Autonomous Sensor-Based Dual-Arm Satellite Grappling

Brian Wilcox Kam Tso Todd Litwin Samad Hayati Bruce Bon

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

Abstract

The NASA Telerobotic Research Project is exploring the feasibility of using robots in space for on-orbit assembly, maintenance, and repair operations. Dual-arm satellite grappling is one of its more challenging tasks.

The task involves the integration of technologies developed in the Sensing and Perception (S&P) Subsystem for object acquisition and tracking, and the Manipulator Control and Mechanization (MCM) Subsystem for dual-arm control. S&P acquires and tracks the position, orientation, velocity, and angular velocity of a slowly spinning satellite, and sends tracking data to the MCM subsystem. MCM grapples the satellite and brings it to rest, controlling the arms so that no excessive forces or torques are exerted on the satellite or arms.

The demonstration setup includes a 350-pound satellite mockup which can spin freely on a gimbal for several minutes, closely simulating the dynamics of a real satellite. The satellite mockup is fitted with a panel under which may be mounted various elements such as line replacement modules and electrical connectors that will be used to demonstrate servicing tasks once the satellite is docked.

The subsystems are housed in three MicroVAX II microcomputers. The hardware of the S&P Subsystem includes CCD cameras, video digitizers, frame buffers, IMFEX (a custom pipelined video processor), a time-code generator with millisecond precision, and a MicroVAX II computer. Its software is written in Pascal and is based on a locally written vision software library. The hardware of the MCM Subsystem includes PUMA 560 robot arms, Lord force/torque sensors, two MicroVAX II computers, and Unimation pneumatic parallel grippers. Its software is written in C, and is based on a robot language called RCCL.

This paper describes the two subsystems and provides test results on the grappling of the satellite mockup with rotational rates of up to 2 rpm.

1 Introduction

NASA is exploring the possibilities of using autonomous systems in place of astronauts for dangerous or time-consuming operations, such as extra-vehicular activity. The NASA Telerobot Research Project is looking at the use of autonomous systems for performing on-orbit assembly, maintenance, and repair operations. All of these operations present challenges to the field of robotics.

This paper is concerned with one challenge associated with the repair of artificial earth-orbiting satellites. That challenge centers around the fact that most such satellites are spin-stabilized. In

Figure 1: Testbed Setup

order to repair such a satellite, it must first be de-spun. An investigation was made into the use of a dual-arm robot with visual assist to grapple and de-spin a rotating satellite mockup.

The Testbed of the NASA Telerobot Project, housed at the Jet Propulsion Laboratory, is composed of several subsystems. Two of these subsystems, the Sensing and Perception (S&P) Subsystem and the Manipulators and Control Mechanization (MCM) Subsystem, were directly involved in the effort to grapple a spinning satellite. S&P visually acquired and tracked the rotating satellite and transmitted the satellite's state information to MCM in real time. MCM, using the data from S&P, directed the arms to grab the satellite and to stop its motion as shown in Figure 1.

Below we will discuss these two subsystems and provide the details of the satellite grappling technique.

2 Testbed Setup

The Testbed of the NASA Telerobot Project is a facility which is composed of many parts. It is divided into three main sections: the computer facility, the operator control station, and the test area.

In the computer facility there are various computer systems to support the subsystems. Included among them are several MicroVAX II microcomputers, Sun workstations, and a Symbolics Lisp machine. There are also other specialized processors used at low levels of support.

The operator control station provides a place for the operators during Testbed use. It has working surfaces, computer terminals, video monitors, and joysticks. It is situated so as to give a good view of the test area.

The test area contains everything else. It houses the two manipulation arms, plus a third arm used for holding movable video cameras (not used for satellite grappling), and all of the video cameras. The satellite mockup, as well as other targets for telerobotic research, are in this area. In addition, there is a calibration fixture used to calibrate the cameras in S&P and the arms in MCM to a common coordinate system.

Satellite Mockup

The satellite mockup is a 350-pound model of a generic artificial satellite. It is suspended from a counterweight by a long cable which is connected — through a fairly wide opening in the top of the satellite — to a gimbal near its center of mass. The method of suspension allows the satellite to move within a small area in a manner similar to its free-space counterparts.

The sides of the mockup consist of real solar panels, framed in the gold-foil insulating material typical of real satellites. This gives S&P a realistic visual target, complete with visual clutter and specular reflections.

On one side of the mockup there is a removable task panel, under which may be mounted various elements such as line replacement modules and electrical connectors. These can be used to demonstrate servicing tasks once the satellite is grappled and docked.

S&P Hardware

The S&P Subsystem hardware consists of a DEC MicroVAX II microcomputer and several other pieces of special equipment. There is a time-code generator, shared with MCM, which allows time tagging the data to millisecond precision. There are five video cameras. Three of them are mounted to the rear of the test area, yielding good overall views of the work region. The other two are mounted on a robot arm and are used for close-up views; they were not used in the work described here.

The cameras feed their signals into video digitizers, which in turn feed a custom video pipelined processor. The processor is called IMFEX, the Image Feature Extractor, and it uses a thresholded modified-Sobel operator to find high-contrast edges in video images. The output of IMFEX is fed into video buffers for storage, access, and manipulation by the MicroVAX II. The video buffers have internal D/A converters to allow viewing of their contents, which may be a captured frame, graphics generated by the MicroVAX II, or a combination of the two. A 16-by-16 video crossbar

switch is used to route all of the analog video signals between the components mentioned above, and over to the video monitors in the operator control station for display.

S&P Software

The S&P software is composed of two major sections, both written in VAX Pascal. The first section is a 30,000-line general-purpose vision software support library. It contains the software used to control and access the various hardware components of S&P, as well as routines to perform all manner of support services for machine vision. Included in this library are the routines used to perform the functions of acquisition and tracking described below, to perform camera calibration, and to deal with object models. The second major section of the software is a 20,000-line software package which is the S&P-specific application code. It is decomposed into five separate tasks, all running concurrently.

MCM Hardware

The MCM Subsystem hardware consists of two PUMA 560 robot arms. Each arm is equipped with a Unimation pneumatic parallel gripper which has been fitted with simple parallel fingers holding a special grappling tool. Each tool has a flexible handle and a Velcro pad at the end in order to hold to the satellite mockup, which has the two complementary Velcro pads attached on either side of the task panel. Each arm is also equipped with a commercial (Lord Corporation) wrist force/torque sensor. The wrist force/torque sensors are used to read the forces and torques sensed in all six dimensions. The computing hardware consists of two DEC MicroVAX II computers. Each one is interfaced to the LSI 11/73 microprocessors of the Unimate controller via two DRV11 parallel cards. The Unimate controller reads the LORD force/torque sensor data through another DRV11 parallel card. The S&P object state data is obtained through a 9600 baud RS-232 serial line connected between the S&P MicroVAX to the MCM MicroVAXs. Figure 2 shows a schematic drawing of the hardware.

MCM Software

The MCM MicroVAXs run under a modified Berkeley BSD 4.3 Unix operating system. The main robot language is RCCL (Robot Control C Library) which was developed originally at Purdue University by Vincent Hayward [3] based on Richard Paul's robotics textbook [4] and later enhanced by John Lloyd [5] and ported to a MicroVAX II [6]. RCCL is a collection of C functions which provide easy and general robot programming. The software can be partitioned into two main parts: the planning level and the real-time control level.

The planning level consists of functions that allow the programmer to specify desired coordinate frames for the robot end-point target position. The programmer must specify several frames, such as where the robot is in the world frame and in the tool frame. He/she must also specify the velocity, and whether the motion should be carried out in Cartesian or joint-interpolated modes. The relationships between the various frames are entered in the code exactly as one would write them in mathematical notations. A function called Makeposition interprets the code and sets up

Figure 2: Schematic drawing of the hardware

appropriate matrix equations. Its general form is:

$$\text{p} \;=\; \text{Makeposition}\ (\text{“P”}, Z, \ldots, T6, \ldots, R, EQ, A, B, \ldots, U, \text{TOOL}, R)$$

This simply sets up a matrix equation

$$(Z \ldots) T6 (\ldots R) = AB \ldots U \tag{1}$$

The main objective is to solve this equation for T6 as

$$T6 = (Z \ldots)^{-1} (AB \ldots U) (\ldots R)^{-1} \tag{2}$$

where $T6$ represents a homogeneous transformation relating the link 6 frame in the 0th (or shoulder) frame. The planning level runs in a normal time-shared manner. A program can develop many motions and place them in a queue for the control level to execute sequentially. An important feature of RCCL is that it allows one to modify the matrices in equation (1) in the control level.

These modifications can be triggered either by the planning level or by external sources such as from S&P object state data or force/torque data. This feature has been used extensively in our work to implement both the tracking and force/compliant control during grappling the satellite mockup.

The control level which is called RCI (Robot Control Interface) is a general robot programming environment. One can write his/her own robot control programs and create custom trajectories to run the robots. In the normal RCCL operations this level receives the motions from the motion queue and uses a trajectory generator to interpolate between the specified via points. Finally, inverse kinematics is performed to obtain the joint angle set points to be sent to the joint microprocessors via the LSI 11/73 computer.

The system works by a hardware clock (located in the Unimation controller) which periodically sends an interrupt to a communication program in the LSI 11/73. At every interrupt, this program gathers information from the arm — which includes the joint angles and other sensors that have been interfaced to the LSI 11/73 — and makes them available to the MicroVAX in shared memory; it signals the MicroVAX with a hardware interrupt. The MicroVAX reads these data and immediately sends the joint angles that have been computed in the previous cycle back to the LSI 11/73 for execution. Note that in this implementation the VAL language is completely bypassed. The hardware of the Unimate controller remains intact, however, and one can switch between the VAL language and RCCL without any hardware modifications.

3 S&P Acquisition and Tracking Algorithms

The task of watching a moving object is broken down into two stages. the first of which is called acquisition. This is the stage wherein the object of interest is first localized within the views of the cameras, and an initial computation is made as to its location and movement within 3-space. It is computationally intensive, and cannot perform quickly enough on currently available computers to keep up with a moving object in real time.

The second stage is call tracking. Tracking is more computationally efficient than acquisition, and is used to follow the object as it moves, updating the state information that was initially provided by acquisition.

Both stages compute the following time-tagged state information in three dimensions: position, translational velocity, orientation, angular velocity, and a covariance matrix of these values.

Acquisition

A fully autonomous acquisition algorithm is currently under development at the Jet Propulsion Laboratory and was not tested in the grappling experiment. A moment's reflection, however, will reveal that it is not possible to track an object without first acquiring it in some fashion. In order to satisfy this need, a "quick-and-dirty" operator-assisted version called *hand acquisition* was designed for the current work.

In hand acquisition, an *a priori* position is used as a starting point. Using this position, S&P displays a wire-frame projection of the satellite's object model in the display, superimposed on the raw video. While the satellite was held still, the operator uses the joysticks to move the wire-frame overlay — and thus the state of the object model — until it roughly overlaps the satellite mockup

in all camera views. Once complete, the operator signals that tracking may start. Hand acquisition of a moving satellite was attempted with mixed results.

Tracking

The tracking algorithm was designed by Donald B. Gennery. Detailed descriptions of the algorithm are given elsewhere [1,2]. The tracker performs its operations in five major phases. In the first phase it acquires a frame of video and notes the time tag associated with the data. In the second phase the old object state is propagated forward to the time of the new data. In the third phase a projection of the propagated object state's edges is made into the view of the camera which took the new data. In the fourth stage measurements are made of the discrepancies between the locations of edge points in the projected edges and the locations of edge points in the data. In the fifth and final stage the projected object state is adjusted by using a least-squares technique with respect to the measurements taken in the fourth stage. Uncertainties are propagated and determine the effect that any given data set has on the current object state.

These five stages constitute one tracking cycle. Between cycles, the updated state information is sent to MCM. Then the tracker selects the next camera and performs another tracking cycle. It continues in this fashion until told to stop or until it loses track. If track is lost, S&P cycles back into acquisition and repeats the entire process.

4 MCM Tracking and Grappling Algorithms

The two robots are driven independently by two MicroVAXs. They run the same copy of the software except each has its own coordinate frames because of the different locations of the robots and because they grapple different points on the satellite mockup. Two machines are used because the computing power of one MicroVAX is not adequate to control two robots during the tracking phase. The two robots are coordinated because they are basically driven from the same source of data — the S&P object states of the satellite mockup.

The MCM software receives the satellite object state at a rate of about two times per second. The object state consists of a time-stamp, position vector and orientation quaternion of the centroid of the satellite mockup, the translational and angular velocities, and their covariances.

A complete cycle of satellite grappling is comprised of four phases: *approach*, *tracking*, *grappling*, and *docking*.

In the approach phase, the MCM software monitors the orientation and angular velocity of the satellite mockup. It deploys the robots to the approach positions when the satellite mockup is spinning with a rate at or below two rpm. The approach positions are chosen so that the robots have the maximum work space for tracking and grappling. The approach position of the left robot is described by the follow equation:

$$ {}^{w}_{0}T_{L} \ {}^{0}_{6}T_{L} \ {}^{6}_{e}T_{L} = {}^{w}_{a}T_{L} \tag{3} $$

where ${}^{w}_{0}T_{L}$ is a transformation describing the 0th frame, ${}^{6}_{e}T_{L}$ is the tool frame, and ${}^{w}_{a}T_{L}$ is the approach frame of the left robot as shown in Figure 3.

The robots wait in the approach positions until the mockup has rotated such that the pads are facing the tools with a designed distance of 100mm. At that moment the robots start tracking the

Figure 3: Coordinate Frame Assignments

satellite mockup driven by the S&P object state data. At the same time the distances between the tools and the pads are gradually reduced until they contact each other. The following kinematic equation is used to specify the motion in the tracking phase:

$$ {}^{w}_{0}T_L \ {}^{0}_{6}T_L \ {}^{6}_{e}T_L = {}^{w}_{s}T \ {}^{s}_{p}T_L \tag{4} $$

where ${}^{w}_{s}T$ is a transformation describing the centroid of the satellite mockup frame, and ${}^{s}_{p}T_L$ is the transformation from the satellite mockup centroid to the left pad. The distance between the tools and the pads, initially 100mm, is faked in equation (4) by making the tool 100mm longer than its physical length. This distance is reduced during tracking until contact.

In RCCL, one can generate trajectories by using the trajectory generator or by an external means. The latter is made possible since one can modify any of the transformations except $T6$ in equation (1) in real time. The satellite tracking uses the latter strategy since it would be too time-consuming if planning is done each time MCM receives an updated object state.

Hence tracking is done in the control level which drives the robots in the following way: Every time an object state is received, the current frame of the satellite ${}^{w}_{s}T$ is predicted according to the received data, and the frame of the tool ${}^{w}_{e}T$ is computed from the joint angles of the robot. The

difference between ${}^{w}_{p}T_L$ and ${}^{e}_{w}T_L$ is computed from

$$\Delta_L = {}^{e}_{p}T_L = {}^{e}_{w}T_L \;\; {}^{w}_{p}T_L \tag{5}$$

where ${}^{w}_{p}T = {}^{w}_{s}T \times {}^{s}_{p}T$. In order to track the satellite, the robots are required to have moved this Δ by the time the next state update is received. This means that, in every sampling period, the robots are moved by $\delta = \Delta \times \frac{t}{T}$, where t is the robot control sampling period, and T is the inter-arrival period of the object state.

The tools approach the grappling pads until a contact is initiated. This is sensed by the force/torque sensors and after the contact the motion is changed from vision-based servoing to force servoing. The following kinematic equation is used to specify the motion in the grappling phase:

$${}^{w}_{0}T_L \;\; {}^{0}_{6}T_L \;\; {}^{6}_{e}T_L = {}^{w}_{s}T \;\; {}^{s}_{p}T_L \;\; \text{COMPLY} \tag{6}$$

where COMPLY is a "small" time-varying transformation and has the following form

$$\text{COMPLY} = \begin{bmatrix} 1 & -\delta\theta_z & \delta\theta_y & \delta x \\ \delta\theta_z & 1 & -\delta\theta_x & \delta y \\ -\delta\theta_y & \delta\theta_x & 1 & \delta z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

Since the COMPLY transformation is placed after the ${}^{s}_{p}T_L$, it will modify the ideal trajectory by a small amount each sample time. Because of integral force control, the COMPLY transformation will be modified based on the force/torque sensor readings and will keep its value even after the forces have been nulled. Since the satellite mockup cannot be stopped instantaneously once it is grappled, the software decelerates the mockup according to a trajectory which is generated based on the initial velocity at the moment of contact.

Once the satellite mockup is stopped, it is pulled to the docking fixture. Active force control is used to nullify the force built up due the dual-arm coordinated motion. If grappling does not occur — detected by the lack of contact between the tools and pads — the whole cycle is repeated by letting the satellite mockup spin one more time.

5 Conclusions and Future Improvements

We have successfully grappled the satellite mockup with rotational rates of up to 2 rpm. With higher speed, due to the communication delay and control inaccuracies, the robots start to miss the pads.

In the present MCM implementation, the computation is performed with two MicroVAX computers, which limits its control rate to once every 28 msec. In the near future, we plan to port our software to a Sun 4/260 computer which will increase the control rate to 200 Hz. This increase will improve the force control capability and hence reduce the build-up of forces at the moment of contact and subsequent grappling. The improved version of RCCL [7,8] can coordinate trajectories for two robots. As such, more precise coordination both at the planning and control levels can be achieved.

6 Acknowledgements

The research described in this document was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

[1] D. B. Gennery, "Tracking Known Three-Dimensional Objects," *Proceedings of the AAAI Second National Conference on Artificial Intelligence*, Pittsburgh PA, August 1982, pp. 13-17.

[2] B. Wilcox, D. B. Gennery, B. Bon, and T. Litwin, "Real-Time Model-Based Vision System for Object Acquisition and Tracking," *Proceedings of the SPIE International Conference*, Los Angeles CA, January 1987.

[3] V. Hayward, and R. Paul, "Robot Manipulator Control Under Unix RCCL," *The International Journal of Robotics Research*, Vol. 5, No. 4, Winter 1987, pp. 94-111.

[4] R. Paul, *Robot Manipulators: Mathematics, Programming, and Control*, MIT Press, 1981.

[5] J. Lloyd, "Implementation of a Robot Control Development Environment," M.S. Thesis, Department of Electrical Engineering, McGill University, Montréal, Québec, 1985.

[6] J. Lee, S. Hayati, *et al.*, "Implementation of RCCL, a Robot Control C Library, on a MicroVAX II," *Proceedings of the SPIE Conference on Advances in Intelligent Robotics Systems*, Vol. 726, October 1986, Cambridge MA, pp 26-31.

[7] J. Lloyd, M. Parker, and R. McClain, "Extending the RCCL Programming Environment to Multiple Robots and Processors," *Proceedings of the IEEE International Conference on Robotics and Automation*, Philadelphia PA, April 1988, pp. 465-469.

[8] S. Hayati, T. Lee, K. Tso, P. Backes, and E. Kan, "The JPL Telerobot Manipulator Control and Mechanization Subsystem (MCM)," *Proceedings of the NASA Conference on Space Telerobotics*, Pasadena CA, January 1989.