DTU Compute
Department of Applied Mathematics and Computer Science

Neuro-Symbolic Planning for Enhancing Coherence and Believability in LLM-Driven Agents

Master Thesis

Neuro-Symbolic Planning for Enhancing Coherence and Believability in LLM-Driven Agents

Master Thesis
February, 2026

By
Alexandre Comas Gispert, Jeremi Wojciech Ledwon

Cover photo: Vibeke Hempler, 2012
Published by: DTU, Department of Applied Mathematics and Computer Science, Richard Petersens Plads, Build. 324, 2800 Kgs. Lyngby Denmark
https://www.compute.dtu.dk/
ISSN: [0000-0000] (electronic version)
ISBN: [000-00-0000-000-0] (electronic version)

ISSN: [0000-0000] (printed version)
ISBN: [000-00-0000-000-0] (printed version)

Approval

This master's thesis was prepared at the Department of Applied Mathematics and Computer Science (DTU Compute) at the Technical University of Denmark, under the supervision of Professor Thomas Bolander. The thesis was carried out over a five-month period, from 1 September 2025 to 1 February 2026, and accounts for 30 ECTS credits, in partial fulfilment of the requirements for the degree of Master of Science in Engineering, Computer Science and Engineering.

It is assumed that the reader has a basic knowledge in the areas of computer science and artificial intelligence.

Alexandre Comas Gispert, Jeremi Wojciech Ledwon - s233148, s232952

.....

Signature

.....

Date

Abstract

[TO BE COMPLETED at the end of the thesis writing process]

Acknowledgements

[TO BE COMPLETED at the end of the thesis writing process]

Contents

1 Introduction

1.1 Background and Context

Believable computational agents that simulate human behavior enable diverse applications: immersive virtual environments and social simulations [1], rehearsal spaces for practicing interpersonal communication, prototyping tools for testing social scenarios, training systems for rare yet difficult social situations, and platforms for validating social science theories. Non-player characters in open-world games can navigate complex social relationships; virtual assistants and social robots can interact more naturally; cognitive models can inform human-computer interaction design. The common requirement is an architecture that produces behavior consistent with past experience, reacts believably to environmental changes, and maintains coherence over extended interactions.

Large Language Model (LLM)-driven generative agents can simulate complex, human-like behavior in virtual worlds, games, and social scenarios by leveraging commonsense reasoning and natural language capabilities [1]. However, purely neural approaches produce logical inconsistencies: agents attempt impossible actions (opening nonexistent doors), violate temporal constraints (simultaneous commitments), or pursue conflicting goals, undermining believability [2].

Park et al. [1] demonstrated emergent social dynamics through memory streams, reflection, and hierarchical planning. Yet their planning component lacks mechanisms to verify logical consistency or enforce environmental constraints, producing plans that are contextually plausible but violate hard constraints or exhibit temporal inconsistencies.

Neuro-symbolic AI can address this by combining neural generation with symbolic planning [3].

Note on the Use of AI

AI-assisted tools, including large language models, were used during the preparation of this thesis for tasks such as literature review, code development, writing assistance, and figure generation. A detailed account of AI usage, including specific tools, tasks, and the extent of human oversight, is provided in Chapter 7.

1.2 Problem Statement

Existing agent architectures face a fundamental tradeoff: purely symbolic systems ensure logical consistency but lack flexibility and commonsense reasoning, while purely neural LLM-driven agents offer adaptability but produce logically inconsistent plans.

Research question: How can a neuro-symbolic planning framework improve the coherence and believability of LLM-driven generative agents?

1.3 Research Aim and Objectives

Aim: To develop and evaluate a hybrid neuro-symbolic planning system in which an LLM generates a hierarchical plan, the plan actions and environment constraints are formalized in PDDL, and a symbolic validator verifies logical consistency, identifies constraint violations, and guides iterative plan refinement.

Objectives:

1. Reimplement the generative agents architecture [1] with a modular, extensible codebase that supports integration of symbolic planning components.
2. Design and implement a PDDL-based validator that formalizes environmental constraints, detects planning violations, and outputs actionable diagnostic feedback.
3. Develop visualization and explanation tools that make agent plans, constraint violations, and repair proposals inspectable to researchers and evaluators.
4. Evaluate the system using (a) quantitative metrics (constraint violation rates, plan success rates, and repair efficiency) comparing a baseline hierarchical planner against the neuro-symbolic approach, and (b) qualitative human evaluation of perceived believability.

1.4 Methodological Overview

This study extends the generative agents architecture [1] by replacing hierarchical planning with a neuro-symbolic framework. The LLM generates hierarchical plans and PDDL action schemas; a symbolic validator detects constraint violations. Evaluation combines:

- **Qualitative assessment**: Within-subjects user study comparing perceived believability of agent behaviors from both systems.
- **Quantitative metrics**: Constraint violation counts and rates on matched scenarios comparing (i) baseline hierarchical planning and (ii) neuro-symbolic planning with validator-guided revision. Metrics include violations per 100 actions, plan success rates, and repair efficiency.

1.5 Scope and Limitations

This project focuses on simulation environments with deterministic action effects and complete observability. Real-world robotics introduces sensing uncertainty and physical dynamics beyond our scope. Evaluation constraint adherence, and perceived believability rather than real-time performance or scalability to large multi-agent systems.

1.6 Thesis Structure

The remainder of the thesis is organized as follows:

- **Chapter 2: Theoretical Background** — Establishes core concepts (LLMs, agents, planning paradigms including PDDL) and reviews relevant literature.
- **Chapter 3: Methodology** — Describes the system design, experimental setup, and evaluation protocols. Details the symbolic validator architecture and the within-subjects user study for assessing believability and constraint adherence.
- **Chapter 4: Results** — Reports quantitative constraint-violation metrics and qualitative believability findings from the user study.
- **Chapter 5: Discussion** — Interprets results, situates findings within the literature, and discusses limitations and implications for agent design.
- **Chapter 6: Conclusion and Future Work** — Summarizes contributions and suggests directions for future research.
- **Chapter 7: Use of AI in this Thesis** — Declaration of AI tools used in the thesis preparation.

2 Theoretical Background

This chapter establishes conceptual foundations and reviews prior work motivating this thesis. It defines core concepts (LLMs, agents, planning paradigms including PDDL), reviews LLM-driven generative agents with emphasis on the seminal Generative Agents paper [1], and examines hybrid neuro-symbolic approaches combining LLM flexibility with symbolic planning guarantees [4, 3]. We emphasize the challenge of maintaining coherence (adherence to environmental constraints) and believability (human-perceived realism) in LLM-driven agents.

Example 2.1 (Running Example: Student NPC)**.** Throughout this chapter, we illustrate concepts using a running example: an NPC simulating a university student managing academic and social commitments. The student must coordinate coursework (attending lectures, completing assignments), part-time work (café shifts), and social activities (meeting friends, attending events) while respecting temporal constraints (no overlapping commitments), location constraints (cannot be in two places simultaneously), and environmental rules (must be enrolled in a course to attend its lectures).

2.1 Core Concepts and Definitions

2.1.1 Large Language Models (LLMs)

Definition 2.1 (Large Language Model)**.** A *large language model* (LLM) is a transformer-based sequence predictor trained on large corpora to estimate conditional token distributions $P(x_t \mid x_{<t})$ [5, 6]. LLMs generate text autoregressively by sampling from next-token distributions, using self-attention to integrate information across prompts, examples, and tool inputs.

LLMs exhibit instruction following, few-shot generalization, and approximate commonsense reasoning [7, 8]. Critically, they perform pattern-conditioned statistical inference, not deductive logical inference with formal guarantees. LLMs can generate structured outputs (PDDL schemas, task specifications) from natural language domain descriptions [3, 4], but these outputs require external validation for logical consistency and constraint adherence.

In this thesis, the LLM (i) generates PDDL schemas from natural language domain descriptions, and (ii) proposes high-level goals and task decompositions grounded in agent memory and social context. The symbolic validator then checks whether proposed plans are executable given domain constraints, producing failure diagnostics (unsatisfied preconditions, invariant violations, temporal/resource conflicts, unsolvability). The LLM produces and sequences action plans; the validator enforces feasibility.

2.1.2 Agents and Believability

Definition 2.2 (Agent)**.** An *agent* receives percepts and produces actions via sensors and actuators, implementing a mapping $f : P^* \to A$ from percept histories P^* to actions A [9].

In simulated environments and games, non-player characters (NPCs) prioritize producing behavior that human observers find plausible, consistent, and engaging rather than maximizing numerical rewards [10, 11].

Definition 2.3 (Believability)**.** *Believability*—the "illusion of life"—is the human-perceived realism of agent behavior: whether actions align with character goals, personality, knowledge, and social norms [2, 11].

Constraining agents with realistic physical and environmental limits increases perceived believability [2, 12]. Park et al. showed that LLM-driven agents with memory, reflection, and planning were rated more believable than human crowdworkers in controlled evaluations [1]. Xiao et al. formalize metrics such as Consistency and Robustness for profile-grounded simulation [13].

Definition 2.4 (Coherence)**.** *Coherence* is the causal and temporal consistency of behavior: whether actions are feasible, properly ordered, and do not contradict prior commitments or environmental constraints [14, 15].

Coherence is measured through constraint adherence: violations of environmental invariants, temporal overlaps, resource limits, and unsatisfied action preconditions.

Example 2.2 (Coherence Violation)**.** In our running example (example 2.1), a coherent agent must not schedule overlapping commitments (attending two simultaneous classes) or attempt impossible actions (submitting an assignment before completing it). A purely LLM-based planner might generate "attend lecture at 10:00" and "work café shift 09:00 to 12:00" without detecting the temporal conflict.

We adopt the NPC perspective where believability is primary. We hypothesize that coherence—enforced through symbolic planning—is necessary but not sufficient for believability. The neuro-symbolic approach aims to maintain naturalistic, context-aware LLM behavior while eliminating logical inconsistencies that undermine realism.

2.1.3 Planning and PDDL

Definition 2.5 (Classical Planning Problem)**.** A *classical planning problem* is a tuple $\langle S, A, T, I, G\rangle$: S is a state set, A is an action set, $T : S \times A \to S$ is a transition function, $I \subseteq S$ is the initial state set, and $G \subseteq S$ is the goal state set. A *plan* is a finite action sequence $\pi = \langle a_1, \ldots, a_n\rangle$ such that executing π from any $s \in I$ via T reaches some $g \in G$ [16].

Actions have preconditions (conditions required before execution) and effects (state changes produced), defining causal structure.

Definition 2.6 (PDDL)**.** *PDDL (Planning Domain Definition Language)* is the standard formalism for encoding planning problems [17, 16]. A PDDL specification consists of two files:

1. **Domain file**: Defines predicates (representing the state space S) and action schemas (representing actions A with typed parameters, preconditions, and effects). This captures universal aspects of the planning problem.
2. **Problem file**: Defines objects, the initial state s_I, and goal conditions G for a specific problem instance.

PDDL extensions support temporal planning (durative actions with start/end conditions and continuous effects) and resource constraints (numeric fluents tracking quantities like time or energy) [18].

Example 2.3 (Student Coursework Domain)**.** For a student managing coursework, predicates might include:

- `(enrolled ?s - student ?c - course)`
- `(completed ?s - student ?a - assignment)`
- `(at-location ?s - student ?l - location)`

An action schema `attend-lecture` would specify preconditions (student must be enrolled, lecture must be scheduled, student must be at the lecture hall) and effects (student gains knowledge of lecture content, updates current location).

These extensions are particularly relevant for simulated agents whose actions have durations and consume resources (e.g., working a shift at a café consumes several hours, traveling between locations requires time proportional to distance).

2.1.4 Symbolic Planning

Symbolic planning uses explicit, compositional representations to algorithmically search for valid plans [19, 17]. Key strengths:

1. **Explainability**: Plans are sequences of named actions with explicit preconditions and effects, enabling causal trace inspection.
2. **Constraint enforcement**: Planners guarantee that plans satisfy all preconditions, avoid violated invariants, and respect temporal and resource bounds.
3. **Optimality**: Many planners find optimal or bounded-suboptimal solutions under well-defined cost models.

These properties directly address coherence: if behavior is synthesized via a symbolic planner, environmental constraints are satisfied by construction.

The primary limitation is **authoring cost**: PDDL domains require manual specification of predicates, actions, preconditions, and effects, which is brittle and labor-intensive for open-ended environments. Symbolic planning also struggles with commonsense reasoning and social nuance unless explicitly encoded [18].

2.1.5 LLM-Based Planning

LLM-based planning uses language models to generate action sequences or subgoal decompositions directly from text [20, 21]. LLMs can draft plausible multi-step procedures via chain-of-thought prompting [7], propose alternatives, and adapt plans to soft preferences without formal domain models.

However, LLM-generated plans lack correctness guarantees and can:

- Omit necessary preconditions (opening a door without checking accessibility),
- Violate environmental invariants (simultaneous presence in two locations),
- Drift temporally (forgetting earlier commitments when context windows truncate),
- Hallucinate actions or states not grounded in the environment [13].

For believability-centric NPCs, these failures manifest as broken commitments, physical impossibilities, and social incoherence. Park et al.'s Generative Agents exhibited emergent social behaviors but lacked constraint enforcement, relying on LLM prompt engineering to maintain temporal coherence heuristically [1].

2.1.6 Hierarchical Planning

Hierarchical task networks (HTN) decompose high-level tasks into ordered or partially ordered subtasks until primitive actions are reached, using methods encoding admissible refinements and constraints [22, 23]. Hierarchy supports abstraction, reuse, and tractable search.

LLMs approximate hierarchical planning by proposing outlines, subgoals, and steps in natural language [7]. Park et al.'s agents generate daily plans with morning, afternoon,

and evening blocks containing embedded tasks, resembling HTN decomposition without explicit HTN semantics or validation [1].

Formal HTN or temporal PDDL planners can validate such decompositions, ensuring high-level commitments refine into feasible, non-overlapping primitive actions. In our running example, “complete coursework this week” might decompose into “attend lectures,” “complete assignments,” and “study for exam,” with temporal constraints preventing overlaps and ensuring deadlines are met.

2.1.7 Neuro-Symbolic Systems for Planning

Neuro-symbolic systems combine learned (sub-symbolic) components with symbolic representations and reasoning to achieve both flexibility and guarantees [24, 25]. Common integration patterns:

1. **LLM-propose/symbolic-plan**: The LLM generates candidate schemas and plans; a symbolic planner forms a sequence of actions that satisfy the constraints [4].
2. **Iterative refinement**: Plans are critiqued by symbolic validators or planners, and feedback improves LLM proposals [26, 27].
3. **Shared world models**: A symbolic state representation is updated from neural perception and queried for decisions.

Recent work explores these patterns for PDDL generation. Tantakoun et al. survey approaches where LLMs construct PDDL domain and problem files from natural language, with symbolic planners validating and executing [3]. Huang et al. propose a pipeline in which multiple LLM instances generate diverse candidate PDDL action schemas, which are filtered for semantic coherence and validated by a symbolic planner to ensure domain solvability [4]. [1]

This thesis follows the iterative refinement pattern with a symbolic verifier. The LLM generates tasks and action sequences grounded in agent memory; these are translated into PDDL schemas and validated by a symbolic planner to enforce temporal, causal, and resource constraints. Validation failures produce diagnostic feedback that can be returned to the LLM for iterative repair, preserving naturalistic behavior while ensuring logical coherence.

Example 2.4 (Neuro-Symbolic Validation of Action Decomposition)**.** The LLM receives a daily task “work café shift (09:00 to 12:00)” and decomposes it into fine-grained actions: “walk to café,” “unlock door,” “turn on lights,” “serve customers,” “clean tables.” The PDDL validator checks each action:

- Preconditions (door must exist and agent must have key before unlocking; lights require door to be unlocked and agent inside),
- Location constraints (agent must be at café to unlock door; cannot serve customers while at storage room),
- Temporal ordering (cannot turn on lights before unlocking door; cannot clean tables before serving customers ends).

Violations are flagged with diagnostics (e.g., “action `unlock_door` fails: precondition `(has-key agent cafe-key)` unsatisfied in initial state; agent inventory empty”).

[1] Unlike the present work, Huang et al. use the symbolic planner not only for validation but also to generate complete plans once a consistent domain schema has been identified.

2.2 Literature Review

This section reviews work on neuro-symbolic planning for LLM-driven agents, focusing on the Generative Agents paper that motivates our system and hybrid approaches combining LLMs with symbolic planning.

2.2.1 Generative Agents: Interactive Simulacra of Human Behavior (Park et al., 2023)

Park et al. introduced generative agents: LLM-driven NPCs simulating believable human behavior in Smallville, a sandbox town environment [1]. The architecture comprises:

1. **Memory Stream**: Time-stamped observations (own actions, others' actions, environment events), retrieved by weighted combination of recency (exponential decay), importance (LLM-rated 1 to 10), and relevance (embedding cosine similarity).
2. **Reflection**: Periodic synthesis triggered when importance scores exceed a threshold (150). The LLM generates questions about recent experiences, produces insights with citations, and creates reflection trees (observations → reflections → meta-reflections).
3. **Planning**: Top-down recursive decomposition into day-level plans (5 to 8 chunks), hour-level plans, and 5 to 15 minute action plans. Agents dynamically re-plan when circumstances change, with the LLM deciding whether to continue or react to new observations.

Emergent behaviors in a 25-agent, two-day simulation:

- **Information diffusion**: Mayoral candidacy news spread from 1 agent to 8 (32%); Valentine's party invitation spread to 13 (52%).
- **Relationship formation**: Social network density increased from 0.167 to 0.74.
- **Coordination**: Isabella planned a party, invited agents, decorated the café; 5 of 12 invited agents attended at the correct time and location.

Evaluation: Controlled study with 100 human participants ranking agents via TrueSkill. Full architecture (memory + reflection + planning) achieved $\mu = 29.89, \sigma = 0.72$, significantly outperforming ablations and human crowdworkers ($\mu = 22.95, \sigma = 0.69$) with effect size $d = 8.16$ (p < 0.001). Interview questions assessed self-knowledge, memory recall, future plans, reactive decisions, and reflections.

Failure modes:

- Memory retrieval errors,
- Hallucination (embellishing details not in memory),
- Overly formal dialogue,
- Over-cooperation.

Relevance: Park et al. demonstrated that LLM agents with memory, reflection, and planning achieve high believability. However, the system lacks explicit symbolic grounding: no formal model of time, resources, or environmental constraints. Temporal coherence is maintained heuristically through textual schedules that can drift or produce overlaps. Actions are not verified against preconditions beyond ad hoc checks, limiting transparency when context windows shift or commitments are forgotten. This motivates our neuro-symbolic approach of integrating PDDL-based validation to detect and repair constraint violations.

2.2.2 LLMs as Planning Modelers (Tantakoun et al., 2025)

Tantakoun et al. survey how LLMs construct and refine automated planning models rather than directly perform planning [3]. They review approximately 80 papers, positioning LLMs as tools for extracting planning models to support reliable planners, addressing the limitation that LLMs struggle with long-horizon planning requiring structured reasoning.

Taxonomy of three paradigms:

1. **LLMs-as-Heuristics**: Enhance planner search efficiency.
2. **LLMs-as-Planners**: Directly generate action sequences or plan proposals.
3. **LLMs-as-Modelers**: Construct PDDL domain and problem files (survey focus).

Within LLMs-as-Modelers:

Task Modeling (approximately 30 papers): LLMs generate PDDL problem files from goal specifications, using few-shot prompting [28] to chain-of-thought techniques [29]. Well-explored but relies on detailed predicate specification.

Domain Modeling (approximately 15 papers): LLMs generate PDDL domain files (predicates and action schemas), more challenging than task specification. Approaches include generate-test-critique loops incrementally building components [30]. Single-domain generation risks misalignment with human expectations due to natural language ambiguity.

Hybrid Modeling (approximately 15 papers): LLMs generate both domain and problem files. Systems use human-in-the-loop approaches with anomaly detection [31]. Coordinating both introduces complexity; linear pipelines risk cascading errors.

Key findings:

1. LLMs generate syntactically valid PDDL but struggle with semantic consistency.
2. Iterative refinement with symbolic planner feedback improves model quality.

This grounds our approach: we position the LLM as a PDDL schema generator (domain modeling) with symbolic validator feedback and iterative refinement.

2.2.3 Planning in the Dark: LLM-Symbolic Pipeline without Experts (Huang et al., 2024)

Huang et al. [4] propose an LLM-symbolic planning pipeline eliminating expert intervention in action schema generation and validation. Natural-language task descriptions are inherently ambiguous; under reasonable assumptions, a single LLM instance has less than 0.0001% probability of generating a solvable action-schema set, but combining multiple LLM instances raises this to over 95% [4].

Three-step architecture:

1. **Diverse Schema Library**: Deploy N LLM instances with high temperature to generate complete action schema sets for M actions. Aggregate into a library with approximately N^M possible combinations.
2. **Semantic Coherence Filtering**: Use sentence encoders to compute cosine similarity between natural language descriptions $E(Z(\alpha))$ and generated schemas $E(\hat{\alpha})$. Apply Conformal Prediction (CP) to calculate threshold $\hat{q}$ at confidence level $1-\epsilon$, filtering schemas below threshold. This reduces combinations from N^M to $\prod_{i=1}^{M} m_i$ where m_i is passing schemas per action. Fine-tune encoder with triplet loss using hard negatives (manipulated schemas with predicate swaps, negations, removals).

3. **Plan Generation and Ranking**: Feed solvable schema sets into symbolic planner (DUAL-BWFS). Rank generated plans by cumulative semantic similarity.

Key findings:

- Layman (ambiguous) descriptions yield 2.35 times more distinct solvable schema sets than detailed expert descriptions (8039 vs. 3419 with 10 LLMs, no CP), reflecting diverse valid interpretations.
- With 10 LLM instances, probability of generating at least one solvable schema set exceeds 95%.
- CP filtering reduces combinations to 3.3% of original (1051/31,483) while increasing solvable ratio from 10.9% to 23.0%.
- Human evaluators ranked pipeline-generated plans (mean 2.97) significantly higher than Tree-of-Thought baselines (3.58); gold standard ranked 1.79.
- Pipeline successfully solved Sussman Anomaly (requiring interleaved subgoal handling); direct LLM approaches (GLM, GPT-3.5, GPT-4o) failed by attempting linear goal ordering.

Relevance: Huang et al. demonstrate that LLM diversity and semantic validation produce solvable PDDL schemas without expert intervention. Their pipeline validates schema feasibility via symbolic planners, ensuring coherence. We adapt this: rather than generating diverse schema libraries, we use iterative refinement with planner feedback to improve schema quality.

2.2.4 Other Neuro-Symbolic Planning Approaches

Additional work combines LLMs with formal planning:

Robotics and grounded planning: SayCan pairs LLM language grounding with value estimates over affordances to select feasible actions in robotic manipulation [20].

Iterative refinement loops: LLM+P and related frameworks prompt an LLM to propose high-level plans, check them with a PDDL planner, and iterate until valid [26, 27, 29]. Critique-and-repair loops such as Reflexion and Tree-of-Thought add self-evaluation and search, while symbolic constraints prune or guide search [32, 33].

Temporal planning integration: Extensions incorporate duration and resource checks to prevent overlaps and enforce deadlines [34].

2.2.5 Summary of Insights and Research Focus

The literature suggests three converging insights:

1. **LLM planning produces human-like behavior but lacks long-horizon coherence** due to missing symbolic grounding, context window limitations, and hallucination [1, 13].
2. **Neuro-symbolic methods offer structure and guarantees** by validating or synthesizing plans against explicit models of actions, time, and resources [4, 3].
3. **Believability and coherence should be assessed jointly**: constraint adherence is necessary for plausibility (coherence), but human evaluation is required to confirm realism (believability) [1, 13].

3 Methodology

3.1 Experimental Setup

[To be completed: overview of simulation environment, agent initialization, scenario design, and logging infrastructure.]

3.2 System Design

Our neuro-symbolic planning system extends a re-implementation of the Generative Agents architecture [1] with modifications enabling controlled comparison between purely neural planning (baseline) and neuro-symbolic planning (our approach).

3.2.1 System Architecture

The implementation transforms the original monolithic Generative Agents codebase into a modular, service-oriented architecture:

1. **Repository Layer**: Abstracts external dependencies (LLM APIs, file storage) behind interfaces. `LLMRepository` supports both OpenAI (production) and mock providers (testing). `EnvironmentRepository` abstracts world state persistence.

2. **Service Layer**: Encapsulates cognitive capabilities in swappable interfaces:

   - `PlanningService`: Daily planning and task decomposition
   - `DialogueService`: Conversation generation
   - `PerceptionService`: Environment observation and memory retrieval
   - `ReflectionService`: Memory summarization
   - `EnvironmentService`: Spatial navigation and object interaction

3. **Orchestration Layer**: The simulation loop consumes services through interfaces, configured via environment variables (`LLM_PROVIDER`, `PLAN_MODULE`) controlling which implementations run.

Key Design Principle: The `PlanningService` abstraction enables side-by-side comparison of baseline (LLM-only hierarchical planning) and neuro-symbolic planning by ensuring both share identical environment state, memory retrieval, and LLM infrastructure. Only the planning logic differs, isolating the independent variable.

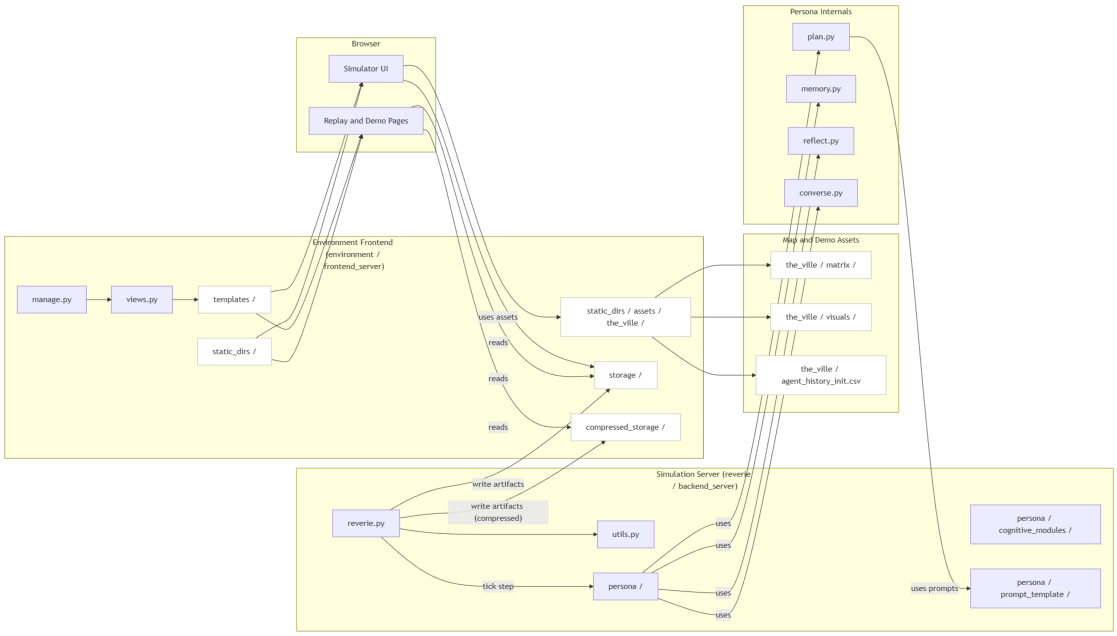

Figure 3.1: Original monolithic architecture from the Generative Agents codebase [1], showing tightly coupled components without clear separation of concerns.

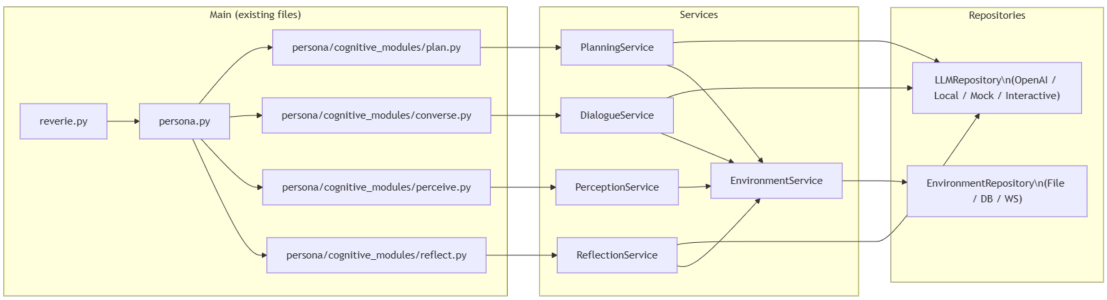

Figure 3.2: Refactored service-oriented architecture with Repository, Service, and Orchestration layers. The `PlanningService` abstraction enables controlled comparison between baseline and neuro-symbolic planning implementations.

3.2.2 Neuro-Symbolic Planning Pipeline

[section is subject to changes after final implementation]

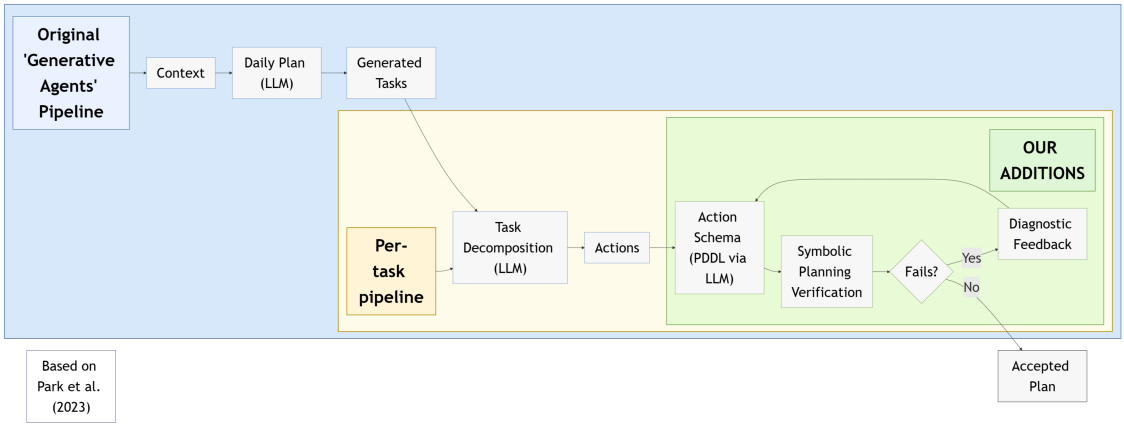

Figure 3.3: Neuro-symbolic planning pipeline showing the three-stage LLM-propose/symbolic-validate framework. The pipeline integrates task generation, action decomposition, and PDDL schema validation with iterative repair feedback.

We implement a three-stage LLM-propose/symbolic-validate framework [35, 3, 4]:

1. **Task Generation**: The LLM generates daily tasks from memory, grounding them in wants, needs, and commitments [1].
2. **Action Decomposition**: Tasks decompose into atomic actions with environment parameters. Example: "complete assignment" → `open-laptop`, `navigate-to-file`, `work-on-document(90min)`, `submit-via-portal`.
3. **Schema Generation & Validation**: The LLM generates PDDL schemas (preconditions, effects, durations) for each action. A symbolic validator checks causal consistency, temporal feasibility, resource limits, and environmental invariants. Violations trigger diagnostic feedback (e.g., "unsatisfied precondition `(at-location student hall)`") for iterative LLM repair until constraints satisfy or iteration budget exhausts.

3.3 Quantitative Evaluation: Constraint Violation Analysis

[To be completed: automated evaluation comparing the hierarchical planning baseline against our validator-augmented system. The validator will automatically detect and flag constraint violations such as attempting to use items that are not available, scheduling overlapping activities, violating location constraints, or executing actions whose preconditions are not satisfied. Metrics will include violation counts at day-level and action-level, violation rates per 100 actions, and success rates after optional validator-guided repair rounds.]

3.4 User Study: Believability Evaluation

This section describes the human-subjects study testing whether our approach improves perceived believability of agent behavior compared to the baseline Generative Agents architecture [1]. We focus on believability of *actions* rather than only personalities or conversations.

3.4.1 Objectives and Hypotheses

Two primary hypotheses:

- **H1 (overall believability)**: Participants judge agents powered by our method as more believable overall than the baseline in matched scenarios.

- **H2 (action believability)**: For the same scenario, participants flag fewer actions as "unbelievable" in our method than in the baseline.

Secondary outcomes: (i) perceived causal coherence when the high-level plan is visible, and (ii) free-text reasons participants provide when deeming actions unbelievable (used for qualitative error analysis) [2, 12, 36, 13].

3.4.2 Conditions

Two within-subject conditions on the same simulated world and character seeds:

1. **Baseline (GA)**: Faithful re-implementation of Generative Agents [1].
2. **Ours (Neuro-symbolic)**: Proposed system with symbolic planning and consistency checks integrated into deliberation and action selection.

Each participant evaluates both conditions on the same character and scenario to enable within-subject comparison. Order is counterbalanced to reduce presentation effects.

3.4.3 Participants

Target 10 to 15 adult participants recruited from the university community and online platforms. Inclusion criteria: English proficiency. We run an initial pilot (3 to 4 participants) to validate timing and interface, then proceed to the main study. All participants provide informed consent and can withdraw anytime without penalty.

3.4.4 Materials and Stimuli

The stimulus for each condition is a replay of a single random character's day. To focus on action believability, we present:

- time-lapse *video replay* of the agent acting in the world (controllable playback speed, pause/seek);
- optional overlay with *high-level plan* (intentions and sub-goals) and *low-level action log*; and
- UI controls to mark an action as unbelievable ("thumbs down"), provide a short reason, and continue.

Replays cover the same scenario (e.g., two simulated in-game days) and use the same character profile and randomness seed across conditions, so any variation is attributable to agent architecture (baseline vs. ours) rather than scenario noise.

3.4.5 Procedure

Each session (approximately 30 minutes):

1. **Introduction.** Scripted briefing introduces the task and believability as coherence, plausibility, and consistency within world rules [12].
2. **Practice.** Participants complete a 2 to 3 minute tutorial on the interface using a neutral example not used in the main study.
3. **Condition A.** Watch the replay, freely scrub, and mark unbelievable actions. For each mark, add a short explanation (optional but encouraged).
4. **Condition B.** Repeat with the other planner. Order varies across participants; assignment is double-blind.
5. **Summary ratings.** For each condition: (i) overall believability rating (7-point Likert), (ii) perceived causal coherence rating (7-point Likert), and (iii) preference judgment (forced-choice which was more believable and why).

We record duration until finished and whether the plan overlay was opened, to analyze how explanations affect believability judgments.

3.4.6 Measures

We operationalize believability with participant-reported and behavior-linked measures. Higher values indicate higher believability unless noted.

Primary Outcomes

1. **Overall believability (Likert).** Single item per condition on a 7-point scale: 1 "not at all believable", 4 "moderately believable", 7 "extremely believable". Prompt: "How believable was the agent's behaviour overall in this replay?"

2. **Action-level unbelievable rate (event-normalized).** Participants flag any action as unbelievable. Let F be flagged action events and A be *atomic actions* viewed (from action log restricted to watched timestamps). The rate is

$$r_{\text{unbel}} = \frac{F}{A} \times 100\,, \tag{3.1}$$

   expressed as flags per 100 atomic actions. Multiple flags within 2 seconds around the same atomic action merge into one.

3. **Pairwise preference.** Forced-choice: "Which replay was more believable overall?" (Baseline vs. Ours).

Secondary Outcomes

- **Causal coherence (Likert).** 7-point rating: 1 "not coherent", 7 "highly coherent".
- **Plan adherence (Likert).** 7-point rating of alignment between visible high-level plan and observed actions.
- **Unbelievable-action categories (coded).** Free-text reasons are open-coded into categories: goal inconsistency, environment rule violation, temporal implausibility, social norm violation, low-level control failure. Two independent coders label a stratified sample ($\geq 30\%$ of flags); disagreements are adjudicated and inter-rater agreement (Cohen's κ) is reported.

Logged covariates (for analysis, not outcomes): condition order, scenario ID, participant playback time, number of overlay openings, and self-reported prior experience with simulations/games. These are used as covariates in exploratory models and to check for order effects.

3.4.7 Data Quality and Exclusion

Sessions are excluded if participants fail an attention check (simple comprehension question about the replay), leave more than half the session unanswered, or complete in less than one-third of median time. We pre-register exclusion rules prior to data collection.

3.4.8 Ethics

The study involves minimal risk. No personal data beyond demographics is collected; all logs are anonymized and stored on encrypted drives.

4 Results

4.1 Experimental Results

Presentation of experimental findings.

4.2 Analysis

Analysis of the results.

5 Discussion

5.1 Interpretation of Results

Discussion and interpretation of findings.

5.2 Limitations

Discussion of study limitations.

6 Conclusion

6.1 Summary

Summary of the thesis work.

6.2 Future Work

Directions for future research.

7 Use of AI in this Thesis

We used artificial intelligence as an assistive tool for three purposes: (i) drafting and editing text, (ii) literature discovery and screening, and (iii) coding assistance. Human authorship and responsibility are preserved throughout; all outputs were reviewed, verified, and, when needed, rewritten by the authors.

7.1 Activities

Writing Large language models were used to improve clarity and structure of human-written drafts.

Literature AI assisted in query refinement, screening, data extraction, and summarization, consistent with evidence that such use can improve the efficiency of systematic reviews when disclosed and auditable [37]. Tools included NotebookLM and ChatGPT with Deep Search.

Coding GitHub Copilot was used to suggest code, refactorings, and edits to agent definitions and prompts; changes were reviewed and tested.

7.2 Workflow Summary

The workflow for AI-assisted work followed these steps:

1. Human draft or code first.
2. Targeted AI pass with explicit constraints.
3. Human verification of facts, citations, and behavior; edits and tests as needed.
4. Iterate selectively.

7.3 Transparency and Compliance

We follow DTU guidance on responsible AI use. We disclose the role of AI tools, preserve key system prompts that materially influenced outputs, together with representative examples and model versions, and do not attribute authorship to AI.

7.4 Limitations and Verification

- Factual statements and references suggested by AI were checked against primary sources.
- Copyright and licensing were verified for any third-party material.
- For reproducibility, model versions and key settings are documented in Appendix A.

Bibliography

[1] Joon Sung Park et al. “Generative Agents: Interactive Simulacra of Human Behavior”. In: *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. UIST ’23. New York, NY, USA: Association for Computing Machinery, Oct. 29, 2023, pp. 1–22. ISBN: 979-8-4007-0132-0. DOI: 10.1145/3586183.3606763. URL: https://dl.acm.org/doi/10.1145/3586183.3606763 (visited on 09/10/2025).

[2] Joseph Bates. “The Role of Emotion in Believable Agents”. In: *Communications of the ACM* 37.7 (July 1994), pp. 122–125. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/176789.176803. URL: https://dl.acm.org/doi/10.1145/176789.176803 (visited on 10/20/2025).

[3] Marcus Tantakoun, Xiaodan Zhu, and Christian Muise. *LLMs as Planning Modelers: A Survey for Leveraging Large Language Models to Construct Automated Planning Models*. Mar. 22, 2025. DOI: 10.48550/arXiv.2503.18971. arXiv: 2503.18971 [cs]. URL: http://arxiv.org/abs/2503.18971 (visited on 09/10/2025). Pre-published.

[4] Sukai Huang, Nir Lipovetzky, and Trevor Cohn. *Planning in the Dark: LLM-Symbolic Planning Pipeline without Experts*. Sept. 24, 2024. DOI: 10.48550/arXiv.2409.15915. arXiv: 2409.15915 [cs]. URL: http://arxiv.org/abs/2409.15915 (visited on 11/04/2025). Pre-published.

[5] Ashish Vaswani et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems*. 2017. arXiv: 1706.03762. URL: https://arxiv.org/abs/1706.03762.

[6] Tom B. Brown et al. “Language Models Are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems*. 2020. arXiv: 2005.14165. URL: https://arxiv.org/abs/2005.14165.

[7] Jason Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. Jan. 10, 2023. DOI: 10.48550/arXiv.2201.11903. arXiv: 2201.11903 [cs]. URL: http://arxiv.org/abs/2201.11903 (visited on 11/10/2025). Pre-published.

[8] Takeshi Kojima et al. “Large Language Models Are Zero-Shot Reasoners”. 2022. arXiv: 2205.11916. URL: https://arxiv.org/abs/2205.11916.

[9] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Fourth edition, global edition. Prentice Hall Series in Artificial Intelligence. Boston: Pearson, 2022. 1 p. ISBN: 978-1-292-40113-3 978-1-292-40117-1.

[10] Michael Mateas. “An Oz-Centric Review of Interactive Drama and Believable Agents”. In: *Artificial Intelligence Today*. Ed. by Michael J. Wooldridge and Manuela Veloso. Vol. 1600. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 297–328. ISBN: 978-3-540-66428-4 978-3-540-48317-5. DOI: 10.1007/3-540-48317-9_12. URL: http://link.springer.com/10.1007/3-540-48317-9_12 (visited on 11/10/2025).

[11] A Bryan Loyall. “Believable Agents: Building Interactive Personalities”. In: ().

[12] Anton Bogdanovych, Tomas Trescak, and Simeon Simoff. “What Makes Virtual Agents Believable?” In: *Connection Science* 28.1 (Jan. 2, 2016), pp. 83–108. ISSN: 0954-0091. DOI: 10.1080/09540091.2015.1130021. URL: https://doi.org/10.1080/09540091.2015.1130021 (visited on 10/20/2025).

[13] Yang Xiao et al. *How Far Are LLMs from Believable AI? A Benchmark for Evaluating the Believability of Human Behavior Simulation*. Version 2. June 15, 2024. DOI: 10.48550/arXiv.2312.17115. arXiv: 2312.17115 [cs]. URL: http://arxiv.org/abs/2312.17115 (visited on 10/20/2025). Pre-published.

[14] R Michael Young. “An Overview of the Mimesis Architecture: Integrating Intelligent Narrative Control into an Existing Gaming Environment”. In: ().

[15] M. O. Riedl and R. M. Young. “Narrative Planning: Balancing Plot and Character”. In: *Journal of Artificial Intelligence Research* 39 (Sept. 29, 2010), pp. 217–268. ISSN: 1076-9757. DOI: 10.1613/jair.2989. URL: https://jair.org/index.php/jair/article/view/10669 (visited on 11/10/2025).

[16] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: Theory and Practice*. Elsevier, May 3, 2004. 665 pp. ISBN: 978-1-55860-856-6. Google Books: eCj3cKC_3ikC.

[17] Drew McDermott et al. “PDDL - The Planning Domain Definition Language”. In: (Oct. 1, 1998). URL: https://www.cs.cmu.edu/~mmv/planning/readings/98aips-PDDL.pdf (visited on 11/10/2025).

[18] Patrik Haslum et al. *An Introduction to the Planning Domain Definition Language*. Morgan & Claypool Publishers, Apr. 2, 2019. 189 pp. ISBN: 978-1-62705-737-0. Google Books: bA6QDwAAQBAJ.

[19] Richard E. Fikes and Nils J. Nilsson. “Strips: A New Approach to the Application of Theorem Proving to Problem Solving”. In: *Artificial Intelligence* 2.3–4 (Dec. 1971), pp. 189–208. ISSN: 00043702. DOI: 10.1016/0004-3702(71)90010-5. URL: https://linkinghub.elsevier.com/retrieve/pii/0004370271900105 (visited on 11/10/2025).

[20] Michael Ahn et al. *Do As I Can, Not As I Say: Grounding Language in Robotic Affordances*. Aug. 16, 2022. DOI: 10.48550/arXiv.2204.01691. arXiv: 2204.01691 [cs]. URL: http://arxiv.org/abs/2204.01691 (visited on 11/10/2025). Pre-published.

[21] Wenlong Huang et al. “Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents”. 2022. arXiv: 2201.07207. URL: https://arxiv.org/abs/2201.07207.

[22] Kutluhan Erol, James Hendler, and Dana S Nau. “UMCP: A Sound and Complete Procedure for Hierarchical Task-Network Planning”. In: ().

[23] D. S. Nau et al. “SHOP2: An HTN Planning System”. In: *Journal of Artificial Intelligence Research* 20 (Dec. 1, 2003), pp. 379–404. ISSN: 1076-9757. DOI: 10.1613/jair.1141. arXiv: 1106.4869 [cs]. URL: http://arxiv.org/abs/1106.4869 (visited on 11/10/2025).

[24] Artur d’Avila Garcez et al. *Neural-Symbolic Computing: An Effective Methodology for Principled Integration of Machine Learning and Reasoning*. May 15, 2019. DOI: 10.48550/arXiv.1905.06088. arXiv: 1905.06088 [cs]. URL: http://arxiv.org/abs/1905.06088 (visited on 11/10/2025). Pre-published.

[25] Artur d’Avila Garcez and Luis C. Lamb. *Neurosymbolic AI: The 3rd Wave*. Dec. 16, 2020. DOI: 10.48550/arXiv.2012.05876. arXiv: 2012.05876 [cs]. URL: http://arxiv.org/abs/2012.05876 (visited on 11/10/2025). Pre-published.

[26] Tom Silver et al. “Generalized Planning in PDDL Domains with Pretrained Large Language Models”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 18. 2024, pp. 20256–20264. DOI: 10.1609/aaai.v38i18.30006. URL: https://ojs.aaai.org/index.php/AAAI/article/view/30006 (visited on 11/10/2025).

[27] Karthik Valmeekam et al. “On the Planning Abilities of Large Language Models: A Critical Investigation”. In: *Advances in Neural Information Processing Systems*. Vol. 36. Curran Associates, Inc., 2023, pp. 75993–76005. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/efb2072a358cefb75886a315a6fcf880-Abstract-Conference.html (visited on 11/10/2025).

[28] Katherine M. Collins et al. *Structured, Flexible, and Robust: Benchmarking and Improving Large Language Models towards More Human-like Behavior in Out-of-Distribution Reasoning Tasks*. May 12, 2022. DOI: 10.48550/arXiv.2205.05718. arXiv:

2205.05718 [cs]. URL: http://arxiv.org/abs/2205.05718 (visited on 11/10/2025). Pre-published.
[29] Qing Lyu et al. *Faithful Chain-of-Thought Reasoning*. Sept. 20, 2023. DOI: 10.48550/arXiv.2301.13379. arXiv: 2301.13379 [cs]. URL: http://arxiv.org/abs/2301.13379 (visited on 11/10/2025). Pre-published.
[30] Lin Guan et al. *Leveraging Pre-trained Large Language Models to Construct and Utilize World Models for Model-based Task Planning*. Nov. 2, 2023. DOI: 10.48550/arXiv.2305.14909. arXiv: 2305.14909 [cs]. URL: http://arxiv.org/abs/2305.14909 (visited on 11/10/2025). Pre-published.
[31] ??? Ye et al. "Domain-Independent Automated Planning with Llms". In: *arXiv preprint* (2024).
[32] Noah Shinn, Federico Labash, and Anna Rumshisky. "Reflexion: Language Agents with Verbal Reinforcement Learning". 2023. arXiv: 2303.11366. URL: https://arxiv.org/abs/2303.11366.
[33] Shunyu Yao, Dian Yu, Jeffrey Zhao Zhao, et al. "Tree of Thoughts: Deliberate Problem Solving with Large Language Models". 2023. arXiv: 2305.10601. URL: https://arxiv.org/abs/2305.10601.
[34] Michael Cashmore et al. "Robustness Envelopes for Temporal Plans". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 7538–7545. DOI: 10.1609/aaai.v33i01.33017538. URL: https://ojs.aaai.org/index.php/AAAI/article/view/4745 (visited on 11/10/2025).
[35] Subbarao Kambhampati et al. *LLMs Can't Plan, But Can Help Planning in LLM-Modulo Frameworks*. June 12, 2024. DOI: 10.48550/arXiv.2402.01817. arXiv: 2402.01817 [cs]. URL: http://arxiv.org/abs/2402.01817 (visited on 11/10/2025). Pre-published.
[36] Fabien Tencé and Cédric Buche. *Automatable Evaluation Method Oriented toward Behaviour Believability for Video Games*. Sept. 2, 2010. DOI: 10.48550/arXiv.1009.0501. arXiv: 1009.0501 [cs]. URL: http://arxiv.org/abs/1009.0501 (visited on 10/20/2025). Pre-published.
[37] Nicholas Fabiano et al. "How to Optimize the Systematic Review Process Using AI Tools". In: *JCPP Advances* 4.2 (June 2024), e12234. ISSN: 2692-9384, 2692-9384. DOI: 10.1002/jcv2.12234. URL: https://acamh.onlinelibrary.wiley.com/doi/10.1002/jcv2.12234 (visited on 10/28/2025).

A AI Tools and Configuration Details

[To be completed]

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Technical
University of
Denmark

Richard Petersens Plads, Build. 324
2800 Kgs. Lyngby
Tlf. 4525 1700

https://www.compute.dtu.dk/