Machine Psychology with the Minimal Sensorimotor Core (MSC)

MSC Documentation Synthesis

November 6, 2025

Contents

1 Introduction

The Minimal Sensorimotor Core (MSC) is a compact implementation of a Non-Axiomatic Reasoning System (NARS) controller. It processes perceptual events, forms temporal implications, and selects operations under resource constraints. The repository provides both the reasoning engine and a battery of experiments that replicate the "Machine Psychology" tasks described in the

accompanying manuscript. This document consolidates the architectural notes, data-structure references, testing instructions, and experimental findings contained throughout the `docs/` directory into a single manuscript.

2 System Architecture

2.1 Build and Entry Points

- `./build.sh` compiles every `src/*.c` module into the `MSC` binary with dead-code elimination.
- Running `./MSC` prints a comprehensive usage string. The binary accepts regression flags (`--list-tests`, `--test`, `--run-all-tests`) and experiment exporters (`--exp1-csv`, `--exp2-csv`, `--exp3-csv`).
- Demo entry points (`pong`, `pong2`, `testchamber`, `alien`, `simple_discriminations`) are dispatched directly from `src/main.c`.

2.2 Reasoning Cycle

One call to `MSC_Cycles(1)` performs:

1. **Input injection:** Belief and goal events, constructed by `MSC_AddInputBelief` or `MSC_AddInputGoal`, enter FIFO buffers with fresh truth values and stamps.
2. **Concept activation:** The newest event activates (or creates) its concept and updates usage priorities.
3. **Link mining:** `Cycle_ReinforceLink` induces temporal implications of the form `<(&/, pre, op, +dt) =/> post>`.
4. **Goal propagation and decision-making:** Goal spikes backchain through stored implications, while `Decision_Suggest` evaluates candidate operations using `Truth_Expectation`. Operations whose expectation surpasses `DECISION_THRESHOLD` are executed.
5. **Cleanup:** Processed spikes are reset and the concept attention queue is rebuilt.

3 Core Data Structures

3.1 Terms, Truth, and Stamps

- **Terms** are fixed-length (two-character) symbol sequences (`src/Term.h`). `Encode_Term` maps strings to unique term IDs.
- **Truth values** couple frequency and confidence. Expectation, revision, deduction, induction, intersection, and projection are implemented in `src/Truth.c`.
- **Stamps** track the evidential base of events and implications, preventing overlap and enabling revision (`src/Stamp.c`).

3.2 Events and FIFOs

- Each **Event** stores a term, truth, stamp, occurrence time, and optional operation ID.
- Belief and goal queues (`FIFO`) maintain recent sequences up to `MAX_SEQUENCE_LEN` (currently 3), enabling longer antecedents such as sample–comparison–operation chains.

3.3 Concept Memory and Implications

- Concepts hold current belief and goal spikes, implication tables per operation, and usage metrics.
- Implications `(pre => post)` include temporal offsets and debug labels. `Table_AddAndRevise` handles insertion and revision with stamp-aware overlap checks.
- A binary-heap priority queue (`PriorityQueue`) yields limited-resource attention by promoting recently used concepts.

3.4 Decision Layer

- **Motor babbling** (`Decision_MotorBabbling`) introduces exploration when no implication meets the decision threshold.
- **Best candidate selection** (`Decision_BestCandidate`) projects goal truth through available implications and selects the highest-utility operation.
- **Assumption of failure** (negative confirmation) decreases the desirability of operations that fail to achieve the goal, stabilising learning in changing conditions.

4 Implementation Parameters

Table 1 summarises the most relevant compile-time or global parameters.

Parameter	Default Meaning
`MAX_SEQUENCE_LEN`	3 event elements per sequence
`FIFO_SIZE`	Depth of belief/goal buffers (1024)
`TABLE_SIZE`	Max implications per concept/operation (32)
`PROPAGATION_THRESHOLD`	Minimum expectation before backchaining (0.501)
`PROPAGATION_ITERATIONS`	Depth of goal propagation (5)
`DECISION_THRESHOLD`	Expectation required to execute an operation (0.6)
`MOTOR_BABBLING_CHANCE`	Exploration probability (0.2, often reduced during experiments)

Table 1: Key MSC configuration parameters.

5 Testing and Demonstrations

MSC bundles deterministic regression helpers plus interactive demos (`docs/testing-guide.md`).

- Regression tests cover stamp handling, FIFO rollover, priority queues, implication revision, multi-step procedure learning, and the machine-psychology experiments.
- Demos showcase online learning in Pong variants, an interactive "Test Chamber," a Space-Invaders-style "Alien" scenario, and a simple discrimination exercise.
- The CLI flags `--list-tests`, `--test <name>`, and `--run-all-tests` orchestrate the suite. Experiment exporters stream per-trial CSVs.

6 Machine Psychology Experiments

The repository reproduces the three tasks from the Machine Psychology manuscript. CSV outputs live in `docs/`, and block accuracy plots may be embedded directly in LaTeX (paths relative to this manuscript).

6.1 Experiment 1: Simple Discrimination

- **Design:** Baseline (3 blocks), Training (3), Testing (3). Agent must choose `^left` when stimulus `A1` is on the left and `^right` otherwise.
- **Outcomes:** Baseline ≈50% accuracy; Training attains 100% by the third block; Testing remains 100%. Expectations for the correct implications converge near 0.89.
- **Reference:** Figure 1 plots block accuracy.

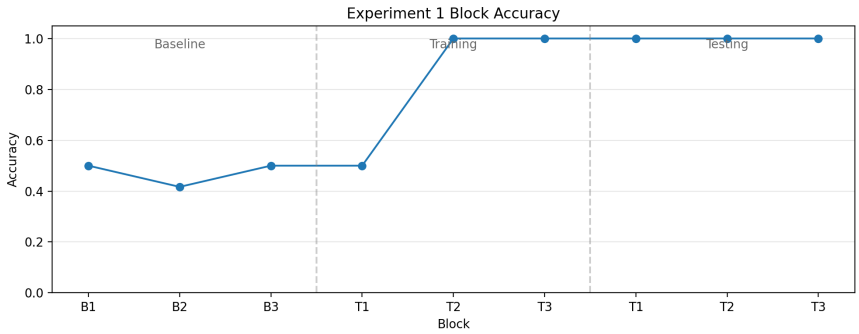

Figure 1: Experiment 1 block accuracy.

6.2 Experiment 2: Changing Contingencies

- **Design:** Baseline (2 blocks), Training 1 (4), Testing 1 (2), Training 2 (reversal, 9 blocks), Testing 2 (2). The mapping flips after Testing 1, requiring relearning.
- **Outcomes:** MSC recovers to perfect accuracy by block 7 of Training 2, maintaining 100% through Testing 2. Expectation logs show the original implications decreasing while the reversed hypotheses accumulate evidence.
- **Reference:** Figure 2 depicts the per-block trajectory.

6.3 Experiment 3: Conditional Discrimination

- **Design:** Baseline (3 blocks), Training (6), Testing (3). Each trial presents a sample stimulus ($A1$ or $A2$) plus comparisons ($B1$, $B2$) on left/right. Successful behaviour requires using the joint context (sample + comparison) before acting.
- **Implementation Highlights:** The experiment harness feeds sample and comparison beliefs sequentially, allowing `Term_Sequence` to build two-element antecedents. Expectations are recorded for both the full sample–comparison implications and the shorter comparison-only rules.

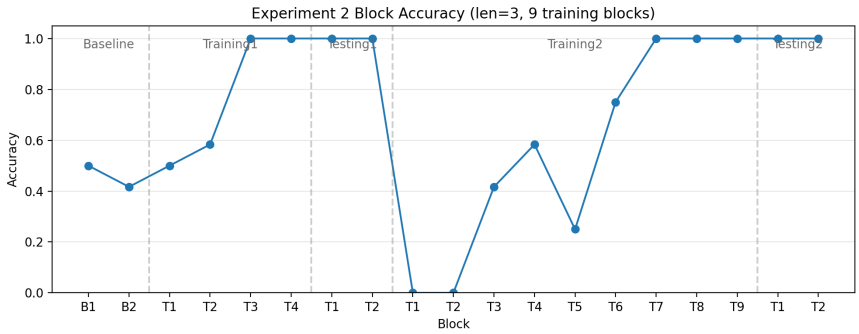

Figure 2: Experiment 2 block accuracy with nine Training 2 blocks.

- **Outcomes:** Training achieves 100% accuracy by block 5 and sustains it through Testing. Sample–comparison implications settle near expectation 0.84 with high confidence, while comparison-only rules remain lower (frequency ≈ 0.74).
- **Reference:** Figure 3 shows block accuracy.

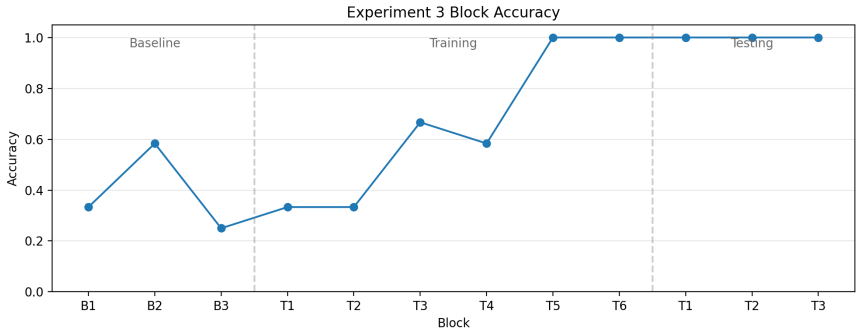

Figure 3: Experiment 3 block accuracy.

7 Observations and Practical Notes

- Raising `MAX_SEQUENCE_LEN` to 3 is essential for Experiment 3; it enables MSC to learn antecedents like `<((A1, B1), ^left) =/> G>`.
- Decision heuristics naturally favour longer antecedents once their expectation exceeds that of shorter, ambiguous rules.
- Motor babbling is throttled (`0.2`) during experiments. Once high-confidence implications exist, exploration rarely fires, allowing deterministic evaluation.
- CSV columns include the trial context and expectation snapshots; the plotting script can visualise any experiment via `scripts/plot_block_accuracy.py`.

8 Reproducibility Checklist

1. Compile with `./build.sh`.
2. Export trials: `./MSC --exp1-csv docs/exp1_trials.csv`, `./MSC --exp2-csv docs/exp2_trials.csv`, `./MSC --exp3-csv docs/exp3_trials.csv`.
3. Generate plots: `venv/bin/python scripts/plot_block_accuracy.py --csv <file> --output <png>`.
4. Optionally rerun regression shortcuts: `./MSC --test exp1`, `--test exp1_training`, `--test exp2`, `--test exp3`.

9 Future Directions

- Extending inference beyond NAL-7/8 primitives—for example, by reintroducing sequence-goal deduction—could support richer planning tasks.
- Increasing `MAX_SEQUENCE_LEN` further would allow MSC to capture longer temporal contexts but would require auditing the FIFO and table memory footprints.
- Tight integration with demos (Pong, TestChamber) could provide online visualisations of implication strength, facilitating real-time debugging.

10 Conclusion

MSC delivers a minimal yet expressive platform for studying non-axiomatic reasoning in sensorimotor environments. The three Machine Psychology experiments confirm that the system can (i) learn simple discriminations, (ii) adapt to changing contingencies, and (iii) solve conditional discriminations, all while maintaining transparent, inspectable data structures. This manuscript captures the current understanding of the codebase and serves as a foundation for future extensions.