Agents Thinking Fast and Slow: A Talker-Reasoner Architecture

Konstantina Christakopoulou, Shibl Mourad, Maja Matarić
Google DeepMind

Abstract

Large language models have enabled agents of all kinds to interact with users through natural conversation. Consequently, agents now have two jobs: conversing and planning/reasoning. Their conversational responses must be informed by all available information, and their actions must help to achieve goals. This dichotomy between conversing with the user and doing multi-step reasoning and planning can be seen as analogous to the human systems of "thinking fast and slow" as introduced by Kahneman [14]. Our approach is comprised of a "Talker" agent (System 1) that is fast and intuitive, and tasked with synthesizing the conversational response; and a "Reasoner" agent (System 2) that is slower, more deliberative, and more logical, and is tasked with multi-step reasoning and planning, calling tools, performing actions in the world, and thereby producing the new agent state. We describe the new Talker-Reasoner architecture and discuss its advantages, including modularity and decreased latency. We ground the discussion in the context of a sleep coaching agent, in order to demonstrate real-world relevance.

1 Introduction

Humans have the ability to do two very different kinds of thinking. On the one hand, we can form snap judgements, such as reacting to a speeding car or recognizing the emotional cues of an upset coworker. On the other hand, we can solve complicated problems, like planning a vacation and doing complex multiplications. The well-known behavioral science theory proposes that two different systems drive those abilities: the fast and intuitive System 1 and the slow and deliberative System 2 [14]. Daniel Kahneman, who introduced the theory, described the two systems for the two modes of thinking as follows: "*System 1 operates automatically and quickly, with little or no effort and no sense of voluntary control. System 2 allocates attention to the effortful mental activities that demand it, including complex computations. It represents the conscious reasoning self that has beliefs, makes choices, and decides what to think about and what to do.*"

Although difficult problems might rely more on System 2 and everyday skills more on System 1, most cognitive processes are a mix of both kinds of reasoning. System 1 continuously generates suggestions for System 2: impressions, intuitions, intentions, and feelings. If endorsed by System 2, impressions and intuitions form the basis of the explicit beliefs of System 2, and intentions turn into the deliberate choices of System 2.

Many reinforcement learning (RL) problems can also benefit from a similar dual-system approach. The rapid advances in large language models (LLMs) [1, 6, 32] have enabled artificial intelligence (AI) agents of all kinds, from AI coding buddies, to tutors and health coaches. These agents are expected to understand the complex patterns of the world via language and potentially perceive other heterogeneous multimodal signals, generating impressions, creating coherent patterns of ideas, and producing dialog (with other modalities being actively added). This resembles the coherent-seeking System 1. On the other hand, AI agents are supposed to perform complex multi-step reasoning, and

NeurIPS 2024 Workshop on Open-World Agents (OWA 2024).

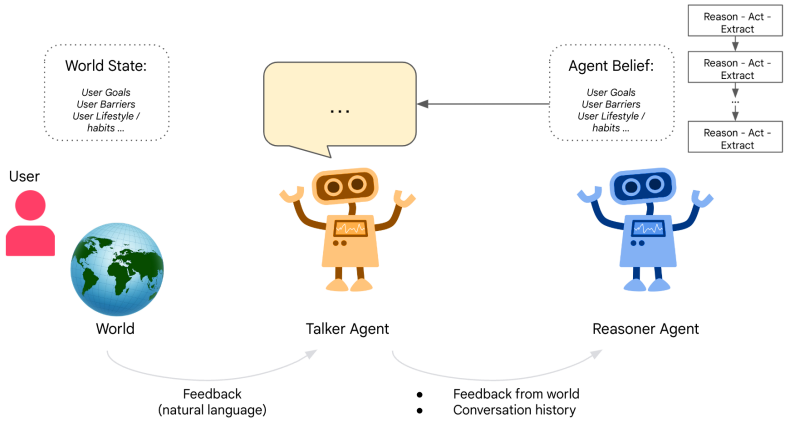

Figure 1: Illustration of the proposed dual-system Talker-Reasoner approach.

make decisions that involve calling tools, actively retrieving information from external data sources, and solving complex problems. This is similar to the slower and more deliberative System 2.

In the context of enabling agents to converse, reason and plan, in this work we consider a dual-system approach that enables those abilities through the two modes of thinking. We therefore divide the agent into two agents: a fast and intuitive Talker agent and a slower and deliberative Reasoner agent. The Talker agent focuses on generating natural and coherent conversation with the user and interacts with the environment, while the Reasoner agent focuses on performing multi-step planning, reasoning, and forming beliefs, grounded in the environment information provided by the Talker. The Talker agent, a la System 1, can access memory, priming its responses.

Similarly to the System 1 and 2 modes of thinking, the division of labor between the Talker and Reasoner agents is efficient: it minimizes effort and optimizes performance. An added benefit of this division is that the Talker can carry out the conversation, while getting more observations from the environment, without needing to wait for the slow reasoning and belief forming of the Reasoner agent. This is analogous to behavioral science dual-System approach, with System 1 always being on while System 2 operates at a fraction of its capacity. Similarly, the Talker is always on and interacting with the environment, while the Reasoner updates beliefs informing the Talker only when the Talker waits for it, or can read it from memory. This division of labor works well most of the time, as the Talker is typically very good at what it does: it can automatically fetch information from memory, effectively priming its underlying model to respond well to familiar situations. However, the framework has limitations. The Talker operates with a more outdated view of the world, which has inherent biases, and can sometimes answer easier questions than the ones asked. Also, it has little understanding of complex problem solving and planning. So, we introduce a variable allowing the Talker to wait the Reasoner, in cases when System 2-thinking is necessary before the Talker forms its response.

To evaluate the proposed dual-system Talker-Reasoner framework, we ground our work on the real world setting of a sleep coaching agent interacting with users through dialog. We discuss success cases of this division of labor, including fast and intuitive conversations driven by the Talker and complex plans and belief states developed by the Reasoner. We also discuss cases where, similar to the dual-system thought machinery, the Reasoner (System 2) might need to override the Talker (System 1). In the AI coaching context, this could be useful when the user is requesting a complex coaching plan the Reasoner needs to finish before the Talker is able to respond.

2 Related Work

Large Language Models for Agent Planning. Inspired by the strong emergent capabilities of LLMs [6], such as zero-shot prompting [15], in-context learning [5], and complex reasoning [37, 43], research into LLM-driven agents is receiving a great deal of attention [36, 40, 2, 25, 21, 34, 35, 24, 4]. The work most relevant to this paper is on *text-based agents* [25, 39, 22], although a great deal of work on real-world embodied agents [16] is increasingly relevant as models become truly multimodal

[36, 2, 34, 19, 27, 30, 28]. ReAct [39] uses chain-of-thought (CoT) prompting [37] and generates both reasoning traces and task-specific actions (e.g., tools to call) with LLMs. Reflexion [26] extends ReAct with self-reflection to improve reasoning. AutoGPT [38] is a tool for automating tasks by crafting a curriculum of sub-goals for completing a high-level goal, connecting to hierarchical reasoning. All these works lack (i) talking while reasoning/planning and (ii) explicit belief modeling, which are crucial components in our framework. The (i) *talking* aspect of our agent connects our work to prior work on *natural-language based feedback agents* [26, 40]. Similarly to [40], our agent iteratively incorporates environment feedback to modify subsequent plans in a closed-loop process. However, we do not use RL to update all future plans, but instead only to adapt the cross-session plan for the single user, by augmenting the context of the in-context learned LLM with the feedback. The (ii) *explicit belief modeling* aspect of our work relates to the theory of mind [23, 8, 18] and the large body of work on world modeling [7, 10, 12, 42], as the world in our case encompasses the user, and the agent builds a proxy model of other agents (i.e., humans) to reason about their behaviors. In particular, the `Reasoner` continuously updates its belief about the user's goals, plans, barriers, motivations, in the form of a structured object or schema [43], similarly to world and user models in past work [11, 10, 42]. We do not yet model beliefs about beliefs (e.g., what the user thinks the agent thinks, and so on) or use world models to predict future actions; this is left for future work.

3 The Talker-Reasoner Agent Model

Before we introduce the dual-system Talker-Reasoner agent framework corresponding to the fast and slow thinking respectively (Section 3.2), we start with formalizing a *single* language-based agent capable of *talking* and System 1 reasoning, as well as System 2 *multi-step reasoning and planning* useful for complex problem solving (Section 3.1).

3.1 Single Language-Based Agent Interacting With Humans: Synergizing Talking and Extracting Beliefs With Reasoning and Planning

Let us consider a language-based AI agent that can interact with users through natural language conversation to help them accomplish some task. The agent should be capable of multi-step reasoning and planning to be able to solve the task and also capable of generating a conversational response to the user. This paradigm of agents reasoning/planning and conversing has become more prevalent as a result of the introduction of large language models [41, 32, 1, 31, 33, 29]. We present a Reinforcement Learning (RL) formulation of this talking-and-reasoning paradigm. We also extend the paradigm to include explicit modeling of the beliefs the agent has about the user, such as the user's motivations, emotions, and goals, which guide the talking and reasoning. Figure 2 shows an overview of the overall language-based agent interacting with a user, which we will describe in detail in what follows.

We formulate the language-based agent that can reason, talk, and do explicit belief modeling in a partially-observable RL framework [29]. The agent is continuously interacting with the world $\mathcal{E}$. The world encompasses both the user the agent is interacting with, and the knowledge bases (such as the World Wide Web) that allow the agent to retrieve real-world knowledge. The agent only has a partial view of the world, thus formulating beliefs $b \in \mathcal{B}$ about the current state of the world. It can learn more about the user by interacting with them via language (future work will add other modalities). Assume that $\mathcal{L}$ represents the language space; the agent receives from the user observations $o \in \mathcal{O}$ which live in the language space $\mathcal{O} \subset \mathcal{L}$. Observations can contain both information and feedback/rewards in natural language (e.g., "I don't like this", and "Can you add something else to my plan?"). We formalize this as $\hat{\mathcal{O}} = \mathcal{O} \cup \mathcal{R}$, with observations $o \in \mathcal{L}$ and rewards $r \in \mathcal{L}$. The observations $\hat{o} \in \hat{\mathcal{O}}$ are then used to update both the agent's beliefs and the subsequent planning/reasoning performed by the agent. This can be seen as a form of online policy learning via natural language feedback and relates to natural-language based feedback agents [26].

We now focus on the agent's actions $a \in \mathcal{A}$. The agent can (i) formulate thoughts $\tau \in \mathcal{L}$ around actions it can take, and (ii) decide which tools $a \in \mathcal{A}$ to select (e.g., APIs, engines like `SEARCH`, functions) to fetch external knowledge—this expands the space of tasks it can accomplish. By combining a series of thoughts and tools/actions, along with the results fetched via the tools, the agent can create a plan p for solving a problem. Furthermore, the agent can (iii) formulate beliefs about the user (and potentially other aspects of the world); thus, another key action is `extracting` leading to a new agent belief state. The beliefs are represented as structured language objects living in XML

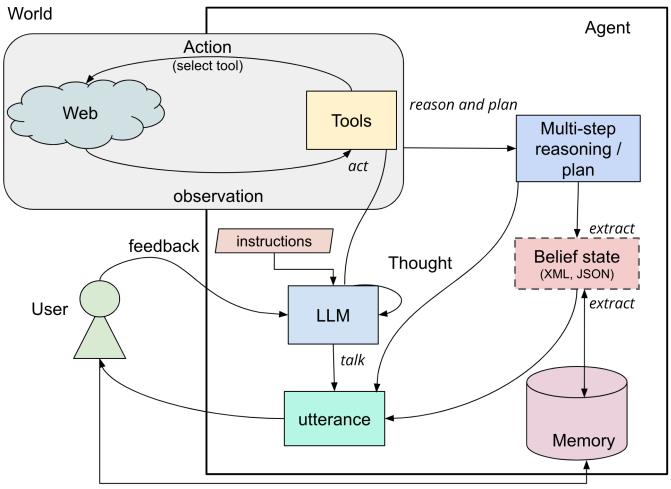

Figure 2: Single LLM-based agent that *talks* and *extracts* belief states while *multi-step reasoning*.

or JSON space, with $b \in \mathcal{SL}$ [42, 43], where structured language can be seen as a subset of the $\mathcal{L}$ language space. The belief state could encode the agent's estimate of the user's goals, needs, thoughts, sentiments, motivations, and barriers (depending on the agent context and use case), also relating this work with theory of minds [18, 42]. Given the plan formed via multi-step reasoning/planning ((i) and (ii) above) and the beliefs formed ((iii) above), the agent can (iv) `talk`, i.e., provide a natural language utterance $u \in \mathcal{L}$ to the user. Putting these together, the space of actions $\mathcal{A}$, which in a classic act-only paradigm contains only the tools, is expanded to $\hat{\mathcal{A}}$. We refer to this as the *augmented space of actions*. The space $\hat{\mathcal{A}}$ encompasses tools $a \in \mathcal{A}$, thoughts/reasoning traces $\tau \in \mathcal{T}$ (as considered in the `ReAct` [39] paradigm), and also beliefs b and utterances u, i.e., $\hat{\mathcal{A}} = \mathcal{A} \cup \mathcal{T} \cup \mathcal{B} \cup \mathcal{U}$. While thoughts and beliefs do not affect the world and lead to no observational feedback, tools and utterances interact with the world, namely the external knowledge bases and the user, respectively.

The language-based agent chooses its augmented action according to a policy π implemented via a large language model (LLM) with parameters Θ, instructed via its prompt/constitution to follow a set of instructions $\mathcal{I} \in \mathcal{L}$. The instructions encode domain knowledge, desired behavior in interactions with the user, and the constitution [3] the agent is supposed to follow. The LLM implementing the policy uses the instructions, the last user's natural-language feedback/utterance o, along with the interaction history $\mathcal{H} \in \mathcal{L}$ in its context window c. Besides the context window, the agent is memory-enabled, allowing it to record all agent-user interactions along with other user information across sessions in its memory $\texttt{mem}$. That is where the agent also stores the most recent belief state $b \in \mathcal{B}$ and the current plan p for how to solve the task. The agent can, at any point, retrieve relevant information from memory $x_{\texttt{mem}}$, augmenting the information in its context window.

We formulate this in a POMDP [20, 29], as follows. At time step $t+1$, the language-based agent chooses actions $\hat{a} \in \hat{\mathcal{A}}$ according to a Θ-parameterized LLM-based policy: $\hat{a}_{t+1} \sim \pi(\hat{a}|c_t, x_{\texttt{mem}}; \Theta)$ where the context $c_t = \text{Concat}(\hat{o}_t, \mathcal{H}_t, \mathcal{I})$ is the concatenation of the last user's utterance, the interaction history, and the overall instructions; $x_{\texttt{mem}}$ represents any information the agent might need to retrieve from memory, including the previous belief state b. For each user-agent interaction, when the agent needs to generate an utterance to talk to the user, there might be series of augmented actions the agent may take before it produces its response. The context captures the series of thoughts/tool selections/results obtained before it generates the action: $c_t = (\tau_1, a_1, o_1, b_1, \ldots, \tau_{t-1}, a_{t-1}, o_{t-1}, b_{t-1})$.

3.2 Proposed Dual-System `Talker-Reasoner` Agent Model

So far, we have formalized an agent that can interact with users to solve tasks via its ability to do multi-step reasoning and planning, talking, and extracting beliefs about the user. However, this can be hard for a single LLM to do, as there are different requirements for talking vs. multi-step reasoning/planning and forming beliefs. In what follows, we propose the dual-system architecture, inspired by the fast and slow thinking Systems 1 and 2, respectively, consisting of:

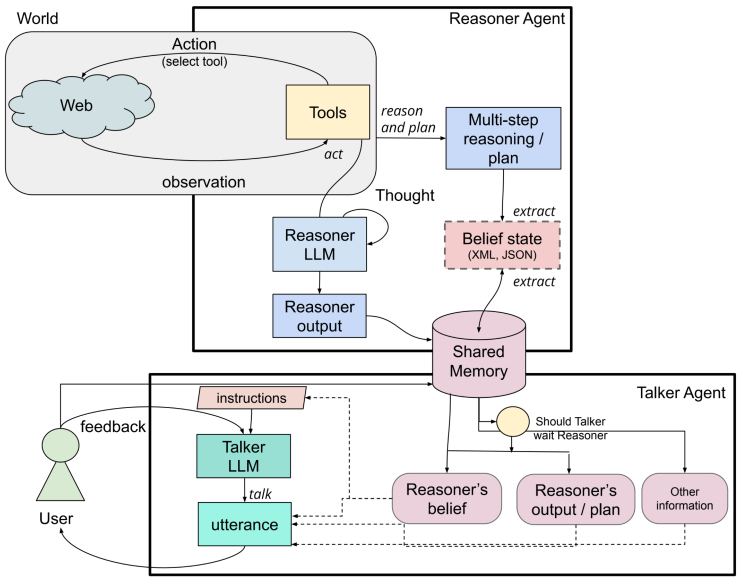

Figure 3: Diagram of Talker-Reasoner architecture.

1. The `Talker`: The fast agent that interacts with the user via language, perceives the world, gets observations and feedback from the user, interacts with memory to prime its responses, and generates the conversational response.
2. The `Reasoner`: The slow and deliberative agent responsible for complex problem solving, which involves synergizing reasoning with taking actions augmenting its knowledge from the real world, such as calling tools or fetching information from external databases [17]. The `Reasoner` is also responsible for making and updating beliefs that drive its decisions, and the `Talker`'s subsequent utterances. The `Reasoner` is typically goal-conditioned, primed to solve a specific problem or goal [9], and hierarchical [36], dividing problems into sub-problems.

As shown in Figure 3, the main way the `Talker` (System 1) and `Reasoner` (System 2) interact is through memory. The `Reasoner` is responsible for generating the next belief state and deriving multi-step reasoning and planning, and storing them in memory. Any time the `Talker` needs the belief, it retrieves the latest one from memory. The `Talker` therefore might operate with a delayed view of the world, as the `Reasoner` might not have had time to generate the new belief and store it in memory. However, because the `Talker` is meant to be intuitive and fast and takes into account what the user just said and the conversation history, the conversational response will still be coherent. In fact, the conversation driven by the `Talker` is processed by the `Reasoner` so that the quick impressions and responses of the `Talker` become sources of explicit beliefs and choices (plans) of the `Reasoner`. The `Talker` can also wait for the `Reasoner` before generating a response; this is equivalent to System 2 taking over and overruling System 1's impulses.

3.2.1 The `Talker` (Thinking Fast) Agent

The `Talker` *interacts with the world*, including the user, and needs to understand language and the interaction/conversation history, and be able to generate natural human-level language to do the needed `talk` action. These criteria are met by implementing the `Talker` agent with a powerful, in-context learned [5] language model. Similar to System 1, the `Talker` strives for coherence, and acts as an associative machine. To ensure the coherence of the `Talker` and a good user experience, instructions $\mathcal{I} \in \mathcal{L}$ are given to the language model to follow, encoding the `Talker`'s constitution [3].

The `Talker` also *interacts with memory* `mem` to prime its responses with relevant information $x_{\texttt{mem}}$, including the latest beliefs that have been formed by the `Reasoner` and stored in `mem`. At every interaction with the user, the `Talker` takes the `Talk` action, and generates a conversational response, i.e., utterance u, conditioned on the context c and the instructions $\mathcal{I}$:

$$u_{(t+1)} \sim \texttt{Talker}(u|c_{t+1}, \mathcal{I}(\cdot|b_{\text{mem}}); \Phi) \tag{1}$$

where Φ are the parameters of the Talker. The context c_{t+1} can include the latest user utterance $\hat{o}$ which acts as both observation and natural language feedback, the b_{mem} is the latest belief produced by the `Reasoner` and stored in memory, and $\mathcal{H}_{\text{mem}}$ is the interaction history:

$$c_{t+1} = \text{Concat}(\hat{o}_{t+1}, b_{\text{mem}}, \mathcal{H}_{\text{mem}}) \tag{2}$$

The instructions $\mathcal{I}$ the `Talker` follows can depend on the belief state. Overall, the `Talker` is supposed to be fast and conversational, minimizing latency. It is "on" whenever the user converses with the system, similar to System 1. It may use beliefs b_{mem} that are not the latest b_{t+1} of the `Reasoner` in order to ensure fast interactivity, meaning that the two systems may at times be decoupled.

3.2.2 The `Reasoner` (Thinking Slow) Agent

The `Reasoner` agent acts like System 2: it enables complex problem solving, deliberate belief forming, and choice making.

The `Reasoner` performs *multi-step reasoning and planning*, entailing series of calls to various in-context learned [5] or Chain-of-Thought (CoT)-prompted language models [37], and calls to different tools [39] or databases [17] for external knowledge fetching. This requires it to synergize `reasoning` (producing thoughts) and `acting` (calling tools to fetch observations), as in retrieval-augmented or tool-enhanced `ReAct`-type agents [39]. The agent can develop plans (e.g., series of tools to call) and reasoning traces to solve complex tasks. It typically decomposes the problem into sub-problems in a *hierarchical fashion*, and tasks each sub-problem to different modules, tools, or LLMs.

It also *forms beliefs about the state of the world*, which can combine multiple intermediate results of multi-step reasoning, and extract from past interaction history all interesting facts about the user model in a structured language object to be stored in `mem`. This aspect of deliberate belief forming is what distinguishes the `Reasoner` from typical `ReAct`-style agents, as it includes deliberate attempt in modeling the world/human, as described in the `extract` action.

Concretely, the actions the `Reasoner` can take are: `reason`, `act`, and `extract`, each resulting in thoughts $\tau \in \mathcal{L}$, intermediate observations as a result of tool use o fetching external knowledge, and beliefs $b \in \mathcal{SL}$ in the form of structured language objects. Thus, the augmented space of actions includes thoughts, tool actions, and belief extractions: $\hat{\mathcal{A}} = \mathcal{A} \cup \mathcal{T} \cup \mathcal{B}$. Since the augmented action space lives in the unlimited language space, learning a policy is difficult and requires strong language priors. Thus, we implement the `Reasoner`'s policy via an in-context learned language model parameterized by Z. The `Reasoner` selects an augmented action:

$$\hat{a} \sim \texttt{Reasoner}(b, \hat{a} | c_{\texttt{Reasoner}}; Z). \tag{3}$$

The context $c_{\texttt{Reasoner}}$ involves the interleaving of a series of n thoughts, actions (e.g., calling tools like `SEARCH`), observations after these actions, and belief extractions, along with the newest observations/language feedback $\hat{o}_t$ provided by the `Talker`:

$$c_{\texttt{Reasoner}} = \text{Concat}(\tau_1, a_1, o_1, b_1, \ldots, \tau_n, a_n, o_n, b_n; \hat{o}_t). \tag{4}$$

When the `Reasoner` finishes its series of n reasoning/planning steps, where n can vary per round depending on the problem, it constructs the belief state b_{t+1} as a combination of intermediate beliefs $b_1, \ldots, b_n$ and stores it in `mem`. Therefore, between two steps of user-`Talker` interaction, there are n steps of slower "thinking" by the `Reasoner`.

4 Evaluation Case Study: Sleep Coaching Agent

We instantiated and validated the `Talker-Reasoner` dual-agent architecture in a sleep coach use case: an AI language agent interacting with users to provide help with sleeping behaviors and challenges.

4.1 Grounding in a Real-World Scenario of AI Coaching for Sleep

We use this real-world scenario to ground the evaluation of our dual-agent architecture. We chose AI coaching because it requires having a model of the user being coached, using sleep coaching expert knowledge to ensure scientifically-supported advice, providing a multi-step coaching plan for the user, and being conversational and empathetic much as a human coach would be. This instantiation allows us to qualitatively test the planning and reasoning capabilities of the `Reasoner` and the interactivity of

the Talker. We also chose sleep because it is a critically important component of human health, with impact on nutrition, activity, and mental health [13]. The AI coach needs to continuously understand the user's needs through dialog, and to accompany the user from understanding them, to helping them set goals, to providing a multi-step plan they can follow, connecting them with resources.

4.2 Instantiating a Talker-Reasoner Dual-Agent Model for Sleep Coaching

Sleep Coaching Talker Agent: We encode expert knowledge about sleep obtained from clinical experts in a set of instructions $\mathcal{I}$ that describe the agent's constitution (e.g., being empathetic, conversational, providing accurate advice) and the desired phases of sleep coaching (understanding, goal-setting, and coaching-plan) with separate instructions for each: $\mathcal{I}_{\text{understanding}}$, $\mathcal{I}_{\text{goal-setting}}$, $\mathcal{I}_{\text{coaching-plan}}$, to guide the Talker through the expert clinically-informed coaching process. We implemented the Talker via a Gemini 1.5 Flash [31] model, conditioned on the instructions, the context including the last user utterance, the interaction history, and the latest available belief state stored in mem, as in Equations 1, 2. The model's strong language prior along with set of instructions allows it to perform complex pattern understanding and provide thoughtful conversational responses akin to System 1, as illustrated in Section 4.3.1.

Sleep Coaching Reasoner Agent: The AI Coaching Reasoner explicitly models beliefs about the user. To do so, with clinician expert input, we encoded a JSON/XML schema of the belief, including fields such as sleep concern, goals, habits, barriers, and sleep environment. The Reasoner infers and updates the schema fields while it performs its multi-step reasoning/planning, thereby creating/updating its belief state. We implemented a *hierarchical* [36] Reasoner— at every turn, it infers which of the three discrete coaching phases the user is in, and employs the corresponding of the three mini-Reasoners, each instantiated by a different combination of CoT-prompted Gemini 1.5 Flash models [31] along with corresponding schemas to extract.

Talker-Reasoner Coordination: Whether the Talker waits for the Reasoner to finish is determined by the belief—in the planning coaching phase the Talker waits, otherwise it does not.

4.3 Qualitative Results

4.3.1 Example Conversation

The following is an example conversation illustrating the interaction between Reasoner and Talker.

USER: Hey, I need your help planning re: how to create a relaxing bedtime environment.
(BELIEF in mem) {updated-context: starting context, updated-title: Sleeping Coaching, coaching-phase: UNDERSTANDING}
TALKER: Great, let's start working on creating a relaxing bedtime environment for you :-) What's one thing in your bedroom that might be disrupting your sleep?
USER: I think noises and light can be too distracting. Can you help create a plan for me for how to eliminate these distractions?
(BELIEF in mem) *unchanged*
TALKER: Absolutely! Here is a plan to eliminate distractions in your bedroom: Noise:

- Identify noise sources: Make a list of all the noise sources in your bedroom
- Block outside noise: Use blackout curtains to block light and reduce noise; Consider noise-cancelling curtains or soundproofing panels.; Try earplugs or a white noise machine.

Light: ... Remember, consistency is key. Stick to this plan every night to create a peaceful and dark sleep environment. Is there anything else I can help you with?
USER: sure, that's useful. However, I would prefer to take it more step-by-step. And also see resources of how to create a serene and relaxing atmosphere in my bedroom.
(BELIEF in mem): {user context: UserCollectedContextData(primary-sleep-concern: Noise and light distractions in the bedroom), coaching phase: CoachingPhase(value=PLANNING)}
REASONER's PLAN: Certainly! Let's take a more in-depth look at the first step of your plan:

- *Step: Choose a calming color palette.* Description: When selecting colors for your bedroom, opt for soothing, neutral tones like soft blues, greens, or grays. These shades have been scientifically proven to promote relaxation and tranquility.
  1. YouTube Video: (link) Title: Wall Color Ideas to Create a Relaxing and Stress-free Bedroom. Channel: roomdsignTV. Reasoning: This video from roomdsignTV provides visual examples

of calming bedroom color palettes. It showcases real-life transformations that demonstrate the impact of soothing colors on creating a serene atmosphere.

2. YouTube Playlist: (link) Title: Berger paints colour combination. Channel: Hresun Interiors. Reasoning: This playlist from Hresun Interiors presents a collection of bedroom color combination ideas. It features a wide range of styles, from traditional to modern, to help you find the perfect color scheme for your bedroom.

TALKER: *copying `Reasoner`'s response*

4.3.2 Adapting planning from feedback

The following is an example of how the `Reasoner` agent adapted its plan based on the feedback collected from user by the `Talker` agent.

USER: thank you, this is quite useful. Could you please add in my plan more steps around any relaxing sounds I should add to my space?
(BELIEF in `mem`): *unchanged*
PLAN by REASONER: Absolutely! Here's an "Explore Natural Sounds" segment of your plan, along with some relevant YouTube content:

- *Explore Natural Sounds:* This segment is designed to help you create a calming and serene ambience in your space using sounds of nature. By incorporating these sounds, you can promote relaxation and reduce stress. Recommended YouTube videos: ...
- *Discover Healing Frequencies:* In this segment of your plan, we'll explore the world of theurapeutic frequencies, which are believed to have a profound impact on relaxation and emotional well-being. ... YouTube recommendations:

4.4 Discussion

As mentioned in Section 3, to minimize latency, the `Talker` uses the *latest available* belief state b from memory, rather than waiting for the `Reasoner` to finish its thinking process. The qualitative results in Section 4.3 illustrate two distinct success and failure modes of this approach:

"Intuitive `Talker`": The asynchronous approach can be effective for tasks where the `Talker` is sufficient even if it operates with an older belief state. These are typically System 1 tasks. For example, when the coaching phase is "understanding", the `Talker` can successfully carry out the conversation without the need for the `Reasoner` to finish the belief updating.

"Snap judgement `Talker`": However, the `Reasoner` must update its belief state before the `Talker` proceeds in complex problem-solving scenarios e.g., when the user is asking for an explicit multi-step plan or for specific resources that require tool calling. In those cases, without waiting for the `Reasoner` to finish, the Talker makes snap judgements. We can see some examples of such "snap judgement `Talker`" behavior when the belief extracted by the `Reasoner` does not yet capture the correct coaching phase, and does not fetch resources. To address this, when the `Talker` reads that the coaching phase is "planning", it is instructed to wait for the `Reasoner` to finish. This corresponds to System 2 taking over and overruling the impulses of System 1.

Finally, although there is a growing interest in AI agents performing more complex System 2 reasoning [14], we believe that our work is the first to formalize the duality of System 1 and System 2 reasoning that our `Talker-Reasoner` architecture offers.

5 Conclusions

This paper introduces the dual-system agent framework as a possible biologically-inspired architecture for foundation-model driven intelligent agents. Inspired by the behavioral science principles behind this framework, directions for future research include deciding when not to probe the Reasoner and how to utilize it in a lower capacity most of the time, when the Talker can handle most situations. Ideally, given a user query, the Talker should automatically determine whether it requires System 2 reasoning, and therefore the Reasoner, or whether it can safely proceed with its System 1 thinking. Another direction is to extend the Talker-Reasoner architecture to multiple Reasoners, each writing belief states to different part of the memory, for different types of reasoning.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691* (2022).

[3] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073* (2022).

[4] Ethan Brooks, Logan Walls, Richard L Lewis, and Satinder Singh. 2024. Large language models can implement policy iteration. *Advances in Neural Information Processing Systems* 36 (2024).

[5] Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

[6] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712* (2023).

[7] Jay W Forrester. 1971. Counterintuitive behavior of social systems. *Theory and decision* 2, 2 (1971), 109–140.

[8] Uta Frith and Christopher D Frith. 2003. Development and neurophysiology of mentalizing. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 358, 1431 (2003), 459–473.

[9] Dibya Ghosh, Abhishek Gupta, and Sergey Levine. 2018. Learning actionable representations with goal-conditioned policies. *arXiv preprint arXiv:1811.07819* (2018).

[10] David Ha and Jürgen Schmidhuber. 2018. World models. *arXiv preprint arXiv:1803.10122* (2018).

[11] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992* (2023).

[12] Zhiting Hu and Tianmin Shu. 2023. Language models, agent models, and world models: The law for machine reasoning and planning. *arXiv preprint arXiv:2312.05230* (2023).

[13] Arianna Huffington. 2016. *The sleep revolution: Transforming your life, one night at a time*. Harmony.

[14] Daniel Kahneman. 2011. Thinking, fast and slow. *Farrar, Straus and Giroux* (2011).

[15] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems* 35 (2022), 22199–22213.

[16] David Kortenkamp, Reid Simmons, and Davide Brugali. 2016. Robotic systems architectures and programming. *Springer handbook of robotics* (2016), 283–306.

[17] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[18] Huao Li, Yu Quan Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Michael Lewis, and Katia Sycara. 2023. Theory of mind for multi-agent collaboration via large language models. *arXiv preprint arXiv:2310.10701* (2023).

[19] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2023. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 9493–9500.

[20] Kevin P Murphy. 2000. A survey of POMDP solution techniques. *environment* 2, 10 (2000).

[21] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332* (2021).

[22] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. (2023), 1–22.

[23] David Premack and Guy Woodruff. 1978. Does the chimpanzee have a theory of mind? *Behavioral and brain sciences* 1, 4 (1978), 515–526.

[24] SP Sharan, Francesco Pittaluga, Manmohan Chandraker, et al. 2023. Llm-assist: Enhancing closed-loop planning with language-based reasoning. *arXiv preprint arXiv:2401.00125* (2023).

[25] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2024. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems* 36 (2024).

[26] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2024).

[27] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2023. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 11523–11530.

[28] Lingfeng Sun, Devesh K Jha, Chiori Hori, Siddarth Jain, Radu Corcodel, Xinghao Zhu, Masayoshi Tomizuka, and Diego Romeres. 2024. Interactive Planning Using Large Language Models for Partially Observable Robotic Tasks. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 14054–14061.

[29] Richard S Sutton, Andrew G Barto, et al. 1999. Reinforcement learning. *Journal of Cognitive Neuroscience* 11, 1 (1999), 126–134.

[30] Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. 2023. Large language models as generalizable policies for embodied tasks. In *The Twelfth International Conference on Learning Representations*.

[31] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).

[32] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239* (2022).

[33] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[34] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291* (2023).

[35] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science* 18, 6 (2024), 186345.

[36] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560* (2023).

[37] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[38] Hui Yang, Sifu Yue, and Yunzhong He. 2023. Auto-gpt for online decision making: Benchmarks and additional opinions. *arXiv preprint arXiv:2306.02224* (2023).

[39] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629* (2022).

[40] Haiteng Zhao, Chang Ma, Guoyin Wang, Jing Su, Lingpeng Kong, Jingjing Xu, Zhi-Hong Deng, and Hongxia Yang. 2024. Empowering Large Language Model Agents through Action Learning. *arXiv preprint arXiv:2402.15809* (2024).

[41] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[42] Pei Zhou, Aman Madaan, Srividya Pranavi Potharaju, Aditya Gupta, Kevin R McKee, Ari Holtzman, Jay Pujara, Xiang Ren, Swaroop Mishra, Aida Nematzadeh, et al. 2023. How FaR Are Large Language Models From Agents with Theory-of-Mind? *arXiv preprint arXiv:2310.03051* (2023).

[43] Pei Zhou, Jay Pujara, Xiang Ren, Xinyun Chen, Heng-Tze Cheng, Quoc V Le, Ed H Chi, Denny Zhou, Swaroop Mishra, and Huaixiu Steven Zheng. 2024. Self-discover: Large language models self-compose reasoning structures. *arXiv preprint arXiv:2402.03620* (2024).