Scene Based Reasoning

Frank Bergmann and Brian Fenton
fraber@fraber.de, brian.fenton@gmail.com

Abstract. This paper describes Scene Based Reasoning (SBR), a cognitive architecture based on the notions of "scene" and "plan". Scenes represent real-world 3D scenes as well as planner states. Introspection maps internal SBR data-structures into 2D "scene diagrams" for self-modeling and meta-reasoning. On the lowest level, scenes are represented as 3D scene graphs (as in computer gaming), while higher levels use Description Logic to model the relationships between scene objects. A plethora of subsystems implement perception, action, learning and control operations on the level of "plans", with scenes acting as planner states.

Keywords: cognitive architecture, self-model, meta-reasoning, description logics, introspection, multi-agent, plan reasoning

1 Introduction

In this paper we describe Scene Based Reasoning (SBR), a cognitive architecture in the tradition of SOAR [14], ACT-R [1] and similar systems [10]. Particular similarities exist with the ICARUS system [16] with respect to the explicit representation of plans with decompositions, the "grounding in physical states", the purpose of controlling a physical agent, the use of observable attributes as a semantic base and spatial roles/ relationships between objects. Both systems share a development roadmap that includes modeling social interaction [16].

The distinctive characteristic of SBR is the use of "scenes", which can be thought of as a generalization of "scene graphs" [24] (as in computer gaming in order to represent 3D world states), Description Logic [2] (in order to represent the relationships between scene objects) and STRIPS style planner states [7] (in order to model time and action). Scenes are also used to represent internal SBR data-structures using a kind of "gödelization" (encoding properties of the reasoning system in object-level language): For example a "plan" (a directed graph composed of nodes and arrows) can be mapped into a 2D "scene diagram", similar to the way that humans draw figures and diagrams in order to gain clarity about complex subject matters. Once the plan is available as a 2D scene, SBR can apply its object recognition and reasoning mechanisms in order to classify the plan, create abstractions, analyze its effects, compare it with other plans and modify the plan. The improved plan can be tested in a "simulation sandbox" or the real world and can finally be converted back to an internal SBR structures for inclusion in the standard inventory of the system.

Applying a similar procedure to the class hierarchy of objects (represented as a Description Logic TBox structure) allows the SBR system to talk about "beliefs" without the need for higher order or modal logic. Updated beliefs can be "written" to a new

TBox, and this TBox can be tested in a sandbox against cases from the Episodic Memory etc. The same mechanism can be applied to belief sets of other agents in order to perform a "what-would-he-do" analysis and other types of social reasoning.

Applying "gödelization" to the recent history of cognitive events ("thoughts") allows the system to talk about it's own cognitive process. Cognitive events include the visual recognition of an object, sensory inputs, a change in attention focus, discovering the missing piece in a planning or reasoning process etc. A separate paper will explore these meta-reasoning properties in the context of the "Self-Model Theory of Subjectivity" [17].

A second distinctive feature of SBR is the use of "plans" as first order objects and as the "unit of analysis" for most subsystems - we could even talk about "Plan Based Reasoning": Perception at the highest abstraction level is plan recognition, action is plan execution, episodic memory is storing past plan executions, planning is plan generation, and plan learning is the acquisition of new plans, sub-plans (task decompositions) and execution statistics. Language comprehension is plan recognition and language generation basically serves to integrate other agents into the subject's plans (to be treated in a future paper). Plan optimization is implemented as a plan itself, allowing the SBR system to improve it's own improvement strategy.

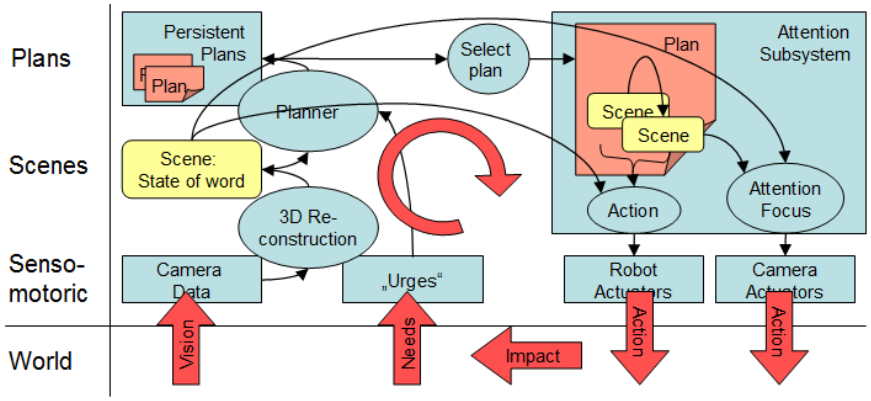

Fig. 1. An overview of SBR subsystems working together for simplified close-loop robot control. 3D reconstruction converts sensor data into a scene, which serves as an initial state for the planner to develop plans. The attention subsystem executes plan actions and controls the attention focus in order to track execution.

In this paper the authors focus on the technical aspects of the SBR architecture and a consistent definition of the SBR subsystems. A prototypical implementation of SBR exists as the "TinyCog" open-source project on http://tinycog.sourceforge.net/. TinyCog currently runs several demos using a scene representation that unifies description logics with planner states.

2 Comparison

SBR shares characteristics with SOAR, ACT-R, ICARUS and a number of lesser known cognitive architectures. The "Jonny Jackanapes" architecture [11] includes a "fusion" of HTN planning with Description Logics. [22] describes a cognitive architecture with a focus on plan recognition designed to infer the intents of competitive agents. PELA [12] describes a probabilistic planner that learns from interactions with the world.

The symbolic "scene" representation resembles [21] semantic networks, while scene graphs are commonly used in computer gaming [24]. [4] combine scene graphs with semantic networks to model human vision and propose this as a representation for "mental images".

[5] surveyed the combination of physics simulation and planning. IJCAI 2015 will host an "Angry Birds Competition" that will require physics simulation.

[9] surveyed the combination of planning and description logics. [20] introduces situation calculus to the FLEX DL system in order to allow for planning with DL ABox structures.

The SBR attention subsystem resembles the [15] "Meander" subsystem for the ICARUS cognitive architecture with similar execution tracking and re-planning properties.

3 Architecture Overview

The proposed architecture consists of four layers with several subsystems each:

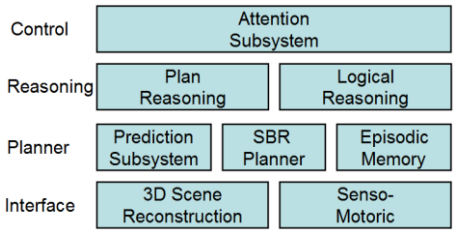

Fig. 2. The SBR Architecture layer stack.

3.1 Subsystems Overview

The "Interface" layer converts sensor data into scenes and executes planner tasks.

3D Scene Reconstruction. Converts 2D sensor data into a 3D scene graph and performs the reverse operation.

Senso-Motoric. Provides the interfaces between the SBR planner and the sensors and actuators available as part of a physical or simulated robot.

The "Planner" layer creates, recognizes and executes plans:

SBR Planner. The core of the SBR system, which takes as input an initial scene and a goal represented by a sub-scene. It returns a number of plans represented by HTN tasks, together with confidence scores.

Prediction Subsystem. Predicts the behavior of objects and agents during planning operations.

Episodic Memory. Stores large amounts of scenes, split into key frames and indexed by the included objects and their properties [19].

The "Reasoning" layer implements reasoning capabilities on top of the planner.

Plan Reasoning. Implements operations on plans that together allow for improving plans and meta-reasoning about plans.

Logical Reasoning. Implements a Description Logic on top of the SBR planner, maintaining beliefs about the world, together with a confidence score.

The "Control" layer provides high-level control of a SBR system.

Attention Subsystem. Controls the "focus of attention" of the SBR system, executes plans and contains the system's "persistent goals".

3.2 Data Structures

Objects. Untyped list of key-value tuples ("attributes") with values that can be integers, real numbers, strings or references to other objects. Symbolic object descriptions are created using the "object configurator" explained below.

Agents. Objects that maintain a "persistent goal hierarchy" and a set of beliefs. Agents represent humans, animals, robots and AGI instances in planning processes.

Relations. Named and directed arrows between two objects.

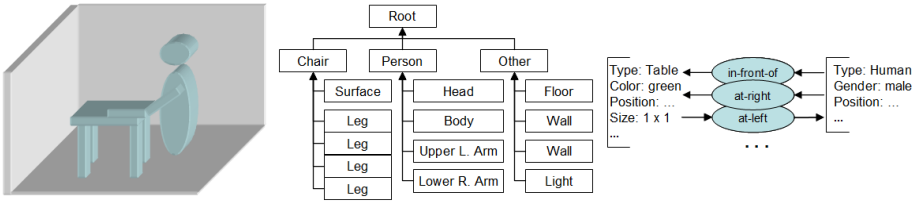

Fig. 3. A sample scene with a scene graph and a symbolic representation.

Scenes. Represent real-world 3D constellations and "mental images" as well as semantic networks for logical reasoning. On the lowest level, scenes are implemented as 3D scene graphs ([24], as in computer gaming) consisting of a number of "objects", together with their position and surface texture so that they can be rendered into a 2D image by a rendering engine. On higher levels, scene graph details are ignored, object characteristics are abstracted into attributes and spatial object constellations are encoded into semantic relations. Finally, scenes are used as an "ABox" for Description Logics reasoning. Scenes provide for self-referentiality and meta-reasoning by representing plans and other internal SBR objects as a 2D diagrams.

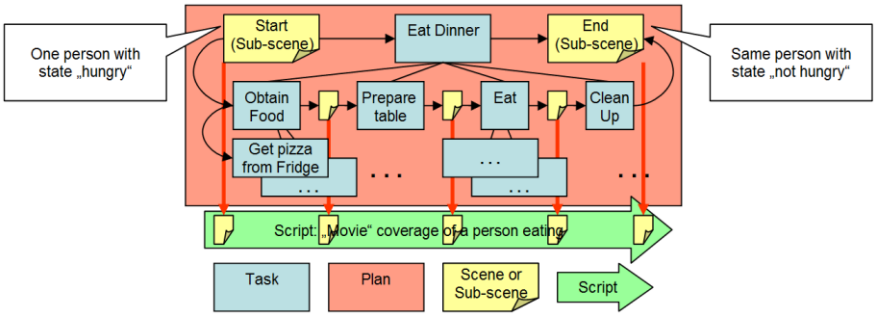

Fig. 4. "Eating dinner" - A plan for eating dinner, explaining the relationship between scenes, scripts and plans. Tasks (in blue) may have multiple learned decompositions that are combined by the planner to create plans with utility and cost.

Sub-Scenes. Scenes with only partially filled object attributes. Sub-scenes are used as rule-heads and to describe the state change effect of an action.

Scripts. Consist of sequences of scenes (similar to [Schank et al 1977]) representing a transition through time of the included objects.

Key frames. Scenes marking the start and end points of important transitions.

Plans. A tree with one root task which is decomposed into a sequence of sub-tasks.

4 3D Scene Reconstruction

This subsystems performs the conversion of 2D sensor data into a 3D scene using an iterative algorithm depicted below.

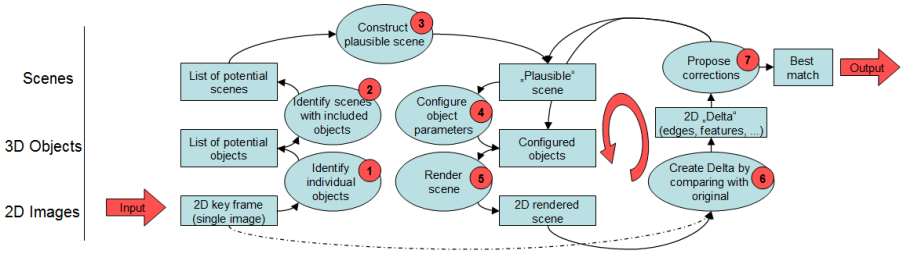

Fig. 5. 3D Scene Reconstruction: 1) Perform classical 2D feature extraction (edges, textures, …) and retrieve episodic memory (EM) object matches. 2) Check the EM for scenes that contain all or part of the objects. 3) Construct a plausible scene with expectations from higher SBR levels. 4) Configure 3D objects (position, textures, …) for best fit with the sensor data. 5) Render the scene including lightning and filters. 6) Calculate "deltas" on edges, textures etc. 7) Use deltas to correct object position and state.

5 Prediction Subsystem or "Sandbox"

This subsystem performs a probabilistic prediction of the behavior of objects and agents in a scene in order to perform a "what-if" simulation of likely outcomes of actions, effectively providing a simulation sandbox to the SBR planner.

- A physics engine [5] predicts the behavior of passive objects.
- An "abstracted physics simulation" predicts object behavior for longer time spans based on spatial relationship and previously observed scripts.
- A "social reasoning simulation" predicts the behavior of agents in a scene as a reaction to SBR actions. This simulation "spawns" a new instance of the SBR system per agent with the agent's parameters and simulates the agent's likely actions similar to [13].

The prediction subsystem can predict the behavior of the SBR system itself, as it can be modeled just like other actors. This can be thought to be part of a SBR "self-model".

6 SBR Planner

The SBR takes as input an initial scene and a goal and returns a number of plans, together with probability scores. The proposed planner includes several features from recent research:

- Spatial-temporal planning language: The SBR planner operates on scenes instead of FOL formulas.
- Probabilistic planning: SBR tasks have multiple outcomes.
- Timeline planning: SBR tasks take a certain time to complete.
- Multi-agent planning: The prediction subsystem predicts the behavior of agents and movable objects.
- Resource-bound operations: The planner will return first plans that are fast to generate, and then propose additional plans if time is available.

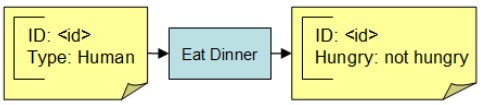

Fig. 6. Eating dinner satisfies hunger: Sub-scenes describe conditions and effects of tasks.

Planning with these characteristics is only partially understood as of today and the authors have not found any reference to practical systems combining stochastic planning and HTNs for more than toy domains. In the face of this situation, the authors sketch below a new approach that relies on "active tasks" and "worst-case analysis" in order to cope with the increased branching factor of probabilistic planning:

Active task. A planner task with a known decomposition, together with statistics about past executions of the task in the episodic memory, along the lines of

PRODIGY [25] and PELA [12]. Past execution may have included re-planning or escalation processes in order to deal with local failures, which have an impact on the cost of the action and its duration.

The statistics of past executions of active tasks are analyzed with respect to the factors leading to success.

Worst-Case Analysis. Undesired outcomes from all tasks in a plan are treated individually, as opposed to calculating probability distributions over "histories" [8]. Combined with the cost of executing the plan and the impact of a failed plan, the SBR planner can calculate a risk compensation and decide whether to pursue this path, develop a better plan or to choose non-action.

6.1 Example: Solving Equations

The following example of solving an equation demonstrates how 2D scene representations can provide the basis for symbolic reasoning.

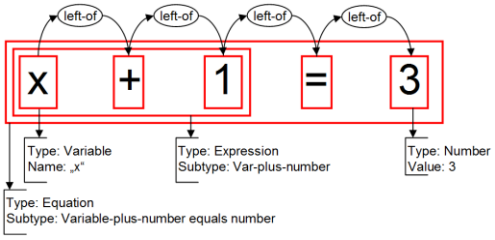

Fig. 7. Solving equations using the SBR planner

1. The 3D reconstruction subsystem passes the 2D sensor data to a statistical algorithm in order to recognize the type and value of each object, resulting in a symbolic representation.
2. The reconstruction subsystem determines the symbolic "left-of" spatial relationships between the objects.
3. The HTN planner then applies "actions" to the symbolic representation in order to simplify and solve the expression. Rules were learned during training sessions.

7 Reasoning About Plans

This subsystem implements several operations on plans that together allow SBR to acquire, simulate, optimize and reason about plans:

Plan recognition. Plan recognition analyzes sensor input in order to determine the plans of all involved agents [3]. This process detects errors during 3D scene recognition and triggers investigation and learning processes.

Plan simulation. The prediction subsystem's "abstracted physics simulation" allows to simulate object behavior in a scene, effectively creating a "sandbox".

Plan statistics. The episodic memory maintains a history of "scripts" of past plan executions, including the initial conditions and outcomes by means of the initial and

last scene. This allows to apply clustering and learning algorithms beyond the scope of this paper.

Plan optimization. Convert plans into 2D scenes using a "pen on paper" representation, compare, tweak, merge, mix and match different plans, pre-validate plans using simulation and execute the new plan in the real world. All of these optimization steps are implemented as meta-plans that can be optimized as well.

8 Logical Reasoning

The logical reasoning subsystem uses Description Logics (DL) [2] to maintains beliefs about the world together with confidence scores in a way similar to FLEX [20]. FLEX inference rules closely resemble SBR planner tasks, allowing the SBR planner to execute inferences rules directly without the need for a separate DL system. The DL "ABox" resembles SBR scenes, allowing to use DL in order to model symbolic object relationships.

Using this architecture, the system can talk about its beliefs ("all birds can fly": confidence=0.9), can test potential new hypotheses against a base of episodic memory cases and track "clashes" (contradictions during reasoning like "penguin P is a bird but doesn't fly") to the their axioms. New beliefs can be acquired via machine learning and checked against the episodic memory for consistency and explanation capability of actor's behavior. All of these operations are performed by "active tasks".

9 Attention Subsystem

The Attention Subsystem maintains a list of "persistent goals", a portfolio of plans and controls a "focus of attention" while tracking the execution of plans.

Persistent Goals. A list of medium and long term goals. Persistent goals are created manually by a human system operator (Asimov's laws of robotics), as a reaction to "urges" or by SBR-Planner as part of a plan that can't be executed immediately.

Attention Focus. Most of the time the attention focus lies with the images from a camera of a robot running SBR, but attention can also be focused on parts of the "self-model". Sensor data are processed by 3D reconstruction and passed on to the episodic memory in order to retrieve "associations", i.e. plans and scripts associated with the focused objects in their context. These "ideas popping up" are matched against active "persistent plans" in order to determine if the idea could contribute to an active plan.

When executing plans or "active tasks", the attention focus tracks the current vs. planned world state and initiates re-planning if necessary.

Portfolio of Plans. A set of plans created in order to satisfy the list of persistent goals. The attention subsystem evaluates the plans according to utility, cost and chance for success and execute the plan with the highest value.

10 Learning

Statistical learning is essential for a cognitive architecture based on a probabilistic planner. However, most references to learning algorithms have been omitted in the previous sections because their role is limited to auxiliary and relatively well understood tasks like calculating task success probabilities, guiding the planner search process or clustering parameter values in order to generate new concepts. Also, the exact choice of algorithms is irrelevant to the general AGI architecture.

This section summarizes the areas where statistical algorithms are employed:

3D Scene Reconstruction. Identify approximately objects and their positions from sensor data.

SBR Planner. Learn and propose applicable planner tasks to given problems, learn task decompositions, learn success factors for executing tasks.

Prediction Subsystem. Predict the behavior of agents as a script.

Episodic Memory. Maintain statistics about object occurrences in scenes, successful execution of tasks, identify scenes leading to successful plan execution.

Plan Reasoning. Classify plans for generalization.

Logical Reasoning. Classify concepts for generalization, learn DL implication links based on example.

Attention Subsystem. Learn the utility function of plans.

Also, non-statistical learning is employed:

Attention subsystem. When trying to "understand" an input 3D script, the plan recognition system will try to classify all objects and to determine the plans of all involved agents. Lack of such understanding may trigger active investigation, including "asking the operator" or getting closer to the agents in order to gather better sensor input.

11 Acknowledgment

The authors are grateful to Ben Goertzel, Sergio Jiménez, Anders Jonsson and José Hernandez-Orallo for their comments on early drafts of this paper.

12 References

1. Anderson, J.R., Lebiere, C.: The newell test for a theory of cognition. Behavioral and Brain Sciences 26(05), 587-601 (2003)
2. Brachman, R.J.: What's in a concept: structural foundations for semantic networks. International Journal of Man-Machine Studies 9(2), 127-152 (1977)
3. Carberry, S.: Techniques for plan recognition. User Modeling and User-Adapted Interaction 11(1-2), 31-48 (2001)
4. Croft, D., Thagard, P.: Dynamic imagery: A computational model of motion and visual analogy. In: Model-Based Reasoning, pp. 259-274. Springer (2002)
5. Davis, E., Marcus, G.: The scope and limits of simulation in cognition and automated reasoning. Artificial Intelligence (2013)

6. Erol, K.: Hierarchical task network planning: formalization, analysis, and implementation. Ph.D. thesis, University of Maryland (1996)
7. Fikes, R.E., Nilsson, N.J.: STRIPS: A new approach to the application of theorem proving to problem solving. Artificial intelligence 2(3), 189-208 (1972)
8. Ghallab, M., Nau, D., Traverso, P.: Automated planning: theory & practice. Elsevier (2004)
9. Gil, Y.: Description logics and planning. AI Magazine 26(2), 73 (2005)
10. Goertzel, B., Pennachin, C., Geisweiller, N.: Engineering General Intelligence, Part 1, Springer (2014)
11. Hartanto, R., Hertzberg, J.: Fusing DL reasoning with HTN planning. In: KI 2008: Advances in Artificial Intelligence, pp. 62-69. Springer (2008)
12. Jim_enez Celorrio, S.: Planning and learning under uncertainty. Ph.D. thesis, Universidad Carlos III de Madrid, Escuela Politécnica Superior (2011)
13. Konolige, K., Nilsson, N.J.: Multiple-agent planning systems. In: AAAI. vol. 80, pp. 138-142 (1980)
14. Laird, J.E., Newell, A., Rosenbloom, P.S.: Soar: An architecture for general intelligence. Artificial intelligence 33(1), 1-64 (1987)
15. Langley, P.: An adaptive architecture for physical agents. In: Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on. pp. 18-25. IEEE (2005)
16. Langley, P.: Altering the ICARUS architecture to model social cognition (2013), http://www.isle.org/~langley/talks/onr.6.13.ppt
17. Langley, P., McKusick, K.B., Allen, J.A., Iba, W.F., Thompson, K.: A design for the ICARUS architecture. ACM SIGART Bulletin 2(4), 104-109 (1991)
18. Metzinger, T.: Being no one: The self-model theory of subjectivity. MIT Press (2003)
19. Nuxoll, A.M., Laird, J.E.: Extending cognitive architecture with episodic memory. In: Proceedings of the National Conference on Artificial Intelligence. vol. 22, p. 1560. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 (2007)
20. Quantz, J.J., Dunker, G., Bergmann, F., Kellner, I.: The FLEX system. KIT Report 124, Technische Universität Berlin (1995)
21. Quillian, M.: A notation for representing conceptual information: An application to semantics and mechanical English paraphrasing, sp-1395. System Development Corporation, Santa Monica (1963)
22. Santos Jr, E.: A cognitive architecture for adversary intent inferencing: Structure of knowledge and computation. In: AeroSense 2003. pp. 182-193. International Society for Optics and Photonics (2003)
23. Schank, R.C., Abelson, R.P.: Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. Erlbaum (1977)
24. Strauss, P.S.: IRIS inventor, a 3d graphics toolkit. In: Proceedings of the Eighth Annual Conference on Object-oriented Programming Systems, Languages, and Applications. pp. 192-200. OOPSLA '93, ACM, New York, NY, USA (1993), http://doi.acm.org/10.1145/165854.165889
25. Veloso, M., Carbonell, J., Perez, A., Borrajo, D., Fink, E., Blythe, J.: Integrating planning and learning: The prodigy architecture. Journal of Experimental & Theoretical Artificial Intelligence 7(1), 81-120 (1995)