Model-based Reactive Programming of Cooperative Vehicles for Mars Exploration

Brian C. Williams, Phil Kim, Michael Hofbaur, Jonathan How,
Jon Kennell, Jason Loy, Robert Ragno, John Stedl and Aisha Walcott
MIT Space Systems and Artifical Intelligence Laboratories
77 Massachusetts Ave., Cambridge, MA 02139 USA
{williams,philkim,hofbaur,jhow,jonk,jloy,rjr,stedl,aisha}@ mit.edu

Keywords: cooperating autonomous agents, planning and execution, embedded programming.

Extended Abstract

In the future networks of unmanned vehicles will act together to robustly achieve elaborate missions within uncertain environments. This network may be a distributed satellite system forming an interferometer, or may be a heterogenous set of rovers and blimps exploring Mars. The creation of robotic networks cannot be supported by the current programming practice alone. Recent mission failures, such as the Mars Climate Orbiter and Polar Landers, highlight the challenge of creating highly capable vehicles within realistic budget limits.

In this paper we advocate the creation of *embedded, model-based programming languages* that support the ability to specify global strategies for multi-vehicle coordination.

First, we argue that the programmer should retain control for the overall success of a mission, by programming game plans and contingencies that in the programmer's experience will ensure a high degree of success.

Second, we argue that model-based programming languages should focus on elevating the programmer's thinking, by automating the process of reasoning about low-level system interactions. Many recent space mission failures, such as Mars Climate Orbiter and Mars Polar Lander, can be isolated to difficulties in reasoning through low-level system interactions. The interpreter or compiler of a model-based program reasons through these interactions using composable models of the system being controlled. We are developing a language, called the *Reactive Model-Based Programming Language (RMPL)*, that supports four types of reasoning about system interactions: reasoning about contingencies, scheduling, inferring a system's hidden state and controlling that state. This paper develops RMPL in the context of contingencies and scheduling.

Third, execution of these programs is a form of mission-level planning that searches through the options for the optimal strategy that operates within the time constraints. Vehicle movement is central to these missions, hence to achieve global optimality we unify mission-level planning with global path planning algorithms. In particular we build upon the rapidly exploring random tree (RRT) algorithm. In addition, these vehicles may operate in highly dynamic situations or situations where maneuvering is extremely tight.

The paper begins by introducing a subset of RMPL that includes constructs from traditional reactive programming plus constructs for specifying contingencies and scheduling constraints. We describe how *Kirk*, an RMPL-based planner/executive, compiles RMPL programs into *temporal plan networks (TPN)*, which compactly represent all possible threads of execution of an RMPL program, and all resource constraints and conflicts between concurrent activities. We present Kirk's online planning algorithm for RMPL that "looks" by using network search algorithms to find threads of execution through the TPN that are temporally consistent and unify this temporal planning capability with randomized, kino-dynamic path planning algorithms. The result is a partially ordered temporal plan. Kirk then "leaps" by executing the plan using plan execution methods.

To develop the model-based programming paradigm in the context of Mars exploration, we have developed a combination of simulation and ground-based testbeds, including a collection of four RWI ATRV rovers and a set of formation flying spacecraft, called Spheres, to be flown on space station.