AUTOMATING LEARNING AND CREATIVITY THROUGH KNOWLEDGE INTEGRATION

From: AAAI Technical Report SS-93-01. Compilation copyright © 1993, AAAI (www.aaai.org). All rights reserved.

STEVEN H. KIM
Lightwell, Inc.
212 Maury Avenue
Charlottesville VA 22903 USA

Abstract. At times a domain is sufficiently tractable that the situations arising within its bounds give rise to structured knowledge in the form of rules, principles, and guidelines. Even in such domains, however, the initial knowledge base usually consists of examples. For a decision maker, the formulation of effective behaviors depends on a judicious mix of principles as well as cases - of deduction as well as induction. Learning and discovery are central features of creative problem solving. To the extent that software can automatically detect known regularities in a knowledge base or through experience, it may be viewed as a learning system. And where the identified patterns were previously unknown to the user, the system may be regarded as engaging in discovery.

A number of critical issues relating to creativity are discussed, as well as methodologies for their incorporation into software. The encapsulation of adaptive capabilities in software should elucidate the nature of human learning and creative behavior. In addition, the embodiment of such abilities in software will result in intelligent systems which can be used to enhance human decision making, or even automate certain tasks requiring creativity. For the sake of concreteness, much of the discussion is conducted in the context of creative problem solving in engineering design and control strategy.

1. Introduction

Creativity has been defined as the resolution of a difficult problem. A difficult problem, in turn, is one in which neither the solution nor its means of attainment, is obvious at the outset (Kim, 1990a).

Although the solution may be elusive, the creative thinker must still rely on his full store of knowledge to address the problem. In the realm of engineering design, for instance, an unprecedented goal such as a manned spacecraft to Mars may offer no simple solution. Even so, the design team must still draw on the spectrum of available knowledge. The relevant knowledge includes technical information of all types as well as precedents such as the experience of manned missions to the Moon and the robotic voyagers to the outer planets.

Much of the learning in people occurs through examples. Sometimes a domain is sufficiently tractable that the examples give rise to more formalized knowledge in the form of rules, principles, and guidelines. This occurs, for instance, in the equations of motion from physics or the laws of supply and demand in economics.

On the other hand, many fields of human endeavor are too complex to permit such ready encapsulation into simple models. Principles of chemistry may be found in textbooks, but the novice physician must serve an apprenticeship as a resident in a hospital; legal codes may be found in tomes, but the student of law must still learn through cases; guidelines for research may be gleaned from manuals, but a neophyte researcher must work alongside a mentor in a graduate training program. Even in our daily lives, we can best learn how to interact effectively with other people, not by reading their resumes, but by observing how they react to specific situations.

In real life, then, the formulation of effective behaviors depends on a judicious mix of principles as well as cases - of deduction as well as induction. Reasoning from precedent cases is a fundamental aspect of dealing with novel situations.

If the available knowledge base is small, then the problem solver may wish to review the entire portfolio. But what if the knowledge base is large? An exhaustive search would be impractical, and the decision maker must rely on a more judicious search through the knowledge.

For instance, a decision maker would find it convenient to discern regularities among precedent cases and retrieve only the most pertinent ones. Such activities may be supported in software by the incorporation of organizing methods such as clustering techniques.

The recognition of regularities in a case base is, in some sense, a macrolevel phenomenon relating to two or more cases. An additional advantage of clustering pertains to the microlevel task of retrieving close precedents. In particular, suppose that a problem solver is faced with a specific task and wishes to examine the closest precedents. Then the clustering of the cases will readily identify the desired items as those lying closest to the target case.

Learning and discovery are central features of creative behavior. To the extent that software can automatically detect known regularities in a knowledge base or through experience, it may be viewed as a learning system. And where the identified patterns were previously unknown to the user, the system may be regarded as engaging in discovery (Kim, 1990b).

This paper addresses a number of critical issues relating to learning and creativity, as well as their incorporation into software. The encapsulation of adaptive capabilities in software should elucidate the nature of learning and creative behavior in people. In addition, the embodiment of such abilities in software will result in intelligent systems which can be used to enhance human decision making, or even automate certain tasks requiring creative problem solving. For the sake of concreteness, much of the discussion is conducted in the context of creative design and learning behavior in control systems.

2. Creative Problem Solving through Knowledge Integration

A creative system should be able to handle unexpected situations as they arise. Moreover, the new experience should be incorporated into future behavior; that is, the system should learn over time. As indicated earlier, learning and discovery are key aspects of creative problem solving.

Intelligent systems often operate in complex environments, rife with uncertainty. The complexity arises from novelty, nonlinearities, and the multitude of interactions which arise when attempting to comprehend various activities. In such a milieu, important process variables can remain unidentified. Even when they are identified, their interactions may remain uncertain. The complexity and the uncertainties which are often its derivatives limit the effectiveness of traditional methods of knowledge representation and system control.

Fortunately, this situation can be remedied by a creative control methodology using knowledge integration. The integration involves a judicious mixture of model based reasoning, information sharing, and case based reasoning.

Over the past decade, a popular methodology for implementing learning systems has lain in the neural network. Despite its many advantages such as autonomous learning in specific contexts, the neural approach has its limitations. Among the limitations are the slow rates of learning and perhaps even more importantly, the implicit nature of the learned skill. More specifically, a neural network may yield the correct response to a query but it cannot explain the result or justify its "reasoning".

In contrast, the use of explicit knowledge allows for explanation and justification for the benefit of other entities, including an interested human observer. Examples of such high-level representation, also called the *knowledge level*, lies with declarative logic or production rules.

A sophisticated system should provide for a fusion of both implicit and explicit methods of knowledge representation. In this way, it can build on the respective advantages of disparate techniques.

When coupled with the ability to reason from precedents, such a learning system embodies a versatile capability. The system can present its learned approach to a human monitor, and both can learn directly from each other. In fact, the process of give and take between two or more people is a standard mode of elucidation in all fields of human endeavor, ranging from scientific discourse to political debate, and from business compacts to legal argumentation.

Given the affinity to human modes of discourse, knowledge level reasoning offers a compelling approach to versatile systems in all fields of endeavor. For the sake of tractability, however, we will focus in this paper on the realm of creative systems for technical design and control automation.

A complex system such as an aircraft operates in a highly dynamic environment, through changing environmental conditions as well as evolving mission requirements. The environmental disturbances may be caused by natural phenomena such as thunderstorms, or human activity such as other aircraft, or automated systems such as collision avoidance systems.

Due to the intrinsic complexity of automating the operation of real vehicles, control systems which have been fielded to date address only specific tasks or limited environmental scenarios. Moreover, exceptional conditions such as damage or failure of components have been handled with limited prowess: the sudden failure of a fuel pump in an aircraft may automatically press a redundant pump into service, but the effect of the discontinuity in thrust on flight stability or a damaged wing may not be explicitly addressed. In general, having a portfolio of disparate systems leads to the problem of suboptimization: in a nonlinear context such as the realm of flight control, the union of local optima determined by the respective controllers is unlikely to lead to the globally optimum action. The need for joint optimization applies not only to systems at a given level of operation, as exemplified by control modules for stabilizing pitch versus roll and yaw, but at different level of supervision, such as trajectory control versus stabilization. In particular, for certain vehicles including supersonic aircraft or hypersonic missiles, the outer and inner control loops must be tightly coupled for effective performance.

The limitations of existing control techniques may be transcended by an integrated approach. The approach takes the form of a generalized framework for creative control which accommodates a portfolio of software and hardware technologies acting in concert. In the realm of software, the framework may be implemented through the declarative techniques of artificial intelligence as well as the procedural methods of traditional computing. The software methods range from rule bases and frames to object oriented programming and case based reasoning. The approach can accommodate not only the methods of declarative logic, but procedural methods such as those from nonlinear adaptive control theory.

The use of a battery of software techniques, especially those of declarative reasoning from knowledge engineering, tends to be computationally expensive. Fortunately, advances in hardware support the rapid execution of the corresponding programs. Depending on the requirements of a program, the computations required by the approach can be designed for execution in real time

using multiple processors and a judicious memory architecture.

The purpose of the creative control system is to optimize performance in real time to accommodate changing functional requirements in the face of routine operating parameters as well as exceptional conditions such as anomalies, damage, and failure.

3. Framework and Methodology

A creative system should have the ability to accommodate a diversity of adaptive techniques among its component subsystems, including those of neural networks and learning classifiers. The general framework is designed to provide a systematic foundation for such functional modules.

This section presents a unified framework for adaptive systems. The framework comprises conceptual and formal constructs based on a modular representation of functional elements. After the presentation of the framework and the multilevel organizing methodology, the overall architecture is interpreted in specific contexts relating to both declarative and implicit types of knowledge representation.

3.1 General Framework

Adaptation refers to the modification of behavior patterns over time (Kim, 1990b, Ch. 1). Such adaptive behavior is useful for dealing with unknown factors in current or future activities. More specifically, they are effective in applications where (a) predictive knowledge of the underlying process is incomplete, and (b) variations in inputs or environmental conditions may exceed the predetermined range of values, or may even assume wholly unexpected characteristics (Kim, 1990b, App. D).

In the past, a machine was considered intelligent if it exhibited a range of behaviors, each action corresponding to specific input stimuli. For a synthetic device, the mapping from input to output was determined by a human programmer or another artifact, and remained fixed unless respecified by external intervention.

More sophisticated systems, however, incorporate autonomous adaptive behavior. By building on experience, an automated system can modify its own repertoire of behaviors over time.

One way to categorize learning systems takes the form of explicit versus implicit representation of knowledge. *Explicit* knowledge representation refers to the encoding of domain knowledge in terms and concepts readily comprehensible to the user. Examples of explicit representation lie in the Eurisko program (Lenat, 1983) for discovering new concepts, or the learning classifier approach (Holland, 1975). Explicit knowledge may be declarative, as in the statement of facts, or procedural, as in the encoding of instructions.

In contrast, *implicit* techniques refer to the generation of appropriate output based on input stimuli, without the direct representation of knowledge for processing the information. The implicit approach is reflected in neural network models. In this scheme, information processing know-how is encoded in terms of multiplicative weights at nodes and on the arcs between pairs of nodes. A key advantage of the neural approach is that the system itself performs the bulk of the work for establishing the implicit knowledge. A second advantage lies in robustness, in that a neural network will degrade gracefully rather than catastrophically when its structure is modified in minor ways.

We turn now to a general framework for innovative systems. A *creative system* L is an entity which operates in some *environment* E. The object L receives certain *input* stimuli Σ from its environment. The domain of Σ is a set of physical quantities such as pressure or temperature; its range is usually some subset of the real numbers.

The purpose of the system is stipulated in terms of a set of functional requirements (Kim, 1990b). The functional requirements define the set F of acceptable final states. The system L, as an adaptive unit, must incorporate mechanisms which specify its own behavior patterns. Hence L encompasses the input transition function M and the output function N, as well as the set F of final states plus the other internal states of the kernel subsystem K.

The kernel K consists of the set F of final states and some data **D**. The terminal cluster F is specified by the functional requirements **F**, while the data **D** is determined by the messages from the input function M.

The main component of the input transition function M is the set of input transition rules R_M. The function M also encompasses the meta-level operators **O** and the plans **T** of the adaptive system. The output function N is specified by the rules R_N which determine the external behavior of the system.

Many phenomena in process control exhibit random characteristics. In such stochastic domains, the automata-theoretic model may be generalized to include probability distributions over state transitions (Kim, 1993). An example of such an application-dependent factor lies in the severity of consequences and the level of risk aversion involved (Kim, 1992). These models can be used as the basis for the determination of stochastic behavior such as mean transition times and rates of adaptation.

3.2 Application to Specific Adaptive Techniques

The general adaptive framework may be tailored to specific approaches to learning systems. In one approach, the knowledge is represented explicitly as a set of rules in terms of triggering conditions and corresponding actions. A specific category of the production approach is found in learning classifier systems.

In a production rule system, the system L operates by utilizing a self organizing strategy or adaptive *plan* **T** in order to attain a set of *functional requirements* **F**. In a composites processing application, **F** might have 2

components relating to density and thickness in a cured part.

The degree to which the functional requirements are met is indexed by a *performance measure* π. If a binary range for π is chosen, for example, then we might use $\pi = 1$ if all the functional requirements are met, and $\pi = 0$ otherwise.

The domain of action of the adaptive plan **T** is the set of *rules* **A**. **A** may be interpreted as the potential knowledge base of decision rules that L can employ. In general, the knowledge base **A** is *potential* rather than *actual*: at any time t, L may have only a small subset of the possible set of rules.

The adaptive plan **T** yields a sequence of selections from a set of *operators* **O**. In other words, **T** invokes particular operators w in **O** to modify structures in **A**. An example of an operator is the splicing and mixing of two rules in **A**; this operator is known as a *genetic algorithm*.

The set of general structures **A** may be partitioned into two subsets: the set of *basic structures* **B** and the *memory* **M**. For some basic rule B in **B**, let $B(t)$ denote the particular item applied at time t in order to fulfill the functional requirements **F**. At each cycle, the basic rules **B** observe the contents of memory **M** and may choose to generate a message. This message may be an output Ω directed to the environment or a memo m sent to augment the memory **M**.

The general adaptive model can be interpreted as well in the context of neural nets, also known as the connectionist model. In general, the connectionist approach does not store input data **D** in explicit form. Rather, the inputs are absorbed into the organizational structure of the system. More specifically, the input is transformed into a set of node values plus arc weights which depict the relationships among the nodes.

At lower levels of representation in the human brain, data is encoded in the form of voltage differences and structural changes, both mediated by chemical action. Yet, at some level of awareness, we are conscious of aggregate concepts (e.g., systems), relationships, (e.g., bondings), and abstractions (e.g., freedom). Hence a practical system must effectively couple the implicit and explicit forms of representing knowledge (Shibazaki and Kim, 1992).

To manage complexity, the reasoning modules of the overall structure can be arranged into a layered architecture (Kim, 1990b). In each layer, a unit may communicate with other units within the same layer, or with the layer immediately above or below it. A unit communicates with an adjacent layer through a standard protocol, and is otherwise unaware of the neighboring layer's internal structure or processes. This convention facilitiates complexity management through information hiding, and supports the modification or upgrade of any cell with minimal impact on other layers.

The framework and concepts for adaptive systems have been tailored to applications in various domains such as autonomous mobile robots and manufacturing systems. An effective control strategy is to incorporate the techniques of closed-loop control, real-time sensing, and adaptive techniques (Egilmez and Kim, 1990, 1992).

4. Case Based Reasoning

A creative system should make increasingly useful decisions as it accumulates experience. This is the express goal of the work in case-based reasoning (CBR). As with any approach, CBR has its advantages and limitations.

Perhaps the most important advantage of CBR is the affinity to human learning. People take account of observations and utilize them for future decision making. Often the extrapolation to new situations is ad hoc, as in modifying a recipe for a beef dish into one for shrimp. In other cases, the extrapolation is more formal and takes the form of inductive propositions such as formulas, principles, laws, or rules of thumb.

Related to the affinity of CBR to human learning is the ease of enhancing system performance. The knowledge in a particular domain can be stored in formats which are conventional for that domain. For instance, a knowledge base for synthesizing photonic sensors can store the information about previous designs in the format used by human designers. This is in contrast to other knowledge level representations such as production rules, in which the system developer is required to extricate the pertinent decision rules used by a human. Obtaining such rules can be a long and arduous affair, since humans often perform admirably in tasks without having made any attempt to formalize the decision process into a set of explicit production rules. Even worse, a person may be competent in a task and, even with effort, be unable to verbalize his decision rules. A case in point lies in overtaking a car in a two-lane highway despite traffic approaching from the opposite direction. A proper formalization would require the use of differential equations to describe the equations of motion; but some drivers have failed to realize that they require college level math before they can perform such everyday feats.

The CBR methodology can be effective even if the knowledge base is imperfect. Certain techniques of automated learning, such as explanation-based learning, work well only if a strong domain theory exists. In contrast, CBR can use many examples to overcome the gaps in a weak domain theory while still taking advantage of the domain theory (Porter et al., 1990). CBR can also be used when the descriptions of the cases, as well as the domain theory, are incomplete (Sycara and Navinchandra, 1991).

Certain techniques for automated learning, such as explanation based learning, rely solely on positive examples of that concept while ignoring negative ones. In resolving difficult problems, however, negative results are the stock in trade for the decision maker (Kim, 1990a). As with a human, a CBR system can take note of the limitations of imperfect solutions and attempt to eliminate them in the quest for a better solution.

A further advantage of CBR is the relative ease of combining techniques with other approaches. This

advantage comes from the high-level representation of knowledge, and the attendant ease of linking information with that in other formats. An example of such compatibility is a system which uses case reasoning to solve problems whenever possible; otherwise it resorts to heuristics to decompose a problem into a simple one (Maher, 1991).

Perhaps the main limitation of CBR is its susceptibility to the misinterpretation of the knowledge in its case base. Like a journalist reporting a quote out of context, any item of information may be misconstrued. This is a perennial hazard in any field of endeavor, automated or otherwise. One way to address the problem in CBR is to encode deeper-level domain knowledge in addition to the surface features of various cases.

Despite the variety of limitations described, CBR offers the advantage of knowledge-level reasoning without the need for explicit translation of experience into special formats for machine interpretation. In a larger sense, CBR is difficult to avoid in practical contexts. Decisions are made against the light of past experience, whether such information is encoded into a machine-compatible format or is available only informally. In addition, each new problem and the attendant solution constitute a case. As a result, case-based reasoning is unavoidable whether or not the techniques are labeled as "CBR".

Case reasoning requires the retrieval of past experience in the form of cases. In this task, two types of difficulties can arise. The *matching* problem refers to the task of associating a new problem to pertinent prior cases. A key issue lies in retrieving prior cases which are similar to the new problem in substantive rather than superficial ways. This relates in part to the issue of *indexing*, which deals with the organization of the case base.

4.1 MATCHING PROBLEM

Problem solving in any arena is dedicated to the resolution of a goal. To this end, the decision maker must find prior cases which resolve the specified or comparable objectives, rather than those that match only surface features having little effect on the effectiveness of the solution. For instance, two structures may be of similar shape and both made of composites; but one is intended to reinforce the legs on an ordinary desk, while the other is a stronger material for reinforcing an aircraft wing. Consequently, a CBR system must search through the base of previous cases by first attempting to find solutions that meet the primary design goals, and subsequently examine them against secondary objectives.

Psychological studies indicate that people, especially novices, often tend to recall analogues on the basis of superficial surface similarities rather than deeper relationships (Forbus and Gentner, 1986). Even so, they believe that similarities of the latter type better reflect the proximity of two cases. As a result, the representation of deeper relationships, such as the intended functionality of different components in a control system, will facilitate the retrieval of close precedents.

The matching problem can be addressed in a number of ways. The default scheme is to perform an exhaustive search through the case base each time a new problem arises. However, system performance can be degraded by such a tedious approach.

A more systematic way is for a human to identify the relevant prior cases. Unfortunately, this technique requires continuing human intervention if a system is to improve its performance over time.

To automate the task of matching in CBR, previous cases can be organized in some fashion to enable the rapid identification of potentially relevant cases. To this end, previous solutions can be indexed by their key attributes and the features which distinguish them from other cases.

4.2 INDEXING PROBLEM

The indexing problem refers to the task of storing cases for effective and efficient retrieval. In terms of efficacy, the subissues are *accuracy* - finding only relevant cases - and *completeness* - identifying all relevant cases.

In general the prior cases retrieved by case reasoning will match the required solution only imperfectly. In particular, the source cases may fail to fulfill some of the requisite objectives. At this point, an analogy can be formed between the functionality of the precedent solutions and the goals of the current problem. The prior solutions may then be modified to eliminate or circumvent the limitations.

To illustrate, consider a novel control problem which largely resembles a prior one in the case base. The use of the previous solution, however, would lead to sluggish response in the current application. This limitation could be addressed by introducing an extra component based on derivative control. But the revised design leads to a spasmodic control signal due to noise in the input signal. The latter difficulty can be resolved in various ways, such as inserting a moving-average smoothing unit immediately before the derivative controller. In this way, a process of iterative refinement can be employed to adapt an old solution to the new problem context (Kim, 1990a).

5. Case Study in Creative Control

The intelligent control of autonomous entities has been a subject of extensive investigation, especially over the past couple of decades. One relevant project is the Pilot's Associate (PA) program sponsored by the United States Air Force and the Defense Advanced Research Projects Agency. The PA is designed to assist the work of a pilot by taking into account external, internal and human factors, including the cognitive load on the pilot. With prior authorization from the pilot, the system considers the environmental context and takes action when appropriate. The action may take the form of informing the pilot of a malfunction, recommending a procedure, or even taking over a mundane

task when the pilot is engaged in more pressing activities such as combat maneuvers.

Another relevant endeavor is the Autonomous Land Vehicle (ALV). Although the ALV program has not quite met the expectations of its sponsors or commentators, its successes and failures are instructive, as in any advanced-technology project. Perhaps the main stumbling block in this project has been the lack of an adequate technology base to support pattern recognition and image understanding in a complex landscape. With improved techniques, the ALV project should meet with greater success. Promising developments along these lines include the use of array-based laser ranging strategies, the use of multiple electromagnetic frequencies for enhanced disambiguation of objects in a scene, improved algorithms for distributed processing, and integration of analog and digital processing through optoelectronics for such tasks as pattern recognition.

The judicious combination of reasoning techniques can lead to practical systems for creative control. The control architecture to be described is explicitly designed to accommodate diverse technologies, including new developments in image processing and pattern recognition as they emerge in the years ahead.

In the past, much of the work on adaptive controllers has focused on specialized sets of tasks. A case in point is the development of a rule-based system for failure diagnosis during flight (Handelman and Stengel, 1988). Although diagnosis is a key function in the monitoring of an aircraft, it is only one of many such critical tasks.

Another example of a constrained control task lies in a reconfigurable flight controller through the use of mutiple explicit models. In this approach, a control scheme is specified beforehand for each set of potential failures or malfunctions, and brought online under appropriate circumstances. More specifically, a control scheme is predefined for each case of a single or double failure in the portfolio of sensors and/or actuators. This approach has worked well in simulation for an aircraft in landing phase (Maybeck and Stevens, 1990).

The preceding approach of multiple models, however, is impractical for a generalized controller. Suppose there are a actuators and s sensors. Then, in general, $a + s$ types of single failure may occur, as well as as double failures, leading to a total of $as + a + s$ scenarios of distress. When a and s are equal to 3 each, as used in the simulation study, the total number of scenarios is only 15. However, a real aircraft has more than 3 sensors and actuators. Even if a and s were only each equal to 10, the total number of scenarios is 120. And even this number is misleadingly low for the following reasons: (1) a sensor or actuator may be subject to multiple levels of degradation, not simply the binary states of "fully functional" versus "fully inoperable", (2) an aircraft may well face damage in 5, 9, or 34 components rather than just one or two, and (3) the impact of sensors and actuators on other system functionality has been ignored. An example of the third limitation is found in the contention between increased thrust to reduce delay during a problematic flight, or decreased power to alleviate stress on damaged subsystems.

These complications highlight the limitations of attempting to specify in advance the entire combinatorial space of potential component states, and generating software for each possible scenario. For these reasons, a modular, self organizing approach to control automation is indicated.

5.1 Some Implementation Issues

The representation of system descriptions and inter-relationships can easily be encoded in a high level language such as Prolog. One of the advantages of using Prolog is the close affinity of constructs in the language to other programming techniques such as frames, and to cognitive structures such as semantic nets (Kim, 1991). The Prolog language has been used extensively over the past decade in applications ranging from design automation to production control and learning systems (Kim, 1993, etc.). In the interest of computational speed, however, it may be advisable to implement some or all of the production version of a creative control system in a procedural language such as C.

The general framework for adaptive systems described previously can be tailored to specific applications. As an illustration, we consider the task of formulating a strategy for flight control. Information from the environment is acquired through the sensors and by observation by the pilot; the input streams are processed by the flight control system in conjunction with dynamic mission requirements and decisions of the pilot. The actions from the system, implemented through the activators, in effect modify the external environment - such as flying over a thunderstorm - or the internal environment - such as the buttressing a weakened structure with a similar component.

The primary modules of the system are organized into a multilevel architecture. In this arrangement, the reasoning phases of observation, prediction, action, and evaluation proceed for each top-level module as well as its subsystems in turn. The inferences for the respective modules are executed in parallel through distributed processors.

The domain knowledge for flight control is encapsulated in the form of area specialists. The primary structure for mutual communication among the modules is a blackboard. In the software implementation, however, communication efficiency can be gained through direct linkages among highly coupled modules, each with the capacity to send messages of variable priority; the highest priority level corresponds to the interrupt feature in tightly coupled processor architectures such as those in conventional mainframe computers.

A versatile system must incorporate input from multiple sources. To illustrate, consider the realm of sensor fusion for mobile objects. The functional requirements on sensor systems for a mobile system such as aircraft may be broadly classified into internal and

external categories. Internal functions deal with proprioceptive tasks such as monitoring for homestasis and malfunctions. External sensory functions, on the other hand, include navigation, feature detection, object identification, and verification.

Any sensory modality exhibits a unique set of advantages and limitations, such as resolving capacity or attenuation in the atmosphere. Moreover, any physical device is also susceptible to malfunction or failure. For this reason, the likelihood of accurate perception is increased by integrating the input from multiple modalities as well as multiple physical units within a single modality. An example of the effective fusion of several modalities lies in the use of passive as well as active modes of infrared and radio frequency emissions for detecting airborne objects.

5.2 Coordination of Subsystems

A creative controller must supervise and coordinate the activities of many subsystems. In this capacity, the control strategy must accommodate different policies for coordination among disparate modules.

In the area of sensor fusion, for instance, one supervisory policy lies in the use of arbitrators. The technique has been developed for the autonomous navigation of airborne vehicles. In this approach, each feature extraction unit submits 10 position estimates for a particular landmark such as a building, bridge, or road, based on the highest correlation between sensed features and a prior map of the terrain. Suppose that at least one of the estimated positions lies within a zone of indifference called a "dynamic envelope"; then among these candidates the arbitrator selects the position corresponding to the highest correlation. On the other hand, if no candidate position lies within the zone of indifference, then the estimate of aircraft location is based on the inertial navigation system (Cameron et al., 1988).

The different subsystems of an aircraft must interact with each other to fulfill mission requirements. In a hypersonic aircraft, for instance, a high rate of descent is best effected by inducing a roll to reduce the vertical component of the lift vector from the wings ("bank to dive"). This procedure must be coordinated not only with the propulsion subsystem, but with the cross-range regulator due to the lateral discrepancy introduced with respect to the initial trajectory. The mediation can be accomplished by the use of a control module called a "resolver": an interface between the outer loop of navigation and guidance, versus the inner loop of longitudinal and lateral trajectory regulation. In particular, the resolver performs a coordinate transformation by converting vertical and lateral acceleration commands into a lift vector command defined by a roll angle and a load factor normal to the coordinate system of the aircraft (Raney and Lallman, 1992).

The coordination of multiple decision making units has also been investigated through the use of game theory and team theory. In particular, the application of coordinative and competitive heuristics to the coordination of mobile robots has been validated through simulation studies (Egilmez and Kim, 1992; Kim, 1993).

5.3 Learning Through Knowledge Integration

A creative system, to be effective, must have the ability to draw on knowledge from diverse sources. The diversity can take the form of different formats for encoding knowledge as well as multiple modules within a single representation format.

The synergism of diverse formats was discussed in an earlier section, such as the integration of production rules with the output of neural networks, or the interplay of case information with rules of thumb. The second major category of integration relates to the enhancement of performance through the fusion of new information with the old in an existing knowledge base.

In human endeavors, failure is often a stepping stone to success. When something unexpected occurs, for instance, then a gap in a person's knowledge is revealed. The failure to predict the outcome of a preliminary solution leads to renewed evaluation and further insight.

In a similar way, a CBR system can be designed to take advantage of failures, including inadequate prediction of the consequences of a proposed solution. The modes of failure as well as the proposed remedies can be stored along with the case for future reference.

The smooth fusion of old and new knowledge is imperative for an effective learning system. To illustrate the process of learning through knowledge integration, we consider the diagnosis of schedule slippage in the construction of a sensor device. The causal relationships among the factors behind schedule slippage can be represented in the form of a fault tree. The system can translate a causal graph representing slippage into natural language explanations by associating each cause with a short verbal statement to be used in the explanations. For instance, the causal factor of *breakdowns* is associated with the message "Equipment is prone to breakdowns."

The knowledge of causal factors for a particular component can be merged with those of other components. The unified causal tree then represents an expansion of the reasoning capability of the system, including the ability to present the integrated knowledge to a human questioner.

Figure 1 shows a sample interaction to illustrate learning through the integration of knowledge into a cohesive structure (Kim and Novick, 1991). In particular, the causes of slippage due to breakdowns pertains to one case while those due to sensor inaccuracy relate to another. In this way a comprehensive base of slippage factors can be constructed from diverse cases without human intervention.

6. Conclusion and Future Directions

Autonomous systems can exhibit creative problem solving through the effective integration of knowledge. The knowledge may be found in formalized structures such as rules as well as informal formats such as case experience.

```
U> What's holding back the scanner
   team?
S> The test equipment is suffering
   from sensor inaccuracy.
U> What's causing it?
S> Sensor inaccuracy is due to these
   factors:
    1. Mechanical vibrations
    2. Excess heat
    3. Voltage surges
U> What do you know so far about
   slippage factors?
S> Would you like to see a cause-effect
   list?
U> Yes.
S> The causes of slippage are as
   follows:
    1. Breakdowns
       a. Poor maintenance
          (1) Manpower
               (a) Exogenous slippage
    2. Sensor inaccuracy
       a. Mechanical vibrations
       b. Excess heat
       c. Voltage surges
   Would you like to see a fishbone
   diagram?
U> No need.
```

Fig. 1. Sample interaction illustrating creative problem solving through knowledge integration. Each statement preceded by a "U>" denotes the user, while "S>" denotes the system.

Learning and creative decision making have been demonstrated in software in various domains. The demonstrations relate to simple tasks in fields such as engineering design and supervisory control. For instance, control systems have been designed to provide continuous, real time, online monitoring of both external and internal functions for automated supervision. In such a context, the system objective is to accommodate dynamic system requirements by attending to routine activities as well as exceptional conditions such as damage, failure, or other anomalies.

The complexity inherent in a creative system can be managed by the judicious use of a spectrum of methodologies in system design and software engineering. In particular, a multilevel architecture can support a layered organization, accommodate modular reasoning, ensure information hiding as appropriate, and provide a cooperative blend of declarative as well as procedural knowledge. The resulting knowledge base incorporates effective techniques from diverse fields, including case reasoning and adaptive control theory.

The approach is designed to support rapid system development and modification. It will readily accommodate future advances in areas such as computing hardware and processing techniques.

A direction for the future is to apply such methodologies for creative systems to more complex environments. For example, a creative controller will be applicable to any type of process control in the industrial sector, such as mobile robots or as an aide to pilots in commercial airliners. The technology of a highly intelligent control system will be useful to the automation of complex plants, whether in the supervision of an automated factory, the management of energy in an office building, or the regulation of a public utility. Other applications include resource management in communication networks as well as service as aides to automobile drivers or as controllers for automated taxis.

In summary, the development of innovative systems can serve to elucidate the nature of human creativity. Moreover, such systems can augment human decision making or even replace manual problem solving in certain tasks requiring creative solutions.

References

Cameron, W.L., Fain, H. and Bezdeck, J.C.: 1988. An intellligent system for autonomous navigation of airborne vehicles. P.S. Schenker, ed., *Sensor Fusion*, Bellingham, WA: SPIE, v. 1003, pp. 451-69.

Egilmez, K. and Kim, S.H.: 1990. A logical approach to knowledge based control. *J. of Intelligent Manufacturing*, v. 1: 59-76.

Egilmez, K. and Kim, S.H.: 1992. Deployment of robotic agents in uncertain environments: game theoretic rules and simulation studies. *Artif. Intel. for Engineering Design, Analysis and Manufacturing*, v. 6(1): 1-17.

Forbus, K.D. and Gentner,D.: 1986. Learning physical domains: toward a rheoretical framework". In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, eds., *Machine Learning: An Artificial Intelligence Approach*, v. 2, Morgan Kaufman, Los Altos, CA: 311-48.

Handelman, D.A. and Stengel, R.F.: 1988. Rule based mechanisms of learning for intelligent adaptive flight control. *Proc. American Control Conf.*, Green Valley, AZ: American Automatic Control Council, pp. 208-213.

Holland, J. H.: 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. of Michigan Press.

Kim, S. H.: 1990a. *Essence of Creativity*. New York: Oxford Univ. Press.

Kim, S. H.: 1990b. *Designing Intelligence*. New York: Oxford Univ. Press.

Kim, S. H.: 1991. *Knowledge Systems through Prolog*. New York: Oxford Univ. Press.

Kim, S. H.: 1992. *Statistics and Decisions*. New York: Van Nostrand Reinhold.

Kim, S. H.: 1993. *Learning and Coordination*. Dordrecht, Netherlands: Kluwer, in press.

Kim, S.H. and Novick, M.B.: 1991. Role of learning in design systems. Cornell Univ., Ithaca, NY: unpublished.

Lenat, D.: 1983. EURISKO: a program that learns new heuristics and design concepts: the nature of heuristics, III: program design and results. *Artificial Intelligence*, v. 21(2), pp. 61-98.

Maybeck, P.S. and Stevens, R.D.: 1990. Reconfigurable flight control via multiple model adaptive control

methods. *Proc. 29th IEEE Conf. on Decision and Control*, NY, IEEE, pp. 3351-56.

Maher, M.L.: 1991. Machine learning in design expert systems. In J. Leibowitz, ed., *Expert Systems World Congress Proc.*, v. 1, Oxford, UK, Pergamon, pp. 728-36.

Porter, B., Bareiss, R., and Holte, R.C.: 1990. Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, v. 45(1-2): 229-63.

Raney, D.L. and Lallman, F.J.: 1992. Control integration concept for hypersonic cruise-turn maneuvers. Hampton, VA: NASA Tech. Pap. 3136, Feb.

Shibazaki, H. and Kim, S.H.: 1992. Learning systems for manufacturing automation: integrating explicit and implicit knowledge. Presented at International Conference on the Manufacturing Science and Technology of the Future, Stockholm, Sweden, June 1989. Also in *Robotics and Computer-Integrated Manufacturing*, v. 9, 1992.

Sycara, K. and Navinchandra, D.: 1991. Index transformation techniques for facilitating Creative Use of Multiple Cases. *Proc. 12th Int. Joint Conf. on AI*, Morgan Kaufman, Los Altos, CA, pp. 347-52.