**FutureSightDrive: Thinking Visually with Spatio-Temporal CoT for Autonomous Driving**

原文出处：<https://arxiv.org/abs/2505.17685>

日期：2025.05

源码：<https://github.com/MIV-XJTU/FSDrive>

**Motivation**

（Abstract）

**Existing VLMs typically utilize discrete text Chain-of-Thought (CoT) tailored to the current scenario, which essentially represents highly abstract and symbolic compression of visual information, potentially leading to spatio-temporal relationship ambiguity and fine-grained information loss.** Is autonomous driving better modeled on real-world simulation and imagination than on pure symbolic logic?

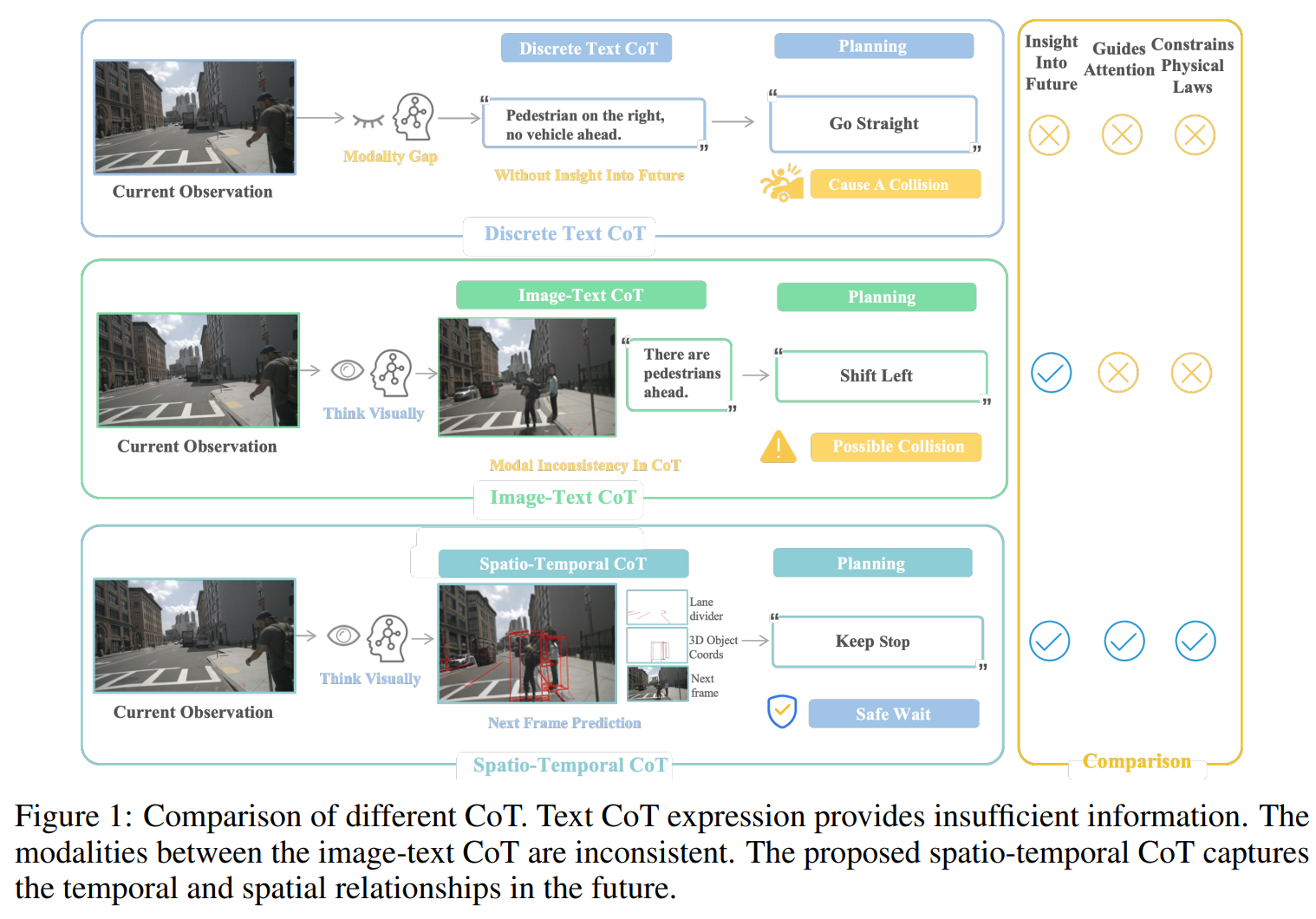
（Introduction）

One promising direction is the end-to-end vision-language-action (VLA) model, which leverages pre-trained vision-language model (VLM) to directly extract scene features from visual observations and language instructions, subsequently generating vehicle control commands (e.g., speed and trajectory).

In the field of language, Chain-of-Thought (CoT) [63, 53] improves reasoning capabilities and interpretability by encouraging step-by-step thinking. **However, existing autonomous driving [18, 29] typically incorporate discrete text CoT (e.g., language descriptions targeting current scenarios and bounding box coordinates) as intermediate reasoning steps.**

**This method is essentially highly abstract and symbolized compression of visual information, which may lead to ambiguous temporal and spatial relationships, loss of fine-grained information, and modality conversion gaps [32], as shown in the top of Figure 1.**

For autonomous vehicles requiring deep physical-world interaction, should their thinking process more closely resemble simulation and imagination of world, rather than merely relying on logical deduction of language?



**Contribution**

（Abstract）

In this paper, we propose a spatio-temporal CoT reasoning method that enables models to think visually.

First, VLM serves as a world model to generate unified image frame for predicting future world states: where perception results (e.g., lane divider and 3D detection) represent the future spatial relationships, and ordinary future frame represent the temporal evolution relationships.

This spatio-temporal CoT then serves as intermediate reasoning steps, enabling the VLM to function as an inverse dynamics model for trajectory planning based on current observations and future predictions.

To implement visual generation in VLMs, we propose a unified pretraining paradigm integrating visual generation and understanding, along with a progressive visual CoT enhancing autoregressive image generation.

（Introduction）

Inspired by the human driver’s cognitive mechanism of directly constructing visual representations of future scenarios in the mind, rather than converting them into language descriptions for reasoning, we propose a more intuitive spatio-temporal CoT method as shown in the bottom part of Figure 1. This method avoids information loss during text abstraction and enables the model to think visually about trajectory planning.

**First, the VLM serves as a world model to generate unified image frame for predicting future world states**: Inspired by visual prompting engineering [38] that draws red circles on images to guide model attention and by VLIPP [61] first predicts future bounding boxes to introduce physical priors when generating future frames, we represent future world spatial relationships through future red lane dividers and 3D detection boxes on the predicted unified frames.

**Subsequently, the spatio-temporal CoT acts as an intermediate reasoning step, enabling the VLM to function as an inverse dynamics model for trajectory planning based on current observations and future predictions.**

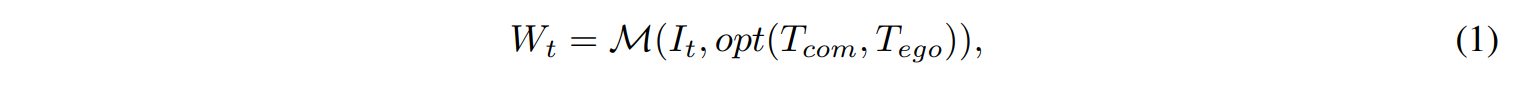
Compared to traditional discrete text CoT, and even image-text CoT methods [18, 68] as shown in the middle of the Figure 1, our method unifies both future scene representations and perception outputs in image format, which more effectively conveys the temporal and spatial relationships.

To endow VLMs with image generation capabilities, we **propose** **a pre-training paradigm that simultaneously preserves the semantic understanding of existing MLLM and activates their visual generation capacity.** Specifically, for the semantic understanding preservation part, we follow previous approaches [50, 18] by incorporating visual question answering (VQA) tasks for current scene comprehension. For the activation of visual generation capabilities, we investigate the shared vocabulary space between image and text, directly unleashing the visual generation potential of existing MLLMs in the field of autonomous driving through minimal data (approximately 0.3% of previous methods [54, 56]) without requiring complex model architecture modifications or redesigns.

**However, directly generating complete detailed future scenes may fail to adhere to physical laws [61]. Thus, we propose a progressive, easy-to-hard generation method. We leverage the world knowledge of VLMs to first infer drivable regions and key object positions in future scenarios, generating coarse-grained future perception images (e.g., lane dividers and 3D detection) to constrain physical laws.**

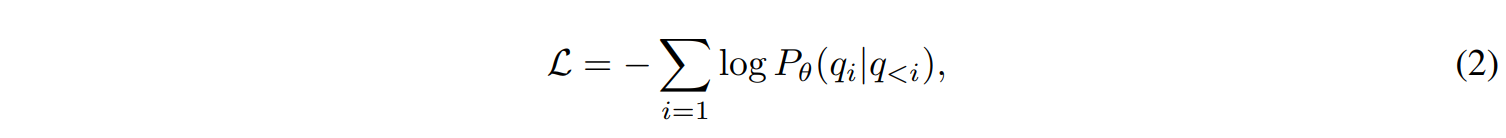
**Method—Preliminary**

**End-to-end trajectory planning.** End-to-end autonomous driving directly generates future trajectory from sensor data, convertible to vehicle control actions like acceleration and steering [18]. given *N* surround-view images *I*t = {*I*1*t*, *I*2*t*, …, *INt*} at timestep *t*, model *M* outputs a BEV trajectory *W*t = {*w*1*t*, *w*2*t*, …, *wnt*}, where each waypoint *wit* = (*xit*, *yit*). The process is formulated as:



*opt*(*Tcom*, *Tego*) denotes optional navigation commands and ego status (e.g., velocity, acceleration).

**Unified visual generation and understanding.** Recent works [48, 55, 45] unify multimodal understanding and vision generation in single LLM. While understanding aligns with standard LLMs, generation methods [43, 27, 57] typically use VQ-VAE [46] to encode images into discrete tokens. First, the image tokenizer quantizes image pixels x∈ℝ*H*×*W*×3 into discrete tokens *q*∈𝒬*h*×*w*, where *h* = *H*/*p*, *w* = *W*/*p*, *p* is the downsampling factor, and *q*(*i*, *j*) represents the index of the image codebook. These *h*·*w* tokens are arranged in raster order to train a Transformer [47]-based autoregressive model. During image generation, a general language modeling (LM) objective is adopted to autoregressively predict the next token, maximizing the likelihood of each image token:



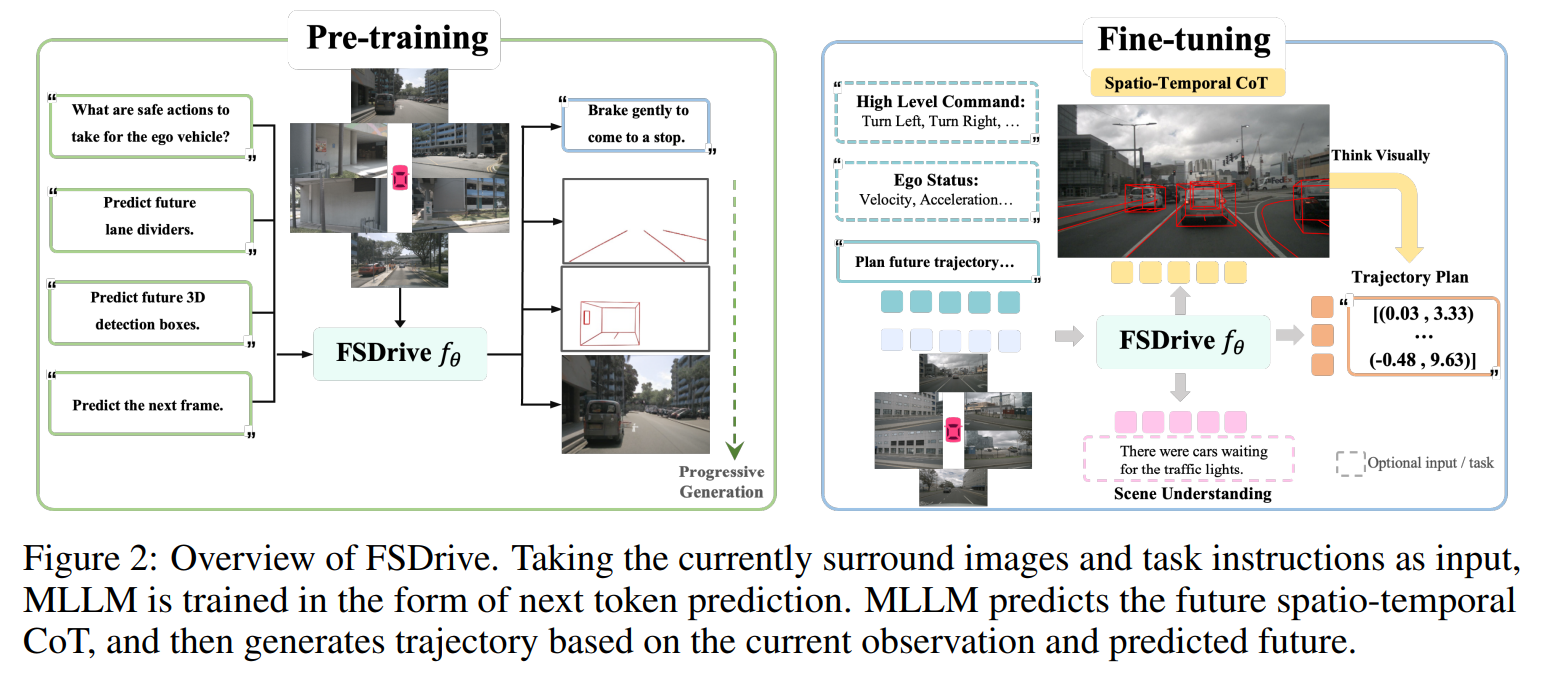
where *qi* denotes the visual token and *θ* represents the LLM parameters. Finally, the VQ-VAE’s detokenizer converts these image tokens back into image pixels.

**Method—Unified pre-training paradigm for visual generation and understanding**

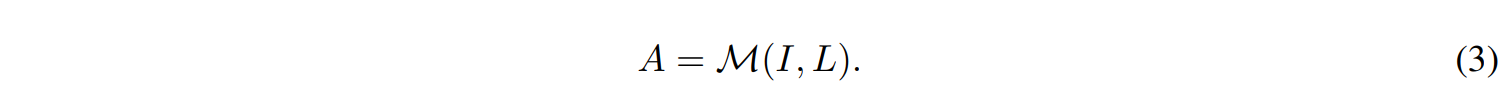
Existing methods (e.g. Lumina-mGPT [26], the visual generation LLM used by Doe-1 [72]) typically employ VQ-VAE to encode images into discrete tokens when extracting visual information. However, these tokens lack semantic information, which hurts downstream understanding performance [57]. Moreover, current methods [54, 9, 57] demand expensive training from scratch on massive billion-scale datasets without leveraging existing LLM knowledge.

**Our method is directly built upon any existing MLLM that employs ViT-based encoders** to convert images into continuous features. **We preserve the original MLLM architecture without altering any structural components** to maintain compatibility with pretrained weights. **The sole modification involves expanding the MLLM’s vocabulary by incorporating image tokens of the VQ-VAE into the text codebook**, thereby extending the vocabulary’s scope from language space to a multimodal space encompassing both visual and textual modalities.

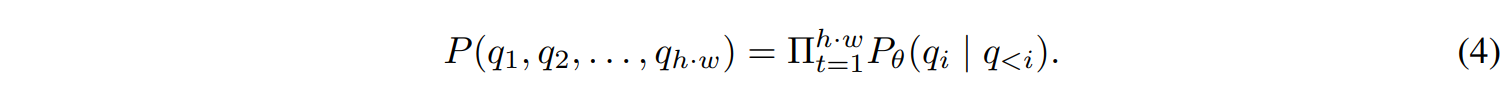
***Pre-training for visual understanding.*** **To effectively** **preserve the semantic understanding capabilities of the native MLLM during the pre-training stage, as shown in the left part of Figure 2**, **we follow previous methods [50, 18] by using a VQA task**, which is crucial for autonomous vehicles to analyze complex driving scenarios.



**Given an image-text question-answer pair (*I*, *L*), where *I* represents the surround-view images of the current scene and *L* denotes the instructional question, model *M* generates a corresponding answer *A*:**



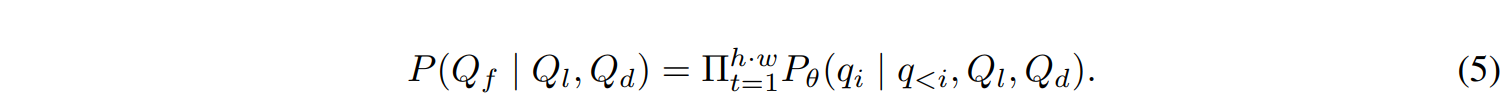
***Pre-training for visual generation*.** Inspired by the world models in autonomous driving [21, 60] that generate future frames to learn physical laws, after activating the visual generation capability, we also enable the VLM to predict future frames, thereby capturing the dynamic evolution of the world. Specifically, **given an image-instruction pair (*I*, *L*), the model predicts the next visual token of the future front-view frame through autoregressive generation:**



**The predicted visual tokens are then converted back into image pixels by VQ-VAE’s detokenizer.** Since future frames naturally exist in video datasets without requiring any labeled data, this approach unlocks the potential to harness abundant video data for improving generation quality.

***Progressive image generation*.** However, directly generating complete detailed future scenes may fail to adhere to physical laws [61]. Therefore, **during pre-training stage, we propose a progressive, easy-to-hard generation method, incorporating annotated data containing lane divider and 3D detection.**

Before generating **visual tokens of future frames *Qf***, we leverage the world knowledge of VLM to first reason about **visual tokens of lane dividers *Ql***, which serve as the skeleton of the road scene and define drivable areas to enforce static physical constraints. Subsequently, we reason about **visual tokens of 3D bounding boxes *Qd***, representing motion patterns of key objects to impose dynamic physical constraints. This progressive method sequence explicitly guides the model to infer structural layouts and geometric details of future scenes while enforcing physical laws. **By leveraging these intermediate visual reasoning steps as context, the model learns to think visually about the dynamic evolution of scenes, ultimately enabling accurate future prediction:**

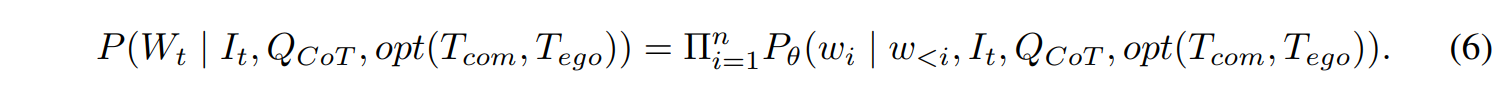


**Method—Think visually with spatio-temporal CoT**

Since our model has already learned physical constraints through the progressive generation during pre-training, and considering efficiency, we no longer separately generate lane dividers, 3D detection, and future frames, but instead integrate all these results into a single unified frame.

**As shown in the right part of Figure 2, here, VLM serves as a world model to generate a unified image frame predicting the future world state**: Inspired by visual prompting engineering [38] that draws red circles on images to guide model attention and by VLIPP [61] first predicts future bounding boxes to introduce physical priors when generating future frames, we represent future world spatial relationships through future red lane dividers and 3D detection boxes on the predicted unified frames. These coarse-grained visual cues direct the model’s attention toward drivable areas and critical objects in future scenes while enforcing physically plausible constraints.

**Subsequently, spatio-temporal CoT *QCoT* serves as an intermediate reasoning step**, allowing the VLM to function as an inverse dynamics model that plans trajectory based on current observations and future predictions:



**Method—Training strategy**

**Our FSDrive can be initialized from any existing MLLM (e.g., Qwen2-VL, LLaVA)**, avoiding training from scratch and saving significant costs. **During training, we fully fine-tune the LLM parameters while freezing all encoders.** The training process is divided into two stages:

***Stage 1: Unified pre-training.*** Our objective is to preserve understanding capabilities of MLLMs through **VQA tasks** and activate their visual generation capabilities to **predict future frames**. **VQA task data originates from OmniDrive-nuScenes** [50]. **We incorporate a large volume of unlabeled image data from nuScenes** [2] **for future frame prediction. To implement progressive easy-to-hard CoT, we integrate nuScenes annotated data to teach the model predicting image-formatted future lane dividers and 3D detection.** **Finally, we add future frame prediction with CoT datas containing intermediate reasoning steps.** All the above understanding and generation tasks are trained together.

**Stage 2: Supervised fine-tuning.** We focus on autonomous driving scene understanding and trajectory planning. **Following OminiDrive [50], scene understanding utilizes DriveLM’s GVQA [39] dataset.** **For trajectory planning, we follow VAD [20, 16] using nuScenes**, **where our spatio-temporal CoT integrates the holistic future scene, explicit lane dividers, and 3D detection results into a single future frame as intermediate reasoning steps.** **We train these tasks simultaneously using a single model**, enabling task-specific predictions during inference through different task prompts.

**Datasets**

Following the previous methods [20, 8], we **evaluate trajectory planning and future frames generation on the** **nuScenes** [2]. The nuScenes contains 1,000 scenes of approximately 20 seconds each captured by a 32-beam LiDAR and six cameras providing 360-degree field of view. Specifically, The dataset provides 28,130 (train), 6,019 (val), and 193,082 (unannotated) samples.

Following the previous methods [5, 50], we **evaluate scene understanding on DriveLM** [39]. This dataset features keyframe descriptions paired with QA annotations covering full-stack autonomous driving (perception, prediction, planning), offering comprehensive language support for development.

**Metrics**

We **evaluate trajectory planning using L2 displacement error and collision rate** following previous methods [16, 20, 14]. **Notably, UniAD** [16] **computes L2 metrics and collision rate at each timestep, whereas ST-P3** [14] **and VAD** [20] **considers the average of all previous time-steps.** **For a fair comparison, we adopted these two different calculation methods.** Following existing methods [51, 60], we **report Fréchet Inception Distance (FID)** [11] **to measure the future frames generation quality.** **DriveLM GVQA** [39] **metrics include language metrics like BLEU, ROUGE\_L, and CIDEr for text generation, the ChatGPT Score for open-ended Q&A and accuracy for multiple-choice questions.**

**Implementation details**

**We initialize our model with Qwen2-VL-2B** [49] **and pre-train it for 32 epochs to enable visual generation while preserving semantic understanding.** **During fine-tuning (12 epochs on 8 NVIDIA RTX A6000), we use 1×10−4 learning rate and batch size of 16. We expand the visual codebook of MoVQGAN** [69] **to the vocabulary of the large language model and use its detokenizer to convert the visual tokens predicted by the large language model to the pixel space.**

**LaV-CoT: Language-Aware Visual CoT with Multi-Aspect Reward Optimization for Real-World Multilingual VQA**

原文出处：<https://arxiv.org/abs/2509.10026>

日期：2025.10

源码：<https://github.com/HJNVR/LaV-CoT?tab=readme-ov-file>

**Motivation**

（Abstract）

Most existing approaches rely primarily on textual CoT and provide limited support for multilingual multimodal reasoning, constraining their deployment in real-world applications.

（Introduction）

**Challenge 1:** Recent Visual CoT methods [42, 65] incorporate visual cues to enable multimodal reasoning, but still suffer from inconsistency across languages. Such limitations constrain the deployment of VLMs in international products, which demand both robustness and interpretable reasoning.

**Challenge 2:** A further challenge lies in constructing high-quality multilingual reasoning data. Manual annotation is costly and often infeasible at scale.

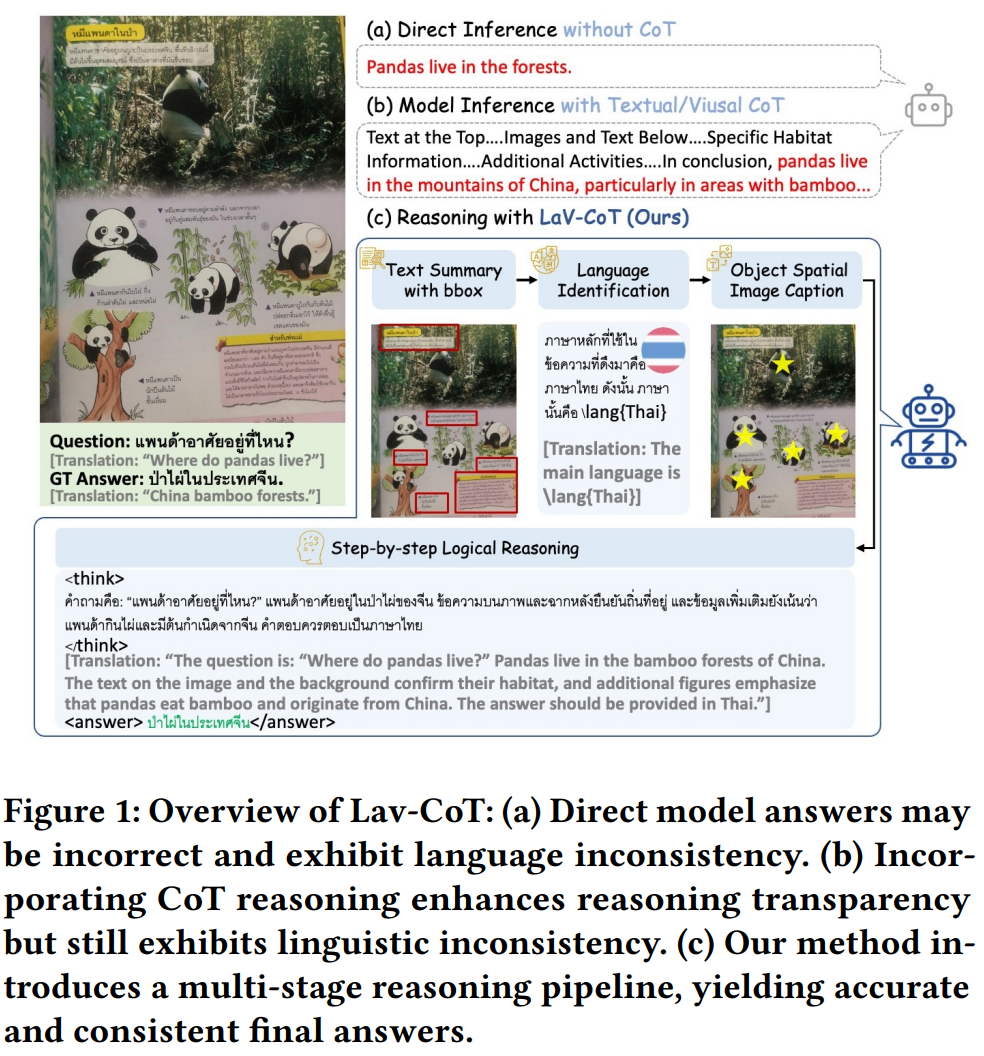
**Contribution**

（Abstract）

**We introduce LaV-CoT, the first Language-aware Visual CoT framework with Multi-Aspect Reward Optimization.** LaV-CoT incorporates an interpretable multi-stage reasoning pipeline consisting of Text Summary with Bounding Box (BBox), Language Identification, Spatial Object-level Captioning, and Step-by-step Logical Reasoning. Following this reasoning pipeline, we design an automated data curation method that generates multilingual CoT annotations through iterative generation, correction, and refinement, enabling scalable and high-quality training data. To improve reasoning and generalization, **LaV-CoT adopts a two-stage training paradigm combining Supervised Fine-Tuning (SFT) with Language-aware Group Relative Policy Optimization (GRPO)**, guided by verifiable multi-aspect rewards including language consistency, structural accuracy, and semantic alignment.

（Introduction）

To address challenge 1, we propose LaV-CoT, a Languageaware Visual CoT reasoning framework with Multi-Aspect Reward Optimization. As illustrated in Figure 1, LaV-CoT introduces an interpretable multi-stage reasoning pipeline that integrates four key components: **(1) Text Summary with Bounding Box (BBox)** [61], **(2) Language Identification**, **(3) Spatial Object-level Captioning** [18], and **(4) Step-by-step Logical Reasoning** [49, 60, 64]. This structured pipeline explicitly disentangles language and visual reasoning, enabling fine-grained cross-modal alignment and improving interpretability in multilingual settings.

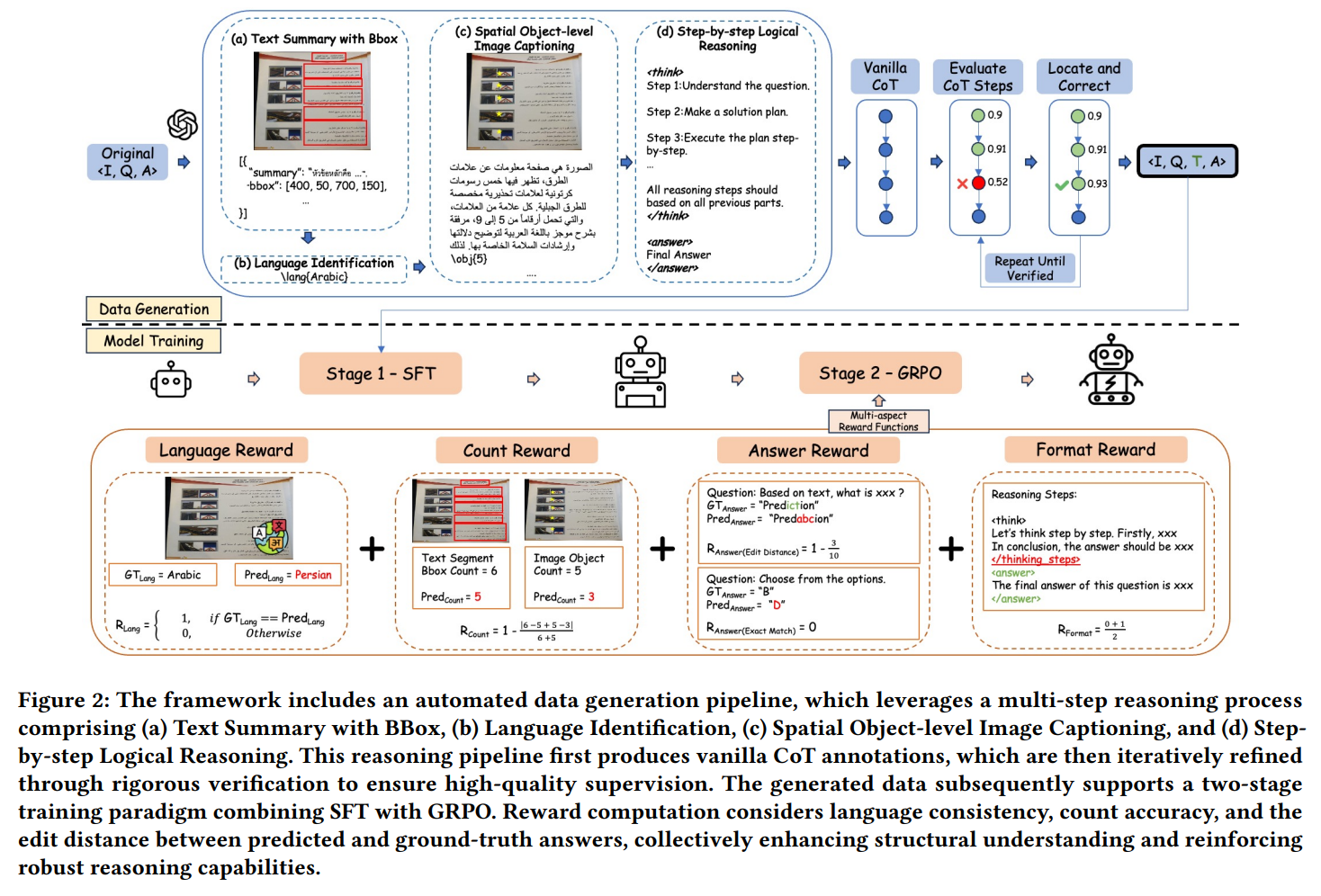


To address challenge 2, **we design an automatic data curation method that generates multilingual CoT annotations via iterative generation, correction, and refinement** [28, 43, 62]. This scalable process produces structured and verifiable reasoning traces, ensuring that training data captures both linguistic fidelity and multimodal reasoning quality.

**On the training side, LaV-CoT adopts a two-stage paradigm combining Supervised Fine-Tuning (SFT)** [5, 9, 24] **with Language-aware Group Relative Policy Optimization (GRPO)** [32, 34, 44]. **Unlike standard optimization methods, our GRPO variant is guided by verifiable multi-aspect rewards**—including language consistency reward, text segments and object count reward, final answer edit distance reward and format reward—yielding stable optimization and robust reasoning across languages and modalities.

**Method—Multi-stage Reasoning**

The multi-stage reasoning design is motivated by the way humans naturally approach image understanding: when examining a document image, we first locate salient text regions and retain their summarized content, then identify the language, recognize objects and their spatial relationships, and finally integrate all information to reason step by step toward an answer. As illustrated in **Figure 2**, our model performs reasoning through four structured stages:



(a) ***Text Summary with Bounding Boxes*.** For text-centric images, we first detect text segments within the image. Since extracting the complete OCR content may be costly in terms of token usage, we apply a summarization strategy to generate concise yet informative representations of the detected text. **The output is formatted as a list of bounding boxes paired with summarized text, where the length of the list is later used as a reward signal during training.**

(b) ***Language Identification*.** Following the saying that the best way to learn a foreign language is to think in that language, **for images containing textual content in any language, we identify the primary language based on the summarized text obtained in the previous step.** **The identified target language is explicitly marked using a \lang{} tag**, which allows reward calculation to directly compare the predicted and ground-truth languages.

(c) ***Spatial Object-Level Image Captioning*.** To capture comprehensive information from the image, **we describe not only the main objects but also their spatial positions**, thereby providing a structured understanding of the visual scene. **In addition, the model outputs a total object count, explicitly marked using an \obj{} tag**, which provides a quantitative signal that can be evaluated during reward computation.

(d) ***Step-by-Step Logical Reasoning*.** **Leveraging the outputs from all previous steps as evidence, we first understand the given question, then devise a detailed solution plan, and finally execute the plan step-by-step until arriving at the final answer.**

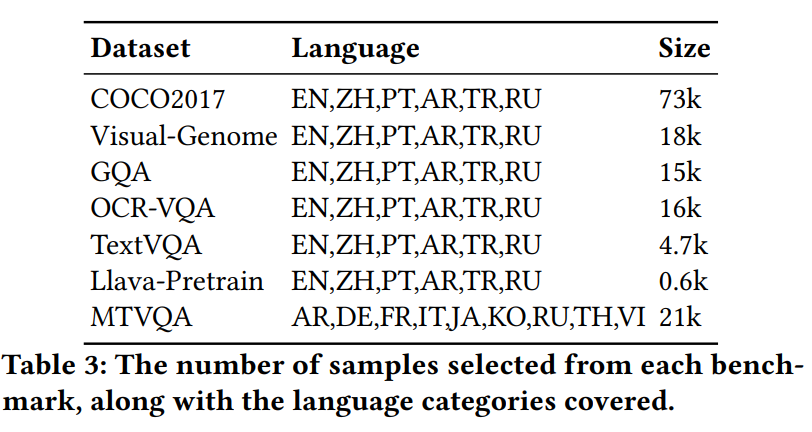
**Once trained, our model can autonomously determine when to initiate each stage, requiring no additional prompts, and complete all stages within a single inference pass.**

This end-to-end structured reasoning process not only improves robustness and effectiveness but also enables reward computation **based on multiple aspects: the length of the bounding-box list, correctness of the \lang{} tag, and accuracy of the \obj{} count**.

**Method—Dataset Curation**

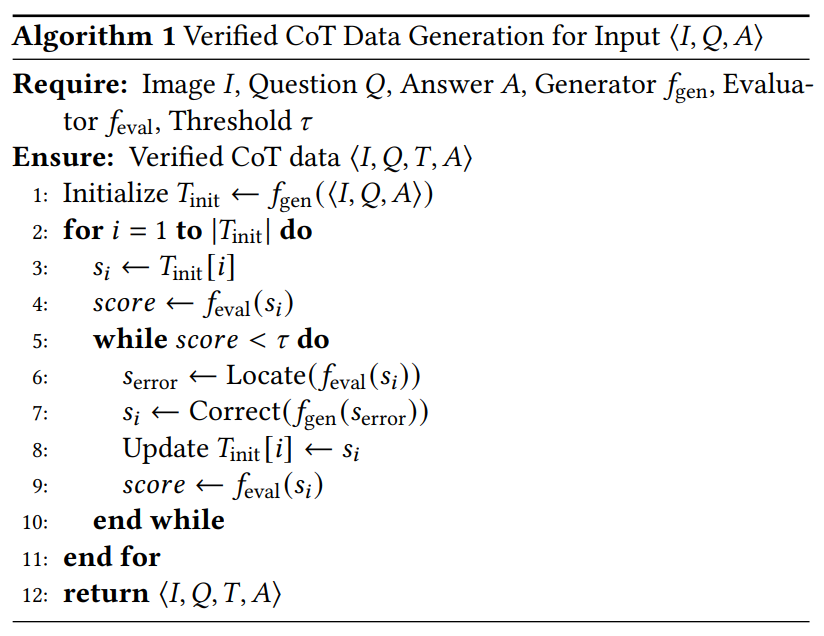
Most existing VQA datasets lack the detailed reasoning processes necessary to effectively train the multilingual reasoning vlm model.

To address this limitation, we compile a new dataset by integrating samples from several widely used VQA benchmarks, resulting in a total of 148k image–question–CoT–answer pairs, as illustrated in Table 3.



Our dataset covers 13 languages, including English (EN), Chinese (ZH), Portuguese (PT), Arabic (AR), Turkish (TR), Russian (RU), German (DE), French (FR), Italian (IT), Japanese (JA), Korean (KO), Thai (TH), and Vietnamese (VI). These languages represent a diverse range of linguistic families.

Specifically, as shown in Algorithm 1, **we start from the original question–answer triplets ⟨𝐼, 𝑄, 𝐴⟩, where 𝐼 denotes the image, 𝑄 the question, and 𝐴 the corresponding answer.**



**We** **first prompt a GPT-based generator 𝑓gen to produce an initial sequence of vanilla Chain-of-Thought (CoT) steps** . **Next, we prompt an evaluator 𝑓eval to score each step 𝑠𝑖∈𝑇init.** **For any step whose score falls below the threshold 𝜏, we iteratively perform the following procedure:** **First,** we apply the evaluator to the step and **then** locate the erroneous part, denoted as 𝑠error, using the function Locate(𝑓eval(𝑠𝑖)). **Next**, a corrected step 𝑠𝑖 is generated by applying the function Correct to 𝑠error, i.e., 𝑠𝑖 is updated as Correct(𝑓gen(𝑠error)). The corrected step then replaces the original step in the sequence 𝑇*init*. **Finally**, the updated step is re-evaluated to obtain the new score using 𝑓eval(𝑠𝑖).

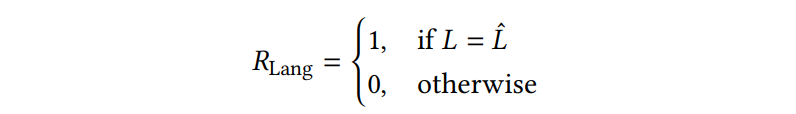
**This evaluation–correction– update loop repeats until all steps in 𝑇init exceed the threshold.** **The final verified Chain-of-Thought sequence is denoted as 𝑇, and the output dataset consists of ⟨𝐼, 𝑄, 𝑇, 𝐴⟩. Detailed prompt design are shown in Appendix B.**

**Method—Model Training**

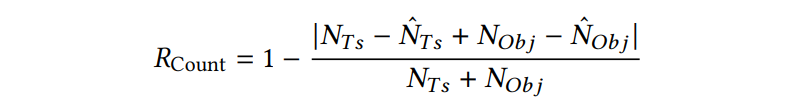
**We adopt Supervised Fine-Tuning (SFT)** to equip the model with robust multilingual and multimodal reasoning abilities. **To further refine output quality, we introduce a composite reward function, 𝑅Multi\_Aspect,** which aggregates multiple rule-based criteria into a single supervisory signal. Instead of relying on an explicit value function or critic, candidate outputs are comparatively evaluated within the same group, and their raw rewards are transformed into relative advantage scores. This pairwise normalization stabilizes optimization and promotes the preference for higher-quality responses.

**The multi-aspect reward 𝑅Multi\_Aspect consists of four complementary components:** **three novel rewards** designed to capture different aspects of multilingual visual-textual reasoning, and **a default format reward** that ensures the generated output adheres to the expected structural conventions.

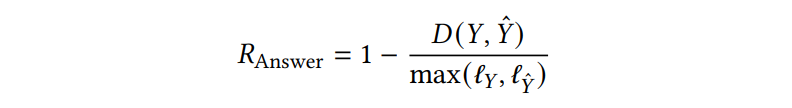
***Language Consistency Reward* (𝑅Lang)**. To encourage the model to perform reasoning in the target language, we **compare the language predicted by the model with the labeled primary language of the input.** **Let 𝐿 denote the ground-truth language label and  represent the language identified by the model. The reward is defined as:**



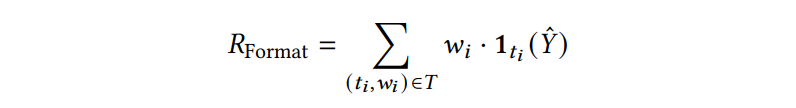
***Text Segments and Object Count Reward* (𝑅Count)**. To ensure accurate text segmentation and object count, l**et 𝑁𝑇𝑠 and  be the numbers of reference and predicted text segments**, and 𝑁𝑂𝑏𝑗 and **** be the numbers of reference and predicted main objects. The reward is defined as:



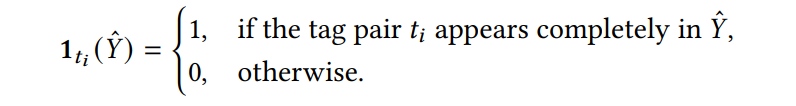
***Edit Distance of Final Answer Reward* (𝑅Answer).** **We compute the normalized Levenshtein distance 𝐷(𝑌, ) between the reference answer 𝑌 and the model prediction :**



***Format Reward* (𝑅Format).** This reward encourages the model to generate outputs that adhere to the prescribed structural format. Let **** denote the model output, and let 𝑇 = {(𝑡𝑖, 𝑤𝑖)} be the set of required tag pairs 𝑡𝑖 with weights 𝑤𝑖∈[0, 1] such that ****, e.g., <think></think> and <answer></answer>. **The format reward is computed as a weighted sum over all tags:**



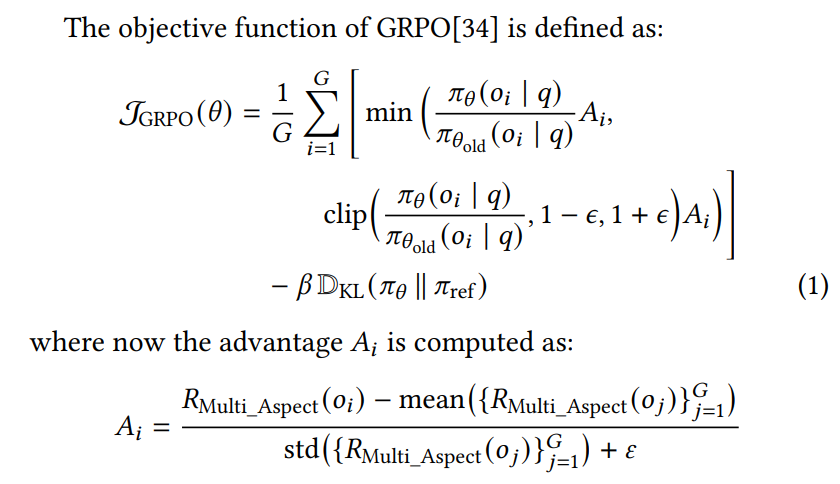
**, where the indicator function is defined as:**



**The final multi-aspect reward is a weighted combination of these four components:**



**where 𝛼, 𝛽, 𝛾, and 𝛿 are non-negative coefficients** that control the relative importance of each reward component.



The GRPO objective evaluates how well the current policy generates outputs compared to the previous policy. Probabilities under the current and previous policies are compared for importance sampling, while a clipping mechanism and KL-divergence penalty ensure stable updates and prevent excessive deviation from a reference policy. Each candidate’s advantage is computed relative to others in the same group using multi-aspect rewards, promoting the selection of higher-quality outputs without a separate value function. By integrating rewards for language consistency, text segmentation, object count, answer correctness, and output format, GRPO provides rich, fine-grained supervision for end-to-end multilingual multimodal reasoning.

**Implementation details**

**During the Supervised Fine-Tuning (SFT) stage,** the model was trained for **1–3 epochs** with a global **batch size of 256** on **8 NVIDIA A100 (40GB) GPUs**. **The image resolution size is set to be 896\*896.** **The learning rate was fixed at 2×10−4** **and kept constant without decay, using the fused AdamW optimizer.** **Mixed-precision training with bfloat16 (bf16) and TensorFloat-32 (tf32) was enabled** to improve computational efficiency while maintaining numerical stability. **Gradient clipping with a maximum norm of 0.3 and a warm-up ratio of 0.03 were applied.** Parameter-efficient fine-tuning was implemented via **LoRA**[16], **targeting the transformer modules q\_proj, k\_proj, v\_proj, o\_proj, gate\_proj, up\_proj, and down\_proj, with rank 32, alpha 64, and a dropout rate of 0.05. In the subsequent GRPO training stage, all reward components were uniformly scaled by 0.25 so that the total reward summed to 1, i.e.,**



**The number of generations per iteration was set to 4**, while other hyperparameters were kept consistent with the SFT stage.

**Evaluation Benchmark**

For our experiments, MMMB[48], Multilingual MMBench[48], and MTVQA[51] serve as the primary evaluation benchmarks.

机器翻译的Metric: BLEU