Can Vision-Language Models See the Future for Robots? Coverage-Aware Human Motion Forecasting with Semantic Intent Residuals

Mujtaba Hasan
AATUN Labs
New Delhi, India
Email: mujtaba@aatun.in

Abstract*—Robots that move around people need forecasts that are valid, metric, and available whenever the robot must plan. Vision-language models (VLMs) are attractive for this setting because they can describe motion and infer intent without trajectory-specific training. We test whether this semantic ability is enough for a robot-facing prediction task: given 8 observed pedestrian positions, can an off-the-shelf VLM output the next 12 metric positions in a form a planner could use? We evaluate GPT-4o, GPT-4o-mini, Gemini 2.5 Flash, and LLaVA-1.5-7B on ETH/UCY with text prompts, rendered bird's-eye-view inputs, multiple prompting strategies, and a strict trajectory parser. The main result is that VLM forecasting is a coverage-aware structured-output problem. Under full-coverage zero-shot text evaluation, GPT-4o-mini obtains ADE 0.548 and GPT-4o obtains 0.561, both statistically indistinguishable from constant velocity at 0.554. Gemini appears much stronger if we score only parsed outputs (ADE 0.127), but it parses only 33/250 samples; constant velocity obtains ADE 0.125 on those same samples, while the failed samples are nearly $5\times$ harder. We formalize this as coverage-conditioned selection bias and recommend coverage, matched-sample baselines, and fallback-aware risk for structured VLM modules in robotics. We also introduce Reliability-Gated Intent-Guided Residual Forecasting (RIGF), a diagnostic hybrid that keeps a safe base forecast and lets semantic intent modify it only through validation-gated residual experts. RIGF-Oracle improves ADE by 9.2% overall and 13.3% on high-nonlinearity trajectories, showing that correct intent is useful. However, GPT-4o-mini intent labels are 100% parseable but poorly calibrated (31.6% proxy intent accuracy), so direct VLM intent degrades forecasting. The takeaway is simple: current VLMs should not be used as direct metric forecasters for robots, but they can be studied as semantic sensors inside prediction stacks that audit, calibrate, and safely gate their outputs.

I. Introduction

A robot that shares space with people must predict human motion before it acts. A mobile robot approaching a doorway, an autonomous vehicle near a crosswalk, or a service robot navigating through a crowd needs a forecast for collision checking, speed selection, and replanning. For human-robot interaction and social navigation, this forecast cannot be just a plausible story about what might happen. It must be a valid trajectory in the robot's coordinate frame, at the required horizon, and it must be available on every control cycle [9, 1, 8, 13].

Robot foundation models and vision-language models (VLMs) are promising because they connect perception, language, spatial relations, and high-level goals [12, 4, 10, 2, 19]. They can often say that a person is slowing down, turning, crossing, or waiting. This raises a concrete robotics question: can an off-the-shelf VLM turn that semantic understanding into a metric future trajectory that a planner can consume? If the answer is yes, VLMs could provide data-efficient forecasting for robots. If the answer is no, the failure mode still matters: a robot designer needs to know whether the model is inaccurate, invalid, poorly calibrated, or simply silent on hard cases.

A simple service-robot example makes the issue concrete. Suppose a robot approaches a hallway intersection and tracks a pedestrian for eight frames. The planner needs a short future path to decide whether to slow down, pass, or wait. If a VLM returns a malformed list, the planner cannot score it. If the VLM returns coordinates only when the pedestrian keeps walking straight, the system falls back exactly when prediction is most valuable. If the VLM confidently says "decelerating" when the person is actually turning, an intent-conditioned controller may choose the wrong residual correction. These are not small benchmarking details; they are robot-interface failures.

We study this question using pedestrian trajectory forecasting. Given 8 observed positions in the ETH/UCY setting, we prompt a VLM to return the next 12 positions. The task looks simple, but it tests several robot-relevant requirements at once: the model must keep the coordinate frame, infer local velocity and curvature, preserve scale, and output exactly 12 two-dimensional coordinates. A missing bracket or an extra coordinate is not a harmless formatting error. For a robot, it means the forecasting module failed and a fallback must be used.

Our first finding is that VLM forecasting should be evaluated as *selective structured prediction*. Many evaluations report error only on outputs that can be parsed. That is unsafe for robot modules. A model can look accurate by producing valid outputs

only on easy states and failing on harder ones. We observe this for Gemini 2.5 Flash: it obtains ADE 0.127 on parsed zero-shot text outputs, but parses only 13.2% of samples. On those same parsed samples, constant velocity obtains ADE 0.125. On the failed samples, constant velocity ADE is 0.619. The low parsed ADE is therefore not a robot-usable improvement; it is a selection effect.

We therefore report four quantities together: parse coverage, parsed ADE/FDE, the fallback baseline on the same parsed samples, and the fallback difficulty of failed samples. This simple protocol changes the interpretation of the results. GPT-4o and GPT-4o-mini parse all zero-shot text samples, but their errors are statistically indistinguishable from constant velocity. Gemini looks strong only because it parses an easy subset. LLaVA parses more often than Gemini but has poor metric grounding. Chain-of-thought prompting, rather than helping, often harms coordinate consistency.

The second part of the paper asks a more constructive question. Even if VLMs are weak direct coordinate generators, can they still help as semantic intent sensors? We introduce Reliability-Gated Intent-Guided Residual Forecasting (RIGF) to test this cleanly. RIGF always keeps a simple geometric forecast available. A semantic intent label can select a residual correction only if that expert improved over the base forecast on held-out validation data. This makes RIGF a diagnostic tool: it tells us whether correct intent would help, and whether the VLM's intent labels are good enough to use.

The answer is mixed but informative. With future-oracle intent, RIGF improves ADE by 9.2% overall and 13.3% on high-nonlinearity trajectories. This shows that semantic intent can be useful for metric forecasting. With GPT-4o-mini intent, however, performance degrades because the labels are biased: the model predicts "decelerating" for 126/250 samples, while the oracle proxy labels only 9/250 as decelerating. A conservative gated variant avoids most harm by falling back to constant velocity, but it does not reliably improve.

Our contributions are:

- a coverage-aware evaluation of off-the-shelf VLMs as robot-facing human-motion forecasters;
- a simple theory and reporting recipe for parse-conditioned selection bias;
- RIGF, a validation-gated semantic residual diagnostic with safe fallback;
- an empirical decomposition showing that correct intent helps, but current VLM intent is not calibrated enough for metric robot prediction.

The main message is not that VLMs are useless for robots. It is that they should not be trusted as direct coordinate forecasters without coverage reporting, matched baselines, calibration, and fallback.

II. Related Work

A. Human motion forecasting for robots

Trajectory forecasting is a core component of autonomous driving, social navigation, and human-robot interaction. Classical social-force models describe attraction, repulsion, and goal-directed motion [9]. Learning-based work includes Social-LSTM [1], Social-GAN [8], graph and transformer predictors, endpoint-conditioned models, and diffusion-style forecasters [13, 5, 18, 11, 7]. These methods are trained for trajectory prediction. Our question is different: can a general VLM, used only through prompts, produce metric forecasts that satisfy a robot prediction contract?

B. Robot foundation models and semantic reasoning

Semantic robotics uses high-level concepts such as goals, affordances, relations, and intent to guide robot behavior. Multimodal foundation models such as GPT-4o, Gemini, and LLaVA provide broad visual-language reasoning abilities [12, 4, 10, 6]. Embodied systems such as PaLM-E and RT-2 use large multimodal models for instruction following and robot-action interfaces [2, 19]. Autonomous-driving work has also explored language models for scene explanation and closed-loop decision support [16, 17, 15]. These systems motivate VLMs as semantic modules, but they do not show that off-the-shelf VLMs can output calibrated metric trajectories.

C. Motion as language

Trajectory-as-language models such as MotionLM tokenize motion and train motion-specific models [14]. They show that language-model architectures can represent trajectories when trained on trajectory data. Our setting is stricter. We ask a general VLM to emit future coordinates from prompts alone. This distinction matters for robotics: a model may correctly state that a pedestrian is turning left, but still fail to output a planner-ready metric trajectory.

D. Prompting and structured numerical outputs

Few-shot prompting, chain-of-thought reasoning, and role prompts can improve many language tasks. Metric forecasting is different because the output is a fixed-length numerical sequence. Extra reasoning text can introduce rounded velocities, inconsistent signs, or invalid final arrays. We therefore evaluate prompting strategies by both coverage and accuracy, rather than assuming that more reasoning text is better.

E. Selective outputs in robot evaluation

Many VLM benchmarks discard malformed outputs. For robot systems, this hides an important failure mode: invalid outputs require fallback, repair, or replanning. The issue is related to selective prediction and shortcut learning [3], but here it appears directly in benchmarking. We ask not only whether a VLM can solve some cases, but which cases remain after parsing and whether the VLM improves over a robot fallback on those same cases.

III. Protocol and Theory

A. Prediction contract

A robot observes a human trajectory $X = (x_1, \ldots, x_T)$, $x_t \in \mathbb{R}^2$, and needs a future trajectory $\hat{Y} = (\hat{y}_1, \ldots, \hat{y}_H)$. We use the standard ETH/UCY setup with $T = 8$ observed steps

TABLE I
REPORTING RECIPE FOR ROBOT-FACING STRUCTURED FORECASTS. PARSED-ONLY ADE IS NOT ENOUGH WHEN COVERAGE IS BELOW 100%.

Quantity	Purpose
Coverage $\Pr(S=1)$	How often the robot receives a valid forecast
Parsed ADE/FDE	Accuracy on valid outputs
Fallback on same samples	Checks whether the VLM beats the robot fallback on the same states
Fallback on failed samples	Shows whether failures occur on harder states
Paired intervals	Tests whether small ADE gaps are meaningful
Operational risk	Scores the VLM plus fallback system

and $H = 12$ future steps at 2.5 FPS [1, 8]. We report average displacement error and final displacement error:

$$\mathrm{ADE}(Y, \hat{Y}) = \frac{1}{H} \sum_{h=1}^{H} \|\hat{y}_h - y_h\|_2, \tag{1}$$

$$\mathrm{FDE}(Y, \hat{Y}) = \|\hat{y}_H - y_H\|_2. \tag{2}$$

A VLM output is valid only if it parses to exactly 12×2 finite coordinates. Invalid outputs remain in the denominator when we report coverage. This matches the robotics use case: if the VLM fails, the robot still needs a fallback forecast.

B. Inputs, prompts, and parser

The text-only input lists the observed (x, y) coordinates and asks for 12 future coordinates in a JSON-like array. The image+text input also includes a rendered bird's-eye-view plot of the target track and nearby observed tracks. We evaluate zero-shot, few-shot, chain-of-thought (CoT), and multi-agent-aware prompts. Few-shot examples are drawn only from training windows. For image+text prompts, the coordinate text is still included, so the image condition tests whether the rendering adds useful information.

The parser is deliberately strict. It looks for the final answer region, extracts the last valid coordinate array, checks shape $(12, 2)$, and rejects non-finite values. We do not repair malformed outputs. A repair layer might be useful in a deployed system, but here we want to measure whether the VLM itself satisfies the structured-output contract.

C. Why parsed-only ADE can be biased

Let $S \in \{0, 1\}$ indicate whether the VLM output parses, and let $c = \Pr(S = 1)$ be coverage. Let L be the loss the VLM would obtain when it produces a valid forecast. Parsed-only evaluation estimates $\mathbb{E}[L \mid S = 1]$, not risk on the full input distribution.

Proposition 1. Let $\mu_1 = \mathbb{E}[L \mid S = 1]$ and $\mu_0 = \mathbb{E}[L \mid S = 0]$. The full-distribution risk is $R = c\mu_1 + (1 - c)\mu_0$, and parsed-only reporting has bias

$$\mathbb{E}[L \mid S = 1] - R = (1 - c)(\mu_1 - \mu_0). \tag{3}$$

If failed cases are harder than parsed cases, parsed-only ADE is optimistic.

In practice, the VLM loss is not observed on failed cases. We therefore use a deterministic fallback, constant velocity, as a difficulty proxy. The fallback can be evaluated on every sample.

Proposition 2. Let B be a deterministic fallback. For a VLM with parse indicator S, the conditional difference

$$\Delta_S = \mathbb{E}[L_{\mathrm{VLM}} - L_B \mid S = 1] \tag{4}$$

compares the VLM and fallback on the same parsed distribution. A low parsed VLM loss is evidence of improvement over B only if $\Delta_S < 0$. This is the matched-sample rule we use throughout the paper.

D. Fallback-aware risk

A robot system usually combines a learned module with a fallback. If invalid VLM outputs revert to fallback B, the operational risk is

$$R_{\mathrm{op}} = c\,\mathbb{E}[L_{\mathrm{VLM}} \mid S = 1] + (1 - c)\,\mathbb{E}[L_B \mid S = 0]. \tag{5}$$

Parsed ADE ignores the second term. This is why a low-coverage model can look strong offline while adding little value to a robot. If the failures occur on easy states, a planner may tolerate them. If they occur on hard states, then the learned module has precisely avoided the states where semantic reasoning was supposed to help. We therefore treat the fallback not as an afterthought but as part of the evaluated robot system.

E. Validation-gated semantic residuals

RIGF keeps constant velocity as the guaranteed output path and uses semantic intent only as a gated correction. For intent z, observed features $\phi(X)$, and residual expert f_z, the prediction is

$$\hat{Y}_z(X) = \hat{Y}_{\mathrm{CV}}(X) + \gamma_z f_z(\phi(X)). \tag{6}$$

The shrinkage γ_z is chosen on validation data from a finite grid that includes zero. If no value improves validation ADE over zero, that expert is disabled and RIGF returns constant velocity for that intent.

Proposition 3. For a fixed intent z, assume ADE is bounded in $[0, B]$ and there are m_z validation samples. Let $\hat{\gamma}_z$ be the validation-minimizing shrinkage over $\Gamma \cup \{0\}$. If the expert is enabled only when its empirical improvement over zero is at least $\tau_z \geq 0$, then with probability at least $1 - \delta$,

$$R_z(\hat{\gamma}_z) \leq R_z(0) - \tau_z + 2B\sqrt{\frac{\log(2|\Gamma|/\delta)}{2m_z}}. \tag{7}$$

If the expert is disabled, RIGF exactly equals the fallback for that intent. The proof is a direct Hoeffding union bound over the finite grid. The proposition does not assume the test-time intent is correct; it only says that, for a fixed intent group, the residual is validation-gated.

If the test-time intent estimate $\tilde{z}$ differs from the correct intent $z^\star$, a useful expert can become harmful. This is the key diagnostic role of RIGF: it separates the value of correct intent from the quality of the VLM's intent labels.

TABLE II
ZERO-SHOT TEXT-ONLY EVALUATION. ADE/FDE ARE OVER PARSED OUTPUTS. FOR LOW-COVERAGE VLMS, "CV SAME" REPORTS CONSTANT VELOCITY ON EXACTLY THE PARSED SUBSET.

Method	Cov.	ADE	FDE	CV same
Constant velocity	100%	0.554	1.177	–
Linear regression	100%	0.639	1.286	–
GPT-4o-mini	100%	0.548	1.152	0.554
GPT-4o	100%	0.561	1.188	0.554
Gemini 2.5 Flash	13.2%	0.127	0.228	0.125
LLaVA-1.5-7B	34.4%	2.499	3.120	0.691

IV. ROBOT-CENTERED EXPERIMENTS

We treat each ETH/UCY window as one robot prediction query. The robot observes a human track and asks for a future trajectory. We ask three questions: Does the VLM return a valid trajectory? If it does, does it beat the fallback on the same states? Do prompts or images improve reliability?

A. Zero-shot text forecasting

Table II is the main full-coverage result. GPT-4o-mini and GPT-4o parse all 250 zero-shot text samples. Their ADEs, 0.548 and 0.561, are statistically indistinguishable from constant velocity at 0.554. The paired bootstrap interval for GPT-4o-mini minus CV is $[-0.034, 0.021]$ ADE; for GPT-4o minus CV it is $[-0.021, 0.043]$. Thus, in the simplest reliable setting, the VLMs mostly match the robot fallback rather than improve on it.

B. Selection bias in low-coverage models

Gemini is the clearest example of why coverage matters. It parses only 33/250 predictions. On those 33 parsed samples, Gemini ADE is 0.127, but constant velocity ADE is 0.125 on the same samples. On the samples Gemini fails to parse, constant velocity ADE increases to 0.619. The parser has selected an easy subset. A table that reports only Gemini's parsed ADE would therefore give the wrong robotics conclusion. The robot still needs to handle the harder failed states.

LLaVA shows a different failure mode. It parses 86/250 samples, but its parsed ADE is 2.499, far worse than constant velocity on the same samples. Thus a structured VLM forecaster can fail because it is selective, because its metric grounding is poor, or both.

C. Four failure modes for robot use

The results separate failure modes that are often collapsed into one score. Table III summarizes them in robot terms. Different failures require different fixes: Gemini needs coverage or fallback-aware abstention handling; LLaVA needs metric grounding; CoT needs a shorter schema-constrained interface; and RIGF-VLM needs calibrated intent rather than just valid JSON. A robot paper should not report only the first valid-output metric that looks good, because that hides which component must be repaired.

D. Prompting and visual inputs

GPT-4o was also evaluated across prompting strategies and modalities (Table V). Few-shot prompting helps on parsed subsets. Few-shot image+text reaches ADE 0.487 at 75.6% coverage, compared with matched-sample CV ADE 0.600. Few-shot text reaches ADE 0.509 at 80.0% coverage, compared with CV ADE 0.595. This is encouraging, but it is not a full-coverage improvement: roughly one-fifth of the states still require fallback or repair.

CoT is consistently harmful. For GPT-4o, CoT text ADE rises to 0.817 at 57.2% coverage, and CoT image+text ADE is 0.784 at 62.0% coverage. GPT-4o-mini shows the same trend. A likely reason is that CoT introduces intermediate velocities, rounded speeds, and alternative hypotheses that are not converted cleanly into the final coordinate list. For robot metric outputs, a short schema-constrained answer is safer than a long verbal chain.

The image+text condition uses rendered bird's-eye-view plots. It does not test raw camera frames, semantic maps, sidewalks, goals, or object-level scene context. The negative visual result should be read narrowly: plot-style images do not reliably help off-the-shelf VLMs output metric coordinates. A robot with richer scene context may still benefit from visual-language semantics if those signals are passed through calibrated motion modules.

From a system perspective, the direct coordinate interface is therefore not the most promising one. The safer interface is a narrow semantic channel: ask the VLM for evidence about intent, goal, crowding, or occlusion; check that evidence against observed kinematics; and let a motion module produce the metric trajectory. Section V implements this idea in the smallest possible form. Figure 1 summarizes these coverage-aware findings, highlighting the gap between raw parsed accuracy and matched-sample performance.

E. Qualitative behavior

Figure 2 shows representative GPT-4o-mini wins and failures against constant velocity. Some wins capture curvature or non-constant motion. Failures include scale drift, over-extrapolation, and implausible shape. This supports the hybrid design in Section V: semantic information may help on some states, but a robot needs a safe metric fallback when the semantic signal is unreliable.

V. RIGF: SEMANTIC INTENT WITH SAFE FALLBACK

The benchmark above tests VLMs as direct coordinate generators. RIGF tests a different use case: VLMs as semantic intent sensors inside a robot prediction stack. The design is deliberately simple. A constant-velocity forecast is always available. A semantic intent label can modify it only through a residual expert that has already improved over the base forecast on validation data.

The intent set is

$$\mathcal{Z} = \{\text{stationary}, \text{linear}, \text{accelerating}, \text{decelerating}, \text{left}, \text{right}\}. \tag{8}$$

TABLE III

OBSERVED FAILURE MODES AND THE CORRESPONDING ROBOT CONSEQUENCE. THE SAME ADE TABLE CAN HIDE VERY DIFFERENT DESIGN PROBLEMS; SEPARATING THE FAILURE MODE MAKES THE RESULT MORE ACTIONABLE FOR ROBOT SYSTEMS.

Failure mode	Evidence in our experiments	Robot consequence
Selective parsing	Gemini parses 13.2%; failed states are $4.95\times$ harder by the CV fallback.	The robot falls back exactly on the states where prediction is hardest and semantic reasoning would be most valuable.
Metric misgrounding	LLaVA parses 34.4% but ADE is 2.499, far worse than CV on the same parsed states.	A valid coordinate array is not necessarily a useful motion forecast; metric grounding must be scored separately from format validity.
Reasoning drift	CoT lowers coverage and raises ADE for GPT-4o and GPT-4o-mini.	Verbal reasoning can corrupt a fixed metric output; schema-constrained answers are safer for planner-facing quantities.
Intent miscalibration	GPT-4o-mini overpredicts deceleration in RIGF despite 100% valid intent JSON.	Semantic labels must be calibrated and gated before they are allowed to control residual corrections.

TABLE IV

SELECTION-BIAS DIAGNOSTICS FOR ZERO-SHOT TEXT. "CV PARSED" AND "CV FAILED" EVALUATE THE FALLBACK ON THE PARSED AND FAILED SUBSETS. GEMINI'S APPARENT STRENGTH DISAPPEARS UNDER MATCHED COMPARISON.

Model	Parsed	VLM	CV parsed	CV failed	Ratio
Gemini 2.5	33	0.127	0.125	0.619	4.95
LLaVA-1.5	86	2.499	0.691	0.481	0.70
GPT-4o-mini	250	0.548	0.554	–	–
GPT-4o	250	0.561	0.554	–	–

TABLE V

GPT-4O CONFIGURATION SWEEP. FEW-SHOT HELPS ON PARSED SUBSETS; COT HURTS. "CV SAME" IS CONSTANT VELOCITY ON THE SAME VALID SUBSET.

Prompt	Input	Cov.	ADE	CV same
zero-shot	text	100.0%	0.561	0.554
zero-shot	image+text	80.0%	0.575	0.595
few-shot	text	80.0%	0.509	0.595
few-shot	image+text	75.6%	0.487	0.600
multi-agent	text	71.2%	0.602	0.616
CoT	text	57.2%	0.817	0.590
CoT	image+text	62.0%	0.784	0.625

TABLE VI

RIGF EXPERT VALIDATION. EXPERTS ARE ENABLED ONLY WHEN VALIDATION ADE IMPROVES OVER CV. Δ IS RIGF MINUS CV ON VALIDATION.

Intent	Val n	γ	CV	Δ
stationary	4743	0.00	0.102	0.000
linear	7257	0.80	0.486	-0.022
accelerating	1402	1.05	0.669	-0.081
decelerating	1153	0.00	0.753	0.000
turning-left	993	0.95	1.472	-0.647
turning-right	1106	0.85	1.111	-0.346

TABLE VII

RIGF ALGORITHM. INVALID, UNSUPPORTED, OR DISABLED SEMANTIC SIGNALS RETURN THE BASE FORECAST.

Input: observed trajectory X, intent source $q(z \mid X)$, enabled experts f_z, shrinkages γ_z.
1. Compute base forecast $\hat{Y}_{\mathrm{CV}}(X)$.
2. Obtain intent $\tilde{z}$ and confidence p from a heuristic, VLM, or oracle proxy.
3. If $\tilde{z}$ is invalid, low-confidence, disabled on validation, or rejected by support/agreement gates, return $\hat{Y}_{\mathrm{CV}}$.
4. Otherwise return $\hat{Y}_{\mathrm{CV}} + \gamma_{\tilde{z}} f_{\tilde{z}}(\phi(X))$.

For each intent, we train a ridge residual expert on ETH/UCY training windows. The expert predicts the residual to constant velocity from observed features: recent velocity, mean velocity, speed variation, signed turning, path tortuosity, and simple crowd-distance statistics. These experts are not meant to be state-of-the-art trajectory models. They are controlled instruments for asking whether semantic intent contains useful information beyond the fallback. This choice is intentional: a stronger black-box forecaster might hide whether the semantic variable itself is useful. By using a simple residual module with explicit gates, we can tell whether failure comes from the residual expert, the intent source, or the decision to override the fallback.

A. Training and gates

We extract 83,271 training windows and split them into 66,617 train and 16,654 validation windows. A future-based proxy assigns an intent label to each training window. This proxy is used for training and for the oracle analysis only. Deployable variants use either observed-history intent or VLM intent.

Each expert predicts $Y - \hat{Y}_{\mathrm{CV}}$. A shrinkage coefficient γ_z is selected on validation data. If the best coefficient does not improve over $\gamma_z = 0$, the expert is disabled. Table VI shows the result. Linear, accelerating, and turning experts improve on validation. Stationary and decelerating experts are disabled because the residual did not help.

B. Intent sources

We evaluate four variants. **RIGF-Heuristic** labels intent from observed kinematics. **RIGF-VLM** asks GPT-4o-mini for a JSON intent label and confidence. **RIGF-VLM-Gated** applies the VLM residual only when the VLM label also passes observed-support and heuristic-agreement checks; otherwise it falls back to CV. **RIGF-Oracle** uses the future-based proxy

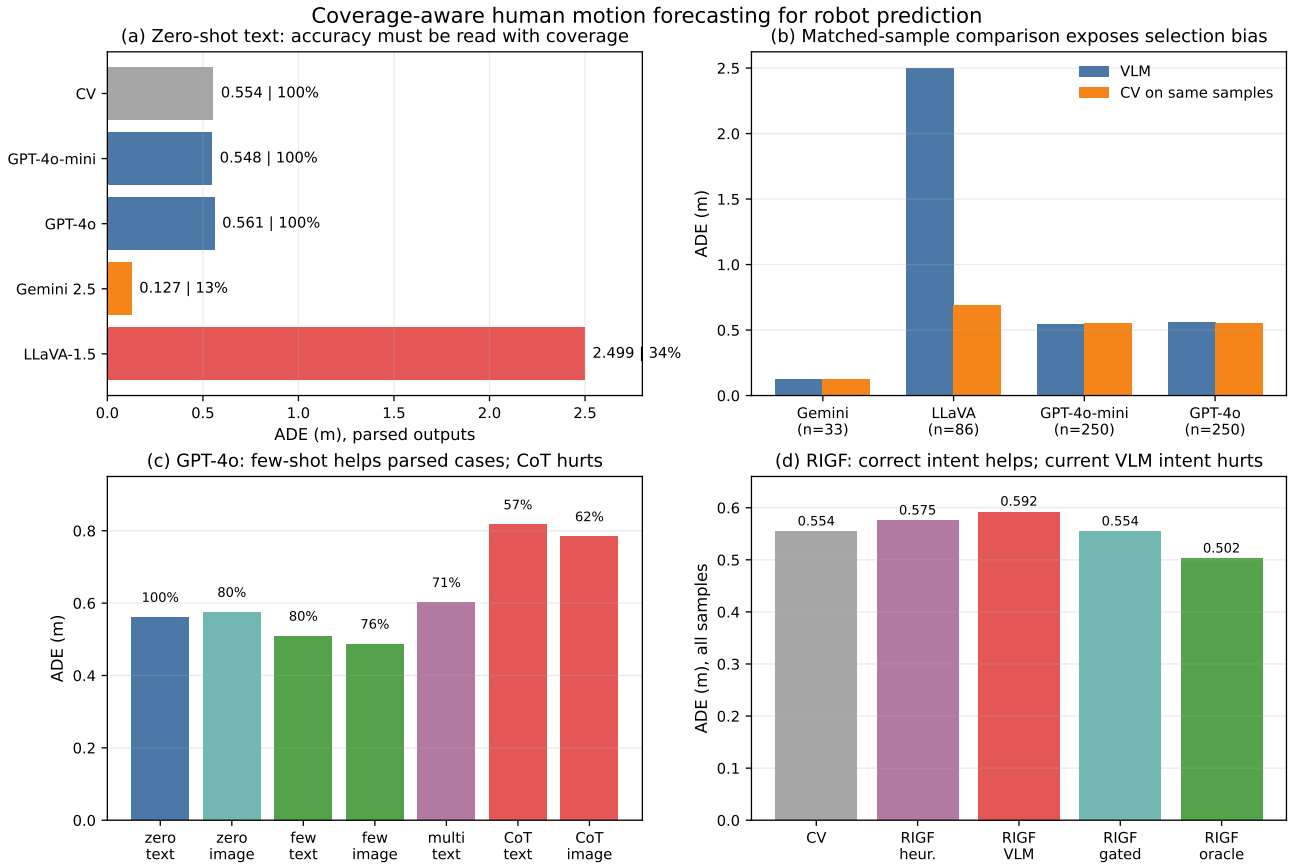

Fig. 1. Main results in one view. (a) Parsed ADE must be read with coverage. (b) Matched-sample comparison shows that Gemini's low parsed ADE is not a gain over the fallback. (c) GPT-4o few-shot improves parsed subsets, while CoT hurts. Percent labels show parse coverage. (d) RIGF shows an oracle intent gap: correct intent helps, but current VLM intent hurts unless gated back to the fallback.

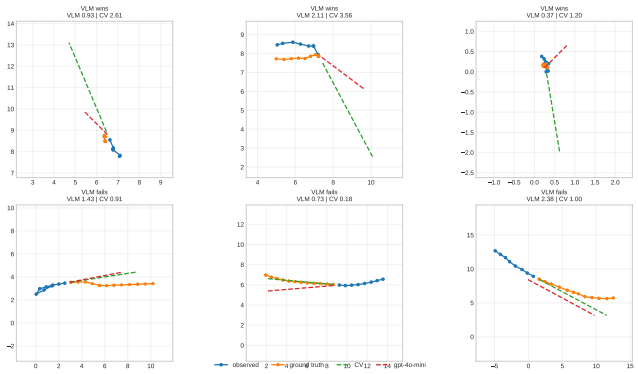

Fig. 2. Qualitative examples. Observed history, ground truth, constant velocity, and GPT-4o-mini predictions. Top: VLM-win cases. Bottom: VLM-fail cases. Axes are independently scaled for visibility.

intent and is an upper bound. This hierarchy answers three questions: can correct intent help, is a hand-coded intent estimator enough, and does the VLM provide a reliable semantic signal? The order matters. If RIGF-Oracle does not improve, then semantic intent is not useful for this residual family. If RIGF-Oracle improves but RIGF-Heuristic does not, then intent estimation is the bottleneck. If RIGF-VLM is worse than RIGF-Heuristic, then the VLM is not yet a reliable semantic sensor for this metric task. This makes RIGF a diagnostic instrument rather than a leaderboard entry.

C. Results

Table VIII gives the diagnostic result. With oracle intent, RIGF improves ADE by 9.2% overall, 13.3% on high-nonlinearity cases, and 12.4% on CV-hard cases. Bootstrap intervals for RIGF-Oracle minus CV are fully below zero. This means correct semantic intent can improve metric forecasting.

The deployable VLM-intent result is negative. GPT-4o-mini returns valid intent JSON for every sample, so parsing is not the issue. The issue is calibration. Its intent accuracy against the future-proxy oracle is only 31.6%, compared with 74.8% for the observed-history heuristic (Table IX). The dominant error is overpredicting deceleration: GPT-4o-mini labels 126/250 samples as decelerating, while the oracle proxy labels only 9/250 that way. As a result, direct RIGF-VLM degrades ADE from 0.554 to 0.592. The gated VLM variant returns to CV-level performance by rejecting unsupported corrections, but it does not improve reliably.

RIGF therefore gives a clear decomposition. Correct intent helps. Current VLM intent is not accurate enough. Conservative fallback prevents harm, but also shows that current VLM intent

TABLE VIII
RIGF DIAGNOSTIC RESULTS. RIGF-ORACLE SHOWS THAT CORRECT INTENT IS USEFUL. DIRECT VLM INTENT IS PARSEABLE BUT BIASED. GATED VLM INTENT SAFELY RETURNS TO CV-LEVEL PERFORMANCE.

Method	All	High-nonlin.	CV-hard
Constant velocity	0.554	0.765	1.091
RIGF-Heuristic	0.575	0.818	1.095
RIGF-VLM	0.592	0.863	1.161
RIGF-VLM-Gated	0.554	0.763	1.089
RIGF-Oracle	**0.502**	**0.663**	**0.956**

TABLE IX
INTENT CALIBRATION DIAGNOSTICS. GPT-4O-MINI PRODUCES VALID INTENT JSON FOR EVERY SAMPLE, BUT THE LABELS DO NOT ALIGN WELL WITH THE FUTURE-PROXY ORACLE.

Diagnostic	Value
VLM intent parse rate	100.0%
VLM vs. oracle-proxy accuracy	31.6%
Heuristic vs. oracle-proxy accuracy	74.8%
VLM/heuristic agreement	36.4%
VLM decelerating labels	126/250
Oracle decelerating labels	9/250

rarely provides actionable signal. The next method target is calibrated intent estimation, not just valid JSON.

For a robot system, this is already actionable. The planner should not accept a VLM's semantic label because it is well formatted or high confidence. It should accept the label only when that label has been validated as useful for the downstream metric module and is supported by current observations. This is the difference between using a VLM as a conversational explainer and using it as a component in a control loop.

VI. DISCUSSION AND LIMITATIONS

A. Robotic implication

A robot must act on every cycle. An invalid forecast is therefore a failure that must trigger fallback, not a sample that can be removed from the benchmark. This is the main reason coverage changes the interpretation of VLM forecasting. Gemini's parsed-only ADE looks excellent, but the parsed subset is easy and the failed subset is hard. A robot designer should care about the full module: VLM plus parser plus fallback.

B. Why full-coverage VLMs match CV

Many ETH/UCY trajectories are smooth, so constant velocity is a strong baseline. A VLM has to infer scale and velocity from the prompt, keep the coordinate frame consistent, and emit 24 numbers. Small scale or velocity errors accumulate over the horizon. The result is not that GPT-4o and GPT-4o-mini fail catastrophically. Rather, under full coverage, they mostly reproduce the performance of a simple kinematic predictor.

C. Few-shot and CoT

Few-shot GPT-4o is the most promising direct VLM result, but its gains are on parsed subsets with reduced coverage. This is useful for diagnosis, not enough for deployment. CoT is worse: it increases error and invalidity. For metric control objects, unconstrained verbal reasoning is a poor interface. A better interface is likely to be short, schema-constrained, and checked against a fallback.

D. Visual inputs

Our visual inputs are rendered BEV plots. They are not raw robot camera images, semantic maps, traversability layers, or goal annotations. The negative image result should therefore be read narrowly. It says that generic plot screenshots do not reliably help off-the-shelf VLMs output metric coordinates. It does not say that visual scene context is useless. Future systems should test richer robot observations and route visual semantics through calibrated motion modules.

E. What RIGF contributes

RIGF is not a state-of-the-art trajectory predictor. Its value is experimental control. It separates four components: base metric prediction, intent estimation, residual correction, and fallback gating. This lets us see where the failure occurs. The oracle result says that correct intent is useful. The VLM result says that current intent labels are biased. The gated result says that fallback can prevent harm.

F. Scope

The experiments use 250 ETH/UCY windows, deterministic single forecasts, and synthetic BEV inputs. This scale is appropriate for controlled API-based probing, but it is not enough to establish broad limits of all VLMs or all robot settings. The RIGF experts are also deliberately simple. A larger follow-up should include more datasets, stronger trajectory backbones, newer frontier VLMs, real robot visual inputs, and multimodal forecast metrics. The most important next comparison is against specialized trajectory models rather than only CV and linear baselines. If a VLM is used at all, it should be tested as an additional semantic sensor on top of such a backbone, not as a replacement for the backbone.

VII. PRACTICAL TAKEAWAYS FOR SEMROB

A. For benchmark designers

Report validity as a main metric. A table with only parsed ADE mixes three abilities: understanding motion, producing a valid structured object, and making an accurate forecast. Coverage and matched fallback baselines should appear in the main results table for any VLM benchmark involving coordinates, plans, tool calls, or object layouts.

B. For robot foundation-model systems

The safest current role for VLMs is not direct coordinate generation. A VLM may propose semantic hypotheses such as "the pedestrian is slowing" or "the agent may turn left". Those hypotheses should pass through a calibrated downstream module with fallback behavior. RIGF implements this pattern in a minimal form: semantic intent can affect the forecast only through a validation-tested residual expert, and unsupported signals return to the fallback.

C. For method developers

The oracle gap is the opportunity. RIGF-Oracle improves over CV, especially on high-nonlinearity and CV-hard cases. This means the right semantic variable can help. What is missing is a reliable test-time estimator of that variable. Promising directions include trajectory-specific intent training, calibrated uncertainty, richer scene evidence, and stronger residual experts.

D. Deployment pattern

A conservative robot architecture is hybrid. Use a guaranteed geometric or learned predictor as the output path. Use a VLM-based semantic module only to propose interpretable corrections. Validate those corrections offline, and require confidence, agreement, or risk gates before overriding the base forecast. This is less dramatic than direct end-to-end VLM forecasting, but it matches the evidence.

E. A checklist for camera-ready robot experiments

For future SemRob-style evaluations, we recommend reporting the following in the main paper: strict parser definition; parse coverage; ADE/FDE on parsed outputs; fallback ADE/FDE on the same parsed outputs; fallback difficulty on failed outputs; and the operational policy used when the VLM fails. If the VLM is used semantically, report label validity separately from label accuracy and from downstream control-relevant error. These small additions make the difference between a capability demo and a robot-system evaluation.

VIII. Conclusion

We evaluated off-the-shelf VLMs as robot-facing metric human-motion forecasters. Under coverage-aware evaluation, GPT-4o-mini and GPT-4o are statistically indistinguishable from constant velocity, while Gemini's apparent advantage is explained by parse-conditioned selection bias. Few-shot GPT-4o improves on parsed subsets, but reduced coverage prevents a full-coverage robot-system claim. We formalized parsed-output bias, introduced fallback-aware reporting, and proposed RIGF as a simple semantic residual diagnostic. RIGF-Oracle shows that correct intent can improve forecasts, but GPT-4o-mini's intent labels are biased and miscalibrated. The practical conclusion is clear: current VLMs should not be trusted as direct coordinate forecasters for robots. They are better viewed as semantic sensors whose outputs must be checked, calibrated, and gated against reliable metric fallbacks.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *CVPR*, pages 961–971, 2016.

[2] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 8469–8488. PMLR, 2023. URL https://proceedings.mlr.press/v202/driess23a.html.

[3] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665–673, 2020.

[4] Gemini Team. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[5] Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. Transformer networks for trajectory forecasting. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 10335–10342, 2021. doi: 10.1109/ICPR48806.2021.9412190.

[6] Google AI for Developers. Gemini 2.5 flash. https://ai.google.dev/gemini-api/docs/models/gemini-2.5-flash, 2025. Model documentation.

[7] Tianyu Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. Stochastic trajectory prediction via motion indeterminacy diffusion. In *CVPR*, pages 17113–17122, 2022.

[8] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *CVPR*, pages 2255–2264, 2018.

[9] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical Review E*, 51(5):4282–4286, 1995.

[10] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[11] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *ECCV*, pages 759–776, 2020.

[12] OpenAI. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. URL https://arxiv.org/abs/2410.21276.

[13] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *ECCV*, pages 683–700, 2020.

[14] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S. Refaat, Rami Al-Rfou, and Benjamin Sapp. MotionLM: Multi-agent motion forecasting as language modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Seff_MotionLM_Multi-Agent_Motion_Forecasting_as_Language_Modeling_ICCV_2023_paper.html.

[15] Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L. Waslander, Yu Liu, and Hongsheng Li. LMDrive: Closed-loop end-to-end driving with large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15120–15130, 2024.

[16] Licheng Wen, Daocheng Yang, Xinyu Fu, Xin Wang, Pinlong Cai, Xin Li, Tao Ma, Pengfei Li, Mingming Xu, Botian Shang, et al. DiLu: A knowledge-driven approach to autonomous driving with large language models. *arXiv preprint arXiv:2309.16292*, 2023.

[17] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K. Wong, Zhenguo Li, and Hengshuang Zhao. DriveGPT4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 9(10):8186–8193, 2024. doi: 10.1109/LRA.2024.3440097.

[18] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M. Kitani. AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *ICCV*, pages 9813–9823, 2021.

[19] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.