NavBind: Diagnosing Instruction-Action Binding at Route Forks in Navigation VLAs

Anonymous ACL submission

Abstract

Vision-language-action (VLA) navigation models translate visual observations and instructions into actions. Navigation success assesses whether an agent reaches the goal. However, reaching the goal successfully does not establish whether the model genuinely follows the instructions or relies on visual cues or navigation history. We introduce NavBind to diagnose instruction-action binding: whether changing the instruction influences the probabilistic distribution of the actions, with the navigation context fixed. We construct 3,252 route-fork contexts by recombining route-aligned instruction segments from R2R-CE and RxR-CE. Each context pairs full instructions that share the description of the route already traveled but specify different branches. Both are evaluated under the same agent state and navigation history. We compare generated local action sequences, the likelihoods of fixed reference action sequences under both instructions, and changes in visual attention and attribution. Across the evaluated models, higher reported navigation success does not imply that the model more faithfully follows the instructions. Our code is available at https://anonymous.4open.science/r/navbind-code-019E/.

1 Introduction

In vision-and-language navigation in continuous environments (VLN-CE) (Krantz et al., 2020), vision-language-action (VLA) models (Zhang et al., 2024) map visual observations and instructions to actions. Success rate measures whether an agent reaches the goal but does not establish whether it follows the instruction. Route-fidelity metrics (Jain et al., 2019) assess the similarity between the generated trajectory and the ground-truth trajectory, providing an indirect measure of instruction following. However, these metrics still cannot establish *whether the model genuinely follows the instructions*, as it may instead rely on visual cues or navigation history alone.

Language interventions, i.e., changing the language instruction and observing how the local actions or navigation performance change, provide a more direct test of language dependence. NavTrust (Jiang et al., 2026) and Contrast Sets (Anwar et al., 2025) rephrase instructions and assess changes in navigation performance. However, these tests keep the intended navigation behaviour unchanged. R2RIE-CE (Taioli et al., 2024) modifies instructions by injecting direction, object, and room errors. However, it neither ensures that the modified instructions describe feasible routes nor provides reference actions corresponding to them. Without matching reference trajectories for the modified instructions, a behavioural change alone does not show that the model followed them. We therefore ask not merely whether the model's behaviour changes, but *whether changing the instruction to describe another feasible route makes the model more likely to choose the corresponding actions*. We keep the agent's state, navigation history, and completed instruction prefix fixed. In Figure 1, both turning left and continuing up the stairs are feasible, and the remaining instruction specifies which branch to take. We use *instruction-action binding* to describe the requirement that the model assigns higher probability to the actions required by the instruction.

We introduce NavBind, a controlled diagnostic built through cross-route recombination of route-instruction examples drawn from R2R-CE and RxR-CE. Using fine-grained route-language alignments, we retain one example's completed instruction prefix and combine it with either its original suffix or a compatible suffix from another route. The resulting full-instruction pairs specify different executable branches under the same replayed history and agent state, yielding 3,252 contexts with reference action sequences for both branches.

Within each fixed context, we first ask: does changing the instruction change the action se-

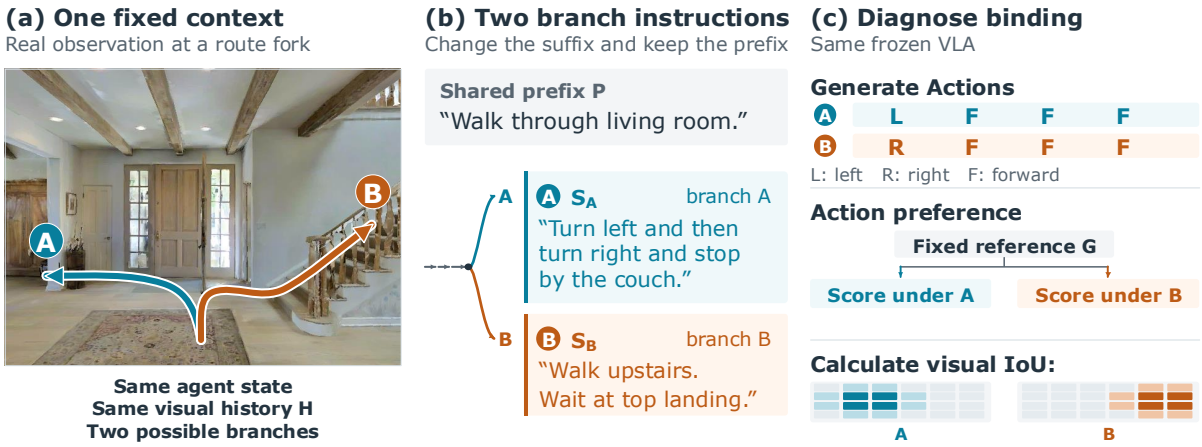


Figure 1: **Does the action follow the instruction?** NAVBIND pairs one fixed navigation history H with two full instructions $P + S_A$ and $P + S_B$. The shared prefix P describes the completed route, and the remaining suffixes S_A and S_B specify different executable branches. This example contrasts turning left with going up the stairs. We jointly examine generated action-sequence changes, fixed-target action likelihood, visual attention, and attribution for each instruction's corresponding reference actions.

quence the model generates? We then ask: does the model assign higher probability to the actions for a branch when instructed to take that branch rather than the other? We test this by comparing the likelihood of each reference action sequence under the two instructions. This comparison can reveal changes in action preference even when the generated sequence stays the same. Finally, we examine the visual response: do attention heatmaps and gradient-based attribution heatmaps highlight different image regions under the two instructions?

We evaluate four navigation VLAs on NAVBIND: Uni-NaVid (Zhang et al., 2025), StreamVLN (Wei et al., 2025), JanusVLN (Zeng et al., 2026), and FutureNav (Zhang et al., 2026a). We find that higher reported navigation success does not imply that the model more faithfully follows the instructions. (Table 1). It also shows that models respond more strongly when the instructions contrast opposite turns than when they contrast forward movement with a turn.

Our contributions are:

- We construct NAVBIND by recombining instruction segments across routes into full instructions for two feasible branches. Each pair preserves the agent state, history, and completed instruction prefix, and provides reference action sequences for both branches.
- We diagnose instruction-action binding using generated sequences to measure output changes, fixed-target likelihoods to assess preferences for the instructed actions, and attention and gradient-based attribution heatmaps to track spatial changes.
- We show that higher reported navigation success does not imply that the model more faithfully follows the instructions. This supports evaluating how language affects local decisions alongside navigation success.

2 Related Work

2.1 Language Diagnostics in VLN

Instruction fidelity beyond success. Goal completion can obscure whether an agent followed the described route. Stay on the Path introduced CLS and longer R4R routes (Jain et al., 2019), while LAW extended language-aligned supervision and evaluation to VLN-CE (Raychaudhuri et al., 2021). Modality ablations further showed that R2R agents could rely strongly on graph structure (Hu et al., 2019). These works motivate language-specific evaluation through task metrics, supervision, and modality ablations.

Language interventions in VLN. Early diagnostics mask object, direction, and numerical cues (Zhu et al., 2022) and swap directional tokens (Hahn et al., 2023). Skill-focused evaluations append atomic instructions to truncated RxR episodes (Yang et al., 2023) and construct human-refined atomic instructions, including same-start left/right pairs (Wang et al., 2024). These studies evaluate agents on discrete navigation graphs and their action tests cannot be directly applied to navigation

VLAs that execute low-level actions.

In VLN-CE, NavTrust evaluates whether agents maintain navigation performance under instruction corruption and stylistic variation (Jiang et al., 2026). Contrast Sets tests instruction rephrasing with unchanged navigation goals on a physical robot (Anwar et al., 2025). However, these tests keep the intended navigation behaviour unchanged. R2RIE-CE injects direction, object, and room errors into instructions to study error detection and localization (Taioli et al., 2024). The modified instructions are not paired with matching reference trajectories. However, it neither ensures that the modified instructions describe feasible routes nor provides reference actions corresponding to them. Without matching reference trajectories for the modified instructions, a behavioural change alone does not show that the model followed them. CAST pairs generated alternative instructions with short action branches for counterfactual training (Glossop et al., 2026), but the intended action behaviour only include a short horizon rather than a long horizon.

NAVBIND constructs full-instruction pairs for long-horizon navigation by recombining original route-aligned instruction spans across source routes. The pairs specify different executable branches under the same agent state, navigation history, and completed instruction prefix. Scoring both reference action sequences under both instructions tests whether the model's local action preferences shift toward the instructed branch.

2.2 Language-Action Coupling in VLAs

Recent work examines whether language and other multimodal signals influence VLA action generation. VLA-Trace connects representation similarity, attention-path knockouts, and rollout-level behavioural probes (Shi et al., 2026), while Häon et al. (2025) identify semantic activation directions and intervene on them to steer actions. Zhang et al. (2026b) diagnose linguistic blindness in manipulation VLAs using contradictory instructions and recalibrate attention at inference time, while Zhan et al. (2026) study robustness to linguistic perturbations and strengthen language guidance through residual steering. These studies establish that language and internal pathways can affect VLA actions, but their evaluations center on manipulation and often combine diagnosis with model modification. NavBind tests instruction-action binding in frozen navigation VLAs: two valid route instructions share the same history but specify different executable branches. We align behaviour with fixed-target likelihood and visual attribution.

3 Problem Setting

3.1 VLN-CE

In vision-and-language navigation in continuous environments (VLN-CE), an agent follows a natural-language instruction I through a continuous 3D environment (Krantz et al., 2020). At time t, it receives an egocentric observation o_t and acts from the navigation history

$$H_t = (o_0, a_0, \ldots, a_{t-1}, o_t). \quad (1)$$

A navigation policy assigns a conditional distribution $\pi_\theta(a_t \mid I, H_t)$ over its next one or few actions. A full episode ends when the agent emits STOP or reaches the maximum steps, and success means stopping near the destination.

We study the widely used four-action interface

$$\mathcal{A} = \{\text{FORWARD}, \text{LEFT}, \text{RIGHT}, \text{STOP}\}, \quad (2)$$

adopted by low-level VLN-CE agents (Krantz et al., 2020) and recent navigation VLAs (Zhang et al., 2025; Wei et al., 2025; Zeng et al., 2026; Zuo et al., 2026; Yang et al., 2026; Zhang et al., 2026a). FORWARD advances the agent by a fixed distance, while LEFT and RIGHT rotate it by a fixed angular increment. This finite action vocabulary provides a common behavioural interface and, for models that expose action logits, permits exact likelihood evaluation of a fixed action sequence.

3.2 Local Route-Fork Diagnosis

Within each source pair, a local diagnostic context consists of a valid history H_h ending at a nonterminal route fork v and its completed instruction prefix P_h, where $h \in \{A, B\}$ identifies the source route (Figure 2(a)). Each remaining suffix S_s, with $s \in \{A, B\}$, specifies one outgoing branch. The full input instruction is $I_{h,s} = P_h + S_s$, and an evaluation condition consists of H_h and $I_{h,s}$.

From the state reached by H_h, the reference action sequence for either executable branch $g \in \{A, B\}$ is

$$G_{h,g} = (a_1^{h,g}, \ldots, a_{K_{h,g}}^{h,g}), \quad K_{h,g} \le K. \quad (3)$$

The diagnostic horizon K limits its length, and STOP ends the sequence.

For each context (H_h, P_h), NAVBIND compares $I_{h,A}$ and $I_{h,B}$, as in a horizontal comparison in Figure 2(b). Both conditions start from exactly the

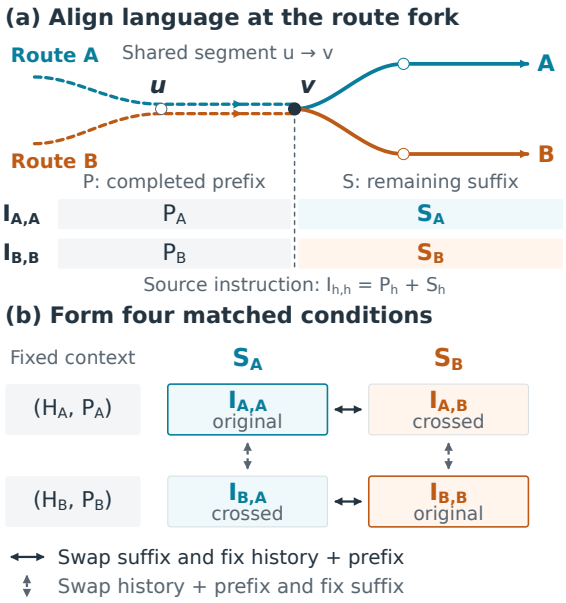


Figure 2: **Route-aligned pair construction.** (a) Original routes A/B (blue/orange) share a directed segment from u to fork v, then diverge. Dashed paths are traversed, and solid paths remain. Each full instruction $I_{h,h} = P_h + S_h$ splits at v into completed prefix P_h and remaining suffix S_h. (b) Horizontal comparisons change only the suffix. Vertical comparisons change history and prefix together with the suffix fixed.

same pose, current observation, and observation-action history. Instruction-action binding requires the model to assign higher probability to the actions required by the suffix. Section 4 describes how route geometry and fine-grained language annotations yield these matched conditions.

4 Counterfactual Language Pairs for Navigation

4.1 Route-Aligned Source Pairs

We construct source pairs from English R2R-CE and RxR-CE routes and instructions (Anderson et al., 2018; Ku et al., 2020; Krantz et al., 2020). For R2R and English RxR, we use released fine-grained annotations from FGR2R (Hong et al., 2020) and Landmark-RxR (He et al., 2021), respectively. These annotations divide full instructions into shorter text segments, called *sub-instructions*, and align each segment with the part of the trajectory it describes.

We first identify route pairs with the structure shown in Figure 2(a): two routes in the same scene share a directed segment from u to the fork v, then take different outgoing edges at v. The routes may differ before u. Both branches beyond v must be executable and nonterminal. We require a sufficiently long shared segment $u \to v$ to align the two routes' positions and orientations at v. Sharing v alone leaves arrival orientation unconstrained, which can change the interpretation of instructions such as "turn left."

For each source route h in a candidate pair, we use the sub-instruction annotations to place the language boundary at v. An edge-exact boundary requires the preceding subpath to end at v and the next subpath to start at v with its first edge entering branch h. We recover the corresponding original-text offsets by monotonic matching with case, spacing, and punctuation normalized. Cutting at the start of that next sub-instruction yields $I_{h,h} = P_h + S_h$: the prefix P_h covers the full route up to v, including any portion before u, and the suffix S_h retains the remaining instruction. Both spans preserve their original wording. Selection uses route annotations alone. Repeated physical forks are deduplicated.

For each retained pair, expert replay of route h from its source start to v yields a history H_h paired with its completed prefix P_h. From each state at v, we construct both local branch targets $G_{h,A}$ and $G_{h,B}$. The resulting history-prefix pairs form the fixed row contexts in Figure 2(b).

4.2 Matched Interventions

Figure 2(b) crosses each context (H_h, P_h) with both suffixes. Each cell gives the instruction $I_{h,s}$, evaluated with the history H_h in its row.

Suffix intervention. Within each row, the diagonal instruction $I_{h,h}$ retains the source text, and the off-diagonal instruction substitutes the other route's suffix. Evaluating $I_{h,A}$ and $I_{h,B}$ from the same replay history and fork state implements the primary fixed-context comparison defined in Section 3.2.

Matched prefix intervention. Across rows, the supplementary comparison between $(H_A, I_{A,s})$ and $(H_B, I_{B,s})$ holds the suffix fixed while jointly changing the history and its source prefix. It compares the model's response to the same branch beyond v across the two source-matched contexts.

Each source pair initially yields two candidate contexts and four conditions, linked by a source-pair identifier. We retain contexts whose two reference sequences differ within four actions at every evaluated native action scale. The resulting NAVBIND set contains 3,252 contexts from 1,648

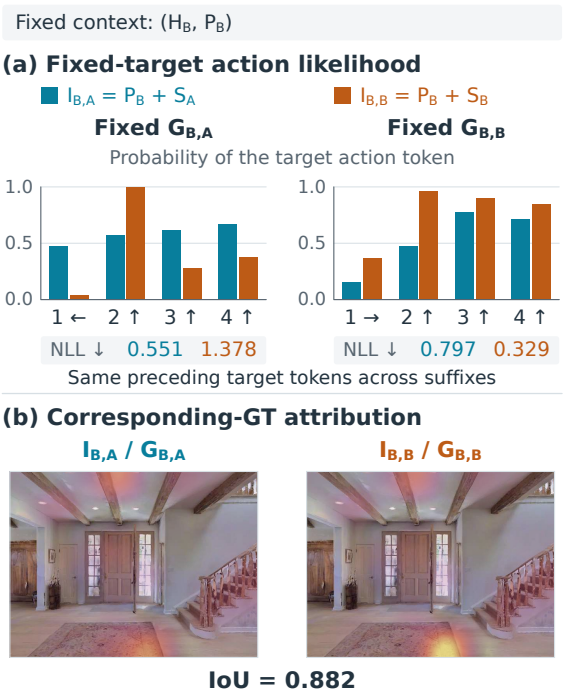


Figure 3: **Action likelihood and visual attribution.** (a) Conditional action-token probabilities for each fixed reference under the two suffixes. Arrows denote left, forward, and right actions. (b) Full attention×gradient heatmaps for each suffix's corresponding reference. Warmer colors indicate higher attribution on a shared scale. IoU uses the original top-25% masks.

source pairs, shared by all models. Appendix B.1 details the selection.

5 Diagnosing Instruction-Action Binding

Given the matched conditions constructed in Section 4, we analyze instruction-action binding from three complementary perspectives. We first measure whether an intervention changes the generated local action sequence. We then measure how it changes the likelihood and action distribution of a fixed reference target. Finally, we examine how visual attention and attribution change across conditions. Likelihood comparisons keep the target fixed, while gradient-based attribution follows each instruction's corresponding GT. Figure 3 illustrates these comparisons on one navigation context.

5.1 Generated Action Sequences

For each context (H_h, P_h), we generate local action sequences under $I_{h,A}$ and $I_{h,B}$ using native inference without teacher forcing. We report the percentage of contexts whose sequences match exactly in action symbols, order, and length within horizon K as Same sequence (Same seq.).

5.2 Fixed-Target Action Likelihood

Discrete generation can remain unchanged even when model's action distribution changes. We therefore measure changes in the likelihood of fixed reference action sequences.

Figure 3(a) illustrates the scoring procedure for context (H_B, P_B), corresponding to the second row of Figure 2(b). The targets $G_{B,A}$ and $G_{B,B}$ are each scored under both instructions, $I_{B,A}$ and $I_{B,B}$, giving four likelihood scores.

In general, $L_{h,s,g}$ denotes the mean negative log-likelihood (NLL) of target $G_{h,g}$ under instruction $I_{h,s}$ and history H_h. Its indices specify the history h, input suffix s, and target branch g, respectively. For chunk policies, we compute the mean NLL over the target's action tokens $z_1, \ldots, z_N$:

$$L_{h,s,g} = -\frac{1}{N}\sum_{k=1}^{N} \log p_\theta(z_k \mid I_{h,s}, H_h, z_{<k})\,. \tag{4}$$

Here p_θ is the model's full-vocabulary token probability. We use natural logarithms throughout. At position k, the model receives the same preceding target tokens $z_{<k}$ under both suffixes (teacher forcing). Only action tokens contribute to the average. Lower $L_{h,s,g}$ means higher target likelihood, and perplexity is $\mathrm{PPL}_{h,s,g} = \exp(L_{h,s,g})$.

Single-step policies instead score each target action at the state reached by executing its preceding GT actions. We sum the native action-token log probabilities across decisions and divide by the total valid token count. Both suffixes use the same GT observation sequence, with independent model memory. This supplies a closed-loop counterpart to the chunk score. The primary suffix effects keep H_h, P_h, and $G_{h,g}$ fixed:

$$d_{h,A} = L_{h,B,A} - L_{h,A,A}, \tag{5}$$
$$d_{h,B} = L_{h,A,B} - L_{h,B,B}. \tag{6}$$

The mean is $\bar{d}_h = \frac{1}{2}(d_{h,A} + d_{h,B})$. We report its average across evaluated samples as $\bar{d}$.

To interpret $\bar{d}$ in Table 1, positive values mean that matching suffixes make their reference actions more likely, negative values mean less likely, and zero means no net advantage. We convert this log-probability difference to a relative percentage change using $(\exp(\bar{d}) - 1) \times 100\%$. Thus, $\bar{d} = 0.0127$ gives $(\exp(0.0127) - 1) \times 100\% \approx 1.28\%$ for reference-action token probabilities, averaged geometrically across tokens, targets, and contexts.

We report Both+ as the percentage of contexts where both $d_{h,A} > 0$ and $d_{h,B} > 0$, so each reference is more likely under its matching suffix.

As a complementary metric, we report symmetric KL divergence. Let $q^{(j)}_{h,s,g}$ denote the action distribution, renormalized over the four legal actions, at reference step j of $G_{h,g}$, scored with $I_{h,s}$ and H_h using the protocol above. With the same preceding target actions, we compute $D_{\mathrm{SKL}}(q^{(j)}_{h,A,g}, q^{(j)}_{h,B,g})$, then average over steps, targets, and contexts. KL is zero for unchanged action probabilities and increases with their redistribution, irrespective of target alignment.

5.3 Visual Attention and GT-Conditioned Attribution

We next compare visual attention and the visual sensitivity of each instruction's corresponding reference action sequence. Let A denote decoder attention from action-query positions to visual tokens. Raw attention is measured under each fixed reference sequence $G_{h,g}$, so both suffixes receive the same preceding target tokens at every prediction position.

For gradient-based attribution, the explained action follows the instruction: $I_{h,s}$ is paired with its corresponding GT sequence $G_{h,s}$. Its score is

$$Q_{h,s} = -L_{h,s,s}. \tag{7}$$

We extract gradient magnitude $|\nabla_A Q_{h,s}|$ and the elementwise product magnitude $|A \odot \nabla_A Q_{h,s}|$. For chunk policies, maps aggregate the teacher-forced action-query positions under fixed observations.

Let $M^e_{h,s,g}$ denote a map computed under instruction $I_{h,s}$ and reference sequence $G_{h,g}$. Raw attention compares the same reference under both suffixes:

$$\mathrm{IoU}^{\mathrm{attn}}_{h,g} = \mathrm{IoU}\big(\mathcal{B}(M^{\mathrm{attn}}_{h,A,g}), \mathcal{B}(M^{\mathrm{attn}}_{h,B,g})\big)\,, \tag{8}$$

and averages the two references within each context. Gradient and attention×gradient compare the corresponding GTs:

$$\mathrm{IoU}^{e}_{h} = \mathrm{IoU}\big(\mathcal{B}(M^{e}_{h,A,A}), \mathcal{B}(M^{e}_{h,B,B})\big)\,. \tag{9}$$

This comparison jointly changes the instruction and the explained action. Here $\mathcal{B}$ selects a fixed top-response fraction on the scoring grid. Multi-resolution visual routes are combined before thresholding as specified in Appendix B.1. Heatmaps are interpolated to RGB resolution for display, while IoU uses the original scoring-grid masks.

For single-step policies, fixed-reference attention is compared at aligned GT states and averaged over steps. Corresponding-GT gradients are compared only at the shared fork observation, using each instruction's first GT action. Later GT trajectories can have different views.

IoU equals one for identical selected regions and zero for disjoint regions. Lower IoU indicates greater spatial change under the respective comparison. Section 7 relates these changes to fixed-target likelihoods and changes in the action sequences.

6 Experimental Setup

6.1 Data and Models

We evaluate Uni-NaVid (Zhang et al., 2025), StreamVLN (Wei et al., 2025), JanusVLN (Zeng et al., 2026), and FutureNav (Zhang et al., 2026a) on the same 3,252 NAVBIND contexts constructed in Section 4, using released checkpoints.

6.2 Evaluation Protocol

We use a four-action horizon ($K = 4$) and each model's native action increments. Uni-NaVid and StreamVLN generate action chunks under fixed observations, while JanusVLN and FutureNav execute each predicted action before the next decision. We retain native decoding: Uni-NaVid uses sampling and StreamVLN uses greedy decoding. Appendices A.3 and B.1 give more details.

7 Results

7.1 Main Results

In Table 1, Uni-NaVid has the highest reported RxR-CE val-unseen SR. However, StreamVLN has the highest $\bar{d}$ and KL values. Uni-NaVid's Both+ is 29.15%, only 4.15 percentage points above a 25% reference. This reference assumes that the two likelihood gains are independent and each has a 50% chance of being positive. StreamVLN and FutureNav have nearly identical reported SR (54.4% and 54.5%), yet StreamVLN has higher $\bar{d}$ (0.8360 versus 0.4411), Both+ (80.38% versus 70.60%), and KL (0.8233 versus 0.6208). Thus, higher navigation success does not necessarily mean that changing the instruction has a larger effect on action probabilities. JanusVLN and FutureNav change their action sequences more often than StreamVLN (lower Same seq.), but show smaller likelihood gains for the instructed action sequences (lower $\bar{d}$). Their attention and attribution heatmaps also highlight more similar visual regions across instructions

Model	RxR-CE SR (%) ↑	Behavior Same seq. (%) ↓	Fixed-target likelihood $\bar{d}$ ↑	Fixed-target likelihood Both+ (%) ↑	Fixed-target likelihood KL ↑	Visual overlap (IoU) Attn. ↓	Visual overlap (IoU) Grad. ↓	Visual overlap (IoU) Attn.×Grad. ↓
Uni-NaVid	56.1	85.79	0.0127	29.15	0.0124	0.860	0.454	0.773
StreamVLN	54.4	32.41	0.8360	80.38	0.8233	0.602	0.420	0.456
JanusVLN*	51.4	30.75	0.3643	72.88	0.4537	0.727	0.760	0.698
FutureNav*	54.5	25.49	0.4411	70.60	0.6208	0.728	0.786	0.702

Table 1: Reported navigation success and NAVBIND diagnostics. SR is the officially reported RxR-CE val-unseen success rate for each released checkpoint used in our experiments. Lower Same seq. and IoU indicate greater output and spatial change. Higher $\bar{d}$ and Both+ indicate stronger or more frequent matching-suffix likelihood gains. Higher KL indicates greater action-distribution change. IoU uses top-25% masks. * marks single-step models.

(higher IoU). One possible explanation is that these models rely on similar visual cues without clearly distinguishing the instructed branch, leaving them less confident about which action to take. Small probability shifts could then be enough to change the selected action.

7.2 Consistency Across Diagnostics

Table 2 shows that StreamVLN's Unchanged group has lower $\bar{d}$ than its Changed group. In this group, changing the instruction produces a smaller increase in the likelihood of the required actions. The Unchanged group also has higher Attn., Grad., and Attn.×Grad. IoU, so the heatmaps highlight more similar regions under the two instructions. Thus, the likelihood and visual results agree with the generated-action results: unchanged action sequences come with smaller changes toward the required actions and smaller changes in the highlighted visual regions.

Output	N	$\bar{d}$ ↑	Attn. IoU ↓	Grad. IoU ↓	Attn.×Grad. IoU ↓
Unchanged	1,054	0.3766	0.6754	0.4831	0.5128
Changed	2,198	1.0563	0.5673	0.3895	0.4292

Table 2: StreamVLN results for unchanged and changed action sequences under the two instructions.

7.3 Weaker Responses with Longer Prefixes

Table 3 shows that StreamVLN has higher Same seq. and lower $\bar{d}$ and KL as the completed prefix contains more tokens. With longer completed prefixes, the model more often generates the same action sequence under the two instructions, and the overall action distribution changes less. Thus, longer completed prefixes are associated with weaker responses to the remaining instruction.

Prefix tokens	Same seq. (%) ↓	$\bar{d}$ ↑	KL ↑
≤ 47	30.80	0.8904	0.8964
48-76	30.90	0.8492	0.8497
≥ 77	35.62	0.7660	0.7205

Table 3: StreamVLN results by prefix token count, with roughly equal group sizes.

7.4 Stronger Responses to Opposite Turns

We compare StreamVLN across three action-pair groups: forward-left (F/L), forward-right (F/R), and left-right (L/R). Table 4 shows that the L/R group has lower Same seq. and higher $\bar{d}$ and KL than the other two groups. In the L/R group, changing the instruction changes the generated action sequence more often. It also produces a larger increase in the likelihood of the required actions and a larger change in the overall action distribution. Thus, StreamVLN responds more strongly when the instructions contrast opposite turns than when they contrast forward movement with a turn.

Action pair	N	Same seq. (%) ↓	$\bar{d}$ ↑	KL ↑
F/L	486	37.24	0.7426	0.6905
F/R	508	33.07	0.8404	0.8248
L/R	1,504	26.66	1.0135	0.9458

Table 4: StreamVLN results by action-pair group. F/L/R denote forward/left/right.

7.5 Attention Changes at Action Divergence

We group contexts by the first step at which the two GT sequences differ. For example, the GT sequences $[F, F, L, F]$ and $[F, F, R, F]$ first differ at step 3. Table 5 reports StreamVLN's attention IoU at each of the four action positions. For contexts whose GT sequences first differ at step 2, 3, or 4, attention IoU is higher at the preceding positions

and reaches its lowest value at the first differing position. In the step-2 and step-3 groups, IoU rises again at the next position. Thus, after shared initial actions, the attention heatmaps under the two instructions differ most where the required actions first differ.

First differing GT position	N	Action position			
		1	2	3	4
1	2,498	0.5911	0.5829	**0.5687**	0.5757
2	409	0.6103	**0.5693**	0.5736	0.5765
3	219	0.6569	0.6183	**0.5777**	0.6021
4	126	0.6792	0.6642	0.6373	**0.6273**

Table 5: StreamVLN attention IoU at each action position. Bold marks the lowest mean IoU in each row.

7.6 Larger Original-Pairing Advantages on Training Sources

StreamVLN has lower NLL under original instructions than under recombined instructions, with a larger gap on train than on val-unseen. For each context (H_h, P_h), we compare $L_{h,h,h}$ under the original instruction with $L_{h,s,s}$ under the recombined instruction ($s \neq h$), scoring each instruction's corresponding reference. We report results separately for train and val-unseen, with both source instructions in each context coming from the same split. The gap is mean recombined NLL minus mean original NLL. A positive gap means lower NLL under original instructions.

Source split	N	Original ↓	Recombined ↓	Gap
Train	2,367	0.4666	0.6007	+0.1341
Val-unseen	385	0.5763	0.6374	+0.0611

Table 6: StreamVLN mean reference-action NLL under original and recombined pairings. Gap is recombined minus original NLL.

Table 6 shows lower NLL under original instructions than under recombined instructions in both splits. The gap is larger on train than on val-unseen. StreamVLN has lower NLL on train than on val-unseen, but this difference is smaller for recombined instructions than for original instructions. This pattern is consistent with overfitting to the original pairings.

7.7 Stability Across Histories

We compare the two source histories and their matched prefixes (Figure 2(b)). With the suffix and four-action reference unchanged, we expect consistent generated actions and associated visual evidence across histories. Here, KL compares the action distributions under the two histories for the same reference.

Model	Same seq. (%) ↑	KL ↓	Attn. IoU ↑	Grad. IoU ↑	Attn.×Grad. IoU ↑
StreamVLN	51.42	0.3131	0.4295	0.3032	0.4088
JanusVLN*	25.18	0.3225	0.2675	0.2278	0.2543
FutureNav*	27.67	0.3150	0.3083	0.2548	0.2894

Table 7: Cross-history stability. With the suffix and reference actions fixed, higher Same seq. and IoU and lower KL indicate greater stability, opposite to Table 1. * marks single-step models. Appendix B.5 shows more details.

Table 7 shows that StreamVLN preserves its generated sequence more often than JanusVLN and FutureNav (51.42% versus 25.18% and 27.67%). It also has higher IoU across all three visual measures and slightly lower KL. Thus, among these three models, generated actions, action distributions, and visual heatmaps give the same stability ordering: StreamVLN, FutureNav, and JanusVLN.

8 Conclusion

NAVBIND diagnoses instruction-action binding through route-fork interventions under fixed navigation contexts, combining generated behavior, fixed-target likelihoods, and visual attention and attribution. Across 3,252 shared contexts, higher reported navigation success does not always mean more frequent action changes or larger shifts in action probabilities. These findings support evaluating how language affects local decisions alongside episode-level navigation success.

Limitations

Recombining source instructions may introduce ambiguity or unnatural transitions at the segment boundaries. The four-action window captures local instruction responses, while later responses and error recovery remain outside the evaluation horizon. Model-native action increments also give this window different physical extents across models. Visual IoU quantifies spatial changes in attention and attribution, without establishing whether the selected regions are relevant to the instructed branch or causally determine the action.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.

Abrar Anwar, Rohan Gupta, and Jesse Thomason. 2025. Contrast sets for evaluating language-guided robot policies. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 2205–2219. PMLR.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3D: Learning from RGB-D data in indoor environments. In *International Conference on 3D Vision (3DV)*.

Catherine Glossop, William Chen, Arjun Bhorkar, Dhruv Shah, and Sergey Levine. 2026. CAST: Counterfactual labels improve instruction following in vision-language-action models. *IEEE Robotics and Automation Letters*, 11(10):11785–11792.

Meera Hahn, Amit Raj, and James M. Rehg. 2023. Which way is "right"?: Uncovering limitations of vision-and-language navigation models. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems*, pages 2415–2417. International Foundation for Autonomous Agents and Multiagent Systems.

Bear Häon, Kaylene Caswell Stocking, Ian Chuang, and Claire Tomlin. 2025. Mechanistic interpretability for steering vision-language-action models. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 2743–2762. PMLR.

Keji He, Yan Huang, Qi Wu, Jianhua Yang, Dong An, Shuanglin Sima, and Liang Wang. 2021. Landmark-RxR: Solving vision-and-language navigation with fine-grained alignment supervision. In *Advances in Neural Information Processing Systems*, volume 34.

Yicong Hong, Cristian Rodriguez, Qi Wu, and Stephen Gould. 2020. Sub-instruction aware vision-and-language navigation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3360–3376. Association for Computational Linguistics.

Ronghang Hu, Daniel Fried, Anna Rohrbach, Dan Klein, Trevor Darrell, and Kate Saenko. 2019. Are you looking? grounding to multiple modalities in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6551–6557. Association for Computational Linguistics.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1862–1872. Association for Computational Linguistics.

Huaide Jiang, Yash Chaudhary, Yuping Wang, Zehao Wang, Raghav Sharma, Manan Mehta, Yang Zhou, Lichao Sun, Zhiwen Fan, Zhengzhong Tu, and Jiachen Li. 2026. NavTrust: Benchmarking trustworthiness for embodied navigation. In *2026 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 2020. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Computer Vision-ECCV 2020*, pages 104–120. Springer.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-Across-Room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 4392–4412. Association for Computational Linguistics.

Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel Chang. 2021. Language-aligned waypoint (LAW) supervision for vision-and-language navigation in continuous environments. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4018–4028. Association for Computational Linguistics.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. 2019. Habitat: A platform for embodied AI research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347.

Haoyuan Shi, Xiancong Ren, Yingji Zhang, Qinfan Zhang, Jiayu Hu, Haozhe Shan, Han Dong, Jinpeng Lu, Yinda Chen, Yi Zhang, Yong Dai, and Xiaozhu Ju. 2026. VLA-Trace: Diagnosing vision-language-action models through representation and behavior tracing. *arXiv preprint arXiv:2605.30117*.

Francesco Taioli, Stefano Rosa, Alberto Castellini, Lorenzo Natale, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, and Yiming Wang. 2024. Mind the error! detection and localization of instruction errors in vision-and-language navigation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12993–13000.

Zehao Wang, Minye Wu, Yixin Cao, Yubo Ma, Meiqi Chen, and Tinne Tuytelaars. 2024. Navigating the nuances: A fine-grained evaluation of vision-language

navigation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4681–4704. Association for Computational Linguistics.

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, Xihui Liu, and Jiangmiao Pang. 2025. StreamVLN: Streaming vision-and-language navigation via SlowFast context modeling. *arXiv preprint arXiv:2507.05240*.

Jiahao Yang, Zihan Wang, Xiangyang Li, Xing Zhu, Yujun Shen, Yinghao Xu, and Shuqiang Jiang. 2026. GA-VLN: Geometry-aware BEV representation for efficient vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 40053–40063.

Zijiao Yang, Arjun Majumdar, and Stefan Lee. 2023. Behavioral analysis of vision-and-language navigation agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2574–2582.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Shichao Xie, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. 2026. JanusVLN: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. In *International Conference on Learning Representations*, pages 33001–33026.

Zhihao Zhan, Yuhao Chen, Jiaying Zhou, Qinhan Lyu, Hao Liu, Keze Wang, Liang Lin, and Guangrun Wang. 2026. Stable language guidance for vision-language-action models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4137–4159. Association for Computational Linguistics.

Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. 2025. Uni-NaVid: A video-based vision-language-action model for unifying embodied navigation tasks. In *Proceedings of Robotics: Science and Systems*.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. 2024. NaVid: Video-based VLM plans the next step for vision-and-language navigation. In *Proceedings of Robotics: Science and Systems*.

Lingfeng Zhang, Zeying Gong, Xiaoshuai Hao, Haoxiang Fu, Qiang Zhang, Mingliang Zhou, Hangjun Ye, Xiaojun Liang, Junwei Liang, and Wenbo Ding. 2026a. FutureNav: Unified world-action modeling for vision-and-language navigation. *arXiv preprint arXiv:2606.30367*.

Ninghao Zhang, Bin Zhu, Shijie Zhou, and Jingjing Chen. 2026b. Restoring linguistic grounding in VLA models via train-free attention recalibration. In *Computer Vision-ECCV 2026*, volume 17068 of *Lecture Notes in Computer Science*, pages 39–56. Springer.

Wanrong Zhu, Yuankai Qi, Pradyumna Narayana, Kazoo Sone, Sugato Basu, Xin Wang, Qi Wu, Miguel Eckstein, and William Yang Wang. 2022. Diagnosing vision-and-language navigation: What really matters. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5981–5993. Association for Computational Linguistics.

Jing Zuo, Lingzhou Mu, Fan Jiang, Chengcheng Ma, Mu Xu, and Yonggang Qi. 2026. FantasyVLN: Unified multimodal chain-of-thought reasoning for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 29674–29683.

A Responsible Research Details

A.1 Ethical Considerations

NAVBIND supports research on reliable instruction following using existing navigation data. Its local diagnostics can help identify errors, but treating them as evidence of safe physical deployment could expose people or property to navigation failures. English-only coverage limits application to other language communities. Indoor imagery may reveal private spaces or identifying details, including if navigation systems are used for surveillance.

A.2 Scientific Artifacts

Artifacts used and created. We use existing instructions, alignments, indoor scenes, simulator software, and pretrained models. We create route-pair manifests, diagnostic code, measurement records, and rule-generated cue labels.

Attribution and sources. Instructions come from R2R (Anderson et al., 2018) and RxR (Ku et al., 2020), with FGR2R (Hong et al., 2020) and Landmark-RxR (He et al., 2021) alignments. Matterport3D provides the scenes (Chang et al., 2017), and replay uses Habitat (Savva et al., 2019) and VLN-CE (Krantz et al., 2020). The evaluation baselines are Uni-NaVid (Zhang et al., 2025), StreamVLN (Wei et al., 2025), JanusVLN (Zeng et al., 2026), and FutureNav (Zhang et al., 2026a).

Rights to release derivative artifacts. We reviewed upstream terms for the planned NAVBIND data release, including source, segmented, and recombined instructions and provenance metadata. RxR annotations use CC BY 4.0. The Landmark-RxR datasheet states CC BY and applicable Matterport conditions. The FGR2R repository carries an MIT notice. Underlying R2R data follow the R2R repository's data terms, which refer to the Matterport agreement. Our data release will retain source attribution and license notices, identify segmentation and recombination as modifications, and include the Matterport agreement with its applicable non-commercial academic-use conditions.

Any StreamVLN adaptations we release will follow its CC BY-NC-SA 4.0 license, including its non-commercial and share-alike terms. We will retain source attribution, copyright and license notices, identify our modifications, and preserve third-party components' applicable licenses.

(a) Source pairs

Source-split combination	Pairs
Train / Train	1,200
Train / Val-seen	241
Val-seen / Val-seen	13
Val-unseen / Val-unseen	194
Total	1,648

(b) Contexts

Source split	R2R	RxR	Total
Train	133	2,468	2,601
Val-seen	11	255	266
Val-unseen	24	361	385
Total	168	3,084	3,252

Table 8: Source provenance of the 3,252-context NAVBIND set. (a) Source-pair split combinations. (b) Contexts by the source history's dataset and original split, counting trajectory reuse across pairs.

Data release scope and privacy. The planned NAVBIND data release contains navigation instructions, reference routes in simulated environments, and dataset provenance metadata. Users will obtain Matterport3D scene data directly from the official source. No annotator identifiers or other personal metadata are included.

Objectionable content in released artifacts. We assessed the risk of objectionable content as low because NAVBIND only segments and recombines existing navigation instructions. We therefore did not consider a dedicated content audit necessary. A lightweight lexicon screen of the text corpus found no matches for objectionable terms.

Artifact documentation and coverage. NAVBIND contains English indoor-navigation instructions derived from R2R-CE and RxR-CE. Its linguistic coverage includes directions, landmarks, rooms and regions, spatial relations and ordinals, and temporal sequencing. The derived metadata does not record annotator demographics.

Dataset sizes and splits. NAVBIND contains 3,252 contexts from 1,648 source pairs across 63 scenes. Table 8 summarizes their source datasets and original splits. All contexts are used for frozen-model evaluation, with split labels indicating provenance in the original datasets.

A.3 Computational Experiments

The main empirical objective is to characterize instruction-action binding through expert replay,

native model inference, teacher-forced action scoring, and attribution backpropagation with frozen model parameters.

Model sizes. Uni-NaVid, StreamVLN, JanusVLN, and FutureNav contain approximately 7.75, 8.03, 9.31, and 5.48 billion parameters, respectively. Counts include all loaded model components, including visual and spatial encoders, multimodal projectors, and auxiliary heads, with shared weights counted once.

Computational budget and infrastructure. We estimate evaluation cost by extrapolating measured per-condition times to 13,992 conditions per model, before primary-cohort filtering. With elapsed times normalized to one GPU, StreamVLN averages 27.59 s on an NVIDIA H100 (80 GB), JanusVLN 98.56 s on an A100 (40 GB), and FutureNav 51.63 s on an H100 (80 GB). These correspond to approximately 107, 383, and 201 GPU-hours, respectively, for the selected production configurations.

For Uni-NaVid, we separate offline rendering from model computation and reuse cached RGB frames across instruction conditions. Shared trajectory-cache construction is estimated at 20.6 H100 GPU-hours from dataset-specific per-frame rendering times. Generation takes an estimated 2.9 H100 GPU-hours, and target scoring with attribution takes 2.6 A100 GPU-hours. The scoring stage uses four 40 GB A100 GPUs with two resident workers per GPU, counting physical GPU time once. These evaluation and rendering stages total approximately 717 GPU-hours, counting the shared cache once. Actual cumulative consumption was higher because earlier inference pipelines were less optimized and pilot experiments, failed attempts, and repeated runs added further compute.

Experimental setup. We evaluate released checkpoints without further training or few-shot instruction-action demonstrations. Model weights use FP16 for Uni-NaVid and BF16 for the other models. The StreamVLN checkpoint is evaluated with a four-action horizon, a 32-frame context setting, eight history frames, seed 0, and greedy decoding with one beam. Native behavior inference uses SDPA attention, while attribution uses eager attention and the final four transformer layers. Each instruction receives its own model context. Target scores exclude prompt and non-action tokens. The primary attribution overlap uses the top 25% of patches. Uni-NaVid uses the released 7B navigation checkpoint with its EVA vision weights and 224-pixel image processor. Its reported inference uses native temperature-0.5 sampling with seed 0 reset for each condition. Appendix B.1 specifies its scoring and attribution extraction. JanusVLN uses `JanusVLN_Base`, and FutureNav uses `FutureNav-4B-Base`.[1] They receive the complete replay history and apply their native frame selection and memory updates, with greedy decoding and one beam. Each instruction and execution window restores an independent model context. Runs use seed 0. Appendix B.1 specifies native action increments, target-scoring rules, and attribution backends.

Implementation. We use PyTorch and Transformers with adapters to the official Uni-NaVid, StreamVLN, JanusVLN, and FutureNav implementations for inference and attribution. Models use their native tokenizers and image processors. Replay uses Habitat, and metric aggregation uses NumPy. Appendix B.1 specifies action scoring, visual-map normalization, and top-25% IoU masks.

A.4 Human Annotation

One author audited 100 recombined instructions sampled uniformly without replacement from the 3,252-context evaluation set.

Full annotation instructions.

> Compare each constructed instruction with its source instructions and extracted spans. Assess (1) fluency at split and join boundaries and (2) semantic coherence, including clarity of references. A span need not form a full sentence if its role in the instruction is clear. For each criterion, choose acceptable, problematic, or uncertain. Mark any problematic or uncertain span and give a brief reason. Evaluate the text without consulting model predictions.

Audit results. The author rated 98 instructions acceptable and two problematic for fluency, and all 100 acceptable for semantic coherence. Neither criterion received uncertain ratings. The two fluency issues involved repeated forward-movement wording and an awkward transition from a left-side landmark description to a right-turn instruction. Both instructions were retained unchanged in the evaluation set, with their fluency flags recorded in the audit.

[1] Table 1 uses the officially reported RxR-CE val-unseen SR for these released checkpoints: Uni-NaVid (56.1%), StreamVLN (54.4%), JanusVLN Base (51.4%), and FutureNav-4B-Base (54.5%).

A.5 AI Assistance

OpenAI Codex assisted with literature review, parts of the analysis implementation, code review, manuscript editing, and appendix drafting and editing. The authors take responsibility for the methods, results, citations, and final text.

B Additional Results and Robustness Checks

B.1 Measurement Implementation and Coverage

Construction and coverage. Source pairs require exact instruction mappings, fully recovered text spans, valid subpath partitions, and an edge-exact boundary at v. These checks use route geometry and fine-grained annotations, with repeated physical forks deduplicated as described in Section 4.

Source histories are replayed with 0.25 m forward motion and 15° turns. From each frozen fork state, we compute both reference targets with the evaluated model's native increments: 0.5 m/30° for Uni-NaVid and 0.25 m/15° for StreamVLN, JanusVLN, and FutureNav.

The preselection pool contains 1,749 source pairs and 3,498 candidate contexts. We retain contexts whose two four-action targets differ at both native action scales. Identical targets occur in 69 contexts at 0.5 m/30° and 234 at 0.25 m/15°, with 57 in both groups. Removing their union excludes 246 contexts, leaving 3,252 contexts from 1,648 source pairs. Each context retains its source-matched prefix, both suffixes, and both target scores, giving 6,504 instruction conditions per model. The full factorial manifest also crosses prefixes, yielding 13,992 conditions before selection. Crossed-prefix conditions are excluded from the reported analyses.

Visual-input verification for StreamVLN. We verified that, in all 3,252 evaluated StreamVLN contexts, the paired suffix conditions have identical per-frame RGB pixel hashes for both current and historical images. Their input-frame indices and ordering are also identical.

Chunk-policy measurements. Uni-NaVid and StreamVLN score each reference action as one native token. NLL uses full-vocabulary probabilities. KL renormalizes the four legal-action logits at each reference prefix and averages $D_{\mathrm{SKL}}(p, q) = \frac{1}{2}(D_{\mathrm{KL}}(p\|q) + D_{\mathrm{KL}}(q\|p))$ over positions and both targets. Generated sequences use up to the first four parsed actions, without STOP padding.

Uni-NaVid extracts gradients of the summed corresponding-GT log probabilities, equivalent to the mean objective up to a scale removed by normalization. Gradient and attention$\times$gradient magnitudes are taken before averaging heads and action-query positions within each of the final four decoder layers, then combining layers. Each map is normalized over 84 visual tokens: 16 history tokens and 68 current-frame tokens (a 2×2 grid and an 8×8 grid). The current-frame scoring grid distributes each coarse token's mass uniformly over its 4×4 support on the 8×8 grid and adds the detailed map. Top-fraction masks retain $\lceil 64f \rceil$ cells for fraction f.

StreamVLN scores each reference in a separate teacher-forced forward pass. Its attribution averages heads and the final four decoder layers after taking gradient-based magnitudes, normalizes over visual tokens at each action position, and then averages the four position-specific maps. Its current-frame grid has 14×14 cells, retaining 49 at 25%. For both models, raw attention averages the two fixed-reference comparisons, while gradient-based maps compare the suffix-matched references.

Single-step measurements. JanusVLN and FutureNav use separate windows for generated actions and each reference trajectory. Generated-action windows contain at most four decisions and three action executions. Fixed-reference windows share GT observations across suffixes and use independent language and visual memory. Action scores multiply native content-token probabilities from a shared autoregressive prefix tree. NLL uses full-vocabulary probabilities, while KL uses the four-action conditional distribution. FutureNav uses native eager attention. JanusVLN uses FlashAttention 2 for generation and scoring and a separate eager decoder pass for attribution. Raw attention is compared at aligned GT states. Corresponding-GT gradients compare each instruction's first reference action at the shared fork view.

Heatmap display. Figures 3 and 4 overlay full maps bicubically interpolated to RGB resolution, sharing a zero-to-joint-maximum scale between each model's suffix conditions. IoU uses native-grid top-25% masks before interpolation.

Action-pair grouping for StreamVLN. For Table 4, we retain StreamVLN contexts whose ref-

erence sequences $G_{h,A}$ and $G_{h,B}$ differ at the first action. We group contexts by these first GT actions, ignoring their A/B order. All metrics retain the full four-action horizon.

Attention by action position for StreamVLN. For Table 5, we compute current-frame attention IoU separately at each action-query position in the shared 3,252 StreamVLN contexts. We keep the observations, reference, and preceding target actions fixed across suffixes and average the two fixed-reference comparisons at each position.

At positions 2 and 3, we also compare the IoU drop from the preceding position between contexts whose GT sequences first differ at that position and contexts whose GT sequences still agree through it. The former groups have larger mean drops by 0.0110 and 0.0137, respectively.

B.2 Source Annotations and Recombination

FGR2R. FGR2R automatically splits R2R instructions into sub-instructions and uses human annotators to align them with subpaths (Hong et al., 2020, Section 4 and Appendix A.1). Annotators watch a trajectory walkthrough and explore the environment in an interactive viewer to mark each subpath's start and end viewpoints. Workers are screened using 15 reference samples, and the interface checks path-partition constraints, including shared endpoints between adjacent subpaths.

Landmark-RxR. Landmark-RxR annotators read full English RxR instructions and follow their trajectories in a 3D viewer, marking subpath endpoints and extracting the corresponding text (He et al., 2021, Section 3.1). Its guidelines require sub-instructions to match their subpaths, express independent meanings, and clearly describe their endpoints. A second annotator reviews the labels and corrects inaccuracies.

Recombination at aligned boundaries. NAVBIND recombines text together with its annotated route segments (Section 4). The prefix P_h remains paired with the replayed route up to v, and the suffix S_s retains its source route after v. Both source instructions have an annotated subpath boundary at v, and their routes share the directed approach $u \to v$. Thus, the text and route are joined at the same boundary, preserving each segment's source alignment. We then construct $G_{h,s}$ along the suffix's branch from the replayed fork state using the evaluated model's native action increments.

Review of recombined instructions. One author reviewed 100 instructions sampled uniformly from the 3,252-context evaluation set, comparing the complete recombined text with its source instructions and extracted spans. The author rated 98 acceptable for fluency and all 100 acceptable for semantic coherence, including clarity of references. The two fluency-flagged instructions were retained unchanged. Appendix A.4 provides the full review protocol and results.

B.3 First-Decision Diagnostics

We add to the main results in Table 1 by comparing the first action each model chooses under the two instructions. This comparison uses the observations available at the fork, before the agent takes another action. From the shared 3,252 contexts, we retain 2,447 in which the two reference sequences begin with different actions for every model. Same first action reports the percentage of contexts with identical generated first actions under the two instructions, using native decoding settings. Both+ and KL follow Section 5, restricted to the first decision.

Model	RxR-CE SR (%) ↑	Same first action (%) ↓	Both+ (%) ↑	KL ↑
Uni-NaVid	56.1	94.12	38.86	0.0153
StreamVLN	54.4	39.89	87.94	2.2897
JanusVLN	51.4	57.34	77.97	0.5453
FutureNav	54.5	49.20	79.61	0.7664

Table 9: First-decision diagnostics on 2,447 shared contexts.

In Table 9, Uni-NaVid has the highest reported RxR-CE val-unseen SR. However, StreamVLN changes its first action most often (lowest Same first action) and has the highest Both+ and KL. All three first-decision diagnostics give the same ordering: StreamVLN, FutureNav, JanusVLN, and Uni-NaVid. This first-decision comparison supports the main finding that reported navigation success and local instruction diagnostics yield different model orderings.

B.4 Val-Unseen Diagnostics

We supplement the main results in Table 1 with the same diagnostics on 385 contexts from 194 source pairs. Both source instructions in each pair

	RxR-CE	Behavior	Fixed-target likelihood			Visual overlap (IoU)		
Model	SR (%) ↑	Same seq. (%) ↓	$\bar{d}$ ↑	Both+ (%) ↑	KL ↑	Attn. ↓	Grad. ↓	Attn.×Grad. ↓
Uni-NaVid	56.1	84.42	0.0052	25.71	0.0123	0.854	0.442	0.773
StreamVLN	54.4	30.65	0.7567	77.14	0.6877	0.595	0.413	0.440
JanusVLN	51.4	24.68	0.4169	75.84	0.4996	0.723	0.755	0.695
FutureNav	54.5	17.92	0.5049	76.62	0.7099	0.725	0.776	0.690

Table 10: Diagnostics on 385 contexts with both source instructions from val-unseen. Metric definitions and reported SR follow Table 1.

Selection	Train N	Val-unseen N	Train gap	Val-unseen gap	Difference
All selected contexts	2,367	385	+0.1341	+0.0611	+0.0731
Complete source pairs	2,334	382	+0.1370	+0.0675	+0.0694
Identical branch targets	736	126	+0.1062	+0.0442	+0.0620

Table 11: StreamVLN original-pairing advantage under nested subset checks. Gap follows Table 6. Difference subtracts the val-unseen gap from the train gap.

come from the original val-unseen split, and all four models use the same contexts.

Table 10 shows that Uni-NaVid has the highest reported SR but the smallest $\bar{d}$ and Both+. StreamVLN has the largest $\bar{d}$ and Both+ and the lowest IoU across all three visual measures. FutureNav changes its generated sequences most often and has the largest KL. Despite nearly identical reported SR, StreamVLN and FutureNav differ in how much the instructions increase the likelihood of the required actions.

B.5 Cross-History Evaluation

Subset construction. Starting from the shared 3,252-context set, we retain the 1,604 source pairs with both histories, giving 3,208 candidate pair-suffix comparisons. For each fixed suffix S_s, we require $G_{A,s} = G_{B,s}$ with exactly four actions. We apply this equality test at both native action scales and intersect the retained pair-suffix identifiers. The resulting subset contains 1,124 comparisons from 880 source pairs. Both suffixes qualify for 244 pairs, and one suffix qualifies for the remaining 636. Selection uses reference actions alone. Each retained pair-suffix comparison receives equal weight in Table 7.

Measurements and state alignment. Each history uses its source-matched prefix and the same suffix, with action distributions and visual maps evaluated on the shared reference. Using the four-action distributions defined in Section 5, we compute $D_{\mathrm{SKL}}(q^{(j)}_{A,s,s}, q^{(j)}_{B,s,s})$ and average over the four positions and retained comparisons. Visual measures compare current-frame top-25% masks on each model's native scoring grid (Appendix B.1).

B.6 Measurement Checks

Original-pairing advantage for StreamVLN. We repeat the comparison in Section 7.6 on two nested subsets within each paired source split. Complete source pairs retain both histories from the shared 3,252-context set, so each suffix appears once in an original pairing and once in a recombined pairing. The identical-branch-target subset additionally requires $G_{A,A} = G_{B,A}$ and $G_{A,B} = G_{B,B}$, with all sequences containing four actions at StreamVLN's native action scale. The two branches remain distinct within each context. This makes the multiset of reference sequences identical between original and recombined conditions. Selection uses source labels and reference actions, and each retained context receives equal weight. Table 11 reports the NLL gaps and their train-minus-val-unseen differences.

Statistical analysis. For Uni-NaVid and StreamVLN, we assess the sensitivity of the NavBind diagnostics in Table 1 to sample composition. Reused trajectories make contexts dependent. We draw 2,000 bootstrap resamples of the 1,648 shared source pairs with replacement (seed 0), keeping each pair's histories, suffix conditions, and target comparisons together. Table 12 reports the 95% percentile confidence intervals for these main-table metrics. The resampling uses the fixed seed-0 model outputs.

Metric	Uni-NaVid		StreamVLN	
	Estimate	95% CI	Estimate	95% CI
Same seq. (%)	85.79	[84.53, 87.03]	32.41	[30.67, 34.29]
NLL advantage $\bar{d}$	0.0127	[0.0089, 0.0167]	0.8360	[0.8066, 0.8655]
Both+ (%)	29.15	[27.51, 30.79]	80.38	[78.80, 81.95]
Symmetric KL	0.0124	[0.0105, 0.0143]	0.8233	[0.7895, 0.8595]
Attention IoU	0.860	[0.856, 0.864]	0.602	[0.597, 0.607]
Gradient IoU	0.454	[0.448, 0.460]	0.420	[0.414, 0.425]
Attn.×Grad. IoU	0.773	[0.768, 0.777]	0.456	[0.451, 0.462]

Table 12: Sensitivity to sample composition for the NAVBIND diagnostics in Table 1, resampling the same 1,648 source pairs for both models.

Attribution thresholds. Table 13 evaluates Uni-NaVid's three visual measures with top-response fractions from 10% to 30%.

Top fraction	Attention	Gradient	Attn.×Grad.
10%	0.864	0.338	0.743
20%	0.857	0.418	0.764
25%	0.860	0.454	0.773
30%	0.866	0.496	0.789

Table 13: Uni-NaVid mean IoU at different mask sizes on the shared 3,252 contexts. Attention averages both fixed references, while gradient-based maps compare the corresponding GTs.

B.7 Completed-Prefix Analysis

Completed-prefix token groups. We tokenize each complete instruction with StreamVLN's native tokenizer without added special tokens, then identify prefix tokens from character offsets and the stored boundary. Joining whitespace is excluded from the prefix span. Across all 6,504 matched conditions, no token spans content from both sides of the boundary, and each context has the same prefix token count under both suffixes.

Table 3 uses sample tertile boundaries, retaining ties in the lower group. In ascending token-count order, the groups contain 1,107, 1,081, and 1,064 contexts, covering the shared 3,252-context set. For the additional comparison below, prefix proportion is the prefix token count divided by the total token count of the original instruction $I_{h,h}$. The same tertile rule gives three groups of 1,084 contexts each before matching.

Matched prefix groups for StreamVLN. Table 14 compares the lowest and highest tertiles of each prefix measure using one-to-one matching without replacement within dataset and split. Matching covariates are traveled distance, historical action count, route progress, and the token count of the source suffix S_h in $I_{h,h}$. Traveled distance sums consecutive replay-state displacements up to the fork. Route progress divides this distance by the full source replay's length.

For distance, action count, and suffix token count, the larger value plus one must be at most 1.15 times the smaller value plus one. Route progress must differ by at most 0.05. Among eligible partners, matching minimizes the sum of squared normalized differences. We log-transform the first three covariates after adding one, divide their differences by $\log(1.15)$, and divide the progress difference by 0.05. Low-prefix contexts with fewer eligible partners are matched first, and context identifiers resolve ties. Matching uses no model outputs. The count and proportion comparisons retain 200 and 106 matched pairs, respectively. Each context retains its original within-context suffix comparison, and table entries average these metrics within each low or high group.

Prefix	Group	N	Same seq. (%) ↓	$\bar{d}$ ↑	KL ↑
Count	Low	200	31.50	0.9073	0.9366
	High	200	41.00	0.7204	0.6575
Proportion	Low	106	31.13	0.8674	0.8531
	High	106	49.06	0.5566	0.4826

Table 14: StreamVLN's lowest and highest prefix tertiles after matching navigation and remaining-text characteristics. N counts contexts per group.

B.8 A Route Fork Through Three Diagnostics

Figure 4 shows a fork where S_A asks the agent to walk straight between two chairs and S_B asks it to turn right toward the sofa. Uni-NaVid generates four left turns under both suffixes. Its mean matching-suffix NLL advantage is 0.0051, and its attention×gradient top-response regions largely overlap (IoU 0.778). StreamVLN changes from

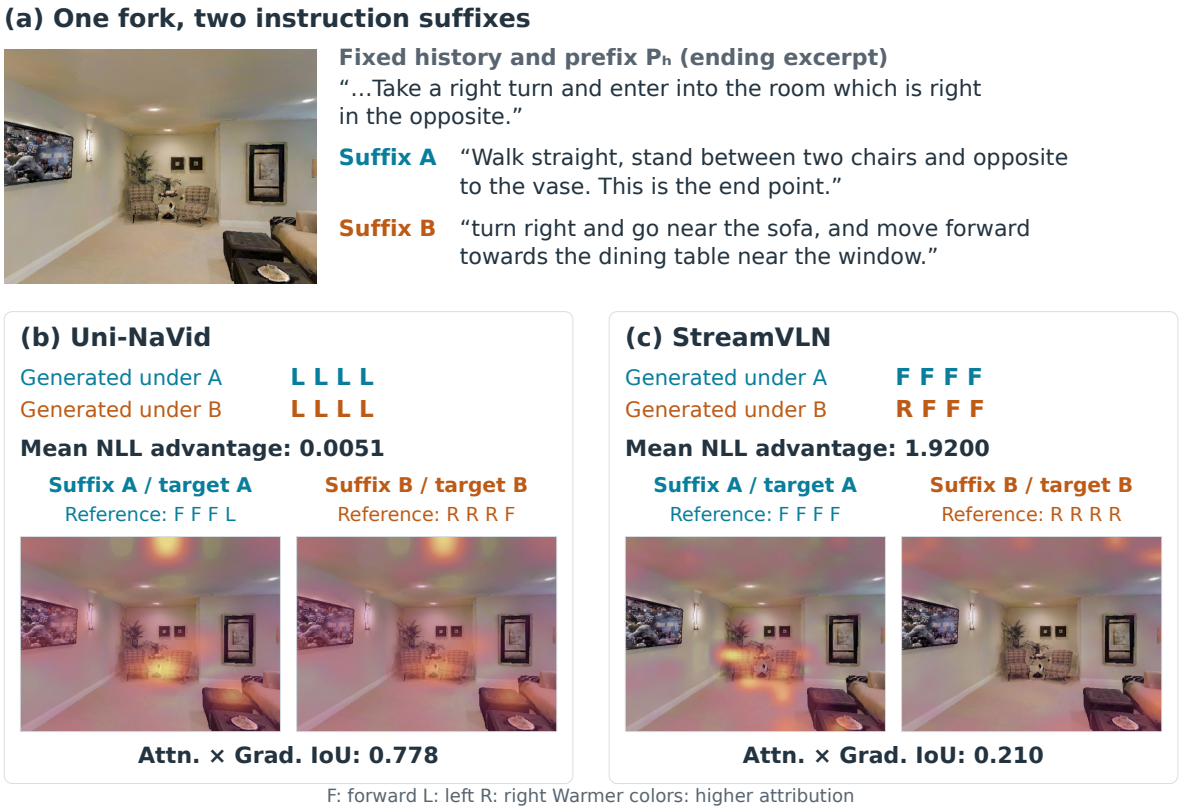


Figure 4: Responses to two suffixes with a fixed history and prefix at one RxR-CE val-unseen fork. Full attention×gradient heatmaps explain each suffix's corresponding reference, with a shared scale within each model. IoU uses native-grid top-25% masks. The prefix is excerpted and both suffixes are shown in full.

four forward actions under S_A to a right turn followed by three forward actions under S_B. The matching suffix lowers NLL for both reference targets, by 1.0912 and 2.7487, giving a mean advantage of 1.9200. Its corresponding-target regions also change more (IoU 0.210). This example connects the three diagnostics at a single fork: StreamVLN's output change accompanies target-aligned likelihood shifts and spatial redistribution of visual attribution, while Uni-NaVid's output invariance coincides with small likelihood shifts and greater attribution overlap.