Languages for Executive Task Level Control

Reid Simmons and David Apfelbaum

Problem: Many autonomous robot systems are developed using a three-tiered architecture that has an executive layer that integrates planning and real-time control. The executive is responsible for hierarchically decomposing high-level tasks into subtasks, dispatching and synchronizing tasks when appropriate, monitoring task execution, and handling exceptions (which may include asking the planner to replan). The resulting control strategies are typically highly conditional and non-linear, and consequently difficult to program and debug. We are developing a higher-level language, called TDL, that directly supports the types of control strategies needed in programming modern robot executives.

Impact: Software for autonomous robots is notoriously difficult to design and debug, often due to complex interactions among concurrent subtasks. The use of a language tailored to support such task-level control capabilities could significantly decrease the time needed to develop and debug robot systems, and significantly increase their reliability. Our initial results show that programs written in TDL are easier to develop, extend, and maintain than those that use conventional programming languages, like C.

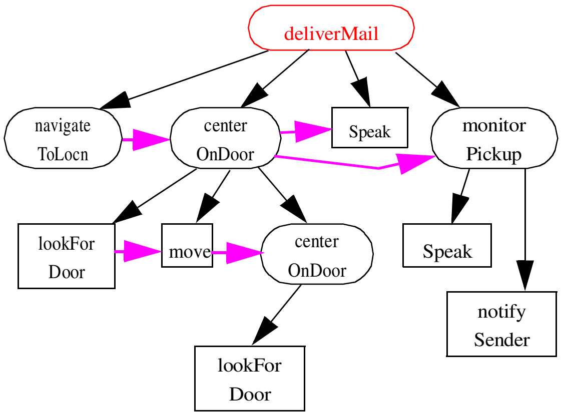
State of the Art: While attempts have been made to develop languages for robot executives (e.g., RAP, ESL, PRS, Colbert), the existing languages are either relatively inexpressive and/or difficult to integrate with existing robot programs (which are typically written in C). In contrast, TDL is very expressive in terms of the inter-task constraints that can be defined and, since it is a superset of C++, can easily integrate with existing robot control programs.

Approach: TDL (Task Description Language) [2] is based on our earlier work with the Task Control Architecture [1]. TDL is a superset of C++ that contains explicit syntax for defining tasks, decomposing tasks into subtasks, constraining and synchronizing the execution of tasks, monitoring the environment, and handling exceptional conditions. A Java-based compiler translates TDL into pure C++ code that includes calls to a Task Control Management (TCM) library. The TCM library maintains hierarchal structures of tasks, called *task trees* (Figure 1) which encode parent/child relationships (vertical arrows in the figure) and inter-task constraints (horizontal arrows). Figure 1 also shows a TDL program for a mail delivery task that is used to generate part of the task tree illustrated.

TDL enables programmers to first encode the nominal part of a task (what should happen if all goes according to plan) and then incrementally add monitors and exception handlers to deal with contingencies. We have found this to be both an effective means of programming robots and one that leads to very reliable systems [1].

TDL is currently being applied for delivery tasks, for coordinating the deployment of multiple small robots, and for mobile construction robotics. It is also planned to be used for a multi-arm workcell at Stanford University and an unmanned airship at CTI, Brazil.

Future Work: Extensions to TDL include adding the ability to deal with resources (joint work with Rich Washington and Greg Dorais at NASA Ames) and creating a distributed version of TDL to better support multi-robot coordination. We are also working on formal models of TDL in order to verify the correctness of task-level control programs using symbolic model-checking (see research summary on "Formal Verification of Autonomous Systems").




```
Goal deliverMail (int room)
      Exception Handler noDelivery
{
 double x, y;
 getRoomCoordinates(room, &x, &y);
 spawn navigateToLocn(x, y);
 spawn centerOnDoor(x, y)
      with sequential execution previous,
           terminate in 0:0:30.0;
 spawn speak("Xavier here with your mail")
      with sequential execution centerOnDoor,
           terminate at monitorPickup completed
 spawn monitorPickup()
      with sequential execution centerOnDoor;
}
```

Figure 1: Partial Task Tree and TDL Program for a Mail Delivery Task

Selected References:

[1] Reid Simmons. Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1):34–43, February 1994.

[2] Reid Simmons and David Apfelbaum. A task description language for robot control. In *Proc. International Conference on Intelligent Robots and Systems*, Vancouver Canada, October 1998.