A Simplifying Diagrammatic Representation of Crisp and Fuzzy Teleo-Reactive Semantic Circuitry for Application in Robotic Agent Task Control

Edward P. Katz

Visiting Scholar
c/o Professor Nils J. Nilsson
Robotics Laboratory, Department of Computer Science
Stanford University, Stanford, CA 94305
katz@robotics.stanford.edu
and
Supply Chain Information Systems
Hewlett-Packard Company
1501 Page Mill Road, MS 5L-A
Palo Alto, California 94304
ed_katz@hp.com

Abstract

*The Teleo-Reactive paradigm for computing and organizing actions for autonomous agents in dynamic environments is intended to enable the construction of semantic circuitry for the continuous computation of parameters and conditions on which agent action is based. However, both crisp and fuzzy semantic circuitry suffers from complexity growth. This paper describes a representation for simplifying semantic circuits capable of realizing both crisp and fuzzy Teleo-Reactive sequences. The new representation supports any combination of primitive actions, action sets, and non-primitive sub-circuit actions while reducing the semantic circuitry complexity. The proposed representation is constructed by identifying the repetition of semantic components and abstracting them into a standard **EISA** model.*

1. Introduction

The Teleo-Reactive (T-R) paradigm [Nilsson94, Nilsson92] for agent control (whether robotic or software agent) is influenced by the need for autonomous agents, such as mobile robots, to *operate in dynamic and uncertain environments*. The T-R paradigm uses continuous sensor processing and durative rather than discrete actions to form a task-level *middleware*. A T-R program consists of an ordered production rule set and is compiled into and executed from the equivalent semantic circuitry. Each rule action may be a *primitive* action, an *action set* composed of parallel actions, or a *non-primitive* T-R sub-sequence (or tree) for hierarchical capability. An increase in T-R program size (i.e. number of levels plus the complexity of actions) results in a complexity increase to the semantic circuitry. Despite the continuous nature of the sensor processing and the continuity of durative actions, the T-R's crisp predicates constrain behavior equivalent to state transition, preventing possible smooth action transitioning and preventing concurrent action execution occurring in different rules.

A fuzzy T-R extension [Katz97] replaces crisp-valued predicates and boolean functions with continuous-valued predicates and logical functions based upon fuzzy sets and relaxes the constraints regarding T-R sequence rules. State transition-like behavior is now optional while supporting the durative execution of several rule's concurrent actions and providing smooth transitions between rules. However, the fuzzy extension also suffers from the same semantic circuitry complexity growth as Nilsson's crisp version.

The proposed representation presented here is constructed by identifying the repetition of semantic components and abstracting them into a standard model.

2. Crisp Teleo-Reactive

An agent control program is a T-R sequence that directs the agent toward a goal (hence *teleo*) while at the same time responding (hence *reactive*) to the perceived dynamic environmental changes in circumstances that the agent encounters. Drawing on the notion of analog circuitry, T-R is intended to provide an agent task execution environment having a more continuous nature than one based on a state transition model.

The T-R paradigm is conceived to be a kind of task-level *middleware* for agent programming (whether robotic or software agent). The assumed *upperware* would contain the high level planning activities needed to determine what the agent should do and to create the corresponding T-R sequence(s). Likewise, the assumed *lowerware* would process the T-R action commands in order to realize the effects of these action commands (e.g. decoding *move* and *rotate* actions into motor voltages to turn the wheels). Should an unanticipated situation arise for which the available T-R sequence(s) can not handle, the upper level planning activities would be invoked to generate a new T-R sequence capable of negotiating the new situation.

A T-R sequence resembles a set of ordered production rules in its simplest form (figure 1a). The conditions (K_i's) are boolean expressions containing predicates about sensory inputs and other information from the world model and are continuously computed with K_1 considered the goal condition. The actions, a_i's, themselves, can each be either a *primitive action*, a concurrent *action set*, or they can be *non-primitive* T-R sequences themselves providing hierarchical capability. A simple T-R sequence can be thought to be *compiled* into and executed from semantic circuitry (figure 1b). In both figures, all conditions, K_i, are continuously computed. As Nilsson describes [Nilsson94],

> *The condition K_1 is taken to be the goal condition, and the corresponding action, a_1, is the null action. The condition K_2 is the weakest condition such that when it is satisfied (and K_1 is not), the durative execution of a_2 will (all other things being equal) eventually achieve K_1. And so on. Each non-null action, a_i, is supposed to achieve a condition, K_j, strictly higher in the list ($j<i$).*

From the diagram in figure 1b, the negation of the condition K_1 signal *squelches* any lower actions from being energized. Thus, if K_1 is true, no lower rule action (higher index) can be active. If K_1 is not true, then likewise, the highest condition, K_j, which is true will inhibit all activations of the lower rules, K_{j+1}, ... , K_n. Thus, in a semantic circuit, a condition, K_j, energizes its corresponding action, a_j, if it is true and there are no higher level conditions (K_1, ..., K_{j-1}) that are also true. K_j also energizes a *squelch* signal to all lower level rules preventing those actions (a_{j+1}, ..., $a_{n)}$ from being energized. The *squelch* signal is the semantic circuit's mechanism for disabling the lower rule action durative executions.

Suppose in Botworld, a two-dimensional simulated world [Nilsson94], bots can move, rotate clockwise, grab bars, and move bars to various locations. Figure 2 is an example of a bar grabbing T-R sequence where a bot is to move to the point *bar center* in order grab bar A. The bot can sense its environment and can evaluate conditions which tell it whether or not it is already grabbing bar A (*is-grabbing*), facing toward bar A (*facing-bar*), positioned with respect to bar A so that it can reach and grab the bar (*at-bar-center*), on the perpendicular bisector of bar A (*on-bar-midline*), and facing a zone on the perpendicular bisector of bar A from which it would be appropriate to move toward bar A (*facing-midline-zone*). The bot is capable of executing primitive actions *grab-bar*, *move*, and *rotate* (clockwise).

Execution of the T-R sequence will result in the bot moving to and grabbing bar A if it is not already holding it. Initially only the last (or default) rule is true (figure 3a) and causes its primitive action, *rotate*, to start the bot executing a clockwise rotation about its axis and continues until the bot is **exactly** facing the midline-zone. Recognition of this zone causes the predicate *facing-midline-zone* to become true. The new rule's corresponding action, *move*, is now energized and duratively executed while the *rotate* action ceases. The bot continues forward until it **precisely** reaches the midline-zone causing the condition *on-bar-midline* to become true. The current *move* action is disabled and the corresponding *rotate* action begins. Processing continues in this way for the other rules until finally the bot is holding the bar.

2.1 Fuzzy-Teleo-Reactive

Binary predicates can sometimes be a constraint when dealing with real world sensor readings and perceived conditions which may not exhibit crisp outputs. Substituting boolean logic and its binary predicates with fuzzy logic and its continuously-valued counterparts provides a more consistent foundation triplet. Since fuzzy logic is a generalization of boolean logic, all desirable aspects of crisp T-R are preserved. A fuzzy T-R extension [Katz97] can be constructed by using continuous-valued predicates, flouncing actions, and action blending.

2.1.1 Continuous-valued Predicates

Suppose predicates now have a range over the continuous interval [0,1], they can also be partially true by exhibiting a degree of truth (degree of fulfillment). For example, this could allow a position predicate (e.g. *at-bar-midline*) to also indicate the degree of nearness if its strength is positive but less than 1 (.i.e. μ(*at-bar-midline*) is a shorthand notation for the degree of truth of the fuzzy predicate, μ(*position* is *at-bar-midline*).

Referring to figure 2b, let the negation ($\sim$) and conjunction ($\wedge$) functions in the semantic circuit be replaced by the corresponding Zadehan negation and T-norm. This time in the bar grabbing example, the predicate values are assigned the truth values of the fuzzy set membership functions. In applying these extensions to our example, the

bot's forward sensor may have some sensitivity distribution which could function as a kind of peripheral vision. When the T-R sequence (figure 2a) begins again as depicted (figure 3b), the bot is initially rotating clockwise on its axis. This time as the bot begins to face the midline zone, the forward sensor begins to ***partially*** sense it. When this happens, the *facing-midline-zone* predicate now has a strength value $0 < \mu(\textit{facing-midline-zone}) \leq 1$, which represents the degree to which the forward sensor registers the midline-zone directly in front of the bot.

2.1.2 Functional Actions

Let actions have the *potential* for executing in a manner related to the corresponding condition's strength $\mu(K_i)$ (i.e. $a_i(\mu(K_i))$). For example, if a_i is a *move* action, then it *might* become a straight-line move at a speed linearly proportion to $\mu(K_i)$ (e.g. linear velocity = $\mu(K_i)$ **x** maximum linear velocity). Binary predicates via fuzzy set singletons emulation are used where needed as in the case of the *is-grabbing* binary predicate (since partially holding a bar has little meaning).

2.1.3 Action Blending

An sample execution snapshot of the example has $\mu(\textit{facing-midline-zone}) = 0.25$ and all other higher conditions are false (i.e. $\mu(K_i) = 0, 1 \leq i < 6$). With continuously-valued predicates and their corresponding "partially" energized actions, the bottom rule will only be partially inhibited (i.e. $\mu(\sim\textit{facing-midline-zone}) = 1\text{-}\mu(\textit{facing-midline-zone}) = 0.75$). Several (or potentially all) rules can be active, each to a different degree. Several *move* and several *rotate* actions can each be energized concurrently and resolved by applying separate defuzzification algorithms. One defuzzifier returns a crisp move velocity, and another a crisp rotate velocity. Multiple actions of the same nature (e.g. several *move* actions) can be ***blended*** [Saffiotti93] via defuzzification. (See [Katz97] for a more detailed description of fuzzy T-R.)

3. EISA Model

Figure 2b illustrates that as the size and complexity of the semantic circuitry grows, so does the number of signal lines regardless of whether crisp or fuzzy T-R. To reduce complexity (e.g. signal line growth), first we must identify the components of a typical semantic circuit.

Every action in the semantic circuit has a control conjunction which determines when the action may be applied (and to what degree in the fuzzy case). Each succeeding circuit layer from the top adds an additional signal line augmenting those from above until the bottom layer (representing the last rule in the T-R sequence) contains a *n*-way conjunction for the last (default) rule. This propagation of parallel signals to lower layers can simplified.

The negated condition signal ($\sim K_i$) transmitted to the lower layer(s) is a *squelch* signal (i.e. suppress any actions in the lower layers representing the lower rules, if K_i is true). This particular signal at the next lower layer is an *enable* signal and in conjunction with all the parallel enable signals from the upper layers together with layer's condition, K_{i+1}, control the action a_{i+1}'s durative execution. This process is repeated for each succeeding lower layer. The nth layer has an *n*-ary conjunction which can be recursively split into an (n-1)-ary conjunction enable signals from the upper layers and a binary conjunction and recombined at each level to form an EISA block (figure 4b) with two inputs *(Enable, Input)* and two outputs *(Squelch, Activate)*. This serialization maintains the same functional semantics while keeping linear complexity growth.

The *EISA* model can be readily used with non-primitive action T-R sub-sequences. A guarded move (figure 5b) replaces the primitive move for the purpose of moving around unexpected obstacles partially blocking the intended path (figure 6). Each guarded move's activation (degree of activation for the fuzzy case) is controlled by the rule's strength (degree of fulfillment) and rotates the bot as it moves forward avoiding an obstacle.

Acknowledgments

This investigation effort has benefited greatly from the collaboration and encouragement of Nils Nilsson.

References

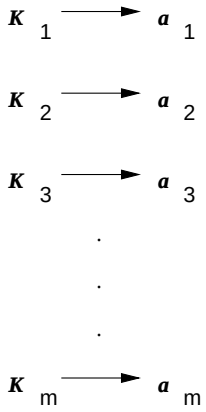
1.[Nilsson92] Nilsson, Nils J., *Toward Agent Programs with Circuit Semantics*, Tech. Rep. STAN-CS-92-1412, **Department of Computer Science**, Stanford University, 1992.

2.[Nilsson94] Nilsson, Nils J., *Teleo-Reactive Programs for Agent Control*, **Journal of Artificial Intelligence Research**, Vol. 1, 1994, pp. 134-158.

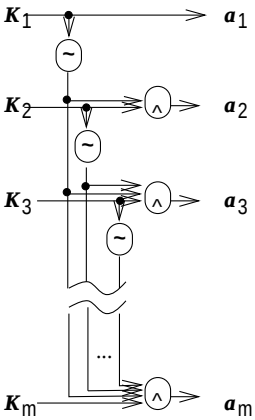
3.[Saffiotti93] Saffiotti, A., Ruspini, E., and Konolige, K. *A Fuzzy Controller for Flakey, an Autonomous Mobile Robot*, Technical Note 529, **AI Center**, **SRI International**, 333 [Zadeh65] Zadeh, L.A., *Fuzzy Sets*, **Information and Control**, Vol. 8, pp. 338-353, 1965.

4.[Katz97] Katz, Edward P., *Extending the Teleo-Reactive Paradigm for Robotic Agent Task Control Using Zadehan (Fuzzy) Logic*, 1997 **IEEE International Symposium on Computational Intelligence and Automation (CIRA '97),** Monterey, California, July 1997, pp. 282-286.

5.[[Bugarin98] Bugarin, A.J., Barro, S., *Reasoning with Truth Values on Compacted Fuzzy Chained Rules*, **IEEE Trans. on Systems, Man, and Cybernetics.**, Part B, V. 28, No.1, Feb 1998, pp 34-46.

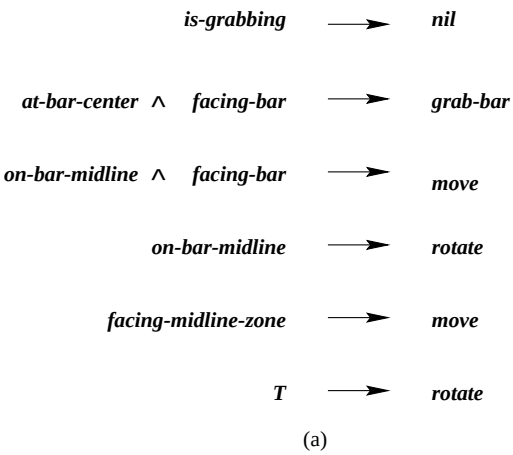


(a)

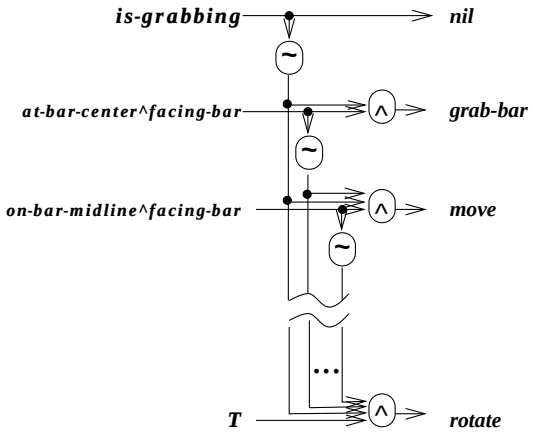


(b)

Figure 1.



(a)



(b)

Figure 2.

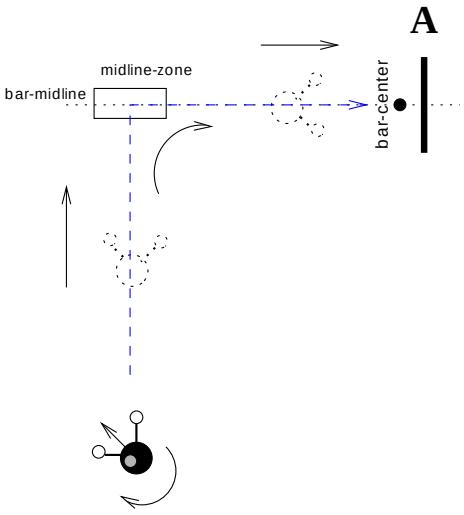


Figure 3a.

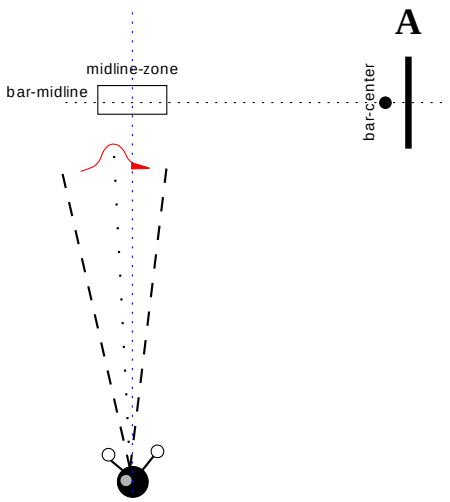


Figure 3b.

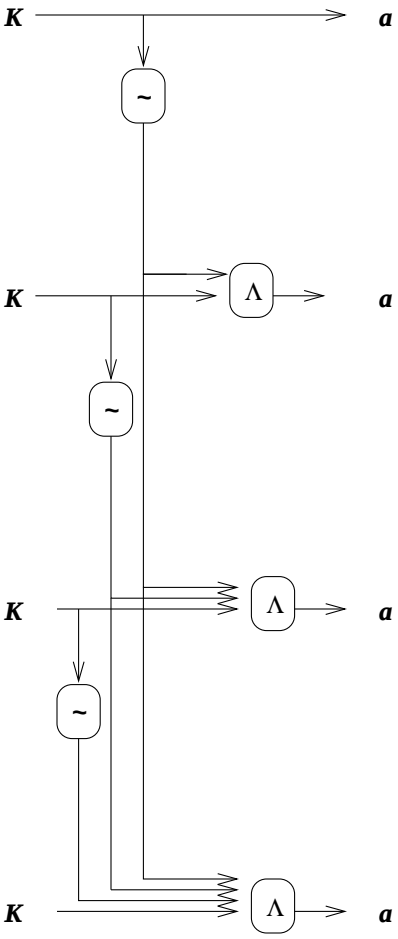


Figure 4a.

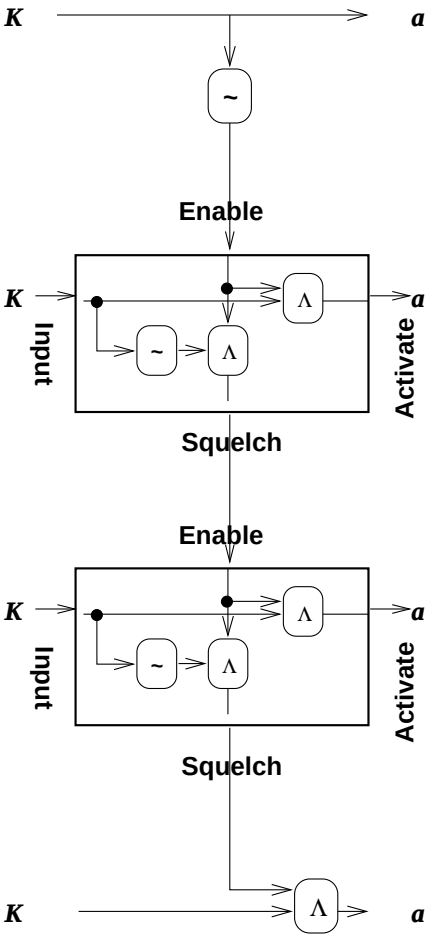


Figure 4b.

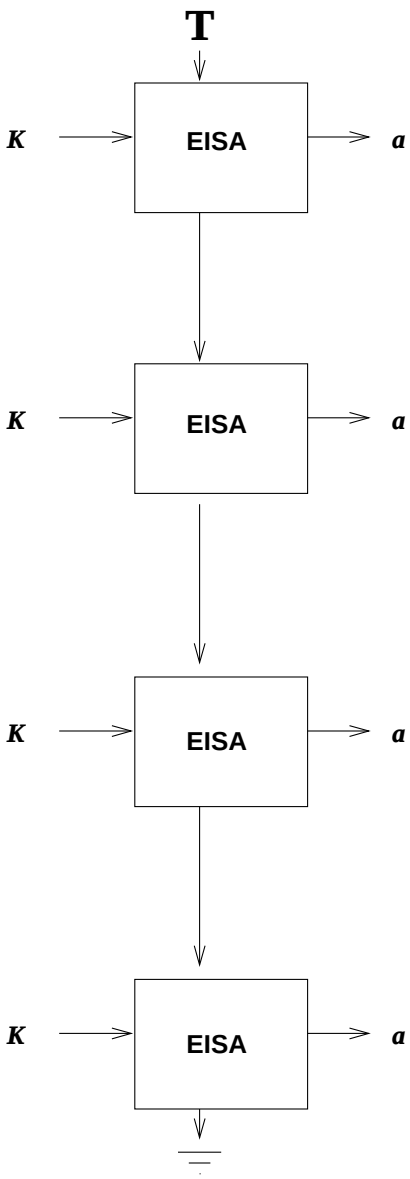


Figure 4c.

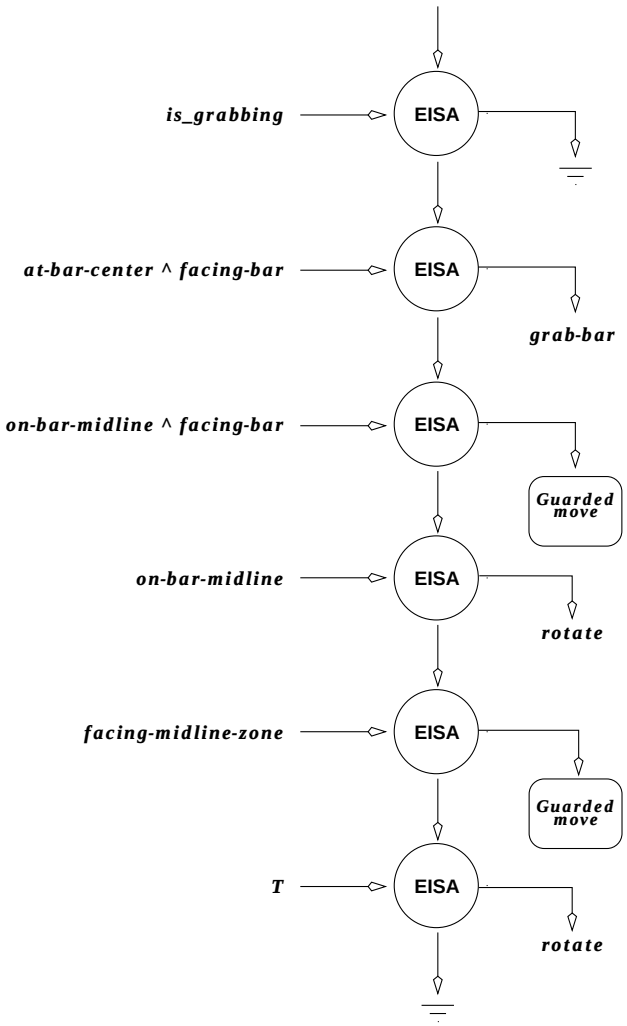


Figure 5a.

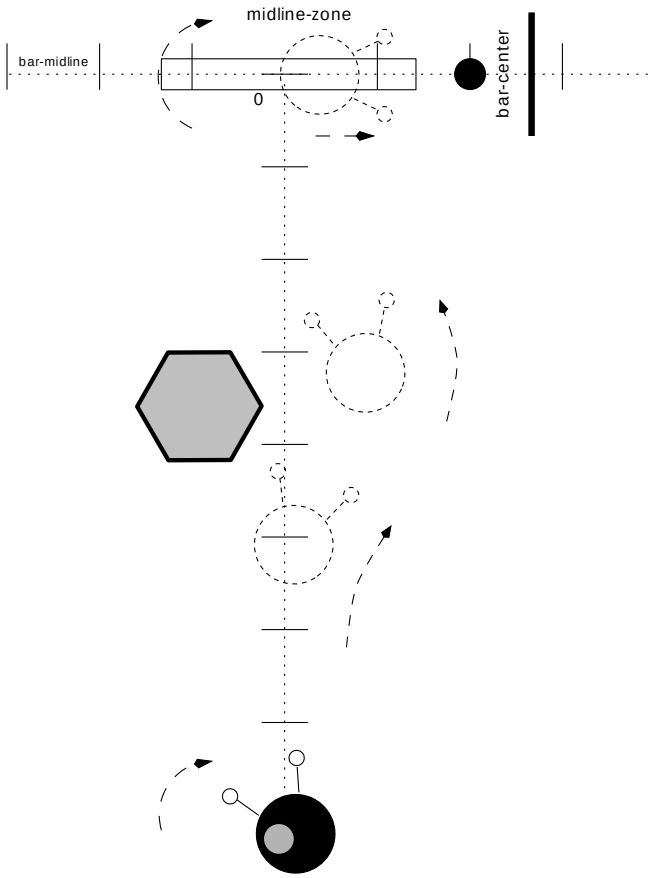


Figure 6.

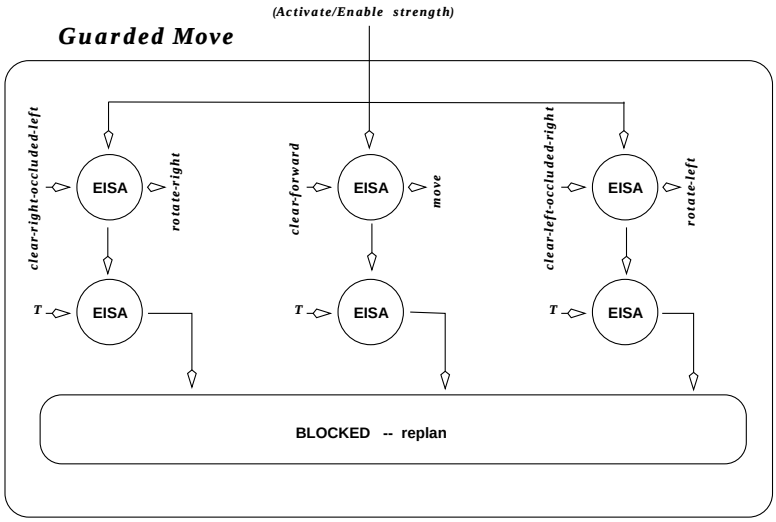


Figure 5b.