Acceleration-controlled teleoperation with variable latency
Nicolas Lee – ME326 Final Report

I. INTRODUCTION
In 2004, a robotic repair mission to the Hubble Space Telescope was proposed as an alternative to servicing via the space shuttle. Despite the cancellation of this robotic mission, interest remains strong in developing the ability to repair satellites on orbit. Future repair missions will likely include a mixture of autonomous and teleoperated robotic systems to accomplish tasks of varying complexity and urgency. Teleoperation from a ground control station would require communication via TDRSS (Tracking and Data Relay Satellite System) in geosynchronous orbit. Past experience with teleoperation through TDRSS during the Japanese ETS-VII (Engineering Test Satellite 7) mission indicates that the latency is approximately 6 seconds [1] but the exact latency is not well characterized. A good understanding of this variable latency and its effect on teleoperator performance is essential for robotic servicing on orbit to be feasible.

II. PREVIOUS RESEARCH
Past studies have characterized the effect of constant delay on specific tasks, such as inserting a peg in a hole [1-3] or tapping the end of a pole [4]. [3] also experimented with navigation through an obstacle course. The metric used to measure performance in these studies has typically been completion time for the given task. The trend, predictably, is that the time it takes to complete a task increases with the latency in the system. The relationship between latency and completion time was found to be roughly linear up to a point. With higher latencies, the operators tend to transition to a “move and wait” control strategy, greatly increasing the completion time.

My study is targeted toward characterizing the effect of variability rather than magnitude of latency on the performance of a teleoperated system.

III. HARDWARE AND SOFTWARE IMPLEMENTATION
The experiments were performed on a Free-Flyer robot in the Aerospace Robotics Laboratory. The robot floats on a smooth granite table using compressed air to generate a near-frictionless cushion for motion in the horizontal plane. Compressed air is also used in eight cold-gas thrusters arranged in pairs around the robot to provide thrust in four directions. The robot is equipped with a momentum wheel for controlling its angular position and rate. An overhead camera and state estimator provides position, heading, and velocities by detecting LEDs on top of the robot. This state information feeds into an off-board computer running the control software, which feeds thruster commands through a wireless connection to the robot. A Logitech Extreme 3D Pro joystick was used as the input device.

The software for the telerobotic system was implemented in Constellation, a C-based real-time environment for control systems. The two main joystick axes were used for x and y thruster control in the robot frame. The momentum wheel was used to keep the robot frame aligned with the table frame so that global x and y forces could be applied without looking at the robot. The joystick software was programmed with a deadband to minimize accidental firings. A 10 Hz pulse width modulation was used to generate forces proportional to the joystick input above the prescribed deadband. The control software polled the joystick at a rate of 100 Hz to be above the maximum human control bandwidth of 50 Hz [5], and introduced a latency before transmitting the input signal to the thruster controller. For feedback, the overhead vision system’s estimate of x and y position and velocity of the robot was displayed on a console window. This state information from the
overhead vision system was delayed by the same amount as the thruster commands before being displayed.

IV. EXPERIMENTAL SETUP

The task performed for this study was a 0.75 meter straight-line traverse along the granite table. The robot started from rest at a fixed initial location, and was brought as close as possible to zero velocity near the fixed target location. The initial and target locations were aligned along the table’s y-axis and the momentum wheel kept the robot aligned with the table frame, so that the traverse required mostly y-axis input, while maintaining a constant position along the x-axis.

The traverse can be partitioned into two phases: a movement phase to bring the robot near the target followed by a station-keeping phase to keep the robot from drifting away. During each traverse, the time history of control inputs, latency, and vehicle state were recorded. This data yielded four metrics of performance:

1) Time to reach the target;
2) Radius of convergence;
3) Effort to reach the target; and
4) Effort to maintain station.

The transition between movement and station-keeping phases was selected to be the second zero-crossing along the y-axis relative to the target position. The time to reach the target is the duration of the movement phase, and the radius of convergence is the maximum distance from the robot to the target in the station-keeping phase. The effort to reach the target is measured using the norm-square of the control input during the movement phase, and the effort to maintain station is measured using the norm-square of the control input during the station-keeping phase. The effort to maintain station is normalized by the duration of the station-keeping phase.

These four performance metrics focus on different aspects of human-in-the-loop control. The time to reach the target and effort to reach the target indicate how aggressively the robot was being driven to the target. The radius of convergence is a measure of how precise the teleoperator could be, and the effort required to maintain station measures teleoperator efficiency.

A typical traverse is shown in Figures 1 and 2. In Figure 1, the trajectory of the robot is shown. In Figure 2, the time histories and phase transition are shown. Of particular note in this plot is the fact that the control during the station-keeping phase was pulsed – this implies that the human teleoperator was acting possibly as a high-gain controller with a deadband or hysteresis.

The signal latency was generated using a linked list buffer in the software. A new Gaussian random number was generated every half-second and was added to the timestamp of the input signal. When the program time passes the timestamp, the signal is released to the thruster controller. As a result, negative latencies were truncated to zero. An artifact of this latency scheme is that signals can overlap – an input sent during a period of high latency can be transmitted after a subsequent input if the latency drops. In each run, a fixed mean and standard deviation were set to generate the random latencies.

In total, 23 runs were performed. This was constrained both by time limitations and the supply of compressed air available. The latency parameters used during these runs were arranged to yield cross-sectional results – all had either a mean latency of 0.2 or 0.4 seconds, or a standard deviation of 0 or 0.06. This arrangement maximized the chance of a trend being detectable given the limited number of data points. In total, six runs had a mean delay of 0.2 seconds, eleven with a mean delay of 0.4 seconds, seven with a standard deviation of 0, and four with a standard deviation of 0.06. The resulting metrics from these runs are presented in four sets of four plots in Figures 3 through 6.
V. RESULTS

From the data series where the standard deviation was held fixed at zero, the latency was constant so trends could be compared to previous research. This data series is summarized in Figure 3. The time and effort to reach target did not show any significant increasing or decreasing trend – this is an indication that the human controller aggressiveness was not correlated with magnitude of delay. However, the radius of convergence and effort to maintain station both increased with increasing latency. This agrees with previous research in the field.

Based on this data, it appears that the radius of convergence and effort to maintain station are reasonable metrics to assess control performance with this task. However, the two metrics are conflicting goals – if the radius of convergence were not a factor, the station-keeping effort could clearly be zero. This indicates that a more suitable metric might be a combination of the two in order to measure overall performance.

Unfortunately, similar trends were not apparent with the standard deviation held fixed at 0.06 (Figure 4), nor with the data series where the latency mean was held constant and the standard deviation varied (Figures 5 and 6). In the latter two data series, the radius of convergence remained roughly constant except for a spurious data point while the effort to maintain station was scattered. This spurious point was the last in a series after a night without much sleep. It is very likely that this was the dominant factor in performance, rather than the latency or its variability.

Figure 7 presents the data obtained in chronological order. The first six trials were performed in a batch the morning after the aforementioned sleepless night. The seventh trial was an isolated run later that night. The remaining trials were all done in a batch three days later when I was better rested. Since the latency parameters were varied throughout these runs, it is difficult to draw conclusions, but the effort to maintain station does appear to be decreasing. This could be evidence of human learning and adaptation, but could also be a function of the knowledge that the supply of compressed air was diminishing.

Aside from the quantitative measurements of performance, it was possible to extract some information on control strategies that were used. In the movement phase, the strategy was generally to apply forward thrust until the velocity reached a comfort threshold, coast, and start applying reverse thrust shortly after half the traverse was completed. After a number of runs, it became possible to anticipate how long to apply each direction of thrust, resulting in an open loop control scheme. This strategy was consciously not employed during the trial runs, in an effort to assess the performance under feedback.

In the station-keeping phase, there were two noticeable control strategies. With low latency, the rate of change in the position display was fairly reliable, so individual thruster pulses were applied to effect changes in velocity. As the velocity decreased, and in high-latency runs, the position indicator yielded less reliable velocity data. In these cases, the control strategy switched to using double pulses in opposing directions to effect a change in position without much effect on velocity. In general, rapid thruster pulses were preferable to sustained lower magnitude inputs. This could be a result of a built-in deadband in the joystick.

Throughout all conditions and during the entire traverse, very little attention was paid to the velocity readout. It is possible that with the feedback data in numeric format, it was not possible to process four numbers at a time. Further testing with displays could allow for more efficient data feedback to the human controller by using graphical indicators.

VI. FUTURE WORK

A consequence of the limited data set is that only qualitative trends could be gleaned from the experimental results. For any quantitative conclusions, a much larger set of data must be collected. Using the ARL Free-Flyer for this task is not the ideal solution due to the high use of compressed air, so a logical next step would be to implement a simulated robot in Constellation to
interface with the joystick. This would allow for much more data collection without using up lab resources. In addition, a software simulation would eliminate the time-varying nature of the robot. Control authority was often affected by the air pressure in the reservoir tanks – as they emptied, the thrusters produced less force.

Also valuable would be a method of controlling for variation in the human controller. Test runs being performed after a night of no sleep were not conducive to comparison against other runs, and there was not enough time to take multiple runs for each set of latency parameters. With a software simulation, some variation in the human can be averaged out. Also, with more data runs, the operator could practice until the learning curve plateaued, eliminating another source of variation.

VII. CONCLUSION

The hardware setup proved to be an impediment to large-scale testing of the nature required to obtain quantitative results. However, promising leads were developed for further research in a simulated environment. The modular nature of software in Constellation will allow for easy replacement of the hardware components by software simulations.

VIII. REFERENCES


IX. FIGURES

Figure 1: Trajectory of a typical run
Figure 2: Time history of data from a typical run
Figure 3: $\mu_d = 0.2$ seconds

Figure 4: $\mu_d = 0.4$ seconds
Figure 5: $\sigma_d = 0$ seconds

Figure 6: $\sigma_d = 0.06$ seconds
Figure 7: In chronological order

- **Time to reach target**
  - Y-axis: Seconds
  - X-axis: Trial number

- **Radius of convergence**
  - Y-axis: Meters
  - X-axis: Trial number

- **Effort to reach target**
  - Y-axis: $|u|^2$
  - X-axis: Trial number

- **Effort to maintain station**
  - Y-axis: $|u|^2$ per second
  - X-axis: Trial number