All gadget and no representation makes Jack a dull environment

David Franklin and **Joshua Flachsbart**
Department of Computer Science
University of Chicago
Chicago, IL 60637
franklin, josh@cs.uchicago.edu

Abstract

In an Intelligent Environment, the user and the environment work together in a unique manner; the user expresses what he wishes to do, and the environment recognizes his intentions and helps out however it can. If well-implemented, such an environment allows the user to interact with it in the manner that is most natural for him personally. He should need virtually no time to learn to use it and should be more productive once he has. But to implement a useful and natural Intelligent Environment, the designers are faced with a daunting task: they must design a software system that senses what its users do, understands their intentions, and then responds appropriately. In this paper we argue that, in order to function reasonably in any of these ways, an Intelligent Environment must make use of declarative representations of what the user might do. We present our evidence in the context of the Intelligent Classroom, a facility that aids a speaker in this way and uses its understanding to produce a video of his presentation.

Introduction

Intelligent Environments can provide an escape from the traditional model of computer-user interaction (i.e. the user enters a command and the computer executes it) that places all of the responsibility of interaction on the user. Also, for many interesting tasks, the computer keyboard is not the most efficient mode of interaction and Intelligent Environments allow their users to use these other modes. Natural interaction should not simply involve substituting your hand for the mouse or substituting complicated (and unintuitive) hand gestures for command-key combinations. Truly natural interaction must allow the user to communicate his intentions just as he would to another person.

Our research is specifically in service of the construction of an "Intelligent Classroom." This facility will use zoomable cameras and directional microphones to observe a speaker and will attempt to coordinate its various automated components with the speaker's lecture (i.e. setting lights, advancing slides and showing videos.) In this way, the Classroom will act as if it were under the control of an audio-visual technician, freeing the speaker from worrying about many of the technical details of giving a presentation. In addition, the Classroom will produce a video feed of the presentation, suitable for closed-circuit viewing (i.e. distance learning) or for storing in presentation archives.

To anyone who attempts to design an Intelligent Environment, it quickly becomes apparent why natural interaction is all but absent from the world of computer systems: it is really hard to implement! In an Intelligent Environment "the user enters a command" becomes "the person physically does something." And "the computer executes it" becomes "the computer senses the person's actions, figures out the person's underlying intentions, figures out what to do about it and then does it." The model of computer-user interaction has become much more complicated because now the computer is responsible for a fair share of the effort of communication. To fulfill its responsibility, the computer must build an understanding of a user's action(s) that agrees with what a human observer would conclude in the same situation. This means that the computer should consider much of what a human observer would:

- the context the person's action(s) occurred in. Often a given action will have very different meaning based on the situation it is used in.
- what goals the person is likely to have, and what actions he might take in service of those goals.
- what sorts of responses are appropriate to different goals and sequences of actions.

To produce a competently intelligent environment, a designer needs to make use of these considerations in helping it to sense, understand and act rationally. This

designer has two choices for how to employ this in his environment: procedurally or declaratively. In a procedural approach, the designer would write programs that explicitly say what to do in any given situation; the considerations of context and reasonableness are encoded into the procedures he writes. Such procedures are often written in the form of a giant decision tree where each intersection performs a test and branches as appropriate. The procedural approach is very useful and produces excellent results in simple domains where it is always easy to decide what the right thing to do is. But, when a designer attempts to extend his program, a primary weakness of the procedural approach is revealed: it scales poorly. To add an additional behavior, the designer must look through all his code to determine how the new behavior interacts with the old. Very often, the designer will have to completely rewrite large portions of his program to implement even a small change to the behavior of the environment.

The declarative approach, on the other hand, yields a very different set of benefits and difficulties. In the declarative approach, behaviors are treated as data structures that the program operates on. Also, information such as context and knowledge about what is likely to happen will also be represented in this way. The designer of this kind of system has two tasks: to design these declarative representations, and to write the program that uses them to decide what to do. If designed well, such a system can easily be extended simply by representing more behaviors; the program itself will not need to be altered. Programming such a system is initially an imposing task, but, once in place, it is easy to add increasingly complicated behaviors.

In the next few sections, we show how declarative representations of these things that human observers consider are invaluable in successful interaction (that is, in sensing, understanding and acting.) Then we discuss how we have implemented these ideas in the Intelligent Classroom. Finally, we examine some related work and summarize our position.

Sensing

For an Intelligent Environment to interact intelligently with its users, it must be able to obtain a reasonable representation of what is happening. Ideally, the environment should acquire a human-level understanding, but, with current technology, it is impossible to even approach this level of understanding. Because of this, most vision researchers abandon this goal and focus on solving smaller subproblems. Using this approach, these researchers are able to produce systems that are highly effective at solving their specific task, but they are unable to obtain all of the information that would be necessary to build a good representation of the world.

We are convinced that, through dynamically applying contextual knowledge, we can build a system that combines these highly specific approaches into a much more general vision system. In such a system, the vision system cannot be viewed as a completely independent component that decides for itself how to best build its representation of what is happening. Instead, it interacts closely with the other components (understanding and acting), using the same declarative representations they use, to decide not only what to sense, but very often how to sense. This means that the vision system does not waste effort doing irrelevant sensing, and, because it utilizes context provided by the other components, it is able to sense more effectively.

For example, for the Intelligent Classroom to understand what the speaker is doing, it needs to carefully track the motion of his hands and body. If the Classroom recognizes that the speaker is walking across the room, the vision system can use information about his current motion to determine where he is likely to go, and so the vision system can greatly restrict the region that it looks for the speaker in. However, if the Classroom turns out the lights while it plays a video, it is no longer necessary to try to track the person: it would be very difficult and the information would not be useful to the understanding.

Context tells us what to sense

With current computer vision techniques and existing computer hardware, it is impossible to sense all the potentially relevant characteristics of a physical domain all the time. However, through close interaction with the understanding and acting components of an Intelligent Environment, the vision system can determine exactly what characteristics are relevant at any moment. Based on the current context (specifically, "what is happening at this moment") the vision system decides what sensing will be useful to the environment as a whole and so is able to eliminate a great deal of wasted effort. Using context, the environment is able to acquire all the information it needs without the effort of acquiring anything else. So, when the lights were turned out in the Classroom, it was able to determine that it would not acquire any useful information if it attempted to track the speaker, so it stopped trying to track him.

Context tells us how to sense

If an Intelligent Environment has a good representation of its world, it will be able to change the way it senses particular characteristics; it can use context to

tell it how to sense. Vision researchers that tackle specific tasks always make strong use of context; they use the very specific constraints associated with their task to make it tractable. In doing so, they sacrifice general applicability so that their algorithm will be highly successful in its particular domain. However, through the use of context, a vision system can tailor the vision algorithms it uses to make use of these very specific constraints. Then, because it uses only the constraints that are relevant to a particular situation, it can be very general, but because it uses very specific constraints, it is also able to be effective.

For example, in the Intelligent Classroom we use a technique of static background subtraction to locate and track the speaker. This technique will only work if the camera is stationary and there is not too much other motion in the camera's view. So, if the speaker moves out of the view, the camera cannot pan to follow him and still use this technique. Instead, a color-based tracking technique could be used because this technique can be successful with a panning camera. The sensing system uses the context of the situation to recognize that the current technique will fail and to select a technique that will succeed.

Understanding

For an Intelligent Environment to behave rationally, it must act based on an understanding of what is going on within. To gain this understanding, the environment must have some technique for mapping a sequence of actions into an explanation. For a trivial environment, where there are only a few activities that a person might perform, designers may successfully employ simple approaches such as purely reactive systems. However, as environments (and the expectations for them) become more complex, these simple approaches will fall short; every time the designer adds an additional behavior to the environment, he must deal with the myriad interactions the behavior produces with the existing environment. Expanding such an environment requires essentially reprogramming it.

If it employs declarative representation, the environment may utilize Plan Recognition to build its understanding. The advantage of this approach lies in its natural scalability: to add a new behavior, the designer can simply add the appropriate representation of the behavior. The existing representation need not be touched at all. But, with a larger body of behaviors, the environment is inevitably faced with the problem of ambiguity. Where with a small set of behaviors, the environment may be able to determine what the person is doing (understand what is going on) after witnessing a single action, with a larger set, the environment may need to observe several actions to gain this understanding. This ambiguity is inevitable; a human observer would not be able to state with certainty what the person is doing in those situations either. The advantage of using declarative representation and plan recognition is that this ambiguity is made explicit and the designer can decide how the environment will deal with it.

Plan recognition in an Intelligent Environment can be viewed as the process of continually refining the set of possible explanations for the actions it has observed. When the environment observes an action that does not fit any of its possible explanations of what is going on, it will propose new explanations. When the environment observes an action that contradicts some of its possible explanations, it will remove those explanations from consideration. Through these two activities, the environment tries to arrive at the simplest explanation for what it observes. If it uses declarative representation, the environment can determine what observations it could make that would disambiguate a given situation quickest. ("If, after the speaker walks over to the chalkboard, he picks up a piece of chalk, then I know that he intends to write on the board.") When an environment recognizes that it is in such a situation, it can then direct its sensing system to make the appropriate observations.

In the next section, we look at how the environment uses its understanding of what is going on to determine what it should do. Often, the environment will need to take action before it has fully disambiguated what is going on. For example, in the Intelligent Classroom, it must always video-tape the presentation. Even if it is unsure of exactly what the speaker is doing, it must keep filming. In almost any Intelligent Environment, it will often be the case that the environment will need to take action before it is able to completely determine exactly what is happening.

For an environment that uses declarative representations, the designer may choose to employ one of at least two possible ways of addressing this problem. The first approach involves using any of a number of probabilistic methods to determine which of the possible explanations is most likely, and then acting on that. Such an approach could be as simple as maintaining knowledge of the *a priori* likelihood of each possible explanation, or as complicated as using full-blown Bayesian Networks. The second approach is to use the principle of least-commitment: in cases of ambiguity, act on a more general explanation that is in agreement with all the possible explanations. (When the speaker walks to the chalkboard, I know that he is at least going somewhere to do something.) If the set of all possible

explanations is arranged in a specificity hierarchy, the general explanation can be found by finding the most specific explanation that is a generalization of all the candidate explanations. In practice, a combination of the two approaches will probably produce the best results. Without the use of declarative representations, a designer cannot even make these sorts of decisions.

Acting

In the previous sections, we discussed how, by using declarative representation, an Intelligent Environment can determining how to sense, what to sense, and how to interpret what has been sensed. Now we look at how their use can aid an environment as it decides what to do based on its understanding of what is going on. An Intelligent Environment has the special characteristic that it invariably tries to aid its users in their tasks; it a wholly cooperative agent. This provides the designer of such an environment with a unique opportunity: for any activity a user might be involved with, the designer can specify how the environment can cooperate.

In the simplest of environments, successful cooperation may be considered to be of the form: the user does something, and then the environment performs a cooperative action. But sometimes cooperative behavior requires a much more involved interaction, with the actions of the user and the environment intricately interleaved. In such situations, the environment must carefully synchronize its actions with those of its user. For example, in the Intelligent Classroom, when the speaker walks up to the chalkboard to write, the Classroom should adjust the lights to illuminate the appropriate portion of the chalkboard, and should change the camera framing technique for the video to show what the speaker writes on the board. If either of these actions occur at inappropriate times, the Classroom's behavior will not appear to be particularly intelligent.

We have developed a language for declaratively representing plans that have sequences of actions performed by one or more agents. This language allows a designer to specify precisely how the steps in such plans need to be synchronized. With such a representation, the environment follows along with the user as he executes his part of a plan, and it acts at the precise moments specified in the plan. By using declarative representation of plans, an Intelligent Environment is able to be cooperative to a degree that would be very difficult (if not impossible) to achieve otherwise.

A glance inside the Intelligent Classroom

The previous sections discuss our many reasons for adopting a strong representation approach to designing

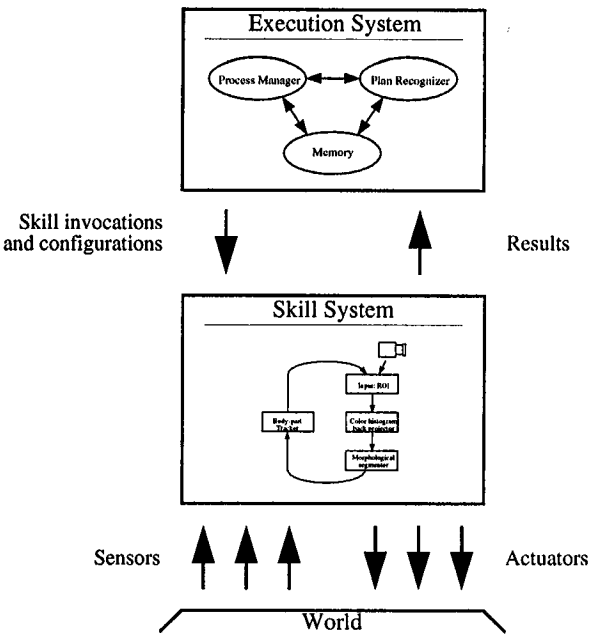

Figure 1: Environment Architecture

the Intelligent Classroom. In this section, we will discuss how it has been (or is being) implemented in our laboratory. Our motivation for writing this section is to establish that it is reasonable to try to use such representations (it might seem easy to argue that more *ad hoc* approaches would be, at least initially, much easier to construct.) We will first describe the overall architecture for our environment, and then examine in detail a few key components and representations.

Architecture

Our design for the Intelligent Classroom borrows much from research in autonomous robotics. We view the Classroom as a robot, where its interior is the world and it uses sensors and actuators in a manner analogous to that of any other robot. As a result, we have designed the Classroom to follow the multi-layered approach promoted by Firby and others in the field of robotics. At the lowest level, there is a Skill System that links together reactive skills and vision modules to form tight control loops. And, above that, there is an higher-level Execution System that directs the Skill System, builds an understanding of what is going on, and figures out (at a high level) what it ought to do. Figure 1 shows the system architecture we use for the Intelligent Classroom. The Execution System interacts with the Skill System in two ways: it tells it which skills to activate (and how to connect them together) and it dynamically sets parameters for the active skills (as suggested by context). The Skill Sys-

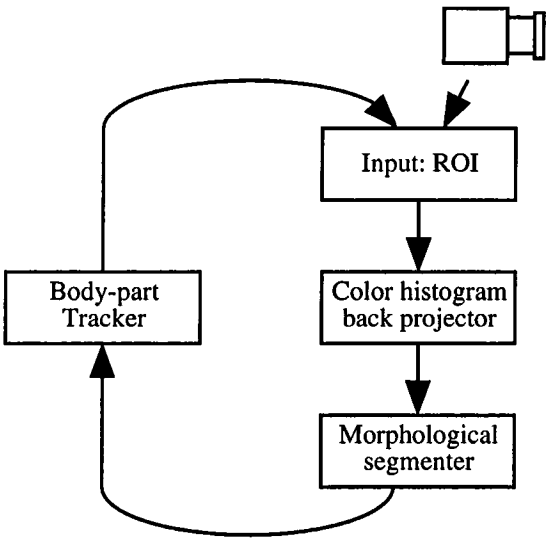

Figure 2: An example pipeline

tem interacts with the physical world: it uses cameras and microphones to sense and it controls the actuators to cause changes.

Skill system: Gargoyle modules and other skills

The Skill System is responsible for all physical interaction with the world. It uses cameras and microphones to sense, and it controls the actuators that set the lights, aim the cameras, play the videos, etc. This interaction is accomplished through the running of sets of modules, connected as tight control loops. Each module in one of these "pipelines" performs a simple task; it uses sensor inputs and outputs from other modules to compute its own outputs. These outputs may be used as inputs to other modules or to actuators for controlling the environment. In the pipeline in Figure 2, the modules on the right work together to locate key features of a speaker in the Classroom and pass the results to the module on the left which then computes the likely future locations of those features. The "Input" module uses a sensor (a digital camera) and a region of interest to construct a small image for the "Color histogram back projector" module to work with.

As context warrants, these pipelines can be dynamically changed through passing new parameters to individual modules (subtly influencing how they operate) and even by adding and removing modules from the pipeline. For example, if the lights are dimmed, it may be necessary to change the threshold used in an edge-detecting visual module (by passing it a lower threshold value.) But if, due to a flashing light, there is a lot of small-scale motion, it may be necessary to substitute a color-based segmentor for an edge-based one.

```
(define-plan (move-to-cboard-and-lecture)
  :main-actor
    (person ?lector)
  :roles
    ((intelligent-classroom ?classroom))
  :accomplishes
    ((do ?lector (lecture-at-cboard)))
  :processes
    ((_p1 ?lector
       (lector-move-to-cboard-and-lecture))
     (_p2 ?classroom
       (observe-lector-move-to-cboard-and-lecture ?lector)))
  :synchronization
    ((starts (_p1 _1) (_p2 _1))
     (equals (_p1 _3) (_p2 _2)))


(define-process (lector-move-to-cboard-and-lecture)
  :main-actor
    (person ?lector)
  :roles
    ((chalkboard ?cboard)
     (chalk ?chalk))
  :steps
    ((_1 (achieve (at ?lector ?cboard))
         (wait-for (at ?lector ?cboard) _2))
     (_2 (achieve (holding ?lector ?chalk))
         (wait-for (holding ?lector ?chalk) _3))
     (_3 (do _write (write-on-cboard-and-lecture))
         (wait-for (_write :done) :done)))
  :time-constraints
    ((duration (30 300 3000))
     (process-duration (_1) (0 5 30))
     (process-duration (_2) (0 5 30)))


(define-process (observe-lector-move-to-cboard-and-lecture ?l)
  :main-actor
    (classroom ?class)
  :roles
    ((person ?l))
  :steps
    ((_1 (do (track-moving-person ?l))
         (wait-for (and (at ?l ?cboard)
                        (chalkboard ?cboard))
                   _2))
     (_2 (do _track (track-person-write-and-lecture ?l))
         (wait-for (_track :done) :done))))
```

Figure 3: Plan and process definitions for the Intelligent Classroom

Plan recognition and cooperation

For the Intelligent Classroom, plans are of a somewhat different nature: because the Classroom needs to interact closely with the speaker, we have decided to use a plan representation to includes both the speaker's actions and what he would like the Classroom to do in response. To accomplish this, we have chosen to represent a plan as a set of "processes" where the Classroom's role in the plan is expressed in one or more of these processes. The plan representation holds temporal constraints that dictate how the Classroom must synchronize its actions (expressed in its processes) with those of the speaker. Figure 3 shows the plans and processes we use to represent the speaker's plan for going to the chalkboard and writing. The plan refers to two processes, one that the speaker runs and one that the Classroom should run. The plan also expresses how the steps in the two processes should be synchronized.

To recognize what plan the speaker is involved in,

the Classroom must recognize that the speaker is running his process in a particular plan. Each process in a plan can essentially be viewed as a sequence of actions (the traditional way of thinking of plans) and so we are able to use existing plan recognition techniques to determine what the speaker is doing. Our approach is, for a sequence of actions, to maintain a list of all potential speaker processes that are consistent with the actions, and to eliminate processes as the speaker's later actions contradict them. This technique allows the Classroom to always know what the speaker might be doing, and to know when all other possibilities have been ruled out.

Our representation for plans has been designed to help the Classroom easily decide how to cooperate with the speaker. The Classroom needs to keep track of where the speaker is in his process in the plan, and then use the temporal constraints to determine when to execute its own actions. To allow the Classroom to take action before it has determined exactly which plan the speaker is involved in, we use a hierarchy of decreasingly general plans so that the Classroom can act on a plan that is a generalization of all the possible plans (this method is described earlier in the paper.)

Related work

The overall architecture for the Intelligent Classroom closely adheres the architecture for vision and action described in (Firby *et al.* 1995). We have altered the representations of plans (RAPs) to explicitly represent the actions of other agents, thereby facilitating the level of cooperation we demand of the Intelligent Classroom. The skill system is made up of the Gargoyle modular visual system(Flachsbart 1997)(Prokopowicz *et al.* 1996) and a modified CRL system (a robotic control system developed in our laboratory.)

The algorithms for recognizing plans have been greatly influenced by Wilensky's PAM system(Wilensky 1981). Both incrementally build representations of what is going on: first trying to explain new input based on current hypotheses of what is happening and then, only when that fails, proposing new hypotheses. Also, both represent what the different agents are doing and why. Our representations of plans and processes have been influenced in many ways by Schank's representation of scripts(Schank & Abelson 1977).

The idea of viewing the operation of the world as the interaction of a set of concurrent processes is adapted from the work of Earl and Firby (Earl & Firby 1997), who also have looked at ways to learn what events should be observed (and when) through the repeated execution of processes.

Our representation of processes as sequences of steps where each step has a number of conditions that it waits for (signals from other processes or memory propositions becoming true) is taken from Firby's work on RAPs(Firby 1994).

Conclusion

Intelligent Environments provide us, as AI researchers, an exciting opportunity to establish new, more natural ways of interacting with computers. We had better live up to the opportunity. Flashy "gadgets" (technology for the sake of technology) may impress people for a ten-minute research demonstration, but to actually change the way people interact with computers we must design our environments to truly cooperate with their users.

In this paper, we argue that to achieve the level of interaction that will truly achieve this, the environment needs to explicitly represent the plans that its users might have. In the Intelligent Classroom, we are taking the use of declarative representation seriously; the sensing, understanding and acting components all make heavy use of the same declarative representations. As our research progresses, we intend to demonstrate that this approach is not only possible, but that it allows for levels of computer-user interaction that cannot be rivaled using procedural approaches.

References

Earl, C., and Firby, R. J. 1997. Combined execution and monitoring for control of autonomous agents. In *Proceedings of the 1st International Conference on Autonomous Agents.*

Firby, R. J.; Kahn, R. E.; Prokopowicz, P. N.; and Swain, M. J. 1995. An architecture for vision and action. In *Fourteenth International Joint Conference on Artificial Intelligence.*

Firby, R. J. 1994. Task networks for controlling continuous processes. *Second International Conference on AI Planning Systems.*

Flachsbart, J. 1997. Gargoyle: Vision in the intelligent classroom. Master's thesis, University of Chicago.

Prokopowicz, P. N.; Kahn, R. E.; Firby, R. J.; and Swain, M. J. 1996. Gargoyle: Context-sensitive active vision for mobile robots. *American Association for Artificial Intelligence.*

Schank, R., and Abelson, R. 1977. *Scripts Plans Goals and Understanding.* Lawrence Erlbaum. chapter 3.

Wilensky, R. 1981. *Inside Computer Understanding.* Lawrence Erlbaum Associates, Inc. chapter 7.