Say What? Analyzing the Impact of an Entity-Level Model of Working Memory Forgetting on Referring Expression Generation

Rafael Sousa Silva
rsousasilva@mines.edu
Colorado School of Mines
Golden, Colorado, USA

Tom Williams
twilliams@mines.edu
Colorado School of Mines
Golden, Colorado, USA

ABSTRACT

Working Memory (WM) is a necessary component both for models of human cognition and human-inspired robot cognitive architectures. Yet it is unclear how different parameterizations for Working Memory models might impact robot cognition, especially robots' ability to engage in natural, situated, language-based interactions. In this work, we evaluate an entity-level, feature-based Working Memory framework through an analysis of temporal decay and demonstrate with a set of case studies how different parameterizations for this WM dynamic have fundamentally different error modes in different interaction contexts. Specifically, we formulate rules that inform the selection of appropriate decay rate values to be used in contexts with different *environment dynamics* and *dialogue dynamics*. By formalizing and analyzing these parameterizations within a robot cognitive architecture, we are able to make key design recommendations for robot cognitive architectures.

CCS CONCEPTS

• **Computing methodologies** → **Natural language generation**; • **Computer systems organization** → **Distributed architectures**.

KEYWORDS

robot cognitive architectures, cognitive systems, working memory, referring expression generation, forgetting, entity-level decay

1 INTRODUCTION AND MOTIVATION

Many robots deployed into Human-Robot Interaction (HRI) contexts need to communicate effectively with humans through natural language. A few examples include robot social companions, healthcare robots, and even robots deployed into space contexts [cf. 8, 10, 15]. Cognitive architectures implement human-like cognitive capabilities into robotic systems [11], and can be used to create better language-capable robots. A key enabler of human cognition is Working Memory (WM), which informs key natural language processes like generation [9], understanding [14], and acquisition [2, 5]. Given WM's centrality in cognition and natural language processes, it must play a key role in human cognitive models, and thus should play a similar role in human-inspired robot cognitive architectures. Many cognitive psychological theories of WM assume that it has limited capacity [12]. As such, these theories make precise commitments to the question of how that capacity limitation is enforced, i.e., how to ensure that the most important information remains cached while other information is selectively *forgotten* from WM.

A key theory of forgetting from cognitive psychology is *decay*, which asserts that WM items are forgotten after a short amount of time, if not rehearsed [3, 6]. Given the popularity of decay, there have been attempts to implement it in cognitive architectures. For example, in ACT-R and SOAR [1, 4], memory items have an associated activation value that decays if those items are not reused by the architecture. Activation values for memory items increase when those items enter WM, and if they reach a certain threshold, the architecture will use the contents of WM to retrieve specific knowledge from long-term memory. Furthermore, work performed in DIARC (see Section 2) presented preliminary evidence that when WM models are used as a cache for relevant features that can be referenced during Referring Expression Generation (REG), the use of decay may lead to the generation of accurate, natural, and human-like referring expressions [18]. Yet, it is unclear whether different levels of decay lead to differences in real-world HRI scenarios.

In this work we evaluate an entity-level, feature-based WM system (see Section 2.1) for robot cognitive architectures and investigate the tradeoffs between different decay parameterizations. We argue that the benefits and limitations of decay may emerge in fundamentally different types of interaction contexts, shaped by critical dimensions of those interaction contexts. We demonstrate how the choice of a decay factor can be important for real-world HRI scenarios. We show that decay-based models of WM have fundamentally distinct error modes arising in different interaction contexts and formulate rules that inform the selection of appropriate decay rate values to be used in contexts with *fast environmental dynamics* and contexts with different *dialogue dynamics*.

2 ARCHITECTURAL FRAMEWORK

DIARC is a component-based robotic cognitive architecture that implements key theories of cognitive psychology and linguistics [17]. It is implemented in the Agent Development Environment (ADE) middleware [16], which treats architectural components as autonomous "agents" implemented in a client-server subsystem. Depending on the application, different architectural instances can be assembled, comprised of a particular set of architectural components. For the purposes of this work, an architectural configuration with a minimum of three components must be used. First, DIARC's WM Manager manages a set of WM buffers distributed across the architecture. Second, a knowledge base manages information about objects that the robot knows about. Finally, DIARC's Referential Executive component performs REG.

2.1 Working Memory Manager

The WM Manager can be described as the tuple $W = (F, C)$ where:

- $F = [\delta, \alpha]$ represents the list of parameter values for Decay (δ) and Interference (α) that guide how WM functions. Interference is another popular theory of forgetting from cognitive psychology that will be addressed in future work.

- $C = \{c_1, \ldots, c_n\}$ represents the set of active consultants within the architecture. Each consultant $c \in C$ is described as $c = (c_{domain}, c_{constraints})$, where c_{domain} represents the domain of entities within c (e.g., "objects", "locations", "people") and $c_{constraints}$ represents the set of constraints that c can handle [20, 21].

This WM Manager's behavior then implements an *entity-level, feature-based* resource management strategy, where each entity has its own WM buffer of relevant features. Features are added to an entity's buffer whenever that entity is mentioned in conversation, either by a human or by the robot itself. The removal of constraints from WM buffers happens according to the forgetting parameters specified by the list $F = [\delta, \alpha]$. For this paper, interference is not in use (i.e., $\alpha = \infty$). Through decay, the least recent (lowest activation) constraint is removed from a WM buffer every δ seconds.

2.2 Knowledge Bases

In DIARC, a set of Distributed Heterogeneous Knowledge Bases (DHKBs) [22] decentralizes world knowledge. That is, DHKBs manage different types of knowledge across different architectural components, while providing a standardized interface where first-order logical properties are used as the query language for accessing that knowledge. These queries are handled by *consultants*, a special type of component that provides domain-independent access to the information contained in DHKBs. Each DHKB is augmented with a set of WM buffers that implement an entity-level (i.e., each entity has its own dedicated WM buffer), feature-based (i.e., buffers store features that apply to their corresponding entity) model of WM. That is, for each consultant c, one WM buffer $B_{c,e} = \{f_0, ..., f_n\}$ is allocated for each entity e known of within the DHKB that is queryable through c's Consultant interface, and that buffer is comprised of a set of features $\{f_0, ..., f_n\}$ (i.e., properties and relations represented in first-order logic) recently known to hold for that entity. DHKBs and their associated Consultant interfaces are leveraged to achieve various tasks, such as REG.

2.3 Referring Expression Generation

REG is a classic natural language generation task in which a speaker decides what properties they will use to refer to a target referent [19]. In DIARC, REG is typically performed using the Distributed Probabilistic Incremental Algorithm (Dist-PIA) [23], a variant of the classic Incremental Algorithm designed by Reiter and Dale [13] for use in distributed robot architectures. When a WM manager is available, DIARC instead uses the Short-term memory-augmented, Distributed, Probabilistic, Incremental Algorithm (SD-PIA) [24], which operates similarly to Dist-PIA, with the exception that it's "first stop" to identify properties it might consider using during REG is the set of properties activated within the set of WM buffers for all entities accessible through all active consultants. When describing entity e managed by consultant c, SD-PIA first considers whether the features in $B_{c,e}$ are sufficient to uniquely discriminate it. If not, SD-PIA performs long-term memory queries through c's consultant interface to determine what other features might be used to construct a uniquely discriminating feature set.

3 CASE STUDIES

For consistency, our two case studies utilize the simple domain composed of four objects already known to the robot that are managed by a single "Block Consultant" (i.e., $C = \{blocks\}$). As shown in Table 1, these objects varied in size, shape, color, and material. Information about these objects and their properties was encoded into a DIARC DHKB. The preference order over these properties used by the SD-PIA algorithm was that proposed by Forsyth [7] (i.e., size, shape, color, and material).

Table 1: Objects in the robot's knowledge base

	Size	*Shape*	*Color*	*Material*
Block 1	small	cubic	green	wooden
Block 2	medium	spherical	blue	metallic
Block 3	medium	cubic	blue	rubber
Block 4	large	cylindrical	red	rubber

3.1 Case Study 1: Environmental Dynamics

First, we considered how decay might lead to changes in REG in cases reliant on *environmental dynamics*. When a robot is engaged in an interaction, the environment in which that interaction is taking place may change at different rates. As such, a property that used to hold for an object (or a relationship that used to hold between two objects) might not hold anymore.

3.1.1 Scenario. Consider a human H who is painting blocks blue while assisted by robot R. H makes a request to R: "Please hand me *the green block*." R then grabs *Block 1*, gives it to H, says "Here is the green block that you requested," and waits for further instructions. Since *Block 1* is small, it only takes H 15 seconds to fully paint it. H then asks: "Put this outside to dry and bring me the red block." R takes 15 extra seconds to take *Block 1* outside, return, and grab *Block 4*. Finally, R alerts H that it is ready to hand over the next block. This scenario is depicted in Figure 1.

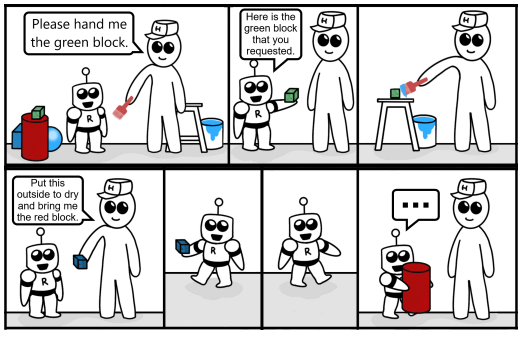

Figure 1: Interaction scenario for case study 1.

3.1.2 Architectural Validation. To assess this scenario, we conducted two decay runs with different parameterizations and started each run with no properties inside the robot's WM buffers. The first run used a decay factor greater than 30 seconds (specifically,

δ = 40 seconds). The second run used a decay factor less than 30 seconds (specifically, δ = 20 seconds). For both runs, the same procedure was followed. First, we simulate *H*'s utterance to describe *Block 1* with the property *green*. Second, we wait for 30 seconds to simulate the time the robot takes to move *Block 1* outside, grab *Block 4*, and alert *H* about the completion of their request. Finally, we ask the Referential Executive component to generate a referring expression for *Block 1* to simulate the utterance used by *R* at the end of the interaction. The lists of *Block 1* descriptors returned by the Referential Executive component are listed in Table 2.

Table 2: Parameterizations and results from Case Study 1

	δ	*List of Predicates Returned*
Decay Run 1	40 seconds	[*green(X)*]
Decay Run 2	20 seconds	[*small(X)*]

When the human describes *Block 1* as "the green block," the property *green(X)* is added to its WM buffer. In the first run, the contents of *Block 1*'s WM buffer do not change, since between the utterances used by *H* and *R* to describe it, not enough time has passed for the WM property to decay. This causes the Referential Executive component to consider the salient property *green(X)* from *Block 1*'s WM buffer first, and since that property is enough to rule out all distractors, the Referential Executive returns the list $L = \{green(X)\}$, suggesting that "the green block" is an appropriate description for *Block 1*. In the second run, however, there is enough time between the utterances used by *H* and *R* for the WM property to decay before the architecture makes a request to the Referential Executive component. When the request goes through, the SD-PIA algorithm defaults to choosing properties from long-term memory following the pre-established order of preference (i.e., size, shape, color, material) because there are no salient properties in WM. The Referential Executive then returns the list $L = \{small(X)\}$ of properties that should be used to refer to *Block 1*, since the property *small(X)* by itself is enough to rule out all other distractors.

3.1.3 Discussion. In situations reliant on *environmental dynamics*, robots may be prone to generating referring expressions containing outdated and invalid information *if* the rate of decay is slower than the rate of environmental change. Based on this observation, we can articulate our first rule of robot memory dynamics:

Rule 1. *Given rate of entity-level environmental change r_e, a rate of decay $\delta \leq r_e$ will avoid use of stale properties.*

If the rate of entity-level environmental change is not known ahead of time, it may be better to rely on a relatively low δ setting (e.g., δ = 10 seconds *if* an interaction context is characterized by fast interaction dynamics). In contrast, if there is little environmental change, it might be reasonable to set δ to a much higher value.

3.2 Case Study 2: Dialogue Dynamics

Next, we considered how decay might lead to changes in REG in cases reliant on *dialogue dynamics*. Remembering and reusing the properties that humans use to refer to specific entities can be helpful for language-capable robots, as their utterances may sound more natural and familiar to their interlocutors. In this case study, we show that while aggressive rates of decay may be needed in contexts with significant environmental change, overly aggressive decay may also erase the benefits of using WM models at all. After all, δ = 0 seconds would certainly avoid the use of stale properties, but would do so by not keeping *anything* in WM. If enough time has passed between the moment when an object is mentioned in conversation and when a robot has to refer to it, the robot's referring expression might differ from the previous significantly.

3.2.1 Scenario. Consider a human *H* who is working on a simple assembly task at home while assisted by a robot *R*. Consider that both *H* and *R* are located in the living room and the four blocks from Table 1 are stored away inside a closet at the opposite end of the house. *H* is building a small structure made only of metallic components with some blocks that they already have available, but realizes they will need *Block 2* to finish it. Promptly, they make a request to *R*: "Please bring me *the metallic block* from the closet." and *R* immediately heads to the closet to grab the requested object. *R* returns one minute later with *Block 2* in hands and alerts *H* about it. This scenario is depicted in Figure 2.

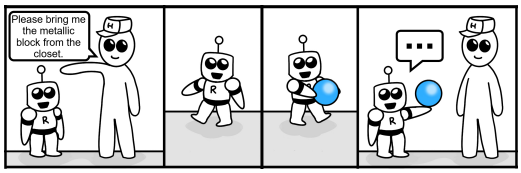

Figure 2: Interaction scenario for case study 2.

3.2.2 Architectural Validation. To assess this scenario, we conducted two decay runs and started each simulation with no properties inside the robot's WM buffers. The first run used a decay factor less than 60 seconds (specifically, δ = 40 seconds). The second run used a decay factor greater than 60 seconds (specifically, δ = 70 seconds). For both runs, the same procedure was followed. First, we simulate the use of the property *metallic(X)* to describe *Block 2*. Second, the architecture waits for 60 seconds to pass in order to simulate the time it takes for the robot to fetch the block and return to the human. Lastly, we make a request to the Referential Executive component for a description of *Block 2* to simulate the utterance used by *R* to describe the object at the end of the interaction. The lists of *Block 2* descriptors returned by the Referential Executive component can be visualized in Table 3.

Table 3: Parameterizations and results from Case Study 2

	δ	*List of Predicates Returned*
Decay Run 1	40 seconds	[*medium(X), spherical(X)*]
Decay Run 2	70 seconds	[*metallic(X)*]

In the first run, we observe a deviation from the utterance previously used by *H* because enough time passes for the salient *metallic(X)* property to leave *Block 2's* WM before a request is made to the Referential Executive component. The Referential Executive

goes through the properties within long-term memory following the order of preference of size, shape, color, and material. Since the properties *medium(X)* and *spherical(X)* are enough to rule out all distractors, they are returned by the Referential Executive as the properties that should be used in a referring expression. In the second run, we observe a consistent referring expression that matches that initially used by H. The contents of *Block 2*'s WM buffer do not change, since between the utterances used by H and R to describe it, not enough time has passed for the WM property to decay. The Referential Executive component is then able to prioritize the property *metallic(X)* from WM, which is sufficient to rule out all distractors on its own and is returned as the only element in the list of properties that should be used to refer to *Block 2*.

3.2.3 Discussion. In HRI contexts, if a robot uses different referring expressions to describe an object than those used by a human, the human may be confused as to why the robot chose a different wording. Conversely, if a robot preserves the referring expressions previously used in conversation, its speech may be perceived as more natural and easy to understand. Therefore, given our results, it seems that large decay rates are well suited for use in interaction scenarios reliant on *dialogue dynamics*, as they will preserve the most recent description for each object in WM and cause the architectural framework to generate consistent referring expressions. If decay is in use with a decay rate δ that is not large enough to accommodate the interval of time between two object references, referring expressions will likely be inconsistent.

From these observations, we are able to articulate our second rule of robot memory dynamics:

Rule 2. *Given the rate of entity-level dialogue dynamics r_d, a rate of decay $\delta > r_d$ will preserve salient properties within WM buffers.*

In cases where the rate of entity-level dialogue dynamics is unknown, it may be better to rely on a relatively high δ setting (e.g., $\delta = 90$ seconds *if* an interaction context is characterized by its dialogue dynamics). In contrast, in contexts where dialogue dynamics do not play a major role, it might be reasonable to set δ to a significantly lower value to account for situations and interaction contexts such as the one described in case study 1.

4 GENERAL DISCUSSION

The results of our case studies provided us with rules for scenarios that are reliant *either* on *environmental dynamics* or on *dialogue dynamics* alone. Putting our rules together, we can also arrive at policies aimed at leveraging decay in interaction contexts that are dependant on *both* environmental and dialogue dynamics. We have two possibilities: First, in scenarios with *slow environmental dynamics* and *fast dialogue dynamics*, such as the experimental context proposed by Sousa Silva et al. [18], the rate of entity-level environmental change will be at least as fast as the rate of entity-level dialogue dynamics (i.e., $r_e \geq r_d$). In such cases, it is easy to satisfy both of our entity-level decay rules, as we are able to choose a decay rate δ such that $r_e \geq \delta > r_d$. However, in our second type of scenarios – with *fast environmental dynamics* and *slow dialogue dynamics* – the rate of entity-level environmental change will be slower the rate of entity-level dialogue dynamics (i.e., $r_e < r_d$). In such cases, following both decay rules will not be possible as they suggest choosing two different decay rates. We must then consider whether preventing the use of outdated, invalid properties in robot referring expressions (i.e., choosing $\delta \leq r_e$ and satisfying Rule 1) outweighs preventing the use of inconsistent, yet valid, referring expressions in conversation (i.e., choosing $\delta > r_d$ and satisfying Rule 2). This shows why robots *must* satisfy Rule 1, as it will lead to lower chances of generating invalid referring expressions.

We note that depending on the context of each interaction, the expected rate of environmental change and/or the expected dialogue rate might vary across different types of objects. In an assembly task, for instance, it might be beneficial for a robot to keep salient features of the objects used in the task in WM for longer than the features of the location where the task is taking place. A dynamic decay policy that can be adjusted to handle different types of objects with different decay rates can lead to more natural and human-like robot dialogue. This type of policy can even be extended to a deeper architectural level and determine different rates of environmental change across different types of *object features*. Future work can thus investigate how different decay rates might be determined according to these different scopes of interaction scenarios.

Finally, the architectural validation of the outcomes for our case study scenarios provides the foundation to design follow-up experiments with the parameters that were used in this work. These experiments can provide valuable feedback regarding how humans will feel about the different referring expressions created by a robot in each scenario (e.g., whether they judge the robot's utterances to be natural and human-like or confusing and out of place).

5 CONCLUSION

In this paper, we evaluated the impact of an *Entity-Level, Feature-Based* WM framework for robotic cognitive architectures on natural, situated, language-based interaction scenarios with humans. Our model features a cognitively-inspired implementation of Decay, a forgetting dynamic that dictates how information can leave WM buffers. We proposed two case studies that investigated how decay can affect a robot's Referring Expression Generation process and validated them within the DIARC cognitive architecture. We identified situations in which using specific decay parameterizations might be problematic for HRI contexts reliant on different *environmental dynamics* and *dialogue dynamics*. We showed that the decay model of forgetting needs to be appropriately parameterized to promote the generation of robot referring expressions that will sound intuitive, natural, and easy to understand for human interactants. Our results can inform future human-subjects experiments aimed at assessing how humans will perceive the referring expressions that are generated through this framework.

ACKNOWLEDGMENTS

This work was funded in part by NSF CAREER grant IIS-2044865.

REFERENCES

[1] John R Anderson, Lynne M Reder, and Christian Lebiere. 1996. Working memory: Activation limitations on retrieval. *Cognitive psychology* 30, 3 (1996), 221–256.
[2] Alan D Baddeley, Susan E Gathercole, and Costanza Papagno. 1998. The phonological loop as a language learning device. *Exploring Working Memory* (1998), 164–198.
[3] John Brown. 1958. Some tests of the decay theory of immediate memory. *Quarterly journal of experimental psychology* 10, 1 (1958), 12–21.

[4] Judy Cantor and Randall W Engle. 1993. Working-memory capacity as long-term memory activation: an individual-differences approach. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 19, 5 (1993), 1101.

[5] Nadiia Denhovska, Ludovica Serratrice, and John Payne. 2016. Acquisition of second language grammar under incidental learning conditions: The role of frequency and working memory. *Language Learning* 66, 1 (2016), 159–190.

[6] Hermann Ebbinghaus. 1885. Memory: A contribution to experimental psychology, trans. *HA Ruger & CE Bussenius. Teachers College.[rWvH]* (1885).

[7] Mark Forsyth. 2014. *The elements of eloquence: Secrets of the perfect turn of phrase.* Penguin.

[8] Goren Gordon, Samuel Spaulding, Jacqueline Kory Westlund, Jin Joo Lee, Luke Plummer, Marayna Martinez, Madhurima Das, and Cynthia Breazeal. 2016. Affective personalization of a social robot tutor for children's second language skills. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.

[9] Jeanette K Gundel, Nancy Hedberg, and Ron Zacharski. 1993. Cognitive status and the form of referring expressions in discourse. *Language* (1993), 274–307.

[10] Deborah L Johanson, Ho Seok Ahn, and Elizabeth Broadbent. 2021. Improving interactions with healthcare robots: a review of communication behaviours in social and healthcare contexts. *International Journal of Social Robotics* 13, 8 (2021), 1835–1850.

[11] Iuliia Kotseruba and John K Tsotsos. 2020. 40 years of cognitive architectures: core cognitive abilities and practical applications. *Artificial Intelligence Review* 53, 1 (2020), 17–94.

[12] Wei Ji Ma, Masud Husain, and Paul M Bays. 2014. Changing concepts of working memory. *Nature neuroscience* 17, 3 (2014), 347–356.

[13] Ehud Reiter and Robert Dale. 1992. A fast algorithm for the generation of referring expressions. In *COLING 1992 Volume 1: The 14th International Conference on Computational Linguistics.*

[14] Jerker Rönnberg, Mary Rudner, Thomas Lunner, Adriana A Zekveld, et al. 2010. When cognition kicks in: Working memory and speech understanding in noise. *Noise and Health* 12, 49 (2010), 263.

[15] Sayanti Roy, Trey Smith, Brian Coltin, and Tom Williams. 2023. I Need Your Help... or Do I? Maintaining Situation Awareness Through Performative Autonomy. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction.* 122–131.

[16] Matthias Scheutz. 2006. ADE: Steps toward a distributed development and runtime environment for complex robotic agent architectures. *Applied Artificial Intelligence* 20, 2-4 (2006), 275–304.

[17] Matthias Scheutz, Thomas Williams, Evan Krause, Bradley Oosterveld, Vasanth Sarathy, and Tyler Frasca. 2019. An overview of the distributed integrated cognition affect and reflection diarc architecture. *Cognitive architectures* (2019), 165–193.

[18] Rafael Sousa Silva, Michelle Lieng, and Tom Williams. 2023. Forget About It: Entity-Level Working Memory Models for Referring Expression Generation in Robot Cognitive Architectures. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, Vol. 45.

[19] Kees Van Deemter. 2016. *Computational models of referring: a study in cognitive science.* MIT Press.

[20] Tom Williams. 2017. A Consultant Framework for Natural Language Processing in Integrated Robot Architectures. *IEEE Intell. Informatics Bull.* 18, 1 (2017), 10–14.

[21] Tom Williams. 2017. Situated natural language interaction in uncertain and open worlds. *AI Matters* 3, 2 (Jul 2017), 20–21. https://doi.org/10.1145/3098888.3098896

[22] Tom Williams and Matthias Scheutz. 2016. A framework for resolving open-world referential expressions in distributed heterogeneous knowledge bases. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

[23] Tom Williams and Matthias Scheutz. 2017. Referring expression generation under uncertainty: Algorithm and evaluation framework. In *Proceedings of the 10th International Conference on Natural Language Generation.* 75–84.

[24] Tom Williams, Ravenna Thielstrom, Evan Krause, Bradley Oosterveld, and Matthias Scheutz. 2018. Augmenting robot knowledge consultants with distributed short term memory. In *International Conference on Social Robotics.* Springer, 170–180.