Steer Where It Matters: Token-Level Visual-Sensitivity Steering for LVLMs Hallucination Mitigation

Ruipeng Zhang[1,3], Zhihao Li[1,2,3], C.L. Philip Chen[1,2,3], Tong Zhang[1,2,3*]

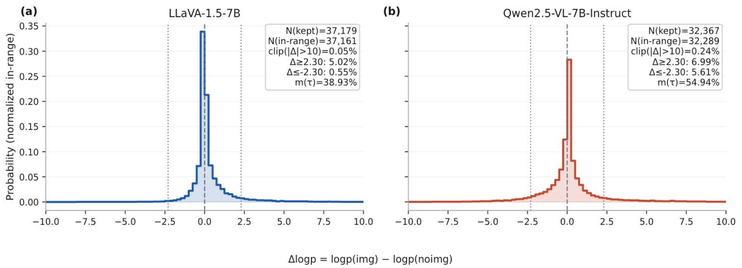
[1]Guangdong Provincial Key Laboratory of Computational AI Models and Cognitive Intelligence, School of Computer Science & Engineering, South China University of Technology

[2]Pazhou Lab, Guangzhou, China

[2]Engineering Research Center of the Ministry of Education on Health Intelligent Perception and Paralleled Digital-Human, Guangzhou, China

ICML International Conference On Machine Learning

Motivation



Our motivation comes from three observations.

- First, visual information only affects a small number of tokens during generation.
- Second, traditional activation steering averages visual signals across all tokens, which weakens the important image-related information.
- Third, applying the same steering signal to every token may change many tokens that do not need adjustment, so a token-level adaptive method is needed.

Problem discussion

Observation 1

We design a token-level diagnosis using teacher forcing with and without the image. For each token, we compute

$$\Delta_t = \log p(y_t \mid x, I, y_{<t}) - \log p(y_t \mid x, \varnothing, y_{<t}).$$

We find that most tokens have (Δt ≈0), while only a few tokens are strongly affected by the image.

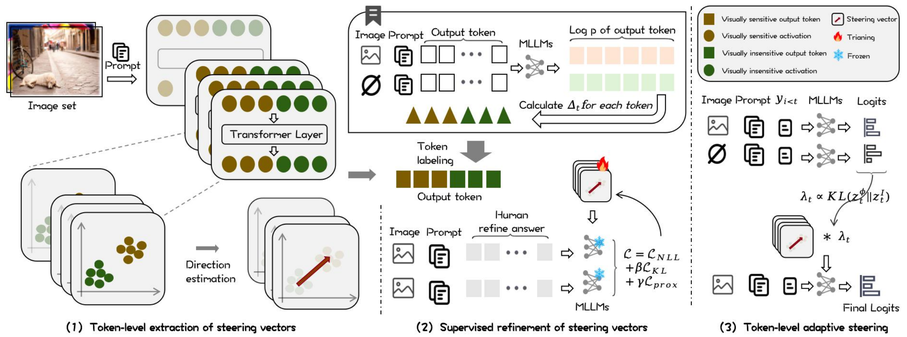
Observation 2

We further test how global steering affects each token using teacher forcing. For each token, we measure

$$\Delta\mathrm{NLL}_t(\text{method} - \text{vanilla}) = \mathrm{NLL}_t^{\text{method}} - \mathrm{NLL}_t^{\text{vanilla}}$$

We find that global steering changes many non-critical tokens, while token-aware steering mainly affects visually important tokens. This shows that global steering misallocates the intervention budget and motivates adaptive token-level steering.

Method



Step 1: Token-level extraction of steering vectors.

We compare token probabilities with and without the image:

$$\Delta_t = \log p_\theta(\hat{y}_t \mid x, I, \hat{y}_{<t}) - \log p_\theta(\hat{y}_t \mid x, \varnothing, \hat{y}_{<t}). \qquad \mathcal{T}^+ = \mathrm{Top}_\rho(\{|\Delta_t|\}_{t\in\mathcal{T}}), \quad \mathcal{T}^- = \mathrm{Bottom}_\rho(\{|\Delta_t|\}_{t\in\mathcal{T}}). \qquad d^{(\ell)} = \frac{1}{|\mathcal{T}^+|}\sum_{t\in\mathcal{T}^+}\tilde{h}_t^{(\ell)} - \frac{1}{|\mathcal{T}^-|}\sum_{t\in\mathcal{T}^-}\tilde{h}_t^{(\ell)}.$$

Tokens with large (| Δt |) are treated as visually sensitive tokens (T+), while tokens with small (| Δt |) are treated as visually insensitive tokens (T–). For each layer, we contrast their normalized hidden states to obtain an initial steering direction:

$$d^{(\ell)} = \frac{1}{|\mathcal{T}^+|}\sum_{t\in\mathcal{T}^+}\tilde{h}_t^{(\ell)} - \frac{1}{|\mathcal{T}^-|}\sum_{t\in\mathcal{T}^-}\tilde{h}_t^{(\ell)}. \qquad W_{\text{init}}^{(\ell)} = \mathrm{PCA}_1\left(\{d_i^{(\ell)}\}_{i=1}^N\right), \qquad W_{\text{init}}^{(\ell)} \leftarrow \frac{W_{\text{init}}^{(\ell)}}{\|W_{\text{init}}^{(\ell)}\|_2}.$$

Step 2: Supervised refinement of steering vectors

We then keep the LVLM frozen and only optimize the steering vectors (W(ℓ)) using correction data:

$$\mathcal{L} = \mathcal{L}_{\text{NLL}} + \beta\,\mathcal{L}_{\text{KL}} + \gamma\,\mathcal{L}_{\text{prox}}. \qquad \mathcal{L}_{\text{NLL}} = -\frac{1}{|\mathcal{A}|}\sum_{t\in\mathcal{A}}\log p_{\theta,W}(y_t \mid x, I, y_{<t}).$$

$$\mathcal{L}_{\text{KL}} = \frac{1}{|\mathcal{A}|}\sum_{t\in\mathcal{A}}\mathrm{KL}(p_{\theta,W}(\cdot \mid x, I, y_{<t}) \,\|\, p_\theta(\cdot \mid x, I, y_{<t})). \qquad \mathcal{L}_{\text{prox}} = \frac{1}{|\mathcal{S}|}\sum_{\ell\in\mathcal{S}}\left\|W^{(\ell)} - W_{\text{init}}^{(\ell)}\right\|_2^2.$$

This lightweight refinement improves the steering direction without updating the model parameters.

Step 3: Adapt steering during decoding.

At each decoding step, we measure visual sensitivity by comparing image-conditioned and image-ablated logits:

$$\mathrm{VS}_t = \mathrm{KL}\left(\mathrm{softmax}\left(\frac{z_t^I}{T_{KL}}\right) \,\middle\|\, \mathrm{softmax}\left(\frac{z_t^\varnothing}{T_{KL}}\right)\right), \qquad \bar{\mathrm{VS}}_t = \frac{\mathrm{VS}_t - \mu}{\sigma + \epsilon}$$

We then map (VS_t) to a token-specific steering strength (\lambda_t), and apply

$$\tilde{\lambda}_t = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min})\, g_t. \qquad \hat{\lambda}_t = \alpha\,\hat{\lambda}_{t-1} + (1-\alpha)\,\tilde{\lambda}_t. \qquad \lambda_t = \min(\hat{\lambda}_t, \lambda_t^{\text{cap}}).$$

Thus, TLVS applies stronger steering to visually important tokens and weaker steering to non-critical tokens.

Partial Experimental Results

Table 1. Main results. TLVS consistently improves hallucination-related metrics across POPE, HallusionBench, CHAIR (COCO), and AMBER on both LLaVA-1.5-7B and Qwen2.5-VL-7B. ↑/↓ indicate that higher/lower is better; **bold**/underline mark the best/second-best within each backbone; gray highlights TLVS, with ↑ / ↓ showing absolute gains over the vanilla baseline.

Model	POPE				HallusionBench			CHAIR (COCO)			AMBER						
	Adv.↑	Pop.↑	Rand.↑	Avg.↑	qAcc↑	fAcc↑	aAcc↑	CHAIRs↓	CHAIRi↓	Recall↑	CHAIR↓	Cover↑	Hal↓	Cog↓	F1-E↑	F1-A↑	F1-R↑	F1↑
LLaVA-1.5-7B	81.80	84.36	89.12	85.09	13.19	16.18	45.70	46.0	12.8	78.9	12.0	50.3	36.4	4.6	83.2	65.6	62.4	74.7
+ VCD	81.33	85.06	87.16	84.52	12.75	19.08	46.41	47.8	14.1	**82.7**	10.1	51.2	43.6	4.3	84.1	63.6	66.4	74.8
+ SPARC	81.03	86.05	89.00	85.36	13.19	17.34	35.25	43.4	16.8	61.1	14.5	44.6	23.1	2.4	88.5	66.7	65.5	77.9
+ ASD	—	—	—	87.87	—	—	—	40.0	11.3	82.0	—	—	—	—	—	—	—	—
+ SHARP	79.00	83.20	85.30	82.50	—	—	—	34.8	10.6	59.7	8.5	**52.1**	39.2	4.8	—	—	—	—
+ VTI	80.43	85.53	88.50	84.82	13.19	15.90	44.11	35.8	11.1	76.8	4.7	47.2	27.0	1.8	85.8	67.5	68.3	77.1
+ VISTA	80.98	88.87	89.25	86.37	14.29	19.36	45.82	44.6	13.4	77.0	6.0	39.9	27.1	2.4	87.6	67.3	55.9	77.6
+ DPO (RLHF-V)	84.03	85.94	89.12	86.36	**16.70**	**20.81**	**51.31**	43.2	11.9	76.4	5.7	49.7	27.3	2.6	90.7	72.6	64.6	80.9
+ TLVS (ours)	**87.85** ↑6.05	**89.42** ↑5.06	**92.08** ↑2.96	**89.78** ↑4.69	15.85 ↑2.66	19.95 ↑3.77	46.50 ↑0.80	**31.2** ↓14.8	**9.7** ↓4.1	82.2 ↑5.3	**3.8** ↓8.2	51.8 ↑1.5	**19.7** ↓16.7	**1.5** ↓3.1	**91.5** ↑5.3	**73.0** ↑7.4	**69.7** ↑7.3	**82.2** ↑7.5
Qwen2.5-VL-7B	84.92	86.78	89.01	86.90	23.74	31.21	55.54	24.0	7.0	**86.7**	5.4	**51.6**	29.0	1.9	97.3	**83.8**	72.6	89.4
+ VCD	83.92	85.80	88.41	86.04	31.43	35.26	61.82	29.8	9.7	58.0	6.2	50.3	36.5	2.1	91.6	78.6	68.0	83.8
+ DPO (RLHF-V)	84.99	86.81	89.72	87.17	**34.95**	**39.60**	62.62	13.4	**3.5**	62.7	**3.1**	50.7	21.9	1.1	**98.5**	82.5	65.9	88.3
+ TLVS (ours)	**85.10** ↑0.18	**87.01** ↑0.23	**89.88** ↑0.87	**87.33** ↑0.43	33.08 ↑9.34	37.19 ↑5.98	**63.23** ↑7.69	**12.0** ↓12.0	5.2 ↓1.8	83.9 ↓2.8	**3.8** ↓2.6	51.2 ↓0.4	**16.7** ↓12.3	**1.0** ↓0.5	97.3	82.7 ↓1.1	**73.0** ↑0.6	**89.8** ↑1.4

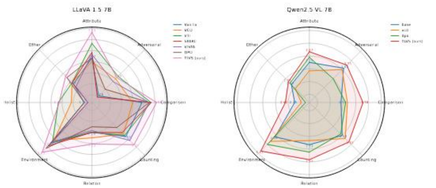


Figure 7. MMHal-Bench performance across question categories (attributes, adversarial objects, comparisons, counting, spatial, environmental, holistic, and others).

Table 2. Ablation on refinement and token-level adaptive steering. ↑ / ↓ indicate higher/lower is better; green/red arrows show absolute gains over the vanilla baseline within each backbone.

Method	Components		POPE	AMBER			
	Refine	Adapt.	Avg↑	CHAIR↓	Cover↑	Hal↓	F1↑
LLaVA-1.5-7B							
Vanilla	–	–	85.09	12.0	50.3	36.4	74.7
TLVS-Init	×	×	86.83 ↑1.74	7.4 ↓4.6	45.4 ↓4.9	35.9 ↓0.5	75.0 ↑0.3
TLVS-Refine	✓	×	88.71 ↑3.62	4.1 ↓7.9	44.5 ↓5.8	28.1 ↓8.3	80.8 ↑6.1
TLVS	✓	✓	89.78 ↑4.69	3.8 ↓8.2	51.8 ↑1.5	19.7 ↓16.7	82.2 ↑7.5
Qwen2.5-VL-7B							
Vanilla	–	–	86.90	5.4	51.6	29.0	89.4
TLVS-Init	×	×	86.83 ↓0.07	5.0 ↓0.4	50.8 ↓0.8	26.8 ↓2.2	87.6 ↓1.8
TLVS-Refine	✓	×	87.00 ↑0.10	4.2 ↓1.2	48.9 ↓2.7	20.0 ↓9.0	88.8 ↓0.6
TLVS	✓	✓	87.33 ↑0.43	3.8 ↓1.6	51.2 ↓0.4	16.7 ↓12.3	89.8 ↑0.4

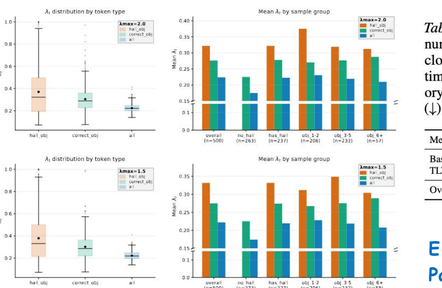


Figure 8. Adaptive steering focuses on critical tokens. λ_t distribution (left) and mean λ_t by token group (right).

Table 3. Efficiency and Overhead. (i) Fwd/Token, the average number of forward passes per generated token; (ii) ms/Token, wall-clock latency per token; (iii) Sec/Sample, end-to-end generation time per sample; and (iv) Peak VRAM, the maximum GPU memory footprint during inference. Lower is better for all four metrics (↓).

Method	Fwd/Token ↓	ms/Token ↓	Sec/Sample ↓	Peak VRAM (GB) ↓
Base	1.00	31.25	3.34	13.95
TLVS	2.00	64.01	6.79	14.03
Overhead (Ours/Base)	2.00×	2.05×	2.04×	1.01×

E-mail:202421044797@mail.scut.edu.cn

Paper:https://arxiv.org/abs/2606.07647

https://zrpchuang.github.io/

Lookingfor PhDopportunities for Fall 2027!