VehicleMemBench

An Executable Benchmark for Multi-User Long-Term Memory in In-Vehicle Agents

MULTI-USER | LONG-TERM | EXECUTABLE

Yuhao Chen[1] Yi Xu[1] Xinyun Ding[2] Xiang Fang[2] Shuochen Liu[1] Luxi Lin[3] Qingyu Zhang[4] Ya Li[2] Quan Liu[2]* Tong Xu[1]*

[1] University of Science and Technology of China [2] iFLYTEK Research
[3] Xiamen University [4] University of Chinese Academy of Sciences * Corresponding authors

COLM 2026
MAIN CONFERENCE

Code | Dataset

Can an agent turn evolving, multi-user memories into the right vehicle actions?

500	50	81.78	92.8k	23	111
executable queries	scenarios · 3 users each	events / history (avg.)	tokens / history (avg.)	vehicle modules	executable APIs

01 Why memory is hard

Vehicle agents must recover who prefers what, when it applies, and the exact setting to execute.

<20% ESM without memory

All 12 evaluated models · Table 6

Five preference dynamics

- **Preference conflict** — Select the active user's preference.
- **Conditional constraint** — Apply only when its trigger holds.
- **Coreference resolution** — Resolve aliases and indirect references.
- **State shift** — Prioritize the latest valid preference.
- **Error correction** — Remove a previously incorrect memory.

Exact values matter

Qwen3-Max + KV Store · Table 1

73.74 Field F1 → **66.26** Value F1

Agents can identify the right attribute yet miss its precise value. Memory must preserve exact settings.

Field–value gap: 7.48 pp (gold: 2.18 pp).

02 Benchmark & evaluation

Interleaved histories → memory → executable tasks

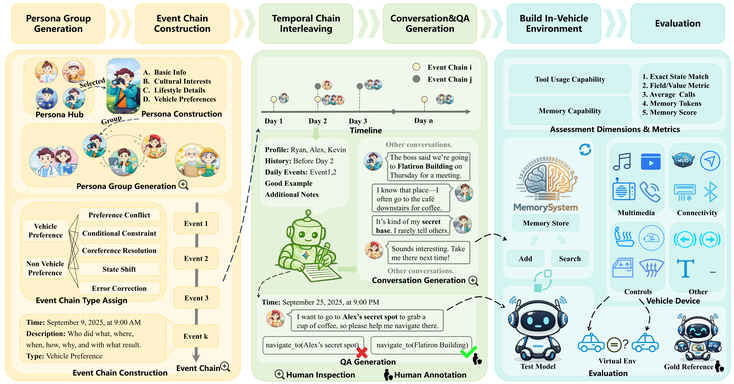


Figure 1 · Event-driven construction and evaluation.

Score the resulting vehicle state

Agent + memory → **Predicted state**

same initial state

Reference calls → **Target state**

= ?

State comparison · no LLM or human scoring

ESM: exact match of the full final state.
Field / value F1: right attributes and exact values.

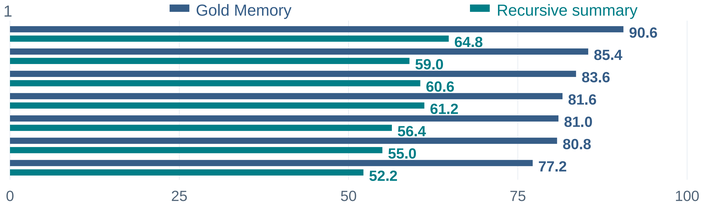
Memory Score = ESM(auto) / ESM(gold)

Gold Memory supplies correct preferences to isolate tool use.

03 Findings and design lessons

1 Memory quality is the main bottleneck

ESM (%) · selected models · Table 1



* Gemini-3-Pro = Gemini-3-Pro-Preview.

−25.8 pp Gemini: 90.60% → 64.80% ESM with summary memory.
63.9% of analyzed errors are memory errors. (Sec. 4.3)

Prioritize memory consolidation and retrieval.

Upgrades can leave memory performance unchanged: GLM-4.7 / GLM-5, both 55.0% ESM with summary.

2 Preference validity depends on context and time

Qwen3-Max · recursive summary
51.96 vs. 67.79 ESM
Conditions vs. preference conflict

Gemini-3-Pro* · state shift
49.38 → 70.79 ESM
KV Store → Supermemory

Bind preferences to their conditions and temporal validity.

3 Domain-specific summaries remain competitive

Memory system	ESM (%)	Tokens / retrieval
Supermemory	65.79	960.79
Recursive summary	**64.80**	**303.25**
LightMem	60.72	170.19
Key Value Store	57.37	29.25
MemOS	56.11	444.91
Mem0	52.10	171.16
Memobase	29.40	993.49

Gemini-3-Pro-Preview · Table 2

+0.99 pp ESM

3.17× retrieval tokens

Supermemory vs. recursive summary

Retrieve compact facts that support exact actions.

TAKEAWAY

Build memory that binds users, conditions, time and exact values to executable actions.

Scope: synthetic histories in a simulator; manually verified. Results: per-configuration point estimates. History tokens: GPT-4o tokenizer.