An Episode Tracker for Cognitive Architectures

Eduardo Yuji Sakabe[1], Anderson Anjos da Silva[1], Luiz Fernando Coletta[1], Alexandre da Silva Simões[2], Esther Luna Colombini[1], Paula Dornhofer Paro Costa[1], and Ricardo Ribeiro Gudwin[1](✉)

[1] University of Campinas (UNICAMP), Campinas, SP, Brazil
gudwin@unicamp.br

[2] University of the State of São Paulo (UNESP), Sorocaba, SP, Brazil

Abstract. This paper introduces the Episode Tracker Module, an encoding mechanism that tracks sensory information through space and time, building up high-level semantic representations called episodes. This module is aimed to extend the Cognitive Systems Toolkit (CST) as a reusable framework for building different cognitive models for episode detection. We created two instances of the episode tracker with two different mechanisms for identifying property categories (geographical regions). Each mechanism correctly induced a different episode detection dynamic. Overall, the Episode Tracker architecture provides a robust and flexible framework for episode detection.

Keywords: Episodic memory · Cognitive architectures · CST

1 Introduction

Episodic memory is an essential component of the human cognitive system. While there is evidence of episodic-like memory in other animals [1], human episodic memory is a cornerstone in the evolution of the species [16].

For example, when searching for our car in a vast parking lot, we use episodic memory to recall where we parked the vehicle and find its current location. We also use it to remember previously inspected places to avoid searching redundant areas. We cannot recollect past experiences or understand our current context without episodic memory. A notable example is the case of Kent Cochrane, widely known as patient K.C., who suffered a brain injury that did not interfere with his understanding of the facts and concepts of the world (semantic memory) but severely damaged his episodic memory, turning impossible for him, for example, to recall his visits to the lab where he was being attended [16]. Therefore, episodic memory is essential to model human intelligence as we know it.

Nuxoll and Laird also highlight several cognitive capabilities that an agent would benefit from episodic memory, such as action modeling, decision-making, retroactive learning, and virtual sensing [13].

One key issue in designing artificial episodic memory relies on how the episodes should be encoded (i.e., the mechanism which assembles the episodes) and, consequently, their representation. Although a few attempts exist to model and implement episodic memory in artificial agents and cognitive architectures, the usual encoding process suffers from a limited scope for semantic interpretation. Typically, an episode representation is just a sequence of snapshots of other memory components, resulting in inappropriate interoperability between episodes and high-level cognitive functions. Our motivation for building yet another artificial episodic memory system is surpassing these limitations and enhancing those capabilities.

In this work, we present a first approach to the problem of modeling episodic memory in cognitive architectures, addressing this concern for semantic interpretability. In particular, we focus on the problem of detecting relevant events and building episodes from perceptual data across time. We employ Cognitive Systems Toolkit (CST) [14] to build an Episode Tracker Module, the most fundamental building block towards a full-featured episodic memory computational model.

The Episode Tracker Module is an encoding mechanism, translating sensory data through space and time into high-level semantic representations called scene-based episodes. Also, as a framework within a cognitive architecture, our implementation has a flexible design that enables the adoption of different mechanisms to identify relevant events that compose an episode.

The paper is organized as follows. We first draw the reader's attention to the difference between a simpler encoding strategy, considering what we call state-based episodes, and a more elaborated (and suitable for semantic interpretation) representation of episodes, which we call scene-based. Then, we describe the main details regarding our Episode Tracker Module, followed by a description of our experiments and their results, before a final conclusion.

2 State-Based and Scene-Based Episodes

There are two ways of representing an episode: state-based or scene-based [5]. A state-based approach represents an episode as a sequence of time-stamped internal states. State-based episodes are easier to encode since they are simply copies of information from other components' states.

This approach has two main downsides. First, it consumes more memory since the episodes are simple copies of the full agent's experiences. Second, these experiences are not interpreted, creating difficulties while interoperating with high-level cognitive functions, as there is no interpretation of what happened during this state sequence.

For instance, suppose a decision-making mechanism relying on previous episodes lived by the agent to choose between actions in a certain context. If these episodes are merely sequences of states, they will not carry the semantic knowledge of what of importance occurred during these states, regarding the agent's actions. This knowledge would require an additional processing cost. This overhead will be necessary whenever a high-level cognitive mechanism is connected with a state-based episodic memory. Scene-based representations bypass this hurdle by directly providing high-level semantic representations.

Unlike state-based episodes, scene-based episodes encode a spatiotemporal segment into a more elaborated representation carrying semantic content, which we term a *scene*. A scene comprises high-level elements, such as objects, their properties, and performed actions. Thus, scene-based episodes require complex perception mechanisms that interpret multi-dimensional sensory data to encode conceptual information (e.g., properties, objects, and actions). In this way, scene-based episodes are analogous to high-level interpretations of state-based episodes. For this reason, developing such a mechanism with a general (task-independent) approach is challenging. As a trade-off, scene-based episodes have the advantage of better interfacing with high-level cognitive functions since they have compatible high-level representations. Nevertheless, state-based episodes are the most frequent approach in cognitive architectures.

For example, in the episodic memory systems developed by Nuxoll and Laird [13], Kuppuswamy et al. [11], and Dodd and Gutierrez [6], all of them present prototypical state-based episodes, where an episode is a sequence of snapshots from other memory components, e.g., working memory.

Also, in the case of Brom et al. [4], the agent's base memory is a tree-like structure with all possible tasks the agent can perform. The episode representation is a path traversing previously executed tasks. This path is analogous to a sequence of the agent's states. Therefore, this can be considered, also, a state-based representation.

The case that most resembles what we call here a scene-based episodic representation is given in the work of Martin et al. [12], which introduces a bio-inspired model of episodic memory. The proposed model interprets the agent's sensory states into a high-level visuo-spatial representation termed a *frame*[1] formed by objects, their spatial configuration, and assigned categories from a determined instant. Two subsequent frames from the agents' experience form frame associations. In this model, an episode is a chain of frame associations. Although the episodes contain high-level visuospatial elements within an instant, there is no temporal interpretation between instants, e.g., change in object's properties. Therefore, we still consider that the episodes have a state-based representation, where each state is a *frame*.

3 The Episode Tracker Module

In this work, we describe the Episode Tracker Module, a sub-system of a cognitive architecture, which is inspired by the cognitive models of Baddeley [2], Tulving [16] and Gärdenfors [9], from Cognitive Psychology. The main task of

[1] Martin et al. [12] originally used the term *scene* to what we are calling here a *frame*. We are using *frame* here to avoid ambiguity with using the term scene in scene-based episodes.

the Episode Tracker Module is to encode perceptual data over time into scene-based episodes, which are made available for further processing in the Working Memory and eventually stored in a future Episodic Memory Module, which is still being developed.

From the neuropsychological point of view, the Episode Tracker Module is analogous to a structure called *Episodic Buffer*, within Baddeley's multi-component Working Memory model, as they both integrate sensory information across time and space into high-level representations [2,3].

Episodes generated by the Episode Tracker Module might be directly stored in the Episodic Memory Module. The Episodic Memory Module is functionally equivalent to Tulving's concept of Episodic Memory [15,16]. Our model presumes that Tulving's long-term Episodic Memory receives the encoded episodes from the Episodic Buffer. We consider two pieces of evidence for proposing this approach. First, the Episodic Buffer presumably interfaces information with long-term memory. Second, the episode representation is similar in both systems. Baddeley [2] also mentions this resemblance. Thus, the Episode Tracker Module is the Episodic Memory Module's encoding mechanism for perceptual data. Our scene-based representation of episodes, using objects and properties, is deeply inspired in Gärdenfors [9] conceptual spaces. Properties are bundles of quality dimensions, and objects are composed of multiple properties. However, we have a different description of events. Our definition of events is similar to actions in conceptual spaces, while events in conceptual spaces are closer to our definition of episodes. We define an event in the Episode Tracker Module as a relevant modification in a single object's property instance between two timesteps. Previous experiences and attentional mechanisms under the effect of conscious awareness select which events are relevant to track. Finally, we define an episode as an interpreted version of multiple events containing context between events from different objects.

3.1 The Episode Tracker Module's Architecture

The Episode Tracker was constructed using the Cognitive Systems Toolkit (CST), and after completion, will be available as a cognitive module in the toolkit. CST is a toolkit implemented in Java for building cognitive architectures under development by our research group [14][2]. A cognitive architecture built under CST is basically constructed relying on two fundamental components: codelets[3] and memory objects[4].

[2] CST's source is available in https://github.com/CST-Group/cst.

[3] Codelets, first introduced in [10] and later enhanced in [7], are small segments of non-blocking code executed in a loop. Codelets run in parallel and are responsible for all data processing within the architecture.

[4] Memory objects in CST hold any type of data structure to store information. Memory Objects are the canonical storage for data in the cognitive architecture. Different knowledge representation schemes might be used in each Memory Object.

The Episode Tracker Module's architecture can be seen in Fig. 1, which describes how information is processed by a sequence of tasks, from receiving perceptual data to the final delivery of scene-based episodes. Rounded rectangles are codelets, yellow circles are memory objects, red circles are memory objects containing long-term memory categories, and the arrows represent the information flow.

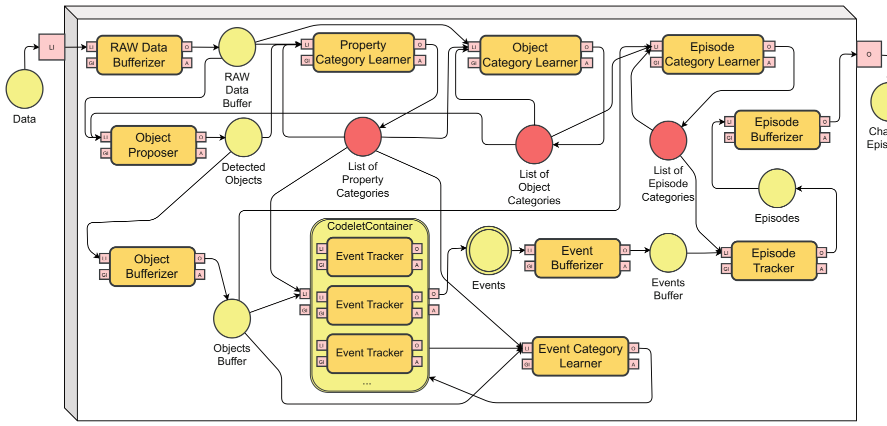


Fig. 1. Episode Tracker Module diagram.

First, the Raw Data Bufferizer receives the sensory data and creates a buffer containing the last n data states; all Bufferizers produce this same behavior. Then, the Object Proposer reads the buffer's information detecting objects and their respective property values. Furthermore, the Object Proposer can assign object and property categories using the information provided by the List of Object Categories and the List of Property Categories, respectively.

The codelet container receives the object's information along time, as a buffer of objects' states. The codelet container is a CST class that allows the operation of a dynamic number of codelets to be included, operating on the same inputs and outputs. All its codelets receive the same input information and manage access to the output. Each codelet container's Event Tracker detects a specific change in property values within one object. It can also track a shift in an object's property categories using the List of Property Categories' information. We consider an event an atomic episode since our episode representation is a chain of multiple events.

Finally, the Episode Tracker codelet interprets sequences of events into episodes. Episodes can group sequences of events and capture the cause-and-effect relations from multiple objects' events. This codelet can also detect pre-existent categorized patterns of episodes through the List of Episode Categories' information.

The resultant episode representation is scene-based since it encodes high-level visuospatial and temporal information. Visuospatial interpretation shapes objects and their properties at each instant of time. Temporal interpretation takes the form of changes in objects' properties through time and their relations (i.e., events and episodes).

4 Experiments

Our experiments used GPS data from mobile devices as sensory input to the Episode Tracker Module. Subjects from our research group commuted in the neighborhood around the UNICAMP campus, collecting GPS data (latitude, longitude, timestamp) from their smartphones using a rate of 1 Hz frequency. The Episode Tracker Module then uses this sensory input to represent the entity *user* and detect how the user changes its property *location* over time, creating relevant episodes. The challenge is inferring and categorizing relevant regions and then evaluating the movement from one relevant region to the next, assembling meaningful movement episodes. In the end, the Episode Tracker Module outputs sequences of episodes describing meaningful episodes describing both staying in a relevant position and moving from one relevant position to the next along the day.

We introduced two methods for recognizing property categories, objects, and episodes. The first method is the pheromone algorithm that learns the relevant regions online using the subject's permanence in a determined position as criteria. The second solution uses an offline clustering solution for generating the relevant regions and then inputs the regions as property categories in the Episode Tracker.

4.1 Pheromone Algorithm for Event Detection

We designed an algorithm, termed the pheromone algorithm, for detecting relevant regions. We employ the algorithm in the Property Category Learner codelet. The codelet outputs the relevant regions as property categories stored in the Property Categories memory object.

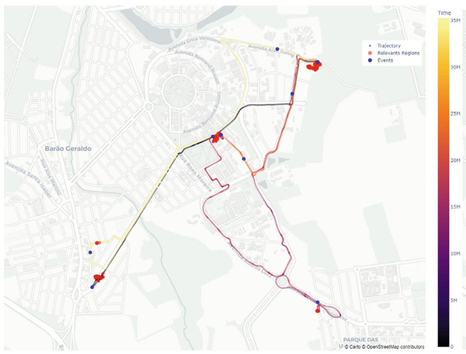
The main idea behind the algorithm is that it deems a region relevant if the subject has spent substantial time parked in that region. The representation of each region is a list of circles with a relevance value. The region is relevant if its relevance value exceeds a pre-setted relevance threshold value. The algorithm receives the subject's current location and updates the detected relevant regions. For each received location, the algorithm creates a circular area centered around the location's coordinates. If there is an overlap between any existing region and the new circle, the overlapping region's relevance increases, and the new circle is appended to the region. If there is no overlap, it creates a new region containing solely the new circle. After this evaluation, it removes regions below a pre-setted minimum value, and regions below the relevance threshold value have their relevance value multiplied by a decay rate parameter ranging from 0 to 1.

4.2 Location Clusterization Algorithm for Event Detection

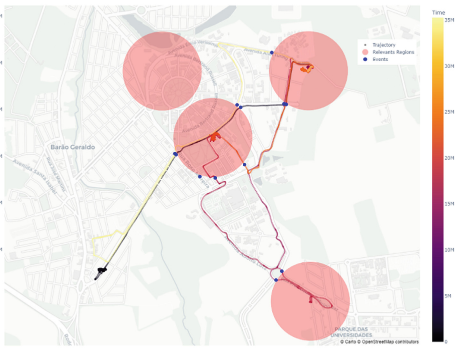
Clustering algorithms are data grouping processes. They are an unsupervised learning method where the goal is to divide a dataset into clusters such that the elements within a cluster are similar to each other and different from the elements in other clusters. There are several clustering algorithms, and in this experiment, we used the MeanShift algorithm [8]. This density-based clustering algorithm works by identifying dense regions of points in the data and assigning each point to the cluster whose density is closest without defining the number of clusters beforehand. The clustering result represents relevant regions through GPS sensor data, represented by latitude, longitude, and timestamp, and processed offline to determine the regions. Latitude, longitude, timestamp, distance, and speed data were defined as input features in the cluster. Based on these characteristics, the MeanShift algorithm calculates relevant regions. After grouping, outliers were removed to keep only the concentrated locations, resulting in the relevant regions, which are the relevant property categories, being stored in the List of Property Categories memory object.

5 Results

The results of the experiments demonstrate the capability of the Episode Tracker Module to generate property categories and utilize them for event detection. The architecture shows its flexibility by employing both online and offline identification of relevant regions through the pheromone and clustering algorithms in Fig. 2a and b, respectively. This allows for tracking events over time, enabling the creation of a trajectory history. The Episode Tracker Module successfully marked the relevant regions for users using both the Pheromone Algorithm and the Clustering Algorithm. Additionally, it effectively detected events such as entering, leaving, and staying within these regions.



(a) Pheromone Algorithm



(b) Clusterization Algorithm

Fig. 2. Event detection algorithms: pheromone and location clusterization.

The offline identification of relevant regions can be seen as a mechanism that queries information in long-term memory. This process is similar to focusing attention on salient features or events that are considered important as a reference. By storing these relevant regions offline, the Episode Tracker Module uses memory information for event detection and tracking. Furthermore, the online identification of relevant regions aligns with the concept of selective attention, where the system dynamically focuses on specific regions of interest in real time. This mechanism enables the detection and immediate response to events, replicating the attentional processes observed in cognitive models. By incorporating both offline and online identification of relevant regions, the Episode Tracker Module combines the advantages of long-term memory and selective attention. This association with attentional models enhances the system's ability to adapt and respond to changing environments, making it a powerful tool for event detection and tracking tasks.

6 Conclusion

Our work introduces the Episode Tracker Module, a cognitive module within the Cognitive Systems Toolkit that interprets sensory information through time and space into scene-based episodes. Its modular design allows the association of different algorithms or learning models with the same cognitive module, which can be effective in identifying events from sensor data on mobile devices. Two algorithms were implemented and tested: a pheromone algorithm that identifies relevant regions online and a clustering algorithm that identifies relevant regions offline. The results of the experiments showed that the architecture can deal with different ways of storing category properties and that both algorithms were able to identify events accurately. The modular cognitive architecture allows different approaches to be experimented with and evaluated, enabling the identification of better solutions for the problem in question.

Acknowledgements. This project was supported by the Ministry of Science, Technology, and Innovation of Brazil, PPI-Softex grant # [01245.013778/2020-21]. The authors also thank CEPID/BRAINN (Proc. FAPESP 2013/07559-3).

References

1. Allen, T.A., Fortin, N.J.: The evolution of episodic memory. Proc. Natl. Acad. Sci. **110**(supplement_2), 10379–10386 (2013)
2. Baddeley, A.: The episodic buffer: a new component of working memory? Trends Cogn. Sci. **4**(11), 417–423 (2000). ISSN: 1364-6613
3. Baddeley, A.D., Allen, R.J., Hitch, G.J.: Binding in visual working memory: the role of the episodic buffer. Neuropsychologia **49**(6), 1393–1400 (2011). ISSN: 0028-3932

4. Brom, C., Lukavský, J., Kadlec, R.: Episodic memory for human-like agents and human-like agents for episodic memory. Int. J. Mach. Conscious. **02**(02), 227–244 (2010). ISSN: 1793-8430. Publisher: World Scientific Publishing Co
5. Castro, E.C., Gudwin, R.R.: A scene-based episodic memory system for a simulated autonomous creature. Int. J. Synth. Emot. (IJSE) **4**(1), 32–64 (2013). ISSN: 1947-9093. Publisher: IGI Global
6. Dodd, W., Gutierrez, R.: The role of episodic memory and emotion in a cognitive robot. In: ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005, pp. 692–697 (2005). ISSN: 1944-9437
7. Franklin, S., Kelemen, A., McCauley, L.: IDA: a cognitive agent architecture. In: SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218), vol. 3, pp. 2646–2651 (1998). ISSN: 1062-922X
8. Fukunaga, K., Hostetler, L.: The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans. Inf. Theory **21**(1), 32–40 (1975)
9. Gärdenfors, P.: The Geometry of Meaning: Semantics Based on Conceptual Spaces. MIT Press (2014). ISBN: 978-0-262-02678-9
10. Hofstadter, D.R., Mitchell, M.: The copycat project: a model of mental fluidity and analogy-making. In: Analogical Connections. Advances in Connectionist and Neural Computation Theory, vol. 2, pp. 31–112. Ablex Publishing, Westport, CT (1994). ISBN: 978-1-56750-039-4
11. Kuppuswamy, N.S., Cho, S.H., Kim, J.H.: A cognitive control architecture for an artificial creature using episodic memory. In: 2006 SICE-ICASE International Joint Conference, pp. 3104–3110 (2006)
12. Martin, L., Jaime, K., Ramos, F., Robles, F.: Bio-inspired cognitive architecture of episodic memory. Cogn. Syst. Res. **76**, 26–45 (2022). ISSN: 1389-0417
13. Nuxoll, A.M., Laird, J.E.: Enhancing intelligent agents with episodic memory. Cogn. Syst. Res. **17–18**, 34–48 (2012). ISSN: 1389-0417
14. Paraense, A.L.O., Raizer, K., de Paula, S.M., Rohmer, E., Gudwin, R.R.: The cognitive systems toolkit and the CST reference cognitive architecture. Biol. Inspired Cogn. Archit. **17**, 32–48 (2016). ISSN: 2212-683X
15. Tulving, E.: Episodic and semantic memory. In: Organization of Memory, pp. xiii, 423. Academic Press, Oxford, England (1972)
16. Tulving, E.: Episodic memory: from mind to brain. Annu. Rev. Psychol. 1–25 (2002)