

On the Behavior-based Architectures of Autonomous Agency¹

Amol Dattatraya Mali (Member, IEEE)
Dept. of electrical engg. and computer science
University of Wisconsin, Milwaukee, WI 53211, USA
Phone - 1-414-229-6762, Fax - 1-414-229-2769
mali@miller.cs.uwm.edu

Abstract

A number of autonomous robots with varying degree of reactive functionality have been built, based on different architectures. We review the foundations, limitations and achievements of a number of architectures of such autonomous agents from the three categories - reactive, deliberative and hybrid. Most of these architectures contain behaviors. The principle of avoiding an explicit representation of goals in the purely behavior-based robots has limited their achievements to simple tasks like box pushing, pipe inspection and navigation. This paper makes two contributions - (i) reviewing autonomous agent architectures and (ii) proposing a new class of architectures where behaviors are coupled and/or markers are introduced in environment, without a planner or sequencer and without an explicit representation of

¹Appears in IEEE Transactions on Systems, Man and Cybernetics (Part C: Applications and Reviews), Vol. 32, No. 3, August 2002, pp. 231-242.

goals and investigating tradeoffs in these architectures. We develop a model of behaviors, environmental modification and goals and then show how the behavior-based robots can be made goal-directed. The tradeoffs in increasing their goal directedness are examined. Defining the notion of coupling that captures dependency within the internal structure of a behavior space, it is shown that more complex goals demand higher coupling or more behaviors or a modification to the environment. These novel tradeoffs show a new spectrum of architectures for integrating goals and the behavior-based reactive functionality.

I. INTRODUCTION

Artificial Intelligence paradigms today are moving towards a more distributed agent-based architecture. It is argued that when intelligence is approached in such an incremental manner, the reliance on global representations and reasoning disappears [1][2][3]. That the agency has had a deep impression on Artificial Intelligence is clear from the fact that several journal issues have recently been devoted to debating this approach (Artificial Intelligence v.47, 1991, Robotics & Autonomous systems, v.6:1, 1990, Cognitive Science v.17, 1993, Artificial Intelligence v.73 1995, Autonomous

Robots, Journal of experimental & theoretical artificial intelligence, Vol. 9, No. 2/3, 1997). Early research on robots assumed their environments to be completely static, predictable and avoided sensory processes, e.g. in the first robot project [4], Shakey executed simplified plans and manipulated blocks, with full a priori knowledge of the types of objects etc. Maintenance of internal representations was considered to be responsible for the failure of this classical planning [4] in more complex environments.

Definitions of agents, multi-agent systems, software agents and purely situated agents are given in [5]. The agents dealt with in our work are close to the purely situated agents defined in [5]. Specifically, the agents in our work are physical or computing entities that are (i) situated in an environment, (ii) possess resources of their own, (iii) can perceive their environment, (iv) have practically no representation of their environment, and, (v) execute behaviors which may offer useful services. Autonomous agents are not directed by commands from a user. Instead, they are directed by a set of tendencies. Our work is about a single agent. Autonomous agency became popular with the work of Brooks [1],[6], who challenged the deliberative paradigm in AI (the classical planning paradigm that was also slow due to brute force search) by building stimulus-response based robots based on his subsumption

architecture, also known as behavior-based robots. A typical behavior-based agent is a collection of several independent task-achieving modules, with a simple distributed control mechanism. Each behavior mediates directly with the external world and the behaviors are in a parallel control structure, as opposed to the traditional serial structure where interaction with the world is processed serially through sensors, reasoners, planners, actuators, etc. All behavior systems attempt to reduce or eliminate the centralized shared memory, relying instead on parameter passing and communication between individual behaviors. Robots in such systems (also called *situated robots*) are largely reactive and communicate through the world by making changes to it that other robots can perceive. It is argued that intelligence emerges from the interaction of the robots with the world, since such an evidence has been found in social insects like ants, wasps and bees. Behavior-based robots avoid explicit planning and an explicit representation of goals. Since some such reactive robots exhibited problems like deadlocks and myopic functionality, hybrid architectures with a deliberative component to fix these problems as in [7] began to be explored. The previous research on autonomous robots has concentrated on adding separate modules to make behavior-based robots goal-directed. The conventional wisdom is - autonomous robot architectures

should be tiered and the lack of tiering places a serious limitation. A large number of hybrid architectures combining planning and agent-environment dynamics were then proposed, with the hope that reactivity would give faster real time functionality and deliberation would provide goal fulfilling capability. However the planning component (which was to be eliminated) dominates the architectures, leaving little scope for the reactive functionality.

This paper makes two contributions - (i) providing a review of autonomous agent architectures and (ii) proposing a new class of architectures where behaviors are coupled and/or markers are introduced in environment, without an explicit planner or sequencer and an explicit representation of goals and investigating tradeoffs in these architectures. The new architectures proposed allow modification of the behavior structure and an environment for fulfilling goals, rather than introducing additional modules like a planner and/or a sequencer. We show that an environment can have an important role to play in increasing the goal-directedness of the behavior-based robots. Some tradeoffs in these architectures are also investigated. It is not claimed that introduction of coupling between behaviors and/or introduction of markers into an environment can completely eliminate a planner. Also, coupled behaviors and a marker-augmented environment are not incompatible with an

explicit representation of goals and a planner. Coupling and/or introduction of markers eliminates explicit representation of some goals and also the planning capability needed to fulfill these goals and other goals can be explicitly represented a planner can be used to achieve these.

The relations between the dependency within a behavior space (defined in terms of *coupling* in section II), goals that can be fulfilled by it and its environment are examined. The mechanisms of externalizing internal states that increase the reactive functionality are discussed. It is shown how an increase in the complexity of goals affects a behavior space and its environment. These tradeoffs indicate a new spectrum of architectures for integrating goals and reactive functionality.

The computational structures of the current architectures of autonomous agency are reviewed (sec. II). We review their achievements (sec. III) and discuss their limitations (sec. IV). A model of environmental and behavioral modification is developed (sec. V) and the model is used to arrive at the results on the tradeoffs in architectures allowing these modifications, without a planner and without an explicit representation of goals (sec. VI). We discuss how a house cleaning robot can be designed using our approach (sec. VII) and compare our approach with several other paradigms (sec. VIII). Conclusions

are presented in section IX.

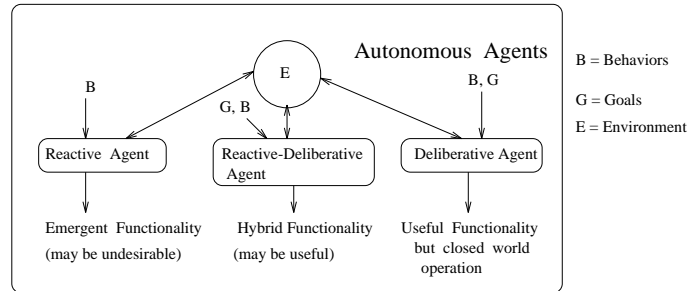


Figure 1: Three dominant types of architectures of autonomous agency

II. ARCHITECTURES OF AUTONOMOUS AGENCY

In this section, we explain three dominant classes of architectures of autonomous agency and then describe key features of several architectures from these classes. Autonomous agents which have a reactive/behavior-based component in their architectures interact with their environments continuously. The current autonomous agent architectures can be classified into three categories (showed in Figure 1) - **1**. Reactive agents that follow the “sense-act” cycle. They exhibit functionality that arises out of sequential and concurrent execution of behaviors. For tasks like simple navigation, the behaviors are generally purely reactive. Behaviors for more complex tasks have a local internal state over which only minimally necessary reasoning is carried out. Behavior-based robotic systems provide a means for a robot to navi-

gate in an uncertain and unpredictable world without planning, by endowing the robot with behaviors and co-ordinating them independently [8] (page 67, chapter 3). Many behavior-based robots resolve conflicts among behaviors by prioritizing them. For example, a can-picking behavior is suppressed by a safety-maintaining behavior since survival of robot is necessary for performing any task. These agents may exhibit undesirable functionality like cyclic behavior. We also refer to these agents as those based only on “agent-environment” dynamics. These architectures were developed to avoid computational complexity of planning and maintenance of a global world model. Brooks considered planning as a way of avoiding figuring out what to do next.

2. Reactive agents augmented with deliberative control to prohibit undesired functionality. Deliberative capability is generally provided in the form of a planner that monitors and controls the execution of reactive behaviors. Agre [9] says that just as planning offered no robust account of moment-to-moment interaction with the world, reaction offered no robust account of how an organism’s or robot’s actions could be guaranteed to work. This concern inspired the development of these hybrid architectures. Hybrid architectures were developed to combine the best of both worlds - fast real-time response from reactive behaviors and reliability of a planner at achieving goals. **3.** De-

liberative agents that use internal world models and make the closed-world assumption. Next we briefly discuss some implementations from these three classes in this order.

Behavior-based/Reactive: Behavior-based robots [10],[11] perform navigation using several navigational behaviors. Behavior-based robot HERBERT [12] picks soda cans and places them at a repository. Just like ants leave pheromone trails, the behavior-based robots [13] encode information into their physical environments (that other robots exploit) to reduce sensing, actuation and computations. Maes [14] proposes action selection that is an emergent property of activation-inhibition dynamics among actions and an environment.

Hybrid: In [15], a high level planning module selects and parameterizes the behavioral sets based on the mission requirements, environmental conditions and internal state of a robot. Bonasso [16] proposes a hybrid reactor-planner architecture. His reaction plans contain an explicit representation of the goals they fulfill and an ordering over the behaviors. The reaction plans are similar to hierarchical task networks [17] because a reaction plan solves the problem of fulfilling an abstract task that requires to be decomposed into lower level less abstract tasks. The agents in [18] too use

hierarchical task network planning. This approach is a hybrid architecture for softbots. Softbots (software robots) are like physical robots in several aspects, instead of motor-driven wheels and effectors, they have commands like “ftp”, “telnet”, “mail”, “rm”. Instead of sensors, they have facilities like “gopher” and “netfind”. Softbot behaviors are controlled by a planner [19] to fulfill more complex tasks, like answering a complex query. Universal plans [20] anticipate every possible action in a domain and prescribe an action for every initial state, so that off line planning can be used as an on line reactivity. Then responses to different situations can be retrieved from the lookup table. Rosenschein & Kaelbling [21] use goal reduction rules that specify how a high level goal is transformed into lower level goals. When given a fixed explicit goal to satisfy and a world model, their Gapps program generates a provably correct reactive program for that goal. Saffiotti et al [22] integrate planning and reactive control. Weights are assigned while combining the outputs of individual behaviors. This architecture differs from other hybrid architectures because of the use of multivalued logics. Stimuli and consequences of these behaviors are fuzzy. Arguing that previous situated agents do not have goals, Planner and reactor interact iteratively and an accurate environmental model is used to identify useful patterns of interaction, in the

hybrid architecture [23]. Simmons [24] proposes a hybrid architecture called task control architecture.

Deliberative: Stanford cart [25],[26] uses a deliberative architecture. It maintains a global world model. It is very slow (moves one meter every ten to fifteen minutes). Its full run lasts about five hours. Obstacles are added to its internal world map as detected and the cart used a graph search algorithm to find the shortest path through this abstract model. Deliberative robot Shakey [27] computed and executed plans for goals, but its knowledge of the world was stored in its knowledge base. The world of robot HILARE [28] contained smooth flat floors found in typical office environment. It conducted planning within a multi-level representational space - geometric models represented the actual distances and measurements of the worlds and a relational model expressed the connectivity of rooms and corridors.

III. ACHIEVEMENTS OF THE AUTONOMOUS AGENT ARCHITECTURES

In this section, we review achievements of autonomous agency that are based on the types of architectures discussed in the previous section. Anderson & Donath [10] report several navigation behaviors for object following, wandering and obstacle avoidance, implemented on real robot SCARE-

CROW. Kadonoff [11] reports several local navigation behaviors for avoiding obstacles and following walls and paths. Mataric [29] describes a behavior-based robot that navigates using a map of the environment and plans paths. Navigation behaviors are implemented on robots in [16]. Office navigation behaviors implemented on a robot are reported in [30]. Arkin [15] reports several behaviors for 3-D terrain navigation. Robots [13] have behaviors for maintaining communication with the previous and the next robot in the row and aligning with the row. The mobile robot [22] navigates indoors. Several robot navigation behaviors are reported in [31]. Connell [32] reports navigation behaviors implemented on a toy car. This shows that there are many autonomous robots whose functionality is limited to navigation. We mention next, autonomous robots that perform some other tasks, in addition to navigation.

A rover Rockey-III that navigates through rough outdoor terrain and collects soil samples has been reported in [33]. Robots [34] collect trash. Kube & Zhang [35] report behavior-based box pushing robots. Decker et al [18] discuss a set of information handling behaviors for autonomous agents, like responding to repetitive queries, monitoring information sources, advertising capabilities and self cloning. Bonasso et al [36] describe a robot

for find and fetch tasks in an outdoor environment.

Interesting results have been achieved using behavior-based approach in the 6 legged walking robot [37], behavior-based flying vehicle that won the aerial robotics competition [38], 12 degree of freedom hyper-redundant serpentine robot that navigates complex pipe structures and inspects them [39], walking robot for exploring volcanic craters [40], drum sampling [41], mobile manipulation [42], non-holonomic robot for box pushing and ball dribbling [43], mobile security guards [44], SONY's entertainment robotic dog [45], modular self-reconfigurable robots [46] and the robot that learns to push boxes [47]. Additional work on behavior-based robots includes [48] and [49]. Many more implementations of autonomous robots are reported in [8] and [50].

IV. LIMITATIONS OF THE AUTONOMOUS AGENT ARCHITECTURES

As shown in section III, most of the work on behavior-based autonomous agents has been successful only in simple tasks like navigation. Box pushing, balloon bouncing, ball dribbling, entertainment and collection of soil samples are some of the very few tasks where capabilities other than navigation are needed, where behavior-based autonomous agents have been shown to be

successful.

In this section, we review limitations of the current architectures of autonomous agency. Cyclic wandering, a typical characteristic of purely reactive robots, is reported in [10]. Robots [35] can get stuck in cycles. Connell [12] reports cyclic behavior of a can collection robot which picked up and dropped the same can repeatedly. Hartley & Pipitone [51] claim that subsumption architecture [52] is not sufficiently modular and that a clean interface between different behaviors is desirable. Several architectures that claim to be reactive contain an explicit planner. We view this as a limitation of these architectures since they are unable to show successful reactive functionality without an explicit planner. Kaelbling [53] claims that her architecture is reactive, but it includes a world model that is used by a planner. To compose behaviors, she uses procedures called “mediators”. The reactive planning architecture in [54] maintains and updates a world model, for synthesizing plans in response to situations. Since it is computationally expensive to construct plans for all possible situations, the universal plans [20] are considered to be infeasible. During the program synthesis phase in [21], the environment is assumed to be static. There is no emphasis on continuous agent-environment interaction. The offline symbolic reasoning

that synthesizes the programs involves the notoriously hard goal regression planning.

Another limitation of hybrid architectures is that there is no methodology to decide which tasks should be assigned to the reactive module and which to assign to the planning module. Narasimhan [55] uses planner and reactor in a non-conventional manner. Planner is used all the time and reactions are used for exceptional situations. This is similar to use of planner and reactor in [24]. Complex goals are achieved in [22] by deliberation technique like goal regression planning that is combinatorially hard.

Yet another limitation of reactive architectures is inability to handle goals without their explicit representation. Maes [14] says that there is a tradeoff between goal orientedness and reactivity. Goals are explicitly represented in her system. If an agent is highly reactive, it is less goal fulfilling and if it is highly goal fulfilling, it is less reactive. The activations of the behaviors can be varied to model various degrees of goal-fulfillment and reactivity. The length of a precondition list, add list or delete list affects the input and output of activation of a behavior. This ad-hoc method of computing activation makes the action selection less rational and less reliable than that of the sophisticated deliberative planners.

Many systems like [36] which are claimed to be reactive contain some sort of explicit planner or sequencer. It is concluded by Firby & Simmons [56] that hybrid architectures do not combine reactive and deliberative paradigms effectively. The interface between deliberation and reactivity is poorly understood (page 234, chapter 6 [8]). It is not clear how the overall functionality should be divided among reactive and deliberative components. Kirsh [57] cogently argues that the duplication of insect behaviors like wandering, avoiding obstacles and following corridors does not prove that mobile robotics is a “royal path to high level behaviors”. Chapman [58] says that combining classical planners with reactive systems combines the worst of both worlds - planning is expensive and reactivity is myopic, their combination is useless. Gat [59] warns that it is possible for the evils of planning to arise in the reactive architectures, if one is not careful. Maes [60] says that because of the task-driven and pragmatic approaches, the autonomous agents are more like a “a bag of hacks and tricks” than an embodiment of general laws and principles. Rosenschein & Kaelbling [21] point out that though reactive architectures have been proposed as an alternative to traditional AI, their theoretical foundations are less developed.

A natural question to ask is - can the behavior-based reactive autonomous

agents scale beyond navigation, preserving highly reactive functionality, without introducing an additional module like the planner in the hybrid architectures? If so, what modifications are needed to the computational structure of an agent and its environment to achieve this? What are the tradeoffs? This is examined in section VI using the model of autonomous behavior-based agency that is built in section V.

V. A MODEL OF BEHAVIOR-BASED AUTONOMOUS AGENCY

First order logic is used to represent the behaviors, since complex goals and constraints can be expressed more easily in logical form [22]. Despite the differences in representations, our results continue to be applicable to real systems. The use of predicate logic to represent preconditions and effects of actions is quite common in the implementations of autonomous agency, as in [52],[12],[53],[19],[20],[21],[14],[32] and [27].

Some notation from first-order logic that we use includes - \exists (existential quantifier), \forall (universal quantifier), \Rightarrow (logical implication), \vee (disjunction), and \wedge (conjunction). \wedge and \vee are also called “logical AND” and “logical OR” respectively. A literal is an atomic formula or negation of an atomic formula. A proposition is an atomic formula. A predicate is also an atomic formula.

• **Behavior** - A behavior β_i is i th behavior of a robot, where $1 \leq i \leq |B|$, modeled as a 2 tuple $\langle s_i, c_i \rangle$ and defined as a mapping from stimulus s_i to consequence c_i , s_i expressed in CNF (conjunctive normal form) and c_i expressed in a purely conjunctive form. B denotes the set of all behaviors of a robot, also called as behavior space. Note that though this representation of behaviors is same as the $\langle pre - condition, effect \rangle$ representation of state transforming actions or operators in the planning literature, the execution criteria for behaviors and operators are different. The behavior of an autonomous robot is generally executed once the stimulus is true, whereas an operator is executed only if its pre-conditions are true and the operator is relevant to the goal that the planner is trying to achieve. Purely behavior-based autonomous robots do not have an explicit representation of goals. The disjunction in the stimuli is not meant to capture the uncertainty in the perception (e.g. inability to determine if there is an obstacle or not, due to noise in the sonars). A stimulus contains disjunctions either because of the clauses required to handle coupling (discussed in this section) or because of generalization of the functionality, e.g. if a behavior can pick up cans as well as cups, its stimulus will be $\exists x((cup(x) \vee can(x)) \wedge graspable(x))$. Stimulus-response is the most intuitive method of expressing behaviors and

any behavior can be represented as a generated response to a given stimulus (page 79, chapter 3 in [8]). It is common to represent behaviors by rules of the form if *condition* then *action*. For example, “if there is an object very close, then move away”. Note that we do not use the term response in our model of behavior, because response is same as action that must be executed to perform the behavior. The action itself is not used in our model because how the behavior changes world is more important as far as assessing capabilities of behaviors is concerned. The consequence of a behavior allows us to know how world changes after a behavior is performed. Representation of consequences also allows us to automatically detect conflicting behaviors and identify which behaviors can occur after a behavior. For example, if consequence of a behavior β_1 contains p and consequence of behavior β_4 contains $\neg p$, where p is a proposition, these behaviors conflict. Similarly, if stimulus of β_2 contains q and consequence of β_5 contains $\neg q$, they conflict too.

- **Stimulus** - It is assumed that each literal in a stimulus corresponds to a number of sensor readings, i.e. sensor readings are processed to extract meaning out of them and there may be a literal to which this meaning is mapped. For example, if readings of 10 sonars are all less than a certain limit, a wall or a big obstacle may be located nearby. Then some literal in

stimulus for the behavior *avoid_obstacle* will then become true. If there are p sensors, each of which can have m distinct readings, we do not consider them to be m^p distinct stimuli, since all these readings can be mapped to fewer literals. As a result, there is a space of sensor readings associated with each condition in stimulus, such that presence of any reading from this space makes the condition true. s_i may contain implications of the form

$$\left(\bigwedge_{m=1}^k A_m \right) \Rightarrow \left(\bigwedge_{s=1}^w P_s \right)$$

(discussed in the definition of coupling in this section), to model parameter passing among behaviors that is necessary to make the construction of certain behavior chains (defined in this section) and hence the fulfillment of certain goals possible (A_m, P_s are predicate symbols). Such an implication can be converted into clauses, thus maintaining s_i in CNF. The stimulus s_i is defined to be at least as *strong* as stimulus s_j if $(s_i \Rightarrow s_j)$. Stimuli for default behaviors like random wandering are assumed to be of the form $\neg(a_1 \wedge a_2 \wedge a_3 \dots \wedge a_h)$. This means that the robot wanders as long as a certain condition is false.

- **Behavior chain** - Complex behavior occurs because a number of primitive behaviors (e.g. β_i) operate sequentially and/or concurrently. Here

we focus on the temporal sequencing mechanism that gives rise to a complex behavior. A behavior chain C is a temporal sequence of behaviors $\{\beta_{i_1} : \beta_{i_2} : \beta_{i_3} : \dots : \beta_{i_k}\}$, where $1 \leq i_m \leq |B|$, $1 \leq m \leq k$. $\beta_{i_m} : \beta_{i_{m+1}}$ is used to denote that these two behaviors occur immediately next to each other in time, with the former behavior preceding the latter. Such a chain is said to be composable from B (denoted by $C \triangleleft B$) if behaviors in the chain are elements of B . All possible temporal chains of behaviors are not programmed a priori, they get composed in real time in a situation driven manner. The actions of earlier behaviors in the chain change the situation in such a way that the newly changed part of the situation in conjunction with the universe U (unaffected part of the situation) implies stimulus for the next behavior. Consider the chain $\{\beta_1 : \beta_3 : \beta_7\}$. Here we may have $c_1 = (a \wedge c)$ and $s_3 = (a \vee b \vee e) \wedge (c \vee d \vee f) \wedge z$. Here $(c_1 \Rightarrow s_3)$ if $U \Rightarrow z$. z is not explicitly listed in c_1 since the list of such universal truths can be arbitrarily long. Desirable properties of behaviors and behavior chains are identified and investigated by the author in his previous work [61], [62] and [63].

- **Task** - A behavior chain transforms one world state into another. A task is defined as a transition from one world state to another, achieved

through a behavior chain. To fulfill a goal, one or more tasks have to be fulfilled.

- **Goal Complexity** - A primitive goal g_i is considered to be specified as a 2-tuple $\langle I_i, F_i \rangle$, where I_i and F_i denote the initial and final states of the relevant objects in the world. These are assumed to be expressed in purely conjunctive form. A primitive goal g_i is said to be at least as complex as the goal g_j if I_j and I_i are same and $(F_i \Rightarrow F_j)$. A non-primitive goal G consists of a set of such primitive goals. A set of goals G' is defined to be more complex than a set G (denoted by $G' >_c G$) if G' is obtained from G by replacing one or more primitive goals $g_i \in G$ by more complex primitive goals and/or adding more goals to G .

- **Marker** - The term marker is used to refer to (a) new objects introduced in an environment or (b) new features added to current objects in an environment or (c) those features of current objects that were not used before but used later, with the intention of externalizing internal state of a behavior or information relevant to a behavior, e.g. if a robot is supposed to collect all tennis balls except those near a cupboard, one way to design the stimulus is to store the absolute location of the cupboard in the form of internal state and design *pickup* behavior of the robot not to pick up balls

within some radius around that location. However one can install a red pole near the cupboard and replace the absolute location of the cupboard in the stimulus of the behavior by presence of red pole that can be sensed by vision. The red pole is a marker. Markers can also serve other important purpose besides externalizing internal state, e.g. for dealing with noisy sensors or reducing perceptual computations (bottom-up visual search where matching is entirely data driven is intractable [64]).

It is common to introduce markers to externalize internal states to enhance the robot-environment interaction, e.g. entries in the mobile robot competitions of the American Association of Artificial Intelligence. The rocks in the event of finding life on Mars [65] were painted black to aid in visual recognition. Black paper indicating the danger zones was spread out on part of the floor. The doors of the lander were painted blue and orange. The life forms consisted of balls and cubes of bright colors. The event of finding the remote [66] which consisted of fetching a known set of objects used textureless surfaces to keep the object on and the objects were guaranteed to be well separated from one another. This approach simplified the visual problem of segmentation (determining which image region corresponds to which object) and thus the shape recognition algorithms did not have to worry about oc-

clusion. Though with some penalty, environmental engineering was allowed in the event of vacuuming home [67]. The object detection mechanism in robotic soccer competition [68] was kept as simple as possible. Different objects had well defined colors that were used as a major cue in object detection. Since a single color or patch was not sufficient to provide orientation information, additional pink patch was added on the top of each robot. The ball was orange and the field was green and the markings on the side were white. Robots' tops were colored either blue or yellow to distinguish the team that they belonged to. The squiggle balls and the tennis balls in the "clean up the tennis court" event were painted black [69]. The vision system that was trained to recognize the yellow tennis balls and pink squiggle balls proved to be extremely reliable during the competition, benefitting from the color cues provided by the objects [70]. In the tennis ball collection event, the gate was marked by two cyan markers that were taped to the ground in front of the gate [70]. The office navigation robot in [71] was trained to recognize red color. If the judges would wear red shorts, the vision system would easily pick them out. The goal area in the "clean up tennis court" event [72] was marked with a blue square. In the implementation of shooting a ball into a goal [73], the ball was painted red and the goal box was painted

blue, to make feature extraction easier.

In our notation, a marker is denoted by M_i and is described by a pure conjunction of its features. For example, a colored cube kept on a flat surface can serve as a marker and be described as

$\exists x \exists y_1, y_2 (cube(x) \wedge is_face_of(x, y_1) \wedge color(y_1, red) \wedge is_face_of(x, y_2) \wedge$

$color(y_2, green))$. A marker M_i is at least as *strong* as a marker M_j if $(M_i \Rightarrow$

$M_j)$. It is *stronger* if $(M_i \Rightarrow M_j)$ and the set of predicates in M_j is a proper

subset of the predicates in M_i . A set of markers becomes *stronger* as more

markers are added to it and/or existing markers from that set are replaced by

stronger ones. The relation *stronger* is intended to indicate that the stronger

marker set can be used in at least as many ways as the weaker marker set, for

the purpose of externalizing internal states. It is assumed that an addition of

markers to an environment does not destroy or hide the stimuli. For example,

the red pole in the example above does not hide the handle of the cupboard

that is used to open it. Some internal states have to be updated whenever

external world changes. Externalizing the states can eliminate such updates

since the most recent information is available in the world itself. Hence

there are reasons for a robot to be more reactive. A detailed formalization

of robot-environment interaction in presence of markers, including desirable

properties of markers, is developed by the author [74].

- **Environment** - $E \rightsquigarrow E'$ denotes that the environment E' is obtained from the environment E by adding zero or more markers to E and/or replacing the existing markers by *stronger* ones.

- **Coupling** - Coupling c_{im} is said to exist between two behaviors β_i and β_m if values of some variables in some literals in s_m are set by conditions of the form $(\bigwedge_{t=1}^k A_t) \Rightarrow (\bigwedge_{s=1}^w P_s)$ (explained later here) in s_i and is defined as the function $f(k, u)$ (f is such that $f(k, u) \geq f(k', u')$ if $k \geq k', u \geq u'$ and $f(k, u) > f(k', u')$ if $(k > k', u \geq u')$ or $(k \geq k', u > u')$, $f(0, 0) = 0$) where k is the number of literals in s_m , one or more variables of which are assigned values by conditions in s_i and u is the number of literals in the corresponding conditions in s_i , e.g. if a robot is to place painting brushes near windows, then there may not be any coupling between the behavior that picks up a painting brush (say β_1) and the behavior that puts it near a window (say β_2). In that case $s_1 = \exists x(\text{graspable}(x) \wedge \text{brush}(x))$ and $s_2 = \exists x, y(\text{in_hand}(x) \wedge \text{brush}(x) \wedge \text{robot_at}(y) \wedge \text{window}(y))$. If it is desired that big brushes should be kept near big windows and small brushes near small windows, s_1 can be modified to be $s_1 = \exists x(\text{graspable}(x) \wedge \text{brush}(x) \wedge (\text{small}(x) \Rightarrow \text{assign}(Y, sm)) \wedge (\text{big}(x) \Rightarrow \text{assign}(Y, bg)))$ where it is assumed

that in evaluating the truth of the condition ($A_m \Rightarrow P_s$) (where A_m is expressed in conjunctive form and P_s is an atomic formula or a conjunction of atomic formulas of arity 2, of type $assign(x, a)$, introduced for the purpose of making assignments), P_s is evaluated to be true only if A_m is true and $assign(x, a)$ sets value of the variable x to a . s_2 can be then modified to be $\exists x, z(in_hand(x) \wedge brush(x) \wedge robot_at(z) \wedge window(z) \wedge size(z, Y))$. Here β_1 and β_2 are behaviors that are coupled. The clauses involved in coupling are $(small(x) \Rightarrow assign(Y, sm))$, $(big(x) \Rightarrow assign(Y, bg))$ and $size(z, Y)$. Hence the actual coupling c_{12} is $f(1, 4)$. The coupling C of a behavior space increases when coupling c_{ij} between some behavior pair β_i, β_j increases, couplings between other pairs staying constant. The coupling captures dependency among stimuli of behaviors and the modifications made to their structure to fulfill a goal. Coupling, marker strength and goal complexity are relative measures. A behavior space is more modular if its coupling is lower.

One can consider the marker introduction and the behavior coupling as means of increasing the goal-directedness of the behavior-based systems, since markers can be used to couple the behaviors through the external world (as we discuss in Theorem 2) and the stimulus modification couples the behaviors internally (without modifying an environment). The marker introduction

and behavior coupling are thus mechanisms of manipulating the dynamics of agent-environment interaction. This is neither pure planning, nor pure reactivity nor their combination in the hybrid architectures. Rather, it involves off-line manipulation of the agent-environment dynamics itself. Introduction of markers and/or coupling can be integrated with current deliberative and hybrid architectures and this will open up several new directions for developing new architectures of autonomous agency. In this paper, we examine tradeoffs in the behavior-based reactive architectures that allow introduction of coupling and/or markers. These architectures do not contain an explicit representation of goals. They do not contain a planner or sequencer. One can keep the coupling and an environment unchanged, and add new behaviors to fulfill more complex goals. But this requires an explicit co-ordination and arbitration of behaviors and is one reason why explicit planners and explicit goals are needed.

VI TRADEOFFS IN INTRODUCTION OF COUPLING AND/OR MARKERS

In this section, we derive results on relations between environment, complexity of goals and coupling. These results reveal tradeoffs in reactive behavior-based architectures that permit introduction of markers and/or

coupling, to increase their goal-directedness, without planner or sequencer or explicit goals.

Theorem 1. As the complexity of a goal increases, the coupling of the behavior space changes from C to C' where $C' \geq C$, when $|B|, E$ are left unchanged.

Proof - Let the goal G' be derived from the goal G by adding tuples of the form $\langle I_i, F_i \rangle$ to G , and/or replacing existing goals by more complex goals, so that $G' >_c G$.

When $|B|$ is kept constant and it is desired that goals not currently fulfilled by the system should be fulfilled, the only option is to modify the stimuli of existing behaviors so that they are chained in a certain way to fulfill the more complex goal. Consider how the stimuli of behaviors in the current system can be modified to fulfill the goals $g \in (G' - G)$, without dropping any literal l from existing stimuli (since dropping literals from existing stimuli to fulfill $(G' - G)$ may result in some $g_i \in G$ not being fulfilled). Dropping literals from stimuli to reduce conjunction can generalize them, e.g. $(a \vee b)$ is more general than $(a \vee b) \wedge c \wedge d$. Behaviors with weaker (more general stimuli) can fulfill more tasks. However this can introduce other conflicts like infinite cycles [63]. Also, dropping literals can make a stimulus stronger,

reducing task-fulfilling capability of a behavior set, e.g. $(a \vee b)$ is less general (stronger) than $(a \vee b \vee c)$.

Case 1. Consider the behaviors β_i and β_m where s_i contains conditions that assign values to variables in literals of s_m . One can modify the values that are being assigned in $assign(x, a)$ to fulfill the elements of $(G' - G)$ or rearrange the existing literals, e.g. if instead of dropping big brushes at big windows and small brushes at small windows, it is desired that the big brushes should be dropped at the small windows and the small brushes should be dropped at the big windows, the assignment conditions $(small(x) \Rightarrow assign(Y, sm)), (big(x) \Rightarrow assign(Y, bg))$ in s_i can be changed to $(small(x) \Rightarrow assign(Y, bg)), (big(x) \Rightarrow assign(Y, sm))$. If it is desired that small brushes should not be moved, s_i can be changed to include $(brush(x) \wedge \neg small(x))$. However these changes leave k, u (in the definition of coupling) unchanged. This argument can be repeated for other pairs of stimuli. Hence $C' = C$.

Case 2. Stimuli are changed by adding new conditions to s_i and/or increasing the number of literals in s_m , variables of which are assigned values. This leads to an increase in k and/or u . Let k, u be increased by k' and u' respectively. In that case, the new coupling between the behaviors is

$f(k+k', u+u') > f(k, u)$. For example, if in the goal of moving brushes, there is a third category of brushes and windows, say *medium*, then an additional condition ($medium(x) \Rightarrow assign(Y, md)$) will be needed. Here u is increased. If it is required that brushes of a particular size should be dropped at only those windows which also have a cupboard of corresponding size near them, the stimuli will have to be modified further, increasing both k and u . This argument can be extended to multiple pairs of behaviors. Since the coupling of at least one behavior pair increases, $C' > C$. Hence the proof. \square

In the example of moving the brushes discussed above, only one robot is responsible for moving and dropping the brushes at the desired location, once it picks up a brush. To eliminate coupling, one will have to design multiple behaviors for picking and dropping, e.g. separate behaviors for picking brushes of different sizes with stimuli $\exists x(small(x) \wedge brush(x))$, $\exists x(big(x) \wedge brush(x))$ and $\exists x(medium(x) \wedge brush(x))$ and corresponding behaviors for dropping the brushes. Then even if a robot picks up a brush and drops it at an incorrect location, other robot can pick it up and drop at the correct location independently. This suggests another dimension of analysis.

Theorem 2. If the coupling of a behavior space is changed from C to C' , $C' < C$, keeping $|B|$ fixed, fulfilling original goals requires introduction of

m markers, $m \geq 1$.

Proof - Since coupling of the space is reduced, there exists a pair of behaviors, β_i, β_m , such that the coupling c_{im} between them is reduced. This means that either the number of literals in s_m , variables of which were assigned values by conditions in s_i were reduced and/or the number of literals in conditions in s_i making truth assignments were reduced (the coupling c_{im} reduces in these cases). These changes will either result in variables in literals in s_m not having any values assigned to them or variables that have some value assigned arbitrarily (these values are not set by conditions in s_i , e.g. instead of letting a condition in s_i set value of Y in $at(Y)$ in s_m , one can force it to some arbitrary value and reduce the coupling) or variables that have incorrect values assigned, resulting in fewer goals fulfilled or an undesired behavior. One has to then create markers that act as substitutes for the values to be assigned, (e.g. one can write the size of window near which a brush is to be dropped on the brush itself and then modify the previously discussed stimuli $s_1 = \exists x(\textit{graspable}(x) \wedge \textit{brush}(x) \wedge (\textit{small}(x) \Rightarrow \textit{assign}(Y, sm)) \wedge (\textit{big}(x) \Rightarrow \textit{assign}(Y, bg)))$, $s_2 = \exists x, y(\textit{in_hand}(x) \wedge \textit{brush}(x) \wedge \textit{robot_at}(y) \wedge \textit{window}(y) \wedge \textit{size}(y, Y))$ to $s_1 = \exists x(\textit{graspable}(x) \wedge \textit{brush}(x) \wedge \textit{has_word}(x))$ ($\textit{has_word}(x)$ means that x has

a word written on it) and $s_2 = \exists x, y, z(in_hand(x) \wedge brush(x) \wedge robot_at(y) \wedge window(y) \wedge on(x, z) \wedge word(z) \wedge size(y, z))$. This introduces $m > 0$ markers, hence the proof. \square

From Theorem 1, Theorem 2 and the definition of environmental transformation, we have

Theorem 3. When a goal G to be fulfilled by a behavior space in environment E is modified to more complex G' ($G' >_c G$) and the coupling C and $|B|$ are kept constant, fulfilling G' requires modification of E to E' , such that $E \rightsquigarrow E'$.

This result also shows that making a behavior-based robotic system goal-directed may also mean making its environment goal directed, since markers are seeded in the environment to bias the behaviors towards fulfilling the goals.

VII. BEHAVIOR-BASED HOUSE CLEANING

We discuss here how the notions of coupling and marker can be used in practice. Consider a robot with following behaviors for house cleaning -

$pick_air_freshener(\beta_1), drop_air_freshener(\beta_2),$

$spray_air_freshener(\beta_3), move(x, y)(\beta_4),$

$pick_floor_mop(\beta_5), sweep_floor(y)(\beta_6),$

drop_floor_mop(β_7), *pick_bowl_scrub_brush*(β_8),

drop_bowl_scrub_brush(β_9), *scrub_bowl*(β_{10}),

pick_tub_scrub_brush(β_{11}), *drop_tub_scrub_brush*(β_{12}), *scrub_tub*(β_{13}).

The stimuli and consequences of various behaviors are shown in parentheses in the description below.

pick_air_freshener (**stimulus:** (*air_freshener*(x) \wedge gripper_free \wedge location(x,y) \wedge at_robot(y)), **consequence:** (has_gripper(x) \wedge \neg gripper_free)),

drop_air_freshener (**stimulus:** (*air_freshener*(x) \wedge has_gripper(x) \wedge at_robot(y)), **consequence:** (\neg has_gripper(x) \wedge gripper_free \wedge location(x,y))),

move(x,y) (**stimulus:** at_robot(x) **consequence:** (at_robot(y) \wedge \neg at_robot(x))),

spray_air_freshener (**stimulus:** (has_gripper(x) \wedge *air_freshener*(x) *wedge* at_robot(y)), **consequence:** *air_fresh_at*(y)),

pick_floor_mop (**stimulus:** (*air_freshener*(x) \wedge gripper_free \wedge location(x,y) \wedge at_robot(y)), **consequence:** (has_gripper(x) \wedge \neg gripper_free)),

drop_floor_mop (**stimulus:** (*air_freshener*(x) \wedge has_gripper(x) \wedge at_robot(y)), **consequence:** (\neg has_gripper(x) \wedge gripper_free \wedge location(x,y))),

sweep_floor(y) (**stimulus:** (has_gripper(x) \wedge mop(x) \wedge dirty(y) \wedge at_robot(y)), **consequence:** \neg dirty(y)),

pick_bowl_scrub_brush (**stimulus:** (*bowl_scrub_brush*(x) \wedge gripper_free \wedge

location(x,y) \wedge at_robot(y)), **consequence:** (has_gripper(x) \wedge \neg gripper_free)),

drop_bowl_scrub_brush (**stimulus:** (bowl_scrub_brush(x) \wedge has_gripper(x) \wedge at_robot(y)), **consequence:** (\neg has_gripper(x) \wedge gripper_free \wedge location(x,y))),

pick_tub_scrub_brush (**stimulus:** (tub_scrub_brush(x) \wedge gripper_free \wedge location(x,y) \wedge at_robot(y)), **consequence:** (has_gripper(x) \wedge \neg gripper_free)),

drop_tub_scrub_brush (**stimulus:** (tub_scrub_brush(x) \wedge has_gripper(x) \wedge at_robot(y)), **consequence:** (\neg has_gripper(x) \wedge gripper_free \wedge location(x,y))),

scrub_tub (**stimulus:** (tub(x) \wedge location(x,y) \wedge at_robot(y) \wedge dirty(x) \wedge has_gripper(z) \wedge tub_scrub_brush(z)), **consequence:** \neg dirty(x)),

scrub_bowl (**stimulus:** (bowl(x) \wedge location(x,y) \wedge at_robot(y) \wedge dirty(x) \wedge has_gripper(z) \wedge bowl_scrub_brush(z)), **consequence:** \neg dirty(x))

It is assumed that the two dimensional workspace of the robot is divided into finite number of locations. Each location is an area. The predicate at_robot(y) is true if the robot is at any point in the area that represents location y. It is assumed that once a robot is at a location, any object in that location is within its reach. It is assumed that when robot sweeps floor when it is at location y, entire area representing y is swept. Actual path

followed by the robot to go from one location to another is irrelevant. x, z and y in the definitions of behaviors are variables. For example, the predicate $\text{fridge}(x)$ is evaluated to true if there is an object in the range of sensors of robot such that it meets the definition of a fridge. Thus $\text{fridge}(x)$ can be interpreted as $\exists x \text{ fridge}(x)$ (which means that there exists some x such that x is a fridge.) What the behaviors do is clear from their names which are same as action part of the behaviors. It is assumed that the number of move behaviors is large enough to allow required movements of the robot. One can either have $O(n^2)$ move behaviors to allow robot movement from any location to any other location, or one can have $O(n)$ behaviors to allow a robot to move only to neighboring locations, n being the total number of locations in the robot's workspace.

Consider the goal of having the floor swept. The chain $\{\beta_5 : \beta_6 : \beta_4 : \beta_6 : \beta_4 : \dots\}$ fulfills this goal. Once the robot picks up the mop, it should sweep the floor. However it is possible that it will drop the mop immediately, since having the mop in hand provides the stimulus for dropping it. To increase the probability of execution of the chain $\{\beta_5 : \beta_6\}$, we can couple the behaviors β_5 and β_4 , so that β_5 sets values of variables in literals in the stimulus of $\text{move}(x, y)$ to wander till dirty floor is not found. Once a dirty

floor is found, the stimulus of β_6 will be triggered and the floor will be swept.

Modified stimuli of β_4 and β_5 are

pick_floor_mop **stimulus:** (*air_freshener*(x) \wedge gripper_free \wedge location(x,y)
 \wedge at_robot(y) \wedge (\neg dirty(y) \Rightarrow assign(R,dirt))),

move(x,y) **stimulus:** at_robot(x) \wedge has(y,R)

This coupling forces the robot to move only to locations which have R, which in this case is dirt. If one decides not to introduce the coupling, one can introduce a marker in the environment and design the stimulus of the behavior *move*(x,y) to move the robot till the marker is found. In this case, marker can be kept at dirty location. If the marker is red paper, modified stimulus of *move*(x,y) will be at_robot(x) \wedge has(y,z) \wedge paper(z) \wedge color(z,red) It is possible that the mop will be dropped at an undesired place after the floor is swept. To prevent that, one can introduce a different move behavior to ensure that the robot moves to the correct place for dropping the mop. One can avoid the introduction of the new *move* behavior by introducing coupling or adding more markers. The chain $\{\beta_1 : \beta_3\}$ freshens air. The chain $\{\beta_8 : \beta_{10}\}$ cleans toilet bowl. The chain $\{\beta_{11} : \beta_{13}\}$ fulfills the goal of cleaning the bath tub. To reliably compose these chains, one can introduce coupling and/or markers, as discussed in the case of sweeping the floor.

VIII. DISCUSSION

As mentioned in the abstract and in the introduction, this paper makes two contributions - (i) reviewing several important architectures of autonomous agency and (ii) investigation of tradeoffs in the architectures that do not have a planner or sequencer and which do not have an explicit representation of goals. The architectures are reviewed in sections II, III and IV, discussing their foundations, achievements and limitations respectively in these sections. The model of behavior-based systems in section V was used to investigate tradeoffs in section VI.

We review several architectures of autonomous robots in this section and show how architectures based on introduction of markers and/or coupling differ from them. Gat & Dorais [75] propose conditional sequencing as a mechanism for navigation. A conditional sequence is similar to a script. They have implemented only navigation behaviors like “dead reckoning”, “avoiding obstacles”, “following walls”. Saffiotti et al [76] argue that engaging in more purposeful activities than wandering requires more than pure reactivity and that an explicit reasoning to fulfill goals is needed. Their work fits in the tradition of the “two level” approaches to robot control in which a strategic planner is used to guide a reactor. Simmons’ methodology in the

task control architecture [77] is to first develop systems having sequential sense-plan-act cycles and then use additional facilities to add concurrency. He says that the decision making for autonomous robots should be a combination of both reactivity and planning. Connell [78] proposes a three layer architecture with symbolic, subsumption and servo layer (which directly interacts with the world). The symbolic layer maintains a world model and takes strategic decisions. Lyons & Hendriks [79] advocate a combination of the ability to plan and react. Their planner maintains a world model and tunes the behavior of reactor to achieve goals. Fujimura [80] presents a distributed approach for multi-robot navigation. Each robot plans reactively and the plans may be aborted, revised or completed, depending upon the situation.

Behavior hierarchies are proposed by Seeliger & Hendler [81] to avoid explicit planning. However these hierarchies provide a detailed guidance for achieving goals. Such hierarchies are provided by a domain expert and may not always be available. The most abstract modules lie at the root of a behavior hierarchy and they are triggered by an explicitly stated goal rather than environmental stimuli. Behaviors like “avoid obstacle”, “decelerate at curve”, “decelerate at narrow road”, “wander”, “follow left edge”, “follow

right edge”, “follow road” are implemented using potential fields in [82]. Huber & Grupen [83] propose a hybrid discrete event dynamic systems approach to robot control in which reactive module and a symbolic reasoner to generate control strategy are used. Ferguson [84] proposes a hybrid architecture combining deliberative and reactive control. Maes [14] models a spectrum of the reaction-deliberation combination rather than making a reactive system goal-directed. Her architecture does not preserve both goal-orientedness and reactivity. Mataric [29] reports an experiment in integrating reactive behaviors and distributed spatial representation for navigation, but the goals are explicitly represented outside the reactive controller.

Though an automated chess player exhibits higher level of intelligence, it is not autonomous. As a result, it is the criterion of autonomy rather than intelligence, that was used to evaluate the success of the early implementations of behavior-based agency. Autonomy does not necessarily entail an extreme level of intelligence. Explicit goals and reasoners were added on the top of behavior spaces, since goal fulfilling functionality did not always emerge from the robot-environment interaction.

Gat [85] argues that it is easy to combine different computational mechanisms, and hence an architecture should be heterogeneous. He describes a

hybrid architecture that integrates classical world models with reactive systems. Despite such claims, the division of functionality between reactor and planner remains a matter of considerable debate. Does learning overcome these limitations? A closer look at the learning in situated robots shows reinforcement learning to be the most widely used form of learning. Dorigo & Schnepf [86] report a simulated robot that learns to follow light and avoid hot dangerous objects. This however turns out to be an even more challenging task, since both the behaviors as well as the co-ordination among them has to be learnt (whereas reactors have a ready to use set of behaviors and deliberators have pre-canned co-ordination strategies). Though learning is more suitable for domains where such pre-canned sets are not available, the difficulty in specifying the right reward and punishment functions has placed a limitation on the success of these architectures. It is not claimed that our approach of introducing coupling and/or markers is better than learning. It is also not claimed that the architectures where introduction of coupling and/or markers is allowed are better than hybrid architectures. It is also not claimed that introduction of markers and/or reactivity can enable a robot to do everything that a planning module can do. Our model of behavior-based systems where coupling and/or markers are allowed does

show a new class of architectures whose potential is worth investigating. It is also clear that introduction of coupling and/or markers does avoid to some extent some problems of uncoupled behaviors and markerless environments, like (i) uncertainty in formation of goal-fulfilling behavior chains and (ii) no representation of goals. This also means that some tasks for which planning module is currently considered to be indispensable, can be fulfilled with the coupling and/or marker-based architectures.

Clearly, higher coupling between behaviors makes a behavior space less modular. Instead of using a planner and/or sequencer to compose goal-fulfilling behavior chains, we modify the stimuli of behaviors and/or the environment. Such a modification increases the probability of goal fulfillment and makes the behavior-based robots more goal directed. It is not argued that such modifications can always replace an explicit planner. However they challenge the belief that reactive behaviors always need planners for tasks more complex than simple navigation. Planners in the current hybrid architectures fulfill goals at the cost of reactive functionality. Since the introduction of coupling and markers does not explicitly select and order behaviors to achieve explicit goals, our approach is different from all other hybrid architectures.

One may argue that in our approach, a change in goals may require a change in the coupling and/or the environment. However, other behavior-based architectures need a change as well, for example, a change in the weights for the behavior outputs or a change in the behavior sequencing strategy or a change in the attractive and repulsive potential fields around the objects may be required. In potential-field based navigation, goal exerts attractive force on a robot and an obstacle exerts a repulsive force. If we change the goal, the previous goal object should be now treated as an obstacle. Hence attractive field must be changed to repulsive and the repulsive field of the object which is the new goal, must be changed to attractive. The world of Shakey [4] was very carefully engineered. Walls were specially constructed and painted with matte finishes. The blocks that Shakey manipulated were polyhedral and each face was painted in a solid matte color. Our idea of modifying an environment by adding markers is different from the above environment engineering. The environment of Shakey was engineered to help the sensors. We modify the environment by adding markers, to externalize internal state. Our main motivation behind introducing markers is to reduce the amount of internal state, avoid the updating of internal state and avoid an explicit representation of goals. No world model is maintained in

the approach of introducing coupling between behaviors and/or introducing markers in an environment. Brooks [52] says that he requires the creatures to do something in the world and maintain multiple goals. However, currently there is no formal and reliable method for integrating goals into the subsumption architecture. Brooks says that behavior-based approach is not same as connectionism, neural networks, production rules, blackboard and German philosophy. We agree with this. Our modifications to the coupling and environment preserve the distinction between behavior-based artificial intelligence and the other paradigms listed above. By introducing coupling between behaviors, we do not sacrifice the “behaviorness” of these computational modules. On the other hand, complex behaviors in [22] are referred to as “behavioral plans”, sacrificing the behavioral nature of the computational modules. Our behavior chains can be viewed as paths in the finite state diagram used in planner-reactor architecture of Arkin & Balch [34]. However we do not a priori group behaviors into state diagrams which explicitly specify all relevant state transitions by reasoning about all relevant sensor readings. Nilsson [87] presents an approach to synthesis of goal fulfilling reactive programs. This is similar to the approach of [21] and Nilsson recommends the use of stored world models and an integration of reactive

programs with planners. One may argue that introducing coupling violates the principle of modularity in behavior-based systems. However, one can see that it is exactly this idea of having completely independent behaviors that invites the planning module and explicit goals. Coupling is introduced among behaviors when goal fulfilling chains cannot be automatically composed through the agent-environment dynamics or when erroneous outcome is highly likely. Since there is no explicit selection and sequencing of goal relevant actions, introducing coupling is not same as satisfying subgoals in planning. Coupling is a mechanism that locally changes behaviors' structure to increase the chances of fulfilling goals.

IX. CONCLUSION

Our work serves two objectives - (i) providing a brief review of autonomous agent architectures and (ii) investigating (a) whether it is possible to have behavior-based autonomous agent architectures without planner and sequencer and without an explicit representation of goals, such that these architectures can have potential to fulfill goals requiring much more than obstacle avoidance and wandering and (b) what the tradeoffs in these architectures are.

We reviewed a number of architectures of autonomous agency from the

three dominant categories - (i) planning-based, (ii) reactive behaviors-based and (iii) hybrid. We discussed their foundations, achievements and limitations in the review. None of these architectures integrates goals into behavior-based design of autonomous agents, without sacrificing the much coveted highly reactive functionality. To overcome this limitation, we proposed the novel approach of introduction of coupling and/or markers. We neither assume a benign environment nor make the closed-world assumption. Goals are not explicitly represented. This opens up directions for development of several new architectures of autonomous agency like those containing

- (i) reactive behaviors, some of which are coupled,
- (ii) reactive behaviors, some of which are coupled, along with markers in environment, that are introduced by the agent itself or other agents,
- (iii) reactive behaviors, some of which are coupled, along with a planner,
- (iv) reactive behaviors, some of which are coupled, along with a planner and markers in the environment introduced by the agent itself or other agents.

There is a major difference between our motivation for introducing markers and conventional motivation for the same. Generally, markers are introduced to simplify perception. We suggest introduction of markers to fulfill goals without explicitly representing goals. We investigated tradeoffs in ar-

chitectures in categories (i) and (ii).

We showed that trying to fulfill more complex goals increases coupling and that trying to eliminate or reduce coupling either increases the number of required behaviors or the number of required markers. Hence it can be concluded that fulfilling more complex goals while curbing the coupling either makes B more complex to debug (due to an increase in the size) or requires the environment to be modified, requiring reasoning about the impact of markers on overall functionality of a robot. The size of a behavior space, an environment and the coupling have an impact on the amount of testing required to verify that a robot will exhibit an acceptable level of reactive and goal fulfilling functionality. The environment affects the perceptual computations and the goal complexities are related to a customer's expectations from a robot. Our mechanisms of introducing coupling and externalizing internal states indicate a rich space of behavior-based architectures that no longer require reactivity and goal fulfilling capabilities to be orthogonal and provide flexible options for integrating goals with behaviors, maintaining the homogeneity of the computational structure. The tradeoffs we revealed make a more flexible design of autonomous agents possible. Our work also opens up directions for developing efficient algorithms for (i) automated introduc-

tion of coupling between behaviors without introducing conflicts and guaranteeing composability of goal-fulfilling behavior chains, without significantly restricting behavior sequencing possibilities and (ii) automated introduction of markers without introducing conflicts and guaranteeing accurate representation of goals, with appropriate modification to stimuli of behaviors.

References

- [1] **Rodney A. Brooks**, Achieving artificial intelligence through building robots, AI memo 899, MIT, May 1986.
- [2] **Rodney A. Brooks, Jonathan H. Connell and Peter Ning**, HERBERT: A second generation mobile robot, AI memo 1016, MIT, Jan. 1988.
- [3] **Rodney Brooks**, Intelligence without reason, Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1991.
- [4] **Nilsson, Nils**, "Shakey the Robot", SRI AI Center Technical Note 323, April 1984.
- [5] **Jacques Ferber**, Multi-Agent Systems: An introduction to distributed artificial intelligence, Addison-Wesley, Chapter 1, pp. 9-13, 1999.
- [6] **Brooks, R. A., 1986**, A robust layered control system for a mobile robot, IEEE journal on robotics and automation, 2(1), 14-23.
- [7] **R. Peter Bonasso and David Kortenkamp**, Characterizing an archi-

tecture for intelligent, reactive agents, Working notes of AAAI spring symposium on lessons learned from implemented software architectures for physical agents, 1995, 29-34.

[8] **Ronald C. Arkin**, 1998, Behavior-Based Robotics, The MIT Press, 1998.

[9] **Philip E. Agre**, Computational research on interaction and agency, Artificial intelligence 72, 1995, 1-52.

[10] **Anderson, T. L.; Donath, M.** 1990. Animal Behavior As A Paradigm For Developing Robot Autonomy, Robotics and Autonomous Systems, 6(1 & 2): 145-168.

[11] **M. B. Kadonoff**, Arbitration of multiple control strategies for mobile robots, SPIE mobile robots, 1986, 727.

[12] **Connell, J.** 1990. Minimalist mobile robotics, A colony style architecture for an artificial creature, Academic press Inc.

[13] **Barry Werger and Maja Mataric**, Robotic food chains: Externalization of state and program for minimal agent foraging, Proceedings of the conference on simulation of adaptive behavior, 1996, 625-634.

[14] **Pattie Maes**, Situated agents can have goals, Robotics and autonomous systems 6, 1990, 49-70.

- [15] **Arkin, R. C.** 1992. Behavior-Based Robot Navigation for Extended Domains, *Adaptive Behavior* 1(2): 201-225.
- [16] **R. Peter Bonasso**, Integrating reaction plans and layered competences through synchronous control, *International joint conference on artificial intelligence*, 1991, 1225-1231.
- [17] **Amol Mali**, Hierarchical task network planning as satisfiability, *Proceedings of European Conference on Planning (ECP)*, 1999.
- [18] **Keith Decker, Anandee Pannu, Katia Sycara and Mike Williamson**, Designing behaviors for information agents, *First international conference on autonomous agents*, Feb. 1997.
- [19] **Daniel S. Weld**, Planning-based control of software agents, *Proceedings of the artificial intelligence planning systems conference*, 1996.
- [20] **M. J. Schoppers**, Universal plans for reactive robots in unpredictable environments, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1987, 1039-1046.
- [21] **Stanley Rosenschein and Leslie Kaelbling**, A situated view of representation and control, *Artificial intelligence* 73, 1995, 149-173.
- [22] **Alessandro Saffiotti, Kurt Konolige and Enrique Ruspini**, A multivalued logic approach to integrating planning control, *Artificial intelligence*

76, 1995, 481-526.

[23] **D.M. Lyons and A.J.Hendriks**, Exploiting patterns of interaction to achieve reactive behavior, *Artificial Intelligence* 73, 1995, 117-148.

[24] **Reid Simmons**, Structured control for autonomous robots, *IEEE transactions on robotics and automation*, Feb. 1994.

[25] **Moravec H.P**, The Stanford cart and CMU Rover, Tech. report, Robotics Institute, Carnegie Mellon univ., 1983.

[26] **Hans P. Moravec**, Towards automatic visual obstacle avoidance, *Proceedings of the fifth international joint conference (IJCAI)*, Cambridge, MA, August 1977, p. 584.

[27] **Fikes R.E & Nilsson N.J.** STRIPS: A new approach to the application of theorem proving to problem solving, *Artificial Intelligence* 2: 251-288, 1971.

[28] **F. Noreils and R. Chatila**, Control of mobile robot actions, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1989, pp. 701-707.

[29] **Maja Mataric**, Integration of representation into goal-driven behavior-based robots, *IEEE transactions on robotics and automation*, Vol. 8, No. 3, June 1992, 304-312.

- [30] **M. Watanabe, K. Onoguchi, I. Kweon and Y. Kuno**, Architecture for behavior-based robot in dynamic environment, Proceedings of IEEE international conference on robotics and automation, 1992, 2711-2718.
- [31] **Luc Steels**, A case study in the behavior-oriented design of autonomous agents, Submitted to the Simulation of adaptive behavior conference, 1994.
- [32] **Jonathan Connell**, Creature design with the subsumption architecture, Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1987, 1124-1126.
- [33] **David Miller, Rajiv Desai, Erann Gat, Robert Ivlev and John Loch**, Reactive navigation through rough terrain: Experimental results, Proceedings of the National Conference on Artificial Intelligence (AAAI), 1992, 823-828.
- [34] **Ronald C. Arkin and Tucker Balch**, AuRA: Principles and Practice in Review, Journal of Experimental and Theoretical Artificial Intelligence, Vol. 9, No. 2, 1997, 175-189.
- [35] **C. Ronald Kube, Hong Zhang**. Collective Robotics: From Social Insects To Robots. Adaptive Behavior, Vol. 2, No. 2, 189-218, 1994.
- [36] **R. Peter Bonasso, H. James Antonisse and Marc Slack**, A reactive robot system for find and fetch tasks in an outdoor environment, Pro-

ceedings of the National Conference on Artificial Intelligence (AAAI), 1992, 801-808.

[37] **Maes Pattie and Brooks Rodney, 1990**, Learning to co-ordinate behaviors, Proceedings of the National Conference on Artificial Intelligence (AAAI), 796-802.

[38] **Montgomery James, Fagg Andrew and Bekey George, 1995**, The USC AFV-I, IEEE Expert, Vol. 10, No. 2.

[39] **Desai Rajiv, Rosenberg Charles and Jones Joseph, 1995**, Kaa: An autonomous serpentine robot uses behavior control, Proceedings of the conference on Intelligent Robots & Systems (IROS), 250-255.

[40] **Wettergreen David, Pangels Henning and Bares John, 1995**, Behavior-based gait execution for the Dante-II walking robot, Proceedings of the conference on Intelligent Robots & Systems (IROS), 274-279.

[41] **MacKenzie Douglas C., and Arkin Ronald C., 1996**, Behavior-based mobile manipulation for drum sampling, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Minneapolis, pp. 2389-2395.

[42] **Cameron Jonathan M., MacKenzie Douglas C., Ward Keith R., Arkin Ronald C., and Book Wayne J., 1993**, Reactive control

for mobile manipulation, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Atlanta, Georgia, Vol. 3.

[43] **Emery Rosemary and Balch Tucker, 2001**, Behavior-based control of a non-holonomic robot in pushing tasks, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea, pp. 2381-2388.

[44] **Birk Andreas and Holger Kenn, 2001**, An industrial application of behavior-oriented robotics, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea, pp. 749-754.

[45] **Arkin, Ronald C., Fujita Masahiro, Takagi Tsuyoshi and Hasegawa Rika, 2001**, Ethological modeling and architecture for an entertainment robot, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea, pp. 453-458.

[46] **Kubica Jeremy, Casal Arancha and Hogg Tad, 2001**, Complex behaviors from local rules in modular self-reconfigurable robots, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea, pp. 360-367.

[47] **Mahadevan, S., and J. Connell, 1992**, Automatic programming of behavior-based robots using reinforcement learning, Artificial Intelligence,

55, 311-365.

[48] **Morignot Philippe, Aycard Olivier and Charpillet Francois, 1997**, A pair of heterogeneous agents in a unique vehicle for object motion, Proceedings of 9th IEEE International Conference on Tools for Artificial Intelligence (ICTAI), California, pp. 508-513.

[49] **Endo Yoichiro and Arkin Ronald C., 2001**, Implementing Tolman's schematic Sowbug: Behavior-based robotics in the 1930s, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Seoul, 2001, pp. 477-484.

[50] **David Kortenkamp, R. Peter Bonasso and Robin Murphy, 1998**, Editors of Artificial Intelligence and Mobile Robots, AAAI Press/MIT Press.

[51] **Hartley, R.; Pipitone, F.** 1991. Experiments with the subsumption architecture, In Proceedings of the IEEE International Conference on Robotics and Automation, 1652-1658.

[52] **Brooks R. A.**, Intelligence without representation, Artificial intelligence 47, 1991, 139-159.

[53] **Leslie Kaelbling**, An architecture for intelligent reactive systems, Readings in planning, 713-728.

[54] **R. James Firby**, An investigation into reactive planning in complex

domains, Proceedings of the National Conference on Artificial Intelligence (AAAI), 1987, 202-206.

[55] **Sundar Narasimhan**, Merging path planners and controllers through local context, Proceedings of the National Conference on Artificial Intelligence (AAAI), 1994, 1285-1290.

[56] **R. James Firby and Reid Simmons**, Mobile Robots II: Architectures for reaction and deliberation, Tutorial presented at the National Conference on Artificial Intelligence (AAAI), 1993.

[57] **David Kirsh**, Today the earwig, tomorrow man? Artificial intelligence 47, 1991, 161-184.

[58] **David Chapman**, Penguins can make cake, AI magazine, Winter 1989, 45-50.

[59] **Erann Gat**, On the role of stored internal state in the control of autonomous mobile robots, AI magazine, Vol. 14, No. 1, Spring 1993, 64-73.

[60] **Pattie Maes**, Modeling adaptive autonomous agents, Artificial life 1: 135-162, 1994.

[61] **Amol D. Mali**, On the synthesis of reactive and robust behavior chains, Proceedings of International Conference on Artificial Intelligence and Soft Computing (ASC), Banff, Alberta, Canada, 2000.

- [62] **Amol D. Mali** and **Amitabha Mukerjee**, Metrics for evaluation of behavior-based robotic systems, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 1998.
- [63] **Amol D. Mali** and **Amitabha Mukerjee**, Robot behavior conflicts: Can Intelligence be Modularized?, Proceedings of National Conference on Artificial Intelligence (AAAI), 1994, pp. 1279-1284.
- [64] **John Tsotsos**, The complexity of perceptual search tasks, Proceedings of the eleventh international joint conference on artificial intelligence (IJCAI), 1989, Detroit, Michigan, pp. 1571-1577.
- [65] **Reid Simmons**, The Find-Life-on-Mars event, AI Magazine, Fall 1998, 19-24.
- [66] **Ian Horswill**, The Find-the-Remote event, AI Magazine, Fall 1998, 25-28.
- [67] **Pete Bonasso** and **Karen Myers**, The Home-Vacuum event, AI Magazine, Fall 1998, 29-32.
- [68] **Manuela Veloso**, **Peter Stone** and **Kwun Han**, CMUNITED-97 - RoboCup-97 small-robot world champion team, AI Magazine, Fall 1998, 61-69.
- [69] **David Kortenkamp**, **Illah Nourbakhsh** and **David Hinkle**, The

1996 AAAI mobile robot competition and exhibition, AI Magazine, Spring 1997, 25-32.

[70] **Sebastian Thrun**, To know or not to know - On the utility of models in mobile robotics, AI Magazine, Spring 1997, 47-54.

[71] **Didier Guzzoni, Adam Cheyer, Luc Julia and Kurt Konolige**, Many robots make short work - Report of the SRI international mobile robot team, AI Magazine, Spring 1997, 55-64.

[72] **Randy Sargent, Bill Bailey, Carl Witty and Anne Wright**, Dynamic object capture using fast vision tracking, AI Magazine, Spring 1997, 65-72.

[73] **M. Asada, S. Noda, S. Tawaratsumida and K. Hosoda**, Vision-based reinforcement learning for purposive behavior acquisition, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), May 1995, pp. 146-153.

[74] **Amol D. Mali**, Marker-Augmented robot-environment interaction, Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 1999.

[75] **Erann Gat and Greg Dorais**, Robot navigation by conditional sequencing, Proceedings of the IEEE international conference on robotics and

automation, 1994.

[76] **Alessandro Saffiotti, Enrique Ruspini and Kurt Konolige**, Blending reactivity and goal-directedness in a fuzzy controller, Proceedings of the second IEEE conference on fuzzy systems, 1993, 134-139.

[77] **Reid Simmons**, An architecture for co-ordinating planning, sensing and action, Presented at DARPA Planning workshop, Nov. 1990.

[78] **Jonathan H. Connell**, SSS: A hybrid architecture applied to robot navigation, IEEE international conference on robotics and automation, 1992, 2719-2724.

[79] **D. M. Lyons and A. J. Hendriks**, Planning for reactive robot behavior, Proceedings of the IEEE international conference on robotics and automation, 1992, 2675-2680.

[80] **Kikuo Fujimura**, A model of reactive planning for multiple mobile agents, Proceedings of the IEEE international conference on robotics and automation, 1991, 1503-1509.

[81] **Oliver Seeliger and James Hendler**, Supervenient hierarchies of behaviors: Lessons learned from a vacuuming robot, Working notes of the AAAI Spring symposium on lessons learned from implemented software architectures for physical agents, 1995, 187-195.

- [82] **Wang Tianmiao and Zhang Bo**, Time-varying potential field-based ‘perception-action’ behaviors of a mobile robot, IEEE international conference on robotics and automation, 1992, 2549-2554.
- [83] **Manfred Huber and Roderic A. Grupen**, A hybrid discrete event dynamic systems approach to robot control, Technical report 96-43, Dept. of computer science, Univ. of Massachusetts, 1996.
- [84] **Innes Ferguson**, Integrating models and behaviors in autonomous agents: Some lessons learned on action control, Working notes of the AAAI Spring symposium on lessons learned from implemented software architectures for physical agents, 1995, 78-91.
- [85] **Erann Gat**, Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real world mobile robots, Proceedings of the National Conference on Artificial Intelligence (AAAI), 1992.
- [86] **Marco Dorigo and Uwe Schnepf**, Genetics-based machine learning and behavior-based robotics: A new synthesis, IEEE transactions on systems, man and cybernetics, Vol. 23, No. 1, Jan/Feb. 1993, 141-154.
- [87] **Nils J. Nilsson**, Teleo-reactive programs for agent control, Journal of artificial intelligence research 1, 1994, 139-158.

Acknowledgement - The author thanks Damian Lyons, Keith Decker, Phil Agre, Paul Rosenbloom, Alessandro Saffiotti, Illah Nourbakhsh and Nicholas Kushmerick for their useful comments on the previous draft of this paper.

Amol D. Mali was born in Pune, India in 1970. He got Ph.D in computer science from Arizona state university in May 1999, in the area of hierarchical planning as satisfiability. Prior to this, he received B.E and M.Tech from VJTI and IIT Kanpur in India respectively. He joined the electrical engineering and computer science department at University of Wisconsin, Milwaukee as an assistant professor in August 1999. His areas of research interest are artificial intelligence (AI), planning, constraint satisfaction and autonomous agents. He has supervised several student projects in the areas of constraint satisfaction and planning. He is a member of AAAI, ACM and IEEE. He presented a half-day tutorial in the area of recent advances in efficient plan synthesis at the IEEE International Conference on Robotics and Automation in Seoul in 2001. He has published 4 journal papers, 1 book chapter and 19 conference papers. He has published in the AAAI, AIPS, ECP and ICRA conferences. Currently he is working on a project in the area of pruning techniques for more expressive planners, funded by National Science Foundation.

He has been a reviewer for several journals and conferences. He has served on program committees of several conferences.