Proceedings of the 2006 IEEE International Conference on Robotics and Automation
Orlando, Florida - May 2006

Autonomous Capture of a Tumbling Satellite

Guy Rouleau, Ioannis Rekleitis, Régent L'Archevêque, Eric Martin, Kourosh Parsa, and Erick Dupuis
Space Technologies, Canadian Space Agency
6767 route de l'Aéroport, Longueuil (St-Hubert), QC, Canada, J3Y 8Y9

Abstract— **In this paper, we describe a framework for the autonomous capture and servicing of satellites. The work is based on laboratory experiments that illustrate the autonomy and remote-operation aspects. The satellite-capture problem is representative of most on-orbit robotic manipulation tasks where the environment is known and structured, but it is dynamic since the satellite to be captured is in free flight. Bandwidth limitations and communication dropouts dominate the quality of the communication link. The satellite-servicing scenario is implemented on a robotic test-bed in laboratory settings.**

I. INTRODUCTION

Over the past few decades, robots have played an increasingly important role in the success of space missions. The Shuttle Remote Manipulator System (also known as Canadarm) has enabled the on-orbit maintenance of assets such as the Hubble Space Telescope. On the International Space Station (ISS), Canadarm2 has been a crucial element in all construction activities. Its sibling, Dextre, will be essential to the maintenance of the ISS. In the context of planetary exploration, robotics has also played a central role on most landed missions. The well-known NASA Mars rovers Spirit and Opportunity are vibrant examples of how robots can allow scientists to make discoveries that will be impossible otherwise.

In light of the missions currently being planned by space agencies around the world, the coming years will only show an increase in the number and the criticality of robots in space missions. Examples include the US Defense Advanced Research Project Agency's (DARPA) Orbital Express mission [1] and the German Space organization (DLR) TECSAS [2] in the area of satellite servicing.

One of the current drawbacks of space robots is that their operation is very human-intensive. Furthermore, all operations carried-out by these robotic systems are thoroughly verified in simulation prior to their execution in space. Different scenarios are run to verify all possible combinations of nominal and anomalous conditions. The verification process spans over several months, taking nominally more than one year. Every hour of operation requires thousands of hours of involvement from support personnel in the planning, verification, and execution phases [3]. The advent of ground control for Canadarm2 on the ISS has relieved the crew from having to perform all robotic operations. Ground-based operators now have the capability to uplink individual commands for automatic execution on the ISS. However, ground control has not reduced the level of effort required for robotic operations: the process to plan, conduct, and support robotic operations remains the same.

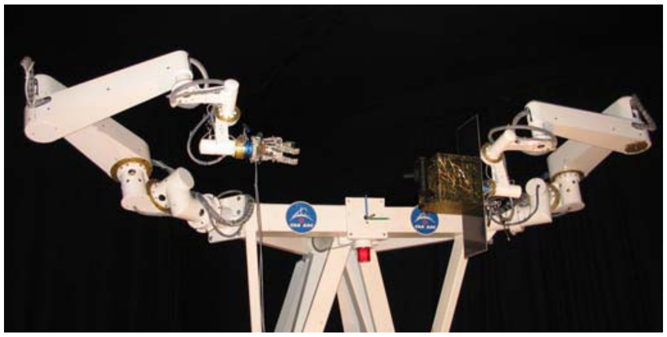


Fig. 1. A dual manipulator system that simulates the tracking and capture scenario. The manipulator on the left is equipped with a hand, and the manipulator on the right is mounted by a mock-up satellite.

One important area for the application of space robotics is autonomous On-Orbit Servicing (OOS) of failed or failing spacecraft. The commercial viability of such operations will require the usage of an efficient process for the planning, verification and execution of operations. In addition, autonomy capabilities will be required for the capture of the target satellite to be serviced. In this paper, we describe the laboratory experiments that verify the feasibility of our approach and demonstrate the autonomous aspect of the process. In particular, a mock-up satellite, the target, is mounted on one manipulator while a second manipulator equipped with a hand fixture, the *chaser*, approaches and captures the *target* (Fig. 1).

In the next section, we present OOS related work. Section III provides an overview of the autonomous aspects of the work. The trajectory planning for the two satellites is outlined in Section IV. Section V contains the experimental results and the last section presents our conclusions.

II. RELATED WORK

For many years, robots such as Canadarm and Canadarm2 have been used in space to service expensive space assets [4]. Canada has also developed another robot called Dextre for the International Space Station (ISS), which should be launched in 2007 and will be used to perform maintenance tasks. The European Space Agency (ESA) has developed the European Robotic Arm (ERA) [5] while the Japan has developed the JEMRMS [6].

In order to speed up the acceptance of OOS and decrease operational costs, a few technology demonstration missions have already been or will soon be conducted. Japan first conducted the ETS-7 mission in 1998-1999 [7]. ETS-7 involved the capture of a target satellite using a chaser satellite equipped with a robotic arm. Both satellites were launched together to minimize risks. The robotic capture was performed while the two satellites were still tied using the latching mechanism, again for reducing the risks [8]. The mission goal was successfully accomplished. DARPA is currently funding the development of the Orbital Express mission to be launched in 2006 [9]. This mission intends to prove the feasibility of OOS and refueling. The Orbital Express servicing spacecraft ASTRO is equipped with a robotic arm to perform satellite capture and Orbital Replacement Units (ORU) exchange operations. Recently, the US Air Force Research Lab demonstrated key elements of extended-proximity operations with the XSS-11 mission [10], [11]. A mission by NASA with similar objectives, DART, flew in 2005 [12]. The objective was to perform an autonomous rendezvous; unfortunately, the mission failed.

The TEChnology SAtellites for demonstration and verification of Space systems (TECSAS) is a mission led by DLR with Canadian participation [2], [13]. The objectives of the mission, planned for 2010, are to prove the availability and advanced maturity of OOS technologies, and the mastering of the capabilities necessary to perform unmanned on-orbit assembly and servicing tasks. For TECSAS, a servicer satellite carrying a robotic subsystem and a client satellite will be launched together. The mission will comprise different phases in which the following features will be demonstrated: far rendezvous, close approach, inspection fly around, formation flight, capture, stabilization and calibration of the coupled system, flight maneuvers with the coupled system, manipulation on the target satellite, active ground control via tele-presence, and passive ground control during autonomous operations.

There also have been several spacecraft designed for transporting logistics to the ISS such as Russia's Progress [14], Japan's HTV [15], and Europe's ATV [14]. Many key technologies required for OOS have already been or will be demonstrated with these missions. For a detailed description of the above missions please refer to [13].

III. AUTONOMY

A. High-level scenario

Using a robot equipped with two 7-degree-of-freedom arms, we demonstrate a fully autonomous free-flyer capture operation. One arm emulates the free-flyer dynamics and the second is the chaser robot equipped with a SARAH hand (developed by University Laval [16]). The Laser Camera System (LCS) from Neptec is used to guide the target robot, and cameras are used to provide video output to the remote operator. A hierarchical finite state machine engine is used to coordinate the autonomous capture task and to provide feedback to the operator. The role of the remote operator is very simple: initiate the capture by submitting the high level command, and monitor the chaser robot while performing the task. The sequencing of events (approach, fly together, and capture) is fully autonomous. In case of an emergency the operator can send a halt or abort command to the autonomy engine.

B. Autonomous Capture

The central tool for the autonomous operation is the *Cortex* Toolbox, which is used to implement command scripts and sets of reactive behaviours. *Cortex* has been developed to ease the development of such behaviour, which rapidly becomes labour intensive even for simple systems when using low level programming languages. *Cortex* is based on the Finite State Machine (FSM) formalism, which provides a higher-level way of creating, modifying, debugging, and monitoring such reactive autonomy engines. Two advantages of this representation are its intuitiveness and the ease with which it can be graphically constructed and monitored by human operators.

The concept of hierarchical FSM allows a high-level FSM to invoke a lower-level FSM. This provides the capability to implement hierarchical task decomposition from a high-level task into a sequence of lower-level tasks. If the FSM is implemented in a modular fashion, it allows the implementation of the concept of libraries that provide the operator with the possibility of the reuse of FSM from one application to another.

In general, FSM's are used to represent a system using a *finite* number of configurations, called *states*, defined by the system parameters or its current actions. The system can *transition* from one state to another based on its current state, conditions, and outside events. Conditions on transitions are often referred to as *Guards*, and are implemented as statements that can be evaluated as being either true or false. Outside events, called *Triggers*, make the FSM evaluate its transition's guards and enable a transition to occur.

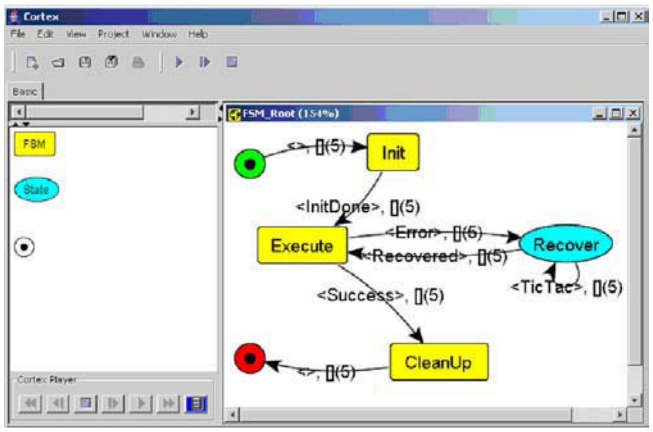
The use of an intuitive graphical representation of FSM (see Fig. 2a,b) by *Cortex* allows the developer and the operator alike to concentrate on the problem to be solved instead of concentrating on the programming skills to implement the solution.

The use of hierarchical FSM can address a wide spectrum of autonomy levels, from sense-and-react behaviours relying on sensor inputs to high-level mission scripts.

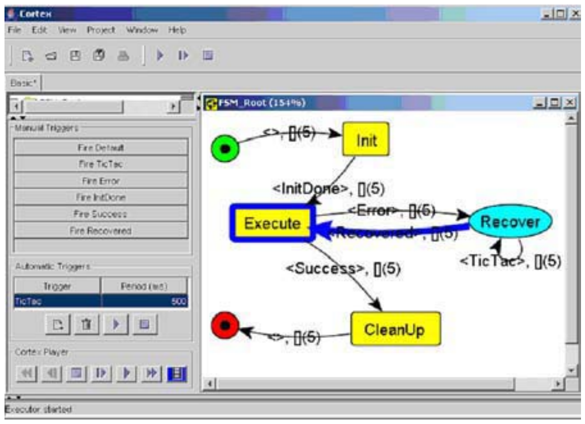
FSM's are assembled graphically using States, sub-FSM, junctions (a construct used to combine transitions), and transitions (Fig. 2a). The operator can provide JAVA code snippets for state actions and transition's guard expressions and can define the inputs, local variables, and output parameters of each sub-FSM. The user can also assign a priority to each transition to force the order in which their guards are tested during execution. Components from other FSM's can also be graphically "cut-and-pasted" to the current FSM to allow for the reuse of existing FSM's.

In this experiment, the *Cortex* Toolbox is used to implement the behaviours required for the autonomous capture of the target satellite. In an on-orbit demonstration, the autonomy engine will take control when the two spacecrafts are distant by a few meters. It will be responsible for performing the final approach of the chaser spacecraft to the target, deploying the manipulator arm and performing the capture of the slowly tumbling target satellite. In our experimental set-up, *Cortex* is used through out the whole process as the chaser manipulator and the target satellite are only a few meters apart.

Transitions between phases of the operation are triggered by sensory events. The *Cortex* engine considers anomalies such as



(a)



(b)

Fig. 2. (a) *Cortex* GUI: FSM Editing Environment. (b)*Cortex* GUI: FSM Monitoring Environment.

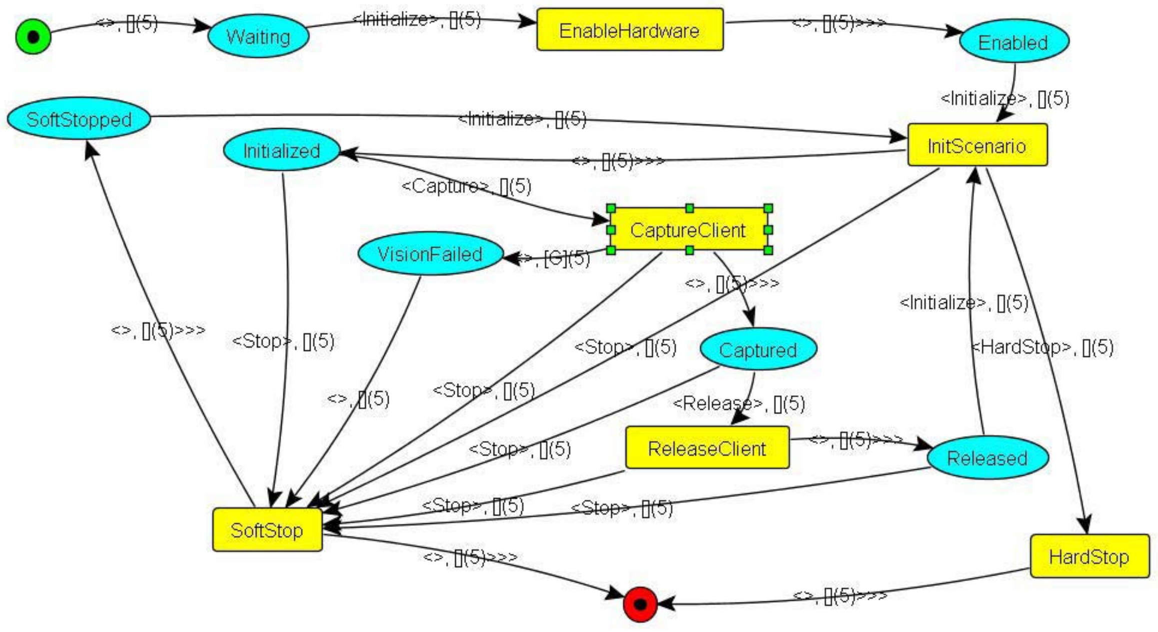


Fig. 3. *Cortex* engine describing the top level scenario of autonomous capture of a tumbling satellite

the possibility of the target spacecraft to drift out of the capture envelope of the manipulator (through translation or rotation); blinding of the vision sensor or loss of sight; reduction of the distance between the manipulator and the target satellite below an acceptable, safe limit; or failed capture which results in the target satellite being sent into a tumbling mode. Fig. 3, shows a *Cortex* implementation of the later phases of a typical OOS scenario after an autonomous far rendezvous has been performed. The operator has overriding control of pausing the process using a soft-stop, and then restarting it, or terminating the process by using a hard-stop. The actual capture sequence is itself a state machine of three stages: approach, align, and capture.

C. Operations and Monitoring

One very important aspect of space robotics is the capability to remotely monitor and/or operate a device. In our architecture, the Ground Control Station (GCS) is used first to develop and monitor the command scripts and autonomous behaviours using the *Cortex* Toolbox. In this context, the GCS is connected to a simulator of the remote system, which includes the dynamics of the two satellites, the robot arm and the vision system. The simulator is then used to validate the command scripts and the autonomous behaviours associated with each phase of the mission. Depending on the predictability of the parameters triggering transitions in the *Cortex* engine, some portions can be simulated in a very deterministic manner. For example, the manipulation of ORU's is performed in a very controlled and static environment, as the target satellite itself is subject to very little uncertainty. On the other hand, other portions of the mission, such as the capture of the target satellite, will be subject to factors such as illumination conditions and initial conditions on satellite attitude, which are much more unpredictable. The validation of the portions of the command script and autonomous behaviours associated with these events will therefore require validation using a broader range of parameters to systematically verify the plausible

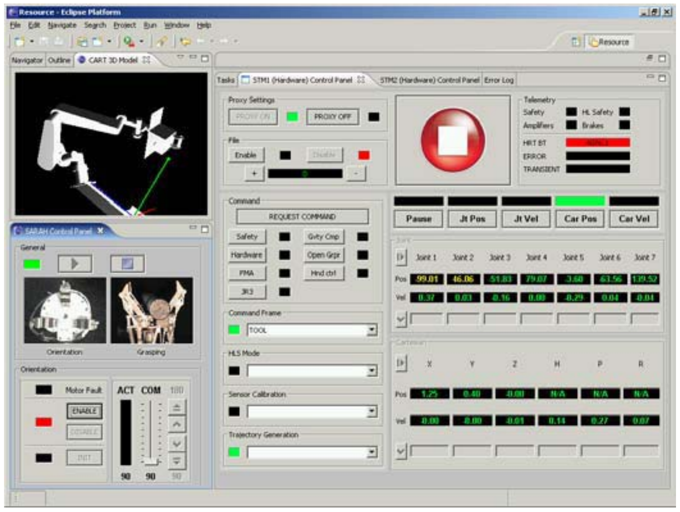


Fig. 4. Snapshot of the Ground Control Station.

outcomes of the mission.

After verification, the GCS is reconfigured to connect to the real system. The validated *Cortex* behaviours and scripts are then uploaded to the system for execution. Depending on the communication windows, it is possible to require operator confirmation before the performance of key steps in the command script. The synergy of the *Cortex* autonomy engine and the GCS allow for operator intervention at different phases of an operation without hindering the autonomous nature of the operation.

In our experiments, we have used the GCS, shown in Fig. 4, to remotely monitor and command the two-arm system from a nearby town and from a different continent (transatlantic) over an Internet connection.

IV. TRAJECTORY PLANNING

A. *Target Satellite*

Many satellites use momentum wheels to stabilize and to control their attitude. When the attitude-control system fails, due to friction, the angular momentum stored in the wheels is transferred to the satellite body over time. This momentum transfer causes tumbling in the satellite. At the same time, the satellite is under the action of small external, nonconservative moments, which make the satellite tumbling motion occur mostly about its major principal axis [17], i.e., the axis about which the satellite moment of inertia is maximum.

Therefore, to mimic the tumbling motion of a satellite, we assume that the satellite is initially rotating about an axis very close to its major principal axis. Then, to create the trajectory, we solve the Euler equations assuming no external moments. It should be noted that, even though there are dissipative moments acting on the satellite, they would not have any significant effect over the short period of capture and the angular momentum of the satellite would be conserved. The ensued motion of the target satellite can be characterized as a major rotation about a spin axis with small precession and nutation [18]. This is similar to the motion described in [19].

By varying the initial conditions of the Euler equations, we have generated different sets of possible trajectories for

Fig. 5. Laser Camera System (LCS) from Neptec.

the target satellite. These scenarios start from simple slow motions, but gradually grow to faster motions with relatively larger nutation angles and precession rates.

Since physical restrictions prevent the robot joints from indefinite rotations, we could only generate a continuous momentum-conserving motion for about thirty seconds. Therefore, to continue the motion, the motion of the satellite mock-up is slowed down towards the end and reversed. By doing so, we can move the satellite as long as we want, without running into the joint limits or the robot singularities.

B. *Chaser Manipulator*

This section describes how the trajectory of the chaser manipulator is generated. A Laser Camera System (LCS), shown in Fig. 5, and the software CAPE for Collision Avoidance and Pose Estimation, both from Neptec, are used to generate the pose (position and orientation) of the target satellite with respect to the LCS Frame. The pose of the tumbling satellite is tracked at about 2 Hz with a 0.5 sec. delay on the pose of the object at a given instant. The location of the LCS sensor with respect to the manipulator inertial frame was precisely calculated using the kinematic model of the robot arm holding the target satellite; thus, the actual pose of the target satellite in the inertial frame can easily be calculated with high accuracy.

An extended Kalman filter is used to filter the LCS raw measurements and to provide a smoothed pose of the target satellite every millisecond. The Kalman filter also includes a predictor to estimate the pose of the target satellite 0.5 sec forward in time. This predictor is used to eliminate the 0.5 sec delay in obtaining the pose of the target satellite using the LCS sensor.

Based on the filtered pose of the capture handle (capture frame) and on the pose of the grasping device (tool frame), a Cartesian velocity command is generated in the tool frame of the manipulator to progressively reduce the distance between them. Generating the command in the tool frame gives the possibility to independently activate and assign each degree of freedom to a different task. In the experiments, the activation of the tasks is based on the distance between the grasping device and the target satellite. Here is a description of the tasks performed during the capture operation.

1) Tracking: The first activated task is the tracking of the target. In the case where a camera is mounted on the end-effector of the manipulator, the goal of this task is to position the visual features in the center of the camera field of view.

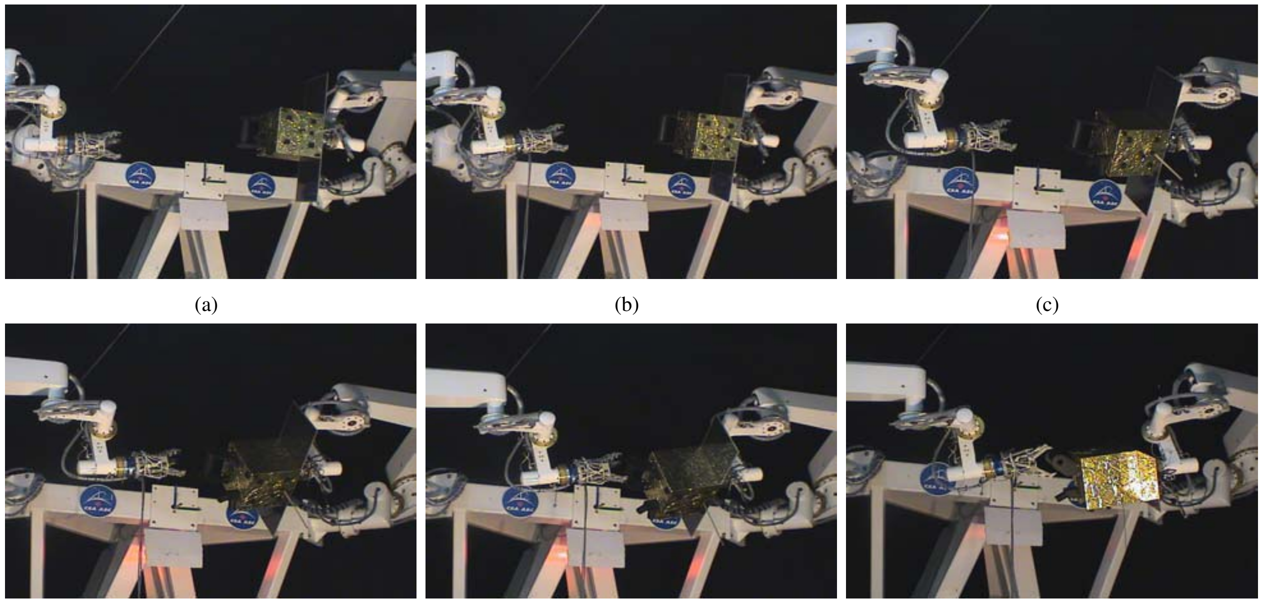


Fig. 6. The chaser satellite approaches the target satellite and tracks its motion. The last figure shows the hand in place to perform the capture.

The pan-tilt motion of the manipulator uses two rotational degrees of freedom and is generated using

$$\omega_y = -k_{R_y} \arctan([r_{z_c}]_t/[r_{x_c}]_t) \tag{1}$$

$$\omega_z = -k_{R_z} \arctan([r_{y_c}]_t/[r_{x_c}]_t) \tag{2}$$

where k_{R_y} and k_{R_z} are control gains for their respective axes, and $[r_{x_c}]_t$, $[r_{y_c}]_t$, and $[r_{z_c}]_t$ determine the position of the origin of the capture frame, attached to the target satellite, in the manipulator tool frame.

2) *Initial Approach:* When the tracking task is activated, the manipulator end-effector points toward the target. Based on that assumption, the desired distance between the grasping device and the target is controlled by the translational degree of freedom that moves the end-effector forward and backward. The command is computed as

$$\dot{\mathbf{r}} = \mathbf{K}\mathbf{R}_t^T \mathbf{R}_c([\mathbf{r}_t]_c - [\mathbf{r}_{des}]_c) \tag{3}$$

where $\mathbf{R}_t$ and $\mathbf{R}_c$ are respectively the rotation matrices representing the orientation of the tool frame and of the capture frame, $\mathbf{K}$ is the control gain matrix which is defined as $\mathbf{K} = \text{diag}(k_x, k_y, k_z)$, $[\mathbf{r}_t]_c$ is the position of the tool frame expressed in the capture frame, and finally $[\mathbf{r}_{des}]_c$ is the desired tool frame position relative to the capture frame.

3) *Translation Alignment:* In order to avoid undesired large end-effector velocities, the grasping device is aligned perpendicular to the target only when they are close to each other. This task uses the two remaining translational degrees of freedom. Equation (3) shows how the translational velocity command is generated.

In the Equation (3), the gains k_x, k_y, and k_z are not activated simultaneously. First, the approach gain (k_x) is activated; when the distance becomes small, the alignment gains (k_y and k_z) are activated. When the three tasks are active, the grasping device is positioned directly in front of the target. Depending on the grasping device used for the capture, a final roll alignment may be required.

4) *Roll Alignment:* In our experiment, the SARAH hand is used to grasp a cylindrical handle. The first task is to align the hand so that the handle fits between the fingers of the hand. In order to achieve that, the remaining rotational degree of freedom is used as

$$\omega_x = -k_{R_x}[\theta_{x_c}]_t \tag{4}$$

where k_{R_x} is the control gain and $[\theta_{x_c}]_t$ is the orientation of the capture frame about the x-axis of the tool frame.

5) *Capture:* With all tasks activated, adjusting the desired relative pose ($[\mathbf{r}_{des}]_c$) to have the handle in the middle of the hand and closing the SARAH hand fingers achieves the capture.

V. EXPERIMENTAL RESULTS

A variety of experiments were performed in the laboratory. Different trajectories were tested for the target satellite. Moreover, the experiments where initiated and monitored remotely from a variety of locations. In all the experiments, the chaser manipulator first approached the target at a safe distance, and then followed the target satellite maintaining the same safe constant distance. In the end, the command was given for the arm to perform the final approach and to grasp the handle mounted on the mock-up satellite using the SARAH hand.

Fig. 6 presents a sequence of images from one of the autonomous capture scenarios performed. In the first three pictures (Fig. 6a,b,c), the chaser manipulator approaches the target maintaining a constant orientation (the thumb of the hand is at the bottom). When the target is close enough, the chaser manipulator adjusts the hand orientation to be in sync

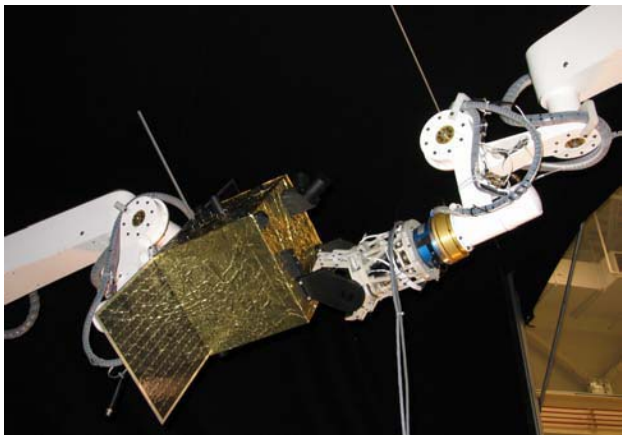


Fig. 7. The SARAH hand grasping the target satellite.

with the handle on the target. This can be seen in the next three pictures (Fig. 6d,e,f) where the SARAH hand is aligned with the handle and follows it. Finally, Fig. 7 shows the capture of the mock-up satellite by the chaser.

Some of the experimental data are presented in Fig. 8. In that figure, the tracking of the x, y, and z coordinates of a point 15 cm in front of the center of the capture handle is illustrated. The dotted (black) lines show the desired tracking position calculated from the raw data provided by the LCS. The solid (blue) lines are the outputs of the Kalman filter, namely, the desired tracking position. Finally, the dash-dotted (red) lines show the actual coordinates of the end-effector of the chaser, which is commanded to track the desired tracking position. As observed in Fig. 8, the tracking is very good.

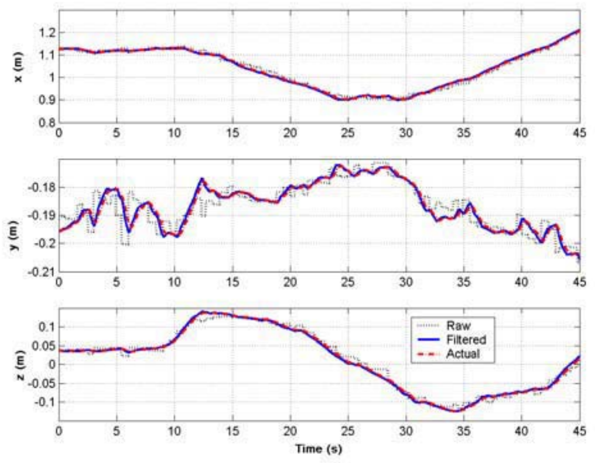


Fig. 8. Experimental results; tracking of the capture handle.

VI. Conclusions

In this paper, we presented missions related to on-orbit servicing of space assets. The Canadian Space Agency framework to perform autonomous operations was described. One central tool in that framework is the *Cortex* Toolbox which is used to implement scripts and sets of reactive behaviours using the Finite State Machine formalism. As an application of the use of Cortex, the successful autonomous tracking and capture of a tumbling satellite was demonstrated. Many trials were remotely controlled and monitored in a process that resembles operations in space. Due to the complex nature of the experiment a variety of technologies were crucial. *Cortex* as mentioned above, trajectory generation for the tumbling satellite and the chaser manipulator, and remote operation capabilities were among the most important contributions of this work.

References

[1] D. Whelan, E. Adler, S. Wilson, and G. Roesler, “Darpa orbital express program: effecting a revolution in space-based systems,” in *Proc. SPIE, Small Payloads in Space*, vol. 4136, November 2000, pp. 48–56.

[2] B. Sommer, “Automation and robotics in the german space program - unmanned on-orbit servicing (oos) & the tecsas mission,” in *The 55th International Astronautical Congress*, Vancouver, Canada, Oct. 2004.

[3] A. Popov, “Mission planning on the international space station program. concepts and systems,” in *IEEE Aerospace Conference*, Big Sky, MT, USA, March 2003.

[4] M. Stieber, S. Sachdev, and J. Lymer, “Robotics architecture of the mobile servicing system for the international space station,” in *Proc. 31st Int. Symposium on Robotics (ISR)*, Montreal, Quebec, May 2000.

[5] F. Didot, M. Oort, J. Kouwen, and P. Verzijden, “The era system: Control architecture and performance results,” in *Proc. 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, Montral, Canada, June 2001.

[6] N. Sato and S. Doi, “JEM Remote Manipulator System (JEMRMS) Human-in-the-Loop Test,” in *Proc. 22nd International Symposium on Space Technology and Science*, Morioka, Japan, 2000.

[7] T. Kasai, M. Oda, and T. Suzuki, “Results of the ETS-7 mission, rendezvous docking and space robotics experiment,” in *Proc. 5th Int. Symposium on Artificial Intelligence, Robotics and Automation In Space (i-SAIRAS)*, Noordwijk, The Netherlands, 1999.

[8] K. Yoshida, “Contact dynamics and control strategy based on impedance matching for robotic capture of a non-cooperative satellite,” in *Proc. 15th CISM-IFToMM Symp. On Robot Design, Dynamics and Control - Romansy*, St-Hubert, Canada, June 2004.

[9] S. Potter, “Orbital Express: Leading the way to a new space architecture,” in *Space Core Tech Conference*, Colorado Spring, November 2002.

[10] E. Grossman and K. Costa, “Small, experimental satellite may offer more than meets the eye,” *Inside The Pentagon*, December 4 2003.

[11] J. Lewis, “Space weapons in the 2005 US defense budget request,” in *Workshop on Outer Space and Security*, Geneva, Switzerland, 2004.

[12] T. E. Rumford, “Demonstration of autonomous rendezvous technology (DART) project summary,” in *Proc. SPIE*, vol. 5088, 2003, pp. 10–19.

[13] E. Martin, E. Dupuis, J.-C. Piedboeuf, and M. Doyon, “The TECSAS mission from a Canadian perspective,” in *Proc. 8th International Symposium on Artificial Intelligence and Robotics and Automation in Space (i-SAIRAS)*, Munich, Germany, Sept. 2005.

[14] T. Boge and E. Schreutelkamp, “A new commanding and control environment for rendezvous and docking simulations at the EPOS facility,” in *Proc. 7th International Workshop on Simulation for European Space Programmes (SESP)*, Noordwijk, Nov. 2002.

[15] *HTV information page: http://www.jaxa.jp/missions/projects/rockets/htv/index_e.html.*

[16] T. Laliberté, L. Birglen, and C. Gosselin, “Underactuation in robotic grasping hands,” *Japanese Journal of Machine Intelligence and Robotic Control, Special Issue on Underactuated Robots*, vol. 4, no. 3, pp. 77–87, 2002.

[17] M. H. Kaplan, *Modern Spacecraft Dynamics and Control*. New York: Wiley, 1976.

[18] H. Goldstein, *Classical Mechanics*, 2nd ed. Reading, MA: Addison-Wesley, 1980.

[19] H. Nagamatsu, T. Kubota, and I. Nakatani, “Capture strategy for retrieval of a tumbling satellite by a space robotic manipulator,” in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, Minneapolis, Minnesota, April 1996, pp. 70–75.