N93-11929

INTEGRATION OF TASK LEVEL PLANNING AND DIAGNOSIS FOR AN INTELLIGENT ROBOT

Amy W. Chan
UFA, Inc.
335 Boylston street,
Newton, MA 02159

Work performed under Contract# NSA8-38420

Abstract

This paper is to diagnose a satellite floating through space with a telerobot attached performing maintenance or replacement tasks. This research included three objectives. The first objective was to generate intelligent path planning for a robot to move around a satellite. The second objective was to diagnose possible faulty scenarios in the satellite. The third objective included two tasks. The first task was to combine intelligent path planning with diagnosis. The second task was to build an interface between the combined intelligent system with Robosim.

The ability of a robot to deal with unexpected scenarios is particularly important in space since the situation could be different from time to time so that the telerobot must be capable of detecting that the situation has changed and the necessity may exist to alter its behavior based on the new situation.

The feature of allowing human-in-the-loop is also very important in space. In some extreme cases, the situation is beyond the capability of a robot so our research project allows the human to override the decision of a robot.

Introduction

This research project is the Phase II SBIR project, "Integration of Task Level Planning and Diagnosis for an Intelligent Robot". This research project is an intelligent system developed for multiple processes running in real time. The intelligent system includes three parts: AI Path Planner, Diagnosis and the interface with Robosim *1. Refer to Figure 1-1 for the relationship.

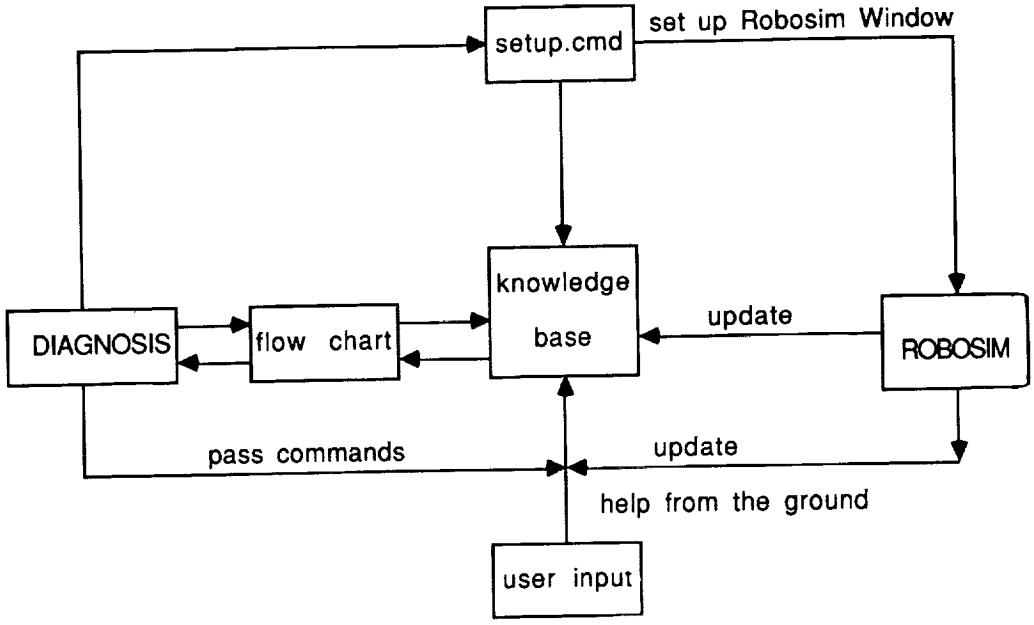

Figure 1-1

In addition to Robosim window, there are two windows in the system: the Satellite Window and the Knowledge Window. A simplified generic satellite model floating through space with a telerobot attached is simulated in the Satellite Window.

There are two knowledge bases in the system: the Satellite Knowledge Base and the Diagnosis Knowledge Base. Both are constantly updated during the simulation and displayed in the Knowledge Window. The Knowledge Window also displays simplified physical architecture and logical diagnosis of satellite subsystems.

The intelligent system provides user friendly environment. It is written in C and runs on a Silicon Graphics Iris workstation with Unix operating system.

AI Path Planner

The AI Path Planner includes three parts: a scenario definition file - "model.dat" and scenario parser; model builder; and Path Planner. Refer to Figure 1-2 for the relationship.

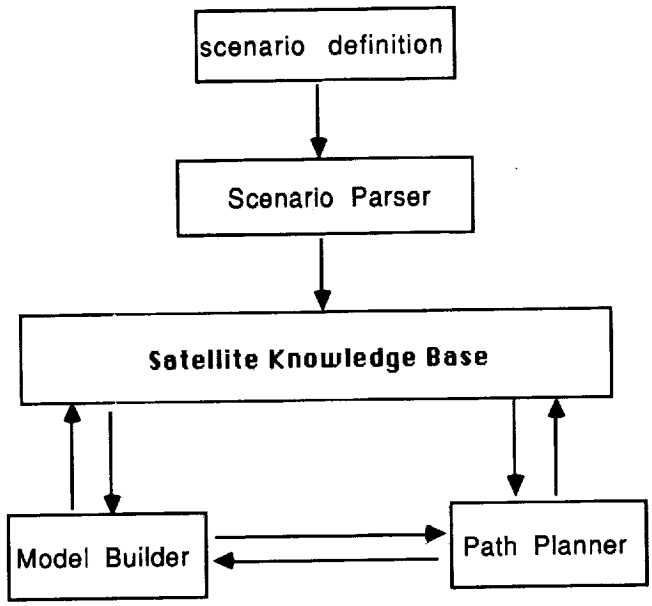

Figure 1-2

Scenario Definition File and Scenario Parser

A scenario definition file called "model.dat" was created to define the geometric configuration of a simplified generic satellite model. The scenario file defines the brief title of each satellite module and assigns each subsystem, panel, antenna and the robot to a separate module. The scenario file also assigns a list of parts to some modules and defines all the possible moves of each module.

There are two parsers in the system: the Scenario Parser and Setup Parser. The Scenario Parser was created to read and store the scenario definition file to the Satellite Knowledge Base. The Setup Parser was created to read a setup file for Robosim.

The Scenario Parser also calculates the center of each module, calculates the distances of each module to all its possible moves, sets up the constraint condition and stores them into the Satellite Knowledge Base.

Model Builder

The model of a simplified generic satellite was constructed in the form of three cylinder-like structures piled up together. Each cylinder-like structure has six sides. The total modules of the satellite model is the eighteen sides plus the top and bottom of the model. The Model Builder displays the model with a pair of solar panels, an antenna and the robot to the Satellite Window.

Path Planner

The Path Planner determines the best path for the robot to move from the module which the robot is standing at to any other target modules. Once a move command is given to the robot, the Path Planner will search dynamically all the adjacent paths one by one and generate the best candidate path set. Each path set includes one or more module elements. The determination of each candidate path set is a complicated task that involves calculating the total distance of the path set, avoiding obstruction, paying extra costs and penalties.

The search algorithm is First Depth plus heuristic and optimal. To be more specific, in order to get the shortest move, when a target module is given, the Path Planner calculates and compares all the distances from the current module to all the adjacent modules one at the time and get the shortest one.

Once the shortest move is found, the Path Planner pushes the target module and its total distance into a result stack. At each dead end, the Path Planner pops the last node off the stack and tries a new path from that point (backtracking).

After the candidate path is determined, the Path Planner pops up the stack element by the rule of "first-in-last-out" and displays them into the Satellite Knowledge Base Display in the Knowledge Window. To simulate the intelligence of the system, the following features had been added to the Path Planner.

- If the robot moves from or to the top or bottom module, a certain "cost" will be charged. For example, if the robot is located at the top module - 'S' and going to move to any adjacent modules, or the robot is standing at any adjacent modules and moving to the top module - 'S', the Path Planner adds an extra cost to the total distance.
- There are a list of parts assigned to some modules. After a move path set is generated, the robot moves one by one according to the path set. While the robot moves to each module element, the Path Planner checks whether a part is assigned to the current module. If there is a part assigned to the current module, the Path Planner will check whether the robot is carrying that part. If the robot is carrying that part, it will skip that part. If the robot is not carrying that part, it will pick up that part.
- If the robot passes some modules with constraints in the center such as panels or antenna, it will be fined with a certain penalty. For example, if the robot passes a module with the panel in the center, the Path Planner will add an extra 15 second penalty to the total penalty. If the robot passes a module with the antenna in the center, the Path Planner will add an extra 20 second penalty to the total penalty.

Unexpected Events

To simulate an unexpected event, an obstruction storm was created. The obstruction storm includes three obstructions. Each obstruction has different sizes. The three obstruction were flying in three different directions in three different speeds in the simulation. The system can handle the following unexpected events:

- An obstruction can be selected to fall on the surface of the model. If the obstruction is falling on the module at which the robot is standing, the robot will detect it and move away to avoid being hit.

- If a given target for the robot to move has an obstruction, the system will detect it , discard the move command and pop up an error message to the Satellite Knowledge Base in the Knowledge Window.

- The Path Planner will detect the module with an obstruction and skip it from the path planning.

Human-in-the-loop

The system allows the human being to take over the control of the robot during the simulation. The system allows the human being to input his path set and also allows the human being to review or change the path set. After the path set is determined, the robot moves one by one according to the path set. The following error checking were performed:

- If one module element of the human input path set is not alphabetic, the system discards it from the path. The new path set remains valid.

- If one module element of the human input path set cannot be found in the Satellite Knowledge Base, the system discards the whole path set and displays an error message to the Satellite Knowledge Base Display in the Knowledge Window.

- If one module element of the human input path set has an obstruction, the system discards the whole path set and displays an error message to the Satellite Knowledge Base Display in the Knowledge Window.

Diagnosis + Path Planner + Robosim

We have combined the AI Path Planner with the Diagnosis and built the interface between the new intelligent system with the Robosim. We pass the Robosim commands through the Unix feature - pipe to the Robosim.

To setup the Robosim, users have to define the color and view point in a file called *"setup.cmd"*. Users can also create, translate or rotate objects in this file. The Robosim initializes the robosim screen according this file.

The Diagnosis includes space experiment lab and four subsystem diagnosis. There is an option in the main menu for each subsystem. Once a particular subsystem is selected, the Diagnosis checks the Satellite Knowledge Base and finds out which module is assigned. Once the module is found, the Diagnosis calls the Path Planner to generate the best path and the robot will move one by one according to the path set. The Satellite Knowledge Base is updated simultaneously.

At the same time, a simplified physical architecture chart of that subsystem is displayed on the left side of the Knowledge Base Window and the complete diagnosis flow is displayed on the right side of the window.

The Diagnosis passes the predefined command files to Robosim to set up the Robosim window simultaneously. Each part of the physical architecture is drawn as a rectangle. As the Robosim does not allow displaying words, the labelling of the Robosim is displayed on the left Knowledge Window.

To cover all the failure scenarios, a random number is generated to give a positive or negative answer at each determination point during the diagnosis of each subsystem. After a complete failure scenario is diagnosed, the system displays it on the right Knowledge Window.

Once a faulty part is detected during the diagnosis, the Diagnosis sends a command file to Robosim to command the robot in the Robosim Window to identify the faulty part. If there are more than one faulty parts found, the robot will identify them one by one. In

some cases, the Diagnosis is not able to find the faulty part, the robot asks for help by displaying a big "HELP" in the Robosim Window.

Conclusions

This scientific AI project includes multiple processes and each process is displayed in a separate window. This artificial program can integrate with a simulator like Robosim.

The breadth-first search plus heuristic and backtracking strategy enables this project to provide a shortest path set between any two modules in the satellite.

This project handles unexpected events and allows human to override the decision of an intelligent robot during the simulation.

Acknowledgments

I appreciate all of the help that made this project possible and a success. Many thanks go to Ms. Cynthia A. Coker, the C.O.R of this research project. I would like to thank Kiet Huang, our previous project manager, for his accomplishments during Phase I and his many inputs and ideas at the beginning of Phase II.

I also appreciate Csaba A. Biegl Ph.D. from Vanderbilt University for his valuable explanation of the ROBOSIM and the integration of our system with Robosim. Finally, I would like to thank Dr. Harold L. Alexander from M.I.T. for his valuable consultation of the satellite subsystem diagnosis.

Reference

1. The initial Robosim was created by Ken Fernandez, Ph.D. of MSFC, NASA. The Robosim which this research project integrated with is an intelligent agent of the Robosim developed by Csaba A. Biegl, Ph. D. from Vanderbilt University.