Experiments

Workspace Memory Research

Target: Clawd-style workspace (`agents.defaults.workspace` , default `~/.openclaw/workspace`) where “memory” is stored as one Markdown file per day (`memory/YYYY-MM-DD.md`) plus a small set of stable files (e.g. `memory.md` , `SOUL.md`).

This doc proposes an **offline-first** memory architecture that keeps Markdown as the canonical, reviewable source of truth, but adds **structured recall** (search, entity summaries, confidence updates) via a derived index.

Why change?

The current setup (one file per day) is excellent for:

- “append-only” journaling
- human editing
- git-backed durability + auditability
- low-friction capture (“just write it down”)

It’s weak for:

- high-recall retrieval (“what did we decide about X?”, “last time we tried Y?”)
- entity-centric answers (“tell me about Alice / The Castle / warelay”) without rereading many files

opinion/preference stability (and evidence when it changes)

time constraints (“what was true during Nov 2025?”) and conflict resolution

Design goals

Offline: works without network; can run on laptop/Castle; no cloud dependency.

Explainable: retrieved items should be attributable (file + location) and separable from inference.

Low ceremony: daily logging stays Markdown, no heavy schema work.

Incremental: v1 is useful with FTS only; semantic/vector and graphs are optional upgrades.

Agent-friendly: makes “recall within token budgets” easy (return small bundles of facts).

North star model (Hindsight × Letta)

Two pieces to blend:

1. **Letta/MemGPT-style control loop**

   keep a small “core” always in context (persona + key user facts)

   everything else is out-of-context and retrieved via tools

   memory writes are explicit tool calls (append/replace/insert), persisted, then re-injected next turn

2. **Hindsight-style memory substrate**

   separate what’s observed vs what’s believed vs what’s summarized

   support retain/recall/reflect

   confidence-bearing opinions that can evolve with evidence entity-aware retrieval + temporal queries (even without full knowledge graphs)

Proposed architecture (Markdown source-of-truth + derived index)

Canonical store (git-friendly)

Keep `~/.openclaw/workspace` as canonical human-readable memory.

Suggested workspace layout:

```
~/.openclaw/workspace/
  memory.md                    # small: durable facts + preferences (core-ish)
  memory/
    YYYY-MM-DD.md              # daily log (append; narrative)
  bank/                        # “typed” memory pages (stable, reviewable)
    world.md                   # objective facts about the world
    experience.md              # what the agent did (first-person)
    opinions.md                # subjective prefs/judgments + confidence + eviden
    entities/
      Peter.md
      The-Castle.md
      warelay.md
      ...
```

Notes:

- **Daily log stays daily log**. No need to turn it into JSON.
- The `bank/` files are **curated**, produced by reflection jobs, and can still be edited by hand.
- `memory.md` remains “small + core-ish”: the things you want Clawd to see every session.

Derived store (machine recall)

Add a derived index under the workspace (not necessarily git tracked):

```
~/.openclaw/workspace/.memory/index.sqlite
```

Back it with:

- SQLite schema for facts + entity links + opinion metadata
- SQLite **FTS5** for lexical recall (fast, tiny, offline)
- optional embeddings table for semantic recall (still offline)

The index is always **rebuildable from Markdown**.

Retain / Recall / Reflect (operational loop)

Retain: normalize daily logs into "facts"

Hindsight's key insight that matters here: store **narrative, self-contained facts**, not tiny snippets.

Practical rule for `memory/YYYY-MM-DD.md` :

- at end of day (or during), add a `## Retain` section with 2–5 bullets that are:
  - narrative (cross-turn context preserved)
  - self-contained (standalone makes sense later)
  - tagged with type + entity mentions

Example:

```
## Retain
- W @Peter: Currently in Marrakech (Nov 27-Dec 1, 2025) for Andy’s birthday.
- B @warelay: I fixed the Baileys WS crash by wrapping connection.update handlers
- O(c=0.95) @Peter: Prefers concise replies (&lt;1500 chars) on WhatsApp; long co
```

Minimal parsing:

- Type prefix: `W` (world), `B` (experience/biographical), `O` (opinion), `S` (observation/summary; usually generated)
- Entities: `@Peter`, `@warelay`, etc (slugs map to `bank/entities/*.md`)
- Opinion confidence: `O(c=0.0..1.0)` optional

If you don’t want authors to think about it: the reflect job can infer these bullets from the rest of the log, but having an explicit `## Retain` section is the easiest “quality lever”.

Recall: queries over the derived index

Recall should support:

- **lexical**: “find exact terms / names / commands” (FTS5)
- **entity**: “tell me about X” (entity pages + entity-linked facts)
- **temporal**: “what happened around Nov 27” / “since last week”
- **opinion**: “what does Peter prefer?” (with confidence + evidence)

Return format should be agent-friendly and cite sources:

- `kind` (`world|experience|opinion|observation`)
- `timestamp` (source day, or extracted time range if present)
- `entities` (`["Peter","warelay"]`)
- `content` (the narrative fact)
- `source` (`memory/2025-11-27.md#L12` etc)

Reflect: produce stable pages + update beliefs

Reflection is a scheduled job (daily or heartbeat `ultrathink`) that:

- updates `bank/entities/*.md` from recent facts (entity summaries)
- updates `bank/opinions.md` confidence based on reinforcement/contradiction
- optionally proposes edits to `memory.md` ("core-ish" durable facts)

Opinion evolution (simple, explainable):

- each opinion has:
  - statement
  - confidence `c ∈ [0,1]`
  - last_updated
  - evidence links (supporting + contradicting fact IDs)
- when new facts arrive:
  - find candidate opinions by entity overlap + similarity (FTS first, embeddings later)
  - update confidence by small deltas; big jumps require strong contradiction + repeated evidence

CLI integration: standalone vs deep integration

Recommendation: **deep integration in OpenClaw**, but keep a separable core library.

Why integrate into OpenClaw?

- OpenClaw already knows:
  - the workspace path (`agents.defaults.workspace`)
  - the session model + heartbeats

  - logging + troubleshooting patterns

You want the agent itself to call the tools:

```
openclaw memory recall "…" --k 25 --since 30d
openclaw memory reflect --since 7d
```

Why still split a library?

- keep memory logic testable without gateway/runtime
- reuse from other contexts (local scripts, future desktop app, etc.)

Shape: The memory tooling is intended to be a small CLI + library layer, but this is exploratory only.

“S-Collide” / SuCo: when to use it (research)

If “S-Collide” refers to **SuCo (Subspace Collision)**: it’s an ANN retrieval approach that targets strong recall/latency tradeoffs by using learned/structured collisions in subspaces (paper: arXiv 2411.14754, 2024).

Pragmatic take for `~/.openclaw/workspace`:

- **don’t start** with SuCo.
- start with SQLite FTS + (optional) simple embeddings; you’ll get most UX wins immediately.
- consider SuCo/HNSW/ScaNN-class solutions only once:
  - corpus is big (tens/hundreds of thousands of chunks)
  - brute-force embedding search becomes too slow
  - recall quality is meaningfully bottlenecked by lexical search

Offline-friendly alternatives (in increasing complexity):

- SQLite FTS5 + metadata filters (zero ML)

- Embeddings + brute force (works surprisingly far if chunk count is low)
- HNSW index (common, robust; needs a library binding)
- SuCo (research-grade; attractive if there’s a solid implementation you can embed)

Open question:

- what’s the **best** offline embedding model for “personal assistant memory” on your machines (laptop + desktop)?
  - if you already have Ollama: embed with a local model; otherwise ship a small embedding model in the toolchain.

Smallest useful pilot

If you want a minimal, still-useful version:

- Add `bank/` entity pages and a `## Retain` section in daily logs.
- Use SQLite FTS for recall with citations (path + line numbers).
- Add embeddings only if recall quality or scale demands it.

References

- Letta / MemGPT concepts: “core memory blocks” + “archival memory” + tool-driven self-editing memory.
- Hindsight Technical Report: “retain / recall / reflect”, four-network memory, narrative fact extraction, opinion confidence evolution.
- SuCo: arXiv 2411.14754 (2024): “Subspace Collision” approximate nearest neighbor retrieval.

‹ Telegram Allowlist Hardening

Model Config Exploration ›