ROSEY the Robot

Christopher Dac Le

Computer Science Dept/Northwestern University
1890 Maple Ave, 3rd floor
Evanston, IL 60201
dac@cs.northwestern.edu

Abstract

The **RO**bot **S**elf-**E**xplains wh**Y** (ROSEY) project is an attempt to achieve greater system transparency on behavior-based robots. By inspecting the structure of the robot's program, the ROSEY system generates causal explanations of the robot's own behavior. In this paper, we describe the initial version of ROSEY the Robot, which was demonstrated at the 2002 American Association for Artificial Intelligence (AAAI) Mobile Robot Exhibition in Edmonton, Alberta.

Introduction

The **RO**bot **S**elf-**E**xplains wh**Y** (ROSEY) project explores how a behavior-based robot can achieve greater transparency to a human user by verbalizing causal explanations of its own behavior. Explanation facilities have long been important components of expert systems for improving the confidence and understanding of a user interacting with the system. We seek a similar facility on a robot; a human observer should be able to ask what the robot is doing at any given moment and ask why the robot is behaving the way it is. During these opportunities, the robot can reveal its internal processes and states to the user.

We begin by offering a design for the explanation task on a behavior-based robot. We describe an explanation process that is based on examining the structure of the robot's program (as opposed to accessing a runtime diagnostic system). We then describe the mechanisms required for supporting explanation. Finally, we describe the initial version of ROSEY the Robot, a behavior-based robot that we demonstrated at the 2002 American Association for Artificial Intelligence (AAAI) Mobile Robot Exhibition in Edmonton, Alberta. We close with preliminary results of ROSEY the Robot's performance at the AAAI demonstration.

Our initial project goal is to build a minimalist system. At the moment, we have not performed any user evaluation studies. We presently ignore issues such as the need to tailor a robot's explanations to the target user; instead, we currently assume that the robot's target user is its programmer.

A process for explaining behavior

Two key observations guide our design approach to the explanation process. First, most behavior-based systems can be described as circuits: signals flow from sensors through wires and processing elements to the control inputs of effectors. Second, we want the robot to answer questions that concern the causes of its current locomotive state, for example "`Why are you turning?`" For the moment, we restrict these "why" questions to pertain only to the robot's current instantaneous state rather than to its past states.

Given that behavior-based systems are circuit-like, answering a "why" question can be reduced to tracing signal flow: start at the signal controlling the motors, and trace backwards through the circuit, discussing the active elements that drive it. The circuit-like structure explicitly captures the causal knowledge required for generating such an explanation.

Tracing causality

The problem however, is that the circuit tends to be complicated. Others have raised similar issues related to the automatic explanation of detailed process models (Williams 1991; Nayak 1992; Mallory, Porter, and Kuipers 1996). If every traced element is discussed, then explanations will become overly detailed and more difficult to grasp. To avoid this problem, we generate descriptions in terms of an *abstract* circuit in which whole sub-circuits are collapsed to single, idealized nodes. In the case of ROSEY, we abstractly describe the circuit in terms of *propositions* (e.g. behavioral states such as "I am turning") and *operators* (e.g. behaviors, plans).

This simplified representation retains the causal properties of the complete circuit that an explanation system can still use to answer the following questions:

- ***What behaviors must be concurrently executing in order to yield the observed actuator behavior?*** We define operators as both the internal and external behaviors that the robot may execute during any single execution cycle. Thus, we want to include behaviors that control the robot's actuators, as well as the cascade of internal behaviors that drive the activation of the current behavior.

- ***What set of conditions must hold true in order to yield the observed actuator behavior?*** We treat propositions as corresponding to internal conditions that the robot may have during any single execution cycle. Propositions include actuator-related states (e.g. "I think I am turning"), sensory-related states (e.g. "I think I see an obstacle ahead of me"), as well as the Boolean-valued states under which an operator will be activated.

Figure 1 illustrates how propositions and operators are causally related: propositions affect operators that then effect other propositions. In other words, operators can change the truth-values of propositions, whereas propositions may directly turn operators on or off. Tracing causality can be reduced to a process of walking this circuit (which we call the *abstract causal circuit*) and finding the path of activated propositions and operators. We currently assume only one active path exists through this circuit.

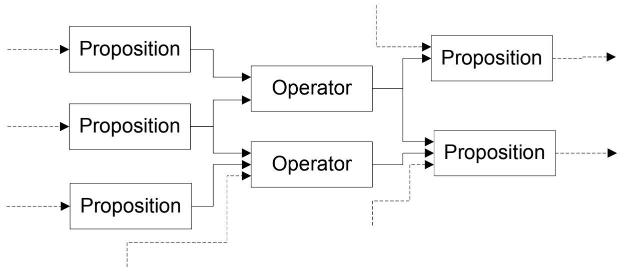

Figure 1. Abstract causal circuit. *A robot circuit is simplified to nodes consisting of either propositions or operators.*

Self-inspection mechanisms

The robot will require some runtime mechanisms to operate over the abstract causal circuit. One is the means to look up and tag an arbitrary node, given its lexical equivalent. For example, the word "turning" should correspond to a particular node in the circuit.

Another set of needed mechanisms involves the extraction of both static and dynamic information pertaining to the node. Required information includes:

- Name
- Type (proposition or operator)
- Current state
    - o If node is a proposition, is it currently true?
    - o If node is an operator, is it currently running?
- Possible causes
    - o If node is a proposition, what operators are known to cause it to be true?
    - o If node is an operator, what propositions are known to activate it?
- The active cause
    - o If node is a proposition, which operator is actually causing it to become true?
    - o If node is an operator, which proposition is actually activating it?

These requirements directly translate into a corresponding set of operations and a simple interface for retrieving this information from the self-inspection (SI) system:

- `get-name`
- `get-type`
- `get-current-state`
- `get-known-causes`
- `compute-active-cause`

A simplified explanation process

Given these mechanisms, we can now sketch a process for generating explanations. Figure 2 illustrates a schematic overview of this process. To initiate an explanation, the user asks a "why" question such as

```
"Why are you in <locomotive-state>?"
```

Here, `<locomotive-state>` is a word describing a particular locomotive state (e.g. "turning" or "moving") and corresponding to a proposition node.

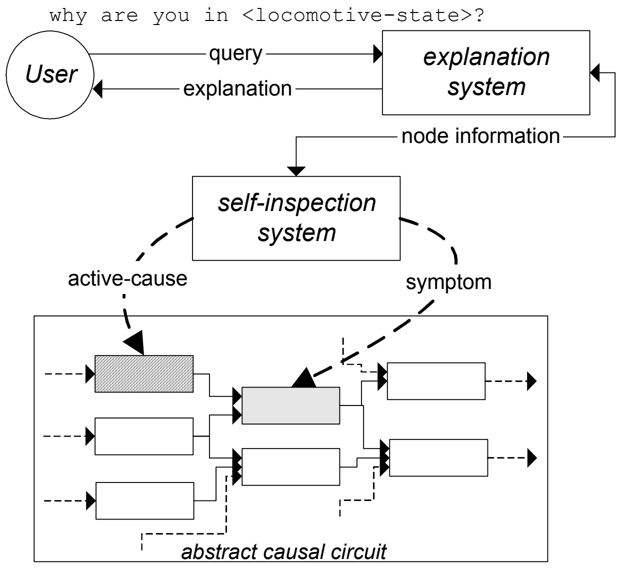

Figure 2. Schematic overview of the explanation process.

The explanation system parses the user's question. Upon recognizing a why question, the explanation system tags the word representing the locomotive state so that the SI system may look up the corresponding node; the SI system references this node as the "symptom."

At this point, the SI system tests whether the symptom node is even active, in other words, whether the symptom is even true. By testing the truth-value of the symptom, the explanation system can comment on the applicability of the user's question. If the symptom turns out to be false, then the explanation system can respond with an utterance such as

```
"I don't believe I am in
<locomotive-state>."
```

On the other hand, if the symptom is true, then the SI system proceeds to identify the active cause. First it looks up the set of possible causes of this symptom (these are identified a priori). It then iterates through each possible cause, testing whether each is true. Upon knowing the actual cause, the explanation system can then respond with an utterance such as

```
"I am in <locomotive-state> because
I am trying to do <active-cause>."
```

where <active-cause> is the name of the identified cause and in this case, corresponds to an operator node.

With this process, we assume only one node can be active among the possible causes, which means we also assume only one active path can be traced through the circuit. In the future, we hope to relax this restriction as we develop methods for constructing the abstract causal circuit.

Follow up questions

Thus far, we have described a process that generates explanations delving only one level deep in the circuit. However, the user may seek further detail in a follow up question such as "`Why?`" This follow up question becomes a contextual one, equivalent to saying

```
"Why are you doing <previous-active-
cause>?"
```

where <previous-active-cause> is the name of the previously identified cause.

In order to repeat the explanation process, the SI system re-references the previously identified active cause node as the new "symptom." The SI system now repeats the same process for testing the possible causes and identifies the new "active cause." A subsequent explanation could then be

```
"I am trying to do <previous-active-
cause> because I think <proposition>
is true."
```

where <proposition> is the name of the proposition node that is currently active and causing the <previous-active-cause>'s operator node to be true.

As the user asks additional follow up questions, the node type of each subsequent active cause will flip-flop between proposition and operator; this reflects the nature of the abstract causal circuit. Eventually, we will explore different ways we may vary the depth—and degree of detail—to which a single explanation response is generated.

AAAI Demonstration

Implementation

For the 2002 AAAI Mobile Robot Exhibition, we implemented a very preliminary version of ROSEY on an indoor 2-wheel differential-drive robot (Figure 3). ROSEY the Robot successfully recognized the following "why" queries:

- `Why are you turning?`
- `Why are you resting?`
- `Why are you reversing?`
- `Why are you moving?`

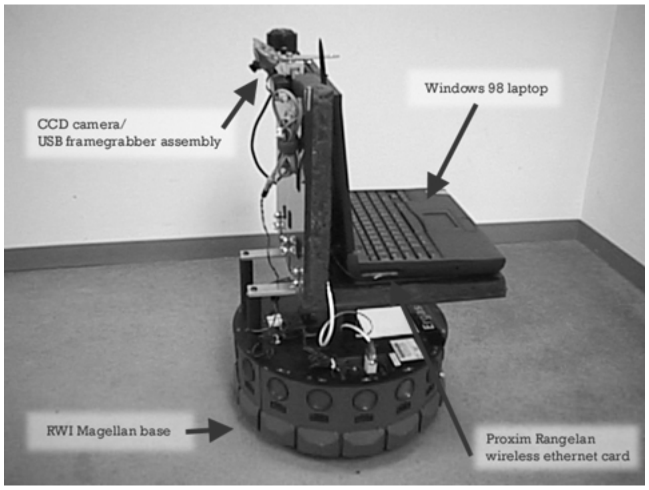

Figure 3. Indoor mobile robot platform

To leverage previous work, we built ROSEY the Robot as an extension to Cerebus (Horswill, Zubek, et. al. 2000), a mobile robot that our lab demonstrated at the 2001 International Joint Conference on Artificial Intelligence (IJCAI) and at the 2000 AAAI Mobile Robot Exhibition. We specifically borrowed the natural language parser module, which handles word recognition and which includes a remote telnet interface; the self-inspection mechanism; and the speech generation mechanism.

We gave the robot the ability to handle two possible high-level locomotive tasks during the Exhibition: wandering around without hitting anything; and following a green ball around without hitting anything. The robot could execute either task when given the command to do so; otherwise, it remained still. A user issued a task-level command (e.g. "track green ball", "wander") through the remote telnet interface.

Given that ROSEY the Robot could be executing either high-level task at any given time, we enabled the robot to also recognize questions about its current task-level goal, specifically the question

```
    "What are you doing?"
```

We wired ROSEY in such a way that when it recognized this particular question, the self-inspection system automatically tested the set of task-level goal operator nodes that existed in the abstract causal tree and proceeded to identify the active one.

To generate the actual explanation responses, we decided to forgo the use of fill-in-the-blank templates (e.g. "`I am executing <symptom>`", where we substitute a node name for `<symptom>`). Instead, every node in the robot's abstract causal circuit had a fixed—but unique—utterance stored as an additional property. When explaining an active cause node, the explanation system retrieved the canned response associated with the particular node. For instance, when asked the question "`Why are you turning?`", ROSEY might respond "`Because ROSEY is trying to get unstuck.`" ROSEY takes this response from the node representing the unwedger motor behavior, which, in this example, is the operator currently causing `turning` to be true.

Sample interaction

The following annotated transcript condenses a sample interaction that took place between a user and ROSEY.

```
  Hello, my name is ROSEY, what would
you like me to do?

  > what are you doing ?
  I am trying not to do anything.
```
[task-level goal operator]

```
  > why ?
  Because you've not told me to
perform a task yet

  > what can you do ?
  ROSEY can answer why questions,
wander, and track certain colors.

  > track green ball
  Okay, ROSEY track green.

  > what are you doing ?
  I am trying to follow the color
green.
```
[task-level goal operator]

```
  > why are you moving ?
  I am trying to find the colored
target.
```
[green ball is not in ROSEY's view]

```
  > why ?
  Because I think I've visually lost
the target.
```

[Now we drop the green ball in its view; ROSEY starts moving towards it]

```
  > why are you moving ?
  I am trying to follow the target.

  > why ?
  Because I think I see the target.
```

[Now we stand in front of ROSEY, but green ball is still in its view]

```
  > why are you resting ?
  I am trying not to hit what I think
is in front of me.

  > why ?
  Because my eyesight thinks there's
an obstacle ahead of me.
```
[ROSEY does not want to advance towards the green ball while an obstacle impedes its progress]

Future extensions

During the AAAI demonstrations, we found the typical interaction to be fairly short. This fact was no surprise, given that ROSEY's question-answering capabilities were limited to dealing primarily with locomotive state.

The four propositions—`turning`, `resting`, `reversing`, and `moving`—attempted to be descriptive of the locomotive qualities that a user may discriminate and ascribe to the robot as it is moving around in the world. We want to continue to expand this set of recognizable symptoms.

To grow this set, we plan to introduce a second class of queries that ROSEY understands. So far, we have described ROSEY handling queries that pertain to its instantaneous locomotive state. We call these queries *instantaneous queries.* However, ROSEY does not handle queries that require it to maintain history or a knowledge of its past execution cycles. We call such queries *history-based queries.* This current limitation explains why ROSEY can handle an instantaneous query like "`Why are you turning?`", while failing to recognize a query like "`Why are you oscillating?`" or "`Why did you turn?`" We acknowledge that maintaining episodic memory will be a valuable part of a robot explanation system.

In addition, we want to introduce questions beyond low-level locomotive states that encompass higher-level plans and goals. Besides being able to answer the question "`What are you doing?`", which points to a task-level goal operator, the ROSEY system should also be able to handle questions pertaining to navigational goals, such as "`Why are you going there?`" In deciding how to generate an explanation to such a question, we find one interesting problem is determining how to dynamically adjust the granularity of the explanation, such that in some contexts, we may want ROSEY to convey only its highest-level goal, while in other contexts, we may want ROSEY to convey a lower-level sub-goal in its explanation.

Related systems

ROSEY is an extension of the Cerebus project (Horswill 2001; Horswill, Zubek, et. al. 2000), a system that is able to discuss its capabilities and internal structures. This work attempts to extend this ability by enabling the robot to explain how its processes and state dictate its current behavior.

ROSEY is part of a larger category of question-and-answer robots. One of those robots includes (Torrance 1994), who describes a robot that can answer questions about navigational plans and about the spatial relationships that hold between known places. KAMRO is a two-armed mobile assembly robot that can explain its own error recovery methods to a human supervisor (Längle, Lüth, Stopp, and Herzog 1996).

ROSEY is also part of a category of work that involves explanation of physical systems. A variety of work has been done, many in the context of diagnosis, instruction, and engineering design. Self-explanatory simulators (Forbus and Falkenhainer 1990) represent one large class of software that permits the user to seek causal explanations of physical processes being modeled. One system that shares similar purposes as ROSEY is described in (Gautier and Gruber 1993). Their task is to respond to behavior-related queries about a physical device by automatically generating causal explanations from a model. They attempt to avoid overly detailed explanations by applying a few heuristics for selecting salient details. We will consider similar heuristics for constructing our circuit.

Conclusion

ROSEY is an attempt to demonstrate how a behavior-based robot can achieve greater system transparency to a human observer through explanation. We view ROSEY as representative of an emerging category of *self-explaining robots.* We are interested in increasing the reliability and cooperativeness of behavior-based robots. By revealing their internal states and processes, such robots will be better understood and accepted by human users. Our pending work will include determining appropriate evaluation criteria for measuring the effectiveness of the explanations.

References

Forbus, K. and Falkenhainer, B. (1990). Self-explanatory simulations: An integration of qualitative and quantitative knowledge. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, Boston, MA., AAAI Press/MIT Press, 1990, 380-387.

Gautier, P.O. and Gruber, T.R. (1993). Generating explanations of device behavior using compositional modeling and causal ordering. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, Washington, D.C., AAAI Press/MIT Press, 1993.

Horswill, I. (2001). Tagged behavior-based architectures: Integrating cognition with embodied activity. *IEEE Intelligent Systems.* September/October 2001: 30-38.

Horswill, I., Zubek, R., et. al. (2000). The Cerebus project. *AAAI Fall Symposium on Parallel Cognition.* Cape Cod, MA, November 2000.

Horswill, I. (1998). Grounding mundane inference in perception. *Autonomous* Robots, 5: 63-77.

Längle, T., Lüth, T.C., Stopp, E., and Herzog, G. (1996). Natural language access to intelligent robots: Explaining automatic error recovery. In *Artificial Intelligence: Methodology, Systems, and Applications* (Ed. A.M. Ramsay). Amsterdam: IOS, 259-267.

Mallory, R.S., Porter, B.W., and Kuipers, B.J. (1996). Comprehending complex behavior graphs through abstraction. In *Tenth International Workshop on Qualitative Reasoning about Physical Systems*, Fallen Leaf Lake, CA, 1996, 137-146.

Nayak, P.P. (1992). Causal approximations. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, San Jose, CA., AAAI Press/MIT Press, 1992, 703-709.

Nilsson, N. (1994). Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research*, 1: 139-158.

Torrance, M. (1994). Natural communication with robots. Master's Thesis. MIT. Cambridge, MA.

Williams, B.C. (1991). Critical abstraction: Generating simplest models for causal explanation. In *Proceedings of the Fifth International Workshop on Qualitative Reasoning about Physical Systems*, 1991.