Fido's Adventures

Marc G. Slack
The MITRE Washington AI Center, Mail Stop z401
7525 Colshire Drive, McLean, Virginia 22102-3481
slack@starbase.mitre.org
(703) 883-7738

Moments in Fido's Life

Fido lives alone in his small house against the living room wall where there is always food and a quite place to rest. Alone in the livingroom at last, Fido listens to make sure that no one is coming. Leaving the warmth of his house Fido starts off across the room. Being mostly blind and having sever neurologic problems, Fido goes across the room until he runs into a wall at which time he turns and goes back across the room. Fido in his impaired state continues this for quite a while, never having totally recovered from the near miss with the car. After a while, Fido tires of this zig-zagging process and spends time tracing around the edges of the living room. Turning a corner Fido is reminded by a sharp pain that he is supposed to remain in the living room. This causes Fido to pass by the door and continue along the edge of the room. All at once Fido feels something large under foot and decides to save it in case it was something of interest to his master. Undaunted Fido continues. By now Fido is feeling the effects of his morning cup of coffee and decides that he should return to his little house in the corner of the living room in order to relieve himself. With a whooshing sound, typical of many central vacuuming systems as well as the space station's toilet, Fido feels relieved. Refreshed Fido heads back out to play one of his favorite games, ring around the may pole. Zipping across the room Fido looks for a pole to travel around. Bumping into something Fido proceeds to move around the object and then is off in another direction looking for another pole to ring around. Fido's games are interrupted, however, when he hears the sound of people entering the room. Remembering the trouble he got into the last time that he was caught out of his house Fido, in his neurotic way, steels away back into his house for a quick snack and a nap before coming back out to play.

Elements of Successful Automation

The following is a brief enumeration some of the desirable elements of vacuuming automation

1. The system should be small to allow it to get under most furniture.
2. The system should attempt to trap anomalous items in a special tray.
3. The system should be low maintenance (e.g., self charging using an AC outlet into which its house is plugged, self emptying into a central vacuum if available or a easy access tray).
4. The system should be low profile. That is it should come out only during user defined periods and only when the room is empty.
5. Detect situations where the room is simply too messy to cleanup.
6. The user should have control. Place an off switch as well as a timed override which will shut the system off for a specified duration (e.g., not the next two hours). The user should also be allowed to specify blackout times during which the system will simply sleep in its house. Finally, the user should be able to run the system when there are people in the room in order to impress their friends. All easily available via a remote control.
7. The system should be able to handle heavy dirt and remain in a small region if there is a location with heavy dirt.
8. The system should also be able to detect situations where it cannot pickup something encountered so that it can stop trying and continue on.
9. Many other things.

Implementation

The following is a very rough pseudo implementation of Fido. The architectural premise is a task sequencing system [Firby, 1989] which manipulates (turns on and off) situated skills [Slack, 1992] in order to manipulate the world and accomplish tasks. The system's

capabilities are implemented using the following set of situated skills:

1. home-skill
2. monitor-power-skill
3. connect-to-charger-skill
4. monitor-stomach-skill
5. connect-to-central-vac-skill
6. monitor-for-humans-skill
7. clean-regions-skill
8. clean-edges-skill
9. clean-islands-skill

Teh above skills are coordinated to accomplish the overall task through the use of a task interpreter which activates the skills based on the situation at hand. The following is a pseudo code implementation defining the task of Fido's life:

```
;;
;;Life of a Vaccuuming Dog
(def-rap (life)
  (method
   (context t)
   (task-net (t1 (stay-charged))
             (t2 (dont-get-too-full))
             (t3 (clandestine))
             (t4 (clean)))))

;;;------------------------

;;
;;Enable skill for returning home
(def-rap-prim (go-home)
  (enable (:homing-skill))
  (disable :above)
  (wait-for (at-home)
            :succeed at-home))

;;;------------------------
;;
;;Wake up every 3-5 minutes and
;; see check the power
(def-rap (stay-charged)
  (succeed nil)
  (monitor-time now 3.0 5.0)
  (method
   (context t)
   (task-net
    (t1 (sense-power => ?state)
        (for t2))
    (t2 (handle-power-state ?state)))))

(def-rap-prim (sense-power => ?state)
  (enable (:monitor-power-skill))
  (disable :above)
  (wait-for (power-sensed ?state)
            :succeed power-sensed))

(def-rap (handle-power-state ?state)
  (method
   (context (= ?state low))
   (task-net (t1 (rest))))
  (method
   (context (not (= ?state low)))
   (task-net nil)))

(def-rap (rest)
  (succeed (stomach :full))
  (suspends (dont-get-too-full)
            (clean)
            (clandestine))
  (method
   (context t)
   (task-net
    (t1 (go-home) (for t2))
    (t2 (sleep)))))

(define-memory-rule (rest) :start
  (rule T  (mem-add (feeding? t))))

(define-memory-rule (rest) :end
  (rule T  (mem-add (feeding? nil))))

(def-rap-prim (sleep)
  (enable (:connect-to-charger-skill
           :monitor-power-skill))
  (disable :above)
  (wait-for (sense-power :full)
            :succeed rested))

;;;------------------------

;;
;;Wakes up every 3-5 minutes
;;  checks to see if it is
;;   time to purge
(def-rap (dont-get-too-full)
  (succeed nil)
  (monitor-time now 3.0 5.0)
  (method
   (context (feeding? nil))
   (task-net
    (t1 (sense-stomach => ?state)
        (for t2))
    (t2 (handle-stomach-state ?state)))))

(def-rap-prim (sense-stomach  => ?state)
  (enable (:monitor-stomach-skill))
  (disable :above)
  (wait-for (stomach-sensed ?state)
            :succeed stomach-sensed))

(def-rap (handle-stomach-state ?state)
  (method
```

```
    (context (= ?state full))
    (task-net (t1 (purge))))
   (method
    (context (not (= ?state full)))
    (task-net nil)))


(def-rap (purge)
  (succeed (stomach :full))
  (suspends (clean)(clandestine))
  (method
   (context t)
   (task-net
    (t1 (go-home) (for t2))
    (t2 (relief)))))

(define-memory-rule (purge) :start
  (rule T  (mem-add (purging? t))))

(define-memory-rule (purge) :end
  (rule T  (mem-add (purging? nil))))

(def-rap-prim (relief)
  (enable (:connect-to-central-vac-skill
           :monitor-stomach-skill))
  (disable :above)
  (wait-for (stomach :empty)
            :succeed hungry))

;;;------------------------

;;
;;Make sure that no humans
;; are in the room go back
;;  to the house if they are
;;  in the room
(def-rap (clandestine)
  (succeed nil)
  (monitor-time now 1.0 2.0)
  (method
   (context (feeding? nil)
            (purging? nil))
   (task-net
    (t1 (detect-humans => ?state)
        (for t2))
    (t2 (handle-humans-state ?state)))))

(def-rap-prim (detect-humans  => ?state)
  (enable (:monitor-for-humans-skill))
  (disable :above)
  (wait-for (humans ?state)
            :succeed human-monitor))

(def-rap (handle-humans-state ?state)
  (method
   (context (= ?state :present))
   (task-net (t1 (hide))))
  (method
   (context (not (= ?state :present)))
   (task-net nil)))


(def-rap (hide)
  (succeed t)
  (suspends (clean))
  (method
   (context t)
   (task-net
    (t1 (go-home) (for t2))
    (t2 (wait-till-quite)))))

(define-memory-rule (hide) :start
  (rule T  (mem-add (hiding? t))))

(define-memory-rule (hide) :end
  (rule T  (mem-add (hiding? nil))))

(def-rap-prim (wait-till-quite)
  (enable (:monitor-for-humans-skill))
  (disable :above)
  (wait-for (not (humans :present))
            :succeed human-monitor))

;;;------------------------

;;
;;Clean the room.
;;  Keep cycling between the three
;;   modes of cleaning.
;;  Note that because the rap cannot
;;   succeed it will be reinvoked
;;   at the end of t3
(def-rap (clean)
  (succeed nil)
  (method
   (context (and (feeding? nil)
                 (purging? nil)
                 (hiding? nil)))
   (task-net
    (t1 (clean-regions 2.0) (for t2))
    (t2 (clean-edges 1.0) (for t3))
    (t3 (clean-islands 1.5)))))


(def-rap-prim (clean-regions ?timeout)
  (timeout ?timeout :timeout)
  (enable (:clean-regions-skill))
  (disable :above))

(def-rap-prim (clean-edges ?timeout)
  (timeout ?timeout :timeout)
  (enable (:clean-edges-skill))
  (disable :above))

(def-rap-prim (clean-islands ?timeout)
  (timeout ?timeout :timeout)
```

```
(enable (:clean-islands-skill))
(disable :above))
```

The Demise of Acme

So in the end ... Fido tries and tries to pickup a nail which had been dropped onto the carpet. Encountering it many times Fido has had to continually give up as it is just too big to swallow. Harry, the home owner, wakes in the middle of the night and needs a drink of water only to encounter the nail in a most unpleasant fashion. Writhing in pain and sure of the fact that he will now have tetanus, Harry (A lawyer) begins his scheming to sue the Acme company for selling a negligent robot. After all Fido did "know" that there was something there that it could not pickup yet failed to inform Fred.

Practical Concerns

While is it beyond the state of the art to implement a general purpose household cleaning robot. It is possible to define niches within which simple technology can do an adequate job so as to make the technology both useful and affordable. The designer of such systems should not be limited by thinking how a person would approach the problem but rather how can simple mechanisms be utilized to fill the niche. The designer should also not be concerned overly much with the implications of the system's lack of cognizance. For example, if people leave valuable dirt on their carpet and forget to turn off Fido then it should simply be too bad. Simple solutions do not negate the integration of vision, speech recognition, or other "complex" systems. it simply places constraints on their use and packaging. As soon as a speech recognition system is cheap enough and small enough, Fido could as easily be responsive to vocal commands as those commands given by the remote control. Start simple and add complexity only as dictated by the task, environment and users. Finally, remember that the there will always be the unforeseen and lawyers.

References

R. J. Firby. *Adaptive Execution in Complex Dynamic Worlds.* PhD thesis, Yale University Department of Computer Science, January 1989. see Technical Report 672.

M. G. Slack. Sequencing formally defined reactions for robotic activity: Integrating RAPS and GAPPS. In *Proceedings of the SPIE Conference on Sensor Fusion*, November 1992.