

[10] Ding, J., Xue, N., Xia, G.-S., Bai, X., Yang, W., Yang, M., Belongie, S., Luo, J., Datcu, M., Pelillo, M., & Zhang, L. (2021). Object Detection in Aerial Images: A Large-Scale Benchmark and Challenges. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1–1. https://doi.org/10.1109/tpami.2021.3117983

[11] Ke Li, Gang Wan, Gong Cheng, Liqiu Meng, and Junwei Han. Object detection in optical remote sensing images: A survey and a new benchmark. ISPRS journal of photogrammetry and remote sensing, 159:296–307, 2020

[12] Li, X., Ding, J., & Elhoseiny, M. (2024). VRSBench: A Versatile Vision-Language Benchmark Dataset for Remote Sensing Image Understanding. ArXiv.org. https://arxiv.org/abs/2406.12384

Appendix A

An Appendix Chapter (Optional)

Appendix B

An Appendix Chapter (Optional)

…