IAC-08-C1.5.1

RESULTS OF SPHERES MICROGRAVITY AUTONOMOUS DOCKING EXPERIMENTS IN THE PRESENCE OF ANOMALIES

Amer Fejzic
Massachusetts Institute of Technology, Cambridge, MA, United States
afejzic@mit.edu

Simon Nolet, Louis Breger, Jonathon P. How, David W. Miller
Massachusetts Institute of Technology, Cambridge, MA, United States
snolet@alum.mit.edu, lbreger@alum.mit.edu, jhow@mit.edu, millerd@mit.edu

Abstract

In 2006 and 2007, the SPHERES testing facility, a series of micro-satellites developed by the MIT Space Systems Laboratory and designed to operate autonomously inside the International Space Station, has been extensively used to demonstrate multiple autonomous docking scenarios. Following the first autonomous docking to a tumbling target ever achieved on-orbit, some off-nominal docking scenarios were attempted. Multiple autonomous docking maneuvers were achieved in the presence of simulated failures. In some situations, the chaser was required to perform a collision avoidance maneuver, which consisted in a simple thrust sequence that brought it on a safe trajectory. Two different algorithms were used to perform some of these experiments. The first one involved a traditional glideslope approach, where the chaser is commanded to reduce its approach velocity with the distance-to-go. The second involved a pre-planned trajectory ensuring a passive abort in case a failure is detected. Following the success of these experiments, interesting conclusions were made on the performance of both algorithms, based on the observations during the experiments and the telemetry. This paper will present the results of these experiments, along with a comparison of the performance of both algorithms for the different scenarios attempted.

I. Introduction

The capability of automated rendezvous and docking is a key enabling technology for many government and commercial space programs [1, 2, 3]. Future space systems will employ a high level of autonomy to acquire, repair, refuel, and reconfigure satellites. In close proximity operations, task sequencing and execution must be robust to unexpected events. These events may be internal (e.g., a component failure), or external (e.g., an obstacle in the desired flight path). A possible docking scenario during routine autonomous servicing of a disabled satellite would be with a target spacecraft that is tumbling. The tumble may have been initiated by a thruster stuck-OFF failure after deployment from a launch vehicle.
This paper compares two different algorithm approaches that address the issues of docking in the presence of anomalies to both fixed (non-rotating) and tumbling target spacecraft. The first algorithm is comprised of a traditional glideslope approach, where the chaser is commanded to reduce its approach velocity with the distance-to-go. The second involves a pre-planned trajectory from a solution to a formulated Mixed-Integer Linear Program [4] that ensures a passive abort in case a failure is detected. The experiments were performed on the Synchronized Position Hold Engage and Reorient Experimental Satellites (SPHERES) testbed aboard the International Space Station (ISS) in November, 2006 and March, 2007.

The paper begins by reviewing the two algorithms: glideslope algorithm and safe trajectory optimization. The review of the latter algorithm covers only the formulation of the safety constraints, while further details may be found in Ref. [4]. Afterwards, a description of the hardware testbed used to demonstrate these docking scenarios is presented. Next, the software implementation of the hardware is discussed that consists of the overall GN&C architecture and the placement of the two algorithms. The results section follows with a total of eight tests aboard the ISS that demonstrate the two algorithms in the presence of anomalies to fixed and tumbling targets. A comparison between how the two algorithms react to anomalies and their performance is discussed. The current status of autonomous docking with SPHERES and potential future work is mentioned before a summary of the paper.

II. Algorithm Review

This section reviews the two algorithms used for autonomous docking in the presence of anomalies: glideslope approach and safe trajectory optimization. Their characteristics and complexity is presented before observing any experimental data.

A. Glideslope Algorithm

The glideslope algorithm [5, 6, 7] is a common and widely used algorithm (Apollo, Shuttle) for trajectory planning in real time. It is used when a straight line approach to the target is required, often because of line-of-sight constraints with docking navigation sensors. It was also shown to be used for docking to a tumbling satellite [8, 9]. It is an algorithm that prescribes a proper approach velocity pattern to follow in the phase plane using a predetermined number of thruster pulses, equally spaced in time (Fig. 1). The approach velocity ($\dot{\rho}$) is commanded to decrease linearly with the distance-to-go ($\rho$). The maneuver has to occur in a period $P$ with a safe commanded arrival velocity ($\dot{\rho}_T$) and using a pre-determined number of thruster firings (Fig. 1).

Figure 1: Typical approach velocity profile computed by the glideslope algorithm [10].

This simple algorithm can be described by the following equation,

$$\dot{\rho} = a \rho + \dot{\rho}_T \tag{1}$$

where $a$ is the glideslope ($< 0$). The solution of the differential equation is,

$$\rho(t) = \rho_0 e^{at} + \frac{\dot{\rho}_T}{a} (e^{at} - 1) \tag{2}$$
while the maneuver period $P$ is

$$P = \frac{1}{a} \ln \left( \frac{\dot{\rho}_T}{\dot{\rho}_0} \right)$$

A glideslope algorithm is a hybrid between a path planning algorithm and a velocity control algorithm. At the appropriate time to apply a velocity correction, the algorithm determines the magnitude of the velocity correction based on the velocity required to reach the next waypoint by the time the next velocity correction is applied (Eq. (2)). The result is an actual trajectory, in the phase plane, that oscillates around the desired velocity pattern as shown in Fig. 1. When performing a straight approach, such an algorithm is appropriate to plan and control the trajectory along the docking axis, while a different control algorithm can be used to maintain alignment transverse to the docking axis.

The glideslope algorithm can easily be adapted to evasive type maneuver by selecting different initial and final commanded velocities. It is very simple, robust, easy to implement and requires low computational power. However, optimality is not guaranteed and the presence of obstacles is not taken into account. Nevertheless, it has the nice property of reducing the $\Delta V$ induced at each burn toward the end of the maneuver, which in some cases can help to reduce the effects of plume impingement.

B. Safe Trajectory Optimization

The following algorithm by Breger and How [4] defines a safe trajectory as an approach path that guarantees passive collision avoidance in the presence of a class of anomalous system behaviors. It guarantees that for a range of faults, no collisions will occur even if the chaser spacecraft cannot use thrusters, computers, or communications equipment. The rationale behind choosing this passive abort strategy is three-fold: (a) passive abort can protect against a large set of possible system failures simultaneously; (b) an abort trajectory that does not require fuel use guarantees that remaining fuel will not be expended rapidly to increase spacecraft separation distance, thereby increasing the likelihood that future docking attempts can occur; and (c) passive abort guarantees thrusting will not be used in close proximity to the target during an anomaly, thereby eliminating the danger of plume impingement during an automatic safe-mode maneuver.

Creating minimal fuel paths for space vehicles using linear trajectory optimization has been previously demonstrated [11, 12]. Breger and How introduced an additional safety formulation addressing the safe drift mode. These additional safety constraints are reviewed while a detailed description is presented in Ref. [4].

The safe trajectory optimization consists of minimizing the discrete fuel-optimal cost function,

$$J = \min_{\mathbf{u}} \sum_{i=0}^{N-1} \| \mathbf{u}_i \|_1$$

where $\mathbf{u} \in \mathbb{R}^3$ is the thrust control input, $\| \cdot \|_1$ is the 1-norm minimizing fuel use, and the trajectory spans $N$ time steps. The additional safety formulation considers the states of chaser after a failure occurs at time $T$. This is referred to as the failure state $\{ \mathbf{x}_k \mid \forall k \geq T \}$. The value of this state is divided into two parts: during the planning horizon for $T \leq k < N$, and after the planning horizon for $k \geq N$. The discrete convolution approach may be used to determine the state during the planning horizon, by considering all inputs to be zero on and after the time step $T$ [4].
\[ x_k = A_d^k x_0 + \begin{bmatrix} A_d^{k-1} B_d & A_d^{k-2} B_d & \ldots & A_d B_d & 0 \end{bmatrix} \begin{bmatrix} u_0 & \vdots & u_{k-1} \end{bmatrix} \]

where \( x_k \) in Eq. (5) is the state of the chaser spacecraft at some \( k < N \). Next, the failure state after the planning horizon is found by an open-loop propagation of the state at time \( N \):

\[ x_k = A_d^{k-N} x_N \quad \text{for } k \geq N \quad (6) \]

The safety horizon is defined to be the period of time after a failure at time step \( T \) and lasts up to \( S \) steps after the nominal planning horizon of \( N \) steps. Both equations (5) and (6) are used to compute the value of the state within the safety horizon. The failure states are constrained to not lie within the target spacecraft space defined by the position states \( T_k \) [4]:

\[ x_k \notin T_k \quad \forall k \in \{T + 1 \ldots N + S\} \quad (7) \]

The constraint in Eq. (7) is imposed for \( T \in \mathcal{F} \) where \( \mathcal{F} \) is the set of all possible failure times. Including this constraint in general linear trajectory optimization ensures a collision-free drift trajectory after a thruster shutoff safety abort. An additional constraint on the terminal conditions is included,

\[ A_{\text{Term}} N x_N \leq b_{\text{Term}} N \quad (8) \]

where \( A_{\text{Term}} k \) and \( b_{\text{Term}} k \) describe the states the spacecraft must occupy at the end of the planning horizon to achieve safe docking. These constraints can be both on position (e.g., enter a region within reach of a grappling arm) or and on velocity (e.g., dock within a velocity range that produces acceptable stress on the docking port). In addition, time-varying bounds are introduced on the maximum thrusting levels in order to ensure large thrusts are not planned for the period immediately before docking.

Lastly, another constraint is added for the trajectory to be within a region defined by a line-of-sight (LOS) cone that protrudes from the target spacecraft docking port. This ensures the rendezvous to remain within this region for any requirements of vision-based navigation. The LOS constraint is:

\[ A_{\text{LOS}} k x_k \leq b_{\text{LOS}} k \quad \forall k = 1 \ldots N \quad (9) \]

where \( A_{\text{LOS}} k \) and \( b_{\text{LOS}} k \) describe the states within the line-of-sight cone at time step \( k \). Figure 2 depicts such a line-of-sight cone on a target spacecraft.

![Line-of-sight cone and target spacecraft docking configuration](image)

The above non-convex constraints are employed into a Mixed-Integer Linear Program (MILP) framework to compute the safe trajectory [13]. The computation requirement and complexity is significantly greater than that of the glideslope algorithm. Generally, trajectory planning using MILP are NP-hard, requiring exponential time to solve in the worst case [12]. However, the authors also introduced a convex formulation that degrades fuel optimality for
improved computation performance. The convex formulation uses a Linear Program (LP) solver which typically converges very quickly. The convex or non-convex formulation requires greater computational power than the glideslope algorithm; however, several features are attained from the additional complexity that the glideslope algorithm cannot perform:

- optimal trajectory, fuel-minimal
- collision avoidance
- plume avoidance [12]
- constrained rendezvous region

No known use of MILP for trajectory planning has been demonstrated in space, but Linear Program (LP) solvers have been flown for purposes of jet selection [14].

III. Hardware Description

The SPHERES testbed was developed by the MIT Space Systems Laboratory to incrementally advance the development, validation, and maturation of control, autonomy, and estimation algorithms. It utilizes the unique space environment provided by the International Space Station (ISS) to offer full micro-gravity dynamics and controlled experimental testing. A more detailed discussion of the facility can be found in [7, 15, 16, 17, 18, 19]. Currently, there are three SPHERES aboard the ISS (shown in Figure 3). They are undergoing experiments in spacecraft formation flying, autonomous docking, reconfiguration, and fragmented configurations.

The testbed SPHERES has a face-to-face distance of 0.21 m while its largest diameter is of 0.25 m. It uses twelve cold-gas thrusters positioned around the spacecraft to maneuver in all six degrees of freedom; thus providing translational and attitude control to the system. Each thruster provides a maximum force of 0.12 N resulting in a maximum axial force of 0.24 N and 0.023 Nm of torque. The metrology system consists of ultrasonic (US) receivers and transmitters. They provide time-of-flight data to the computer at a rate of 5Hz. This data from the US metrology system is augmented with three internal gyroscopes sampled at 1kHz. All this data feeds to the estimation algorithm to provide real-time state relative to a global coordinate frame fixed to the ISS. The US metrology encompasses a testing volume of approximately 1.4 m x 0.9 m x 1.2 m.

With regard to autonomous docking experiments, the SPHERES testbed does not have any mechanical docking ports attached, but instead, a velcro system is used for the capture mechanism. Still, the location of the velcro face is referred to as the docking port face of the spacecraft.

IV. Software Implementation

One of the key characteristics of SPHERES is its modularity in the architecture development. Many functions are needed to operate SPHERES. By modularizing the software, the user can focus on the development of one func-
tion, while borrowing existing modules for the other functions [16, 20]. Figure 4 shows a hierarchical view of the architecture. It is based on Fehse’s review of existing automated docking methodologies [21].

In Fig. 4, the components common to Fehse’s architecture are grayed. The changes begin by using the word *autonomy* instead of *automated* as by Fehse. This is for the purpose of letting the satellite have more responsibility and intelligence to operate by itself. The word *docking* is used instead of *rendezvous* as close proximity maneuvers exhibit higher risk and complexity. Moreover, the architecture focuses on the last few meters of the approach. The remote operator line is dashed in this architecture emphasizing intermittent communication with ground. This is compatible with missions to Mars that have a large time delays between ground and the spacecraft of approximately eight minutes. An introduction of a collision avoidance maneuver (CAM) is added that is handled by the on board computer through the Fault Detection Isolation and Recovery (FDIR) module. The interaction with the target satellite is added by either state communication or remote sensing such as visual navigation. Lastly, a solver is introduced such as a mathematical program solver and any other iterative solvers.

An overview of the GN&C architecture in Fig. 4 begins with the box labeled *autonomous onboard docking control systems*, which contains the software modules that run on the satellite. Each software module is an implementation of an algorithm accomplishing a certain task. External interfaces include the plant (satellite), remote operators, and the target satellite. The software modules are grouped into Fault Detection Isolation and Recovery (FDIR), solver, Mission and Vehicle Management (MVM), and Guidance Navigation & Control (GN&C).

Estimation on the SPHERES testbed is performed by an Extended Kalman Filer (EKF) [7, 22] and the control algorithms that currently exist are proportional-derivative (PD) and proportional-integral-derivative (PID) type position and attitude controllers that compute the desired forces and torques. The control inputs are transformed into thruster ON/OFF times using a pulse-width modulator inside the mixer module. Currently, the attitude controllers are executed at 1 Hz and the position controllers at 0.5 Hz.

The FDIR module is active at several levels and linked to multiple modules to autonomously assess any failure such as invalid state estimation from measurements. This is currently implemented in the estimator to reject any bad measurement data above a certain threshold. The FDIR module adds the robustness to perform autonomous docking in the presence of measurement anomalies. For any critical anomalies detected that require an abort, the FDIR module may issue an active or passive abort measure. An active abort consists of a collision avoidance maneuver (CAM) by a pre-computed thruster se-
quence that avoids the target. The passive abort simply shuts-off all thrusters and allows the spacecraft to drift.

The solver module would solve a mathematical program to compute an optimal state trajectory profile for the PD/PID-type controllers to follow. Currently, there are no solvers implemented on-board SPHERES. The final MVM module is the highest autonomy level module that manages the solver, FDdR, and GN&C modes to accomplish a mission objective such as docking to a spacecraft.

When demonstrating the glideslope algorithm, it is applied only along the docking port axis. A PD/PID controller is used to control the alignment transverse of this axis. Since the SPHERES testbed does not currently have an on-board MILP solver, the “safe” trajectory is computed off-line and stored on-board. A PD/PID controller is used to track the safe trajectory.

The GN&C architecture with its encompassing algorithms is used to demonstrate experimental tests of autonomous docking in the presence of anomalies.

V. Results and Comparison

This section covers the results of autonomous docking tests during the SPHERES 5\textsuperscript{th} and 6\textsuperscript{th} test session aboard the ISS in November, 2006 and March, 2007. The docking scenario dynamics consist of a fixed non-rotating and a steady rotating target satellite with its rotation axis perpendicular to its docking port (DP) axis. The rotation results in the DP axis of the target to sweep a plane in which the chaser lies inside. The dynamics of the docking scenarios are depicted in Figure 5.

Results of a total of eight docking tests are presented. Four that use the glideslope algorithm with two tests of a fixed target and the other two of docking to a tumbling target. One of the two tests for each target spacecraft dynamics is performed with a simulated fault executing an active abort. The abort for the glideslope algorithm may be a passive or active abort with the active being a collision avoidance maneuver. Likewise for the four safe trajectory docking tests where two are with a fixed target and another two tests with a tumbling target satellite. The only abort maneuver here is a passive abort by shutting off all thrusters. A summary of the tests are shown in Table 1. The discussion of the results consist of two tests at a time. Both use the same algorithm and have equivalent target spacecraft dynamics. The difference is that one of tests executes an abort maneuver, either active or passive.

Each test is comprised of two phases, the initialization and docking phase. The initialization phase lets the estimator converge and performs maneuvers to achieve the scenario specific initial configuration.

All of the following tests have not been presented before except of tests 1, 3, and 6 that are discussed in Ref. [8, 9]. The results are shown again for the case of comparison between the new experiments.
Table 1: Autonomous docking experiments aboard the ISS in the presence of anomalies.

<table>
<thead>
<tr>
<th>Test</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Glideslope Fixed</td>
<td>glideslope along DP axis</td>
</tr>
<tr>
<td></td>
<td>PD/PID perpendicular</td>
</tr>
<tr>
<td>2. Glideslope Fixed</td>
<td>Glideslope along DP axis</td>
</tr>
<tr>
<td>w/ Fault</td>
<td>PD/PID perpendicular</td>
</tr>
<tr>
<td></td>
<td>CAM</td>
</tr>
<tr>
<td>3. Glideslope Tumble</td>
<td>glideslope along DP axis</td>
</tr>
<tr>
<td></td>
<td>PD/PID perpendicular</td>
</tr>
<tr>
<td>4. Glidescope Tumble</td>
<td>Glideslope along DP axis</td>
</tr>
<tr>
<td>w/ Fault</td>
<td>PD/PID perpendicular</td>
</tr>
<tr>
<td></td>
<td>Passive abort</td>
</tr>
<tr>
<td>5. Safe Traj. Fixed</td>
<td>pre-planned trajectory</td>
</tr>
<tr>
<td></td>
<td>PD/PID tracking</td>
</tr>
<tr>
<td>6. Safe Traj. Fixed</td>
<td>pre-planned trajectory</td>
</tr>
<tr>
<td>w/ Fault</td>
<td>PD/PID tracking</td>
</tr>
<tr>
<td></td>
<td>Passive abort</td>
</tr>
<tr>
<td>7. Safe Traj. Tumble</td>
<td>pre-planned trajectory</td>
</tr>
<tr>
<td></td>
<td>PD/PID tracking</td>
</tr>
<tr>
<td>8. Safe Traj. Tumble</td>
<td>pre-planned trajectory</td>
</tr>
<tr>
<td>w/ Fault</td>
<td>PD/PID tracking</td>
</tr>
<tr>
<td></td>
<td>Passive abort</td>
</tr>
</tbody>
</table>

A. Glideslope Fixed w/ and w/o Fault

This section compares the results of tests 1 and 2 in Table 1. The tests consist of having the chaser spacecraft dock to a fully cooperative target spacecraft that is actively holding its position and attitude. The first test executes this docking scenario using the glideslope algorithm. The results of this test were previously presented in [8, 9]. The second test is identical with the exception of an active abort executed after a simulated fault. The fault is chosen online at a random time during the docking phase. The active abort of the glideslope algorithm during the case of docking to a fixed target is a collision avoidance maneuver. The CAM thrust sequence is pre-computed to be perpendicular to the approach velocity vector.

The initialization phase of both tests consists of having both satellite acquire a navigation solution and orient their docking faces at each other. Next, the docking phase proceeds with the execution of the glideslope algorithm along the DP axis and PD controllers for the two perpendicular axes for tangential alignment. The results of only the docking phase for the two tests is shown Figure 6.

Figure 6: Glideslope docking to a fixed target with and without fault, approach profile.

The plot in Figure 6 shows the separation magnitude between the two spacecraft. The horizontal capture line indicates when the geometric centers are one diameter apart from each other, resulting in contact. The initial separation after the initialization phase of the first test, glideslope docking to a fixed target, is $\approx 0.85$ meters. The approach trajectory afterwards shows a linear approach rate all the way to capture. Although contact did occur, the velcro mechanism did not latch. The video of the experiment and communication from the crew confirmed precise alignment and capture.

The second test began with a smaller separation distance and followed with a similar approach as the first test. However, at approximately 25 seconds, the random simulated fault occurred. This initialized the collision avoidance maneuver by thrusting perpendicular to the approach velocity for 0.9 seconds, a suffi-
cient amount to divert the chaser from collision. As seen in Figure 6, the trajectory after the CAM shows the chaser drift away from the target spacecraft and avoids collision.

These two tests demonstrate the ability of the glideslope algorithm to docking to a fixed target with and without a fault. The active abort maneuver shows good results in the CAM for safely avoiding the target.

B. Glideslope Tumble w/ and w/o Fault

The tests compared in this section are numbers 3 and 4 from Table 1. These tests consist of having the chaser spacecraft dock to a fully cooperative target spacecraft that is actively executing a steady rotation, for the case with and without a simulated fault. The results of the test without fault were also previously presented in [8, 9]. It is repeated here for comparison with the same conditions for the case with a simulated fault at a random time along the docking phase.

To simulate a tumbling target, the target satellite was commanded to maintain an angular rate of -2.25 degrees per second around its z-body axis during the docking approach phase. The use of a constant rate allows comparison with other docking experiments that use a pre-computed approach trajectory. The results of the tests are shown in Figure 7.

The initialization phase of the two tests consisted in both satellites acquiring an absolute navigation fix, pointing at each other, rolling into position and reducing the separation down to ≈ 40 centimeters (performed by the chaser). The target began its constant rotation about its z-body axis. The approach trajectory in Figure 7 shows an initial separation close to 40 centimeters for both tests. For the first test, glideslope docking to a rotating (tumbling) target, the chaser began its approach toward the target while maintaining tangential alignment using PID controllers. The video of the experiment and the data indicates that contact occurred with both satellites well aligned. The test resulted in the first successful autonomous docking to a tumbling target on-orbit.

The second test began a similar approach trajectory until the simulated fault at the randomly chosen time of 11 seconds. The response to the fault in this case was a passive abort instead of an active. This was performed by shutting down all thrusters, knowing that the chaser will keep its tangential velocity and move away from the target without colliding. The separation magnitude plot in Figure 7 and video of the experiment confirm a collision free drift of the chaser successfully avoiding the target.

The two tests demonstrated the first docking to a tumbling target on-orbit using the glideslope algorithm and a successful passive abort maneuver. The same initial configuration of the spacecraft before the start of the docking phase is used for the safe trajectory docking tests, so proper comparison can be observed.

C. Safe Trajectory Fixed w/ and w/o Fault

These tests consist of following a trajectory demonstrating safe rendezvous [4] with a sim-
ulated detection of a failure toward the end of the trajectory. This is called a safe docking because when detection of a failure occurs, the GN&C system responds by shutting off all thrusters, forcing the satellite to enter a drift mode. This drift mode will either cause the chaser to drift past the target or cause the chaser to dock with the target at an acceptable impact velocity. The constraint on the low impact velocity is formulated in Eq. (8) in addition to the terminal position states. For these tests, the final position is defined to be a box in front of the target spacecraft docking port. The time of the simulated failure detection is programmed to occur randomly within the last few seconds of the test, when the optimized trajectory has been guaranteed to be safe in the presence of failures. Since the optimized trajectory is designed a priori, without knowledge of the failure time, this test demonstrates the ability of the safety algorithm to handle arbitrary failures that occur during that window. A similar GN&C architecture, as in the other docking tests, is used. However, a PID control module, replacing the glideslope module, is used to follow a pre-computed safe trajectory. Also, once the initial location of the target is determined, it becomes the target location for both satellites throughout the experiment (and therefore the expected docking location).

The first test does not simulate a failure and is intended to demonstrate docking by following a safe trajectory. The test with the fault is previously presented in Ref. [8]. Results of the experiments is shown in Figure 8.

During the initialization phase of the two tests, both satellites acquired a navigation fix, pointed at each other, and moved to a separation distance of approximately 90 cm (performed by the chaser). The chaser started to follow the pre-computed approach trajectory using a PID controller. The GN&C shutdown occurred at 51 seconds for the test with fault, at which point the chaser disabled its thrusters and entered a drift mode as expected. Video of the experiment showed that the chaser successfully docked 56 seconds into the test, even in the presence of a detected failure. Since a safe trajectory is defined specifically in terms of collision avoidance, and the terminal docking box was not inside the failure avoidance region, the resulting docking, even with disabled thrusters, is consistent with a safe maneuver [13]. Failures in the terminal docking box result in docking, while failures in the avoidance region result in an avoidance maneuver. The test without the simulated fault demonstrated a successful capture at the end of the trajectory.

D. Safe Trajectory Tumble w/ and w/o Fault

These tests consist of following a trajectory demonstrating safe rendezvous [4] towards a steady rotating target satellite with the same initial configuration as in tests 3 and 4, glideslope docking to a rotating (tumbling) target with and without fault. The results of these tests is shown in Figure 9.

During the initialization phase, both satel-
lites acquired a navigation fix, pointed at each other, and moved to a separation distance of approximately 45 cm (performed by the chaser). The chaser started to follow the pre-computed approach trajectory while the target began a steady closed-loop rotation of -2.25 degrees per second. The start of the trajectory following by the PID controller led to a normal approach towards the target spacecraft. The generated trajectory is “exotic” in terms of the complicated constraints it imposes. The start of the increase in the relative separation at about 40 seconds is most likely caused by the plume impingement from the chaser to the target. The target in this case is actively holding its position and thus returns to its fixed location. Afterwards, the chaser began a smooth approach towards the target, see Figure 9. The pre-computed trajectory has a final position state defined to be anywhere within the terminal box in-front of the target docking port. As this trajectory is pre-computed and followed by a PID controller, the tracking must have the performance of following the path with a maximum deviation of less than the required position tolerance for docking port alignment. The PID tracking controller did not exhibit such a performance and the closest separation reach was at ≈ 5 cm. The first test without fault did not achieve a full capture.

In the second test, a failure was introduced at a random step constrained to be in the last five way-points in the rendezvous trajectory. The failure time for this test ended to be at 86 seconds. This failure resulted in the chaser microsatellite disabling all on-board thrusters. Once the thrusters of the chaser were disabled during the rendezvous trajectory, the two spacecraft did not collide, which is consistent with following a passively safe trajectory.

The experiments demonstrated the flexibility of the GN&C architecture presented in Figure 4. In this case, the simple replacement of a GN&C module with another one permitted the testing of an exotic docking scenario. The results of the experiment exemplify the importance of a good trajectory following controller that exhibits small overshoot characteristics.

E. Comparison Review

Two algorithm approaches were used for docking to a fixed and tumbling spacecraft in the presence and without anomalies. The first one is the glideslope algorithm, a simple and robust controller that has been used to successfully demonstrate the first autonomous docking to a tumbling satellite in micro-gravity [6]. The second approach used a pre-computed trajectory that is fuel-optimal, avoids obstacles, and has a “safe” trajectory towards the end of the path that ensures a safe passive abort [4]. Since the glideslope algorithm is simple enough, it is placed on-board the SPHERES testbed and runs online. On the other hand, the safe trajectory is computed offline because a complex MILP solver is not in place on the SPHERES. Therefore, it is taken into account that the experimental test depended on accurate following of the path. Each algorithm approach has several advantage and disadvantages from one
• The glideslope controller is simple enough to compute in milliseconds [7] on the SPHERES testbed. The offline solution to the non-convex MILP formulation for a safe trajectory takes \( \approx 9 \) seconds to compute on a 3 GHz computer [4] with a 20 step safety horizon. This computation load limits the potential use of such a complex planner in a closed-loop manner for the current control rate of SPHERES being 1 Hz. However, the convex formulation in Ref. [4] using an LP solver computes the safe trajectory in 0.17 seconds with a loss in fuel-optimality. By trading off the optimality for speed, a closed-loop implementation of the safe trajectory optimization is possible.

• The glideslope controller does not take into account any fuel minimization nor obstacle avoidance unlike the safe trajectory. Therefore, the glideslope should only be applied once the chaser is aligned with the docking port axis of the target. This limits the use to realistic docking scenarios as any initial configuration of the two spacecraft could occur. This random initial configuration was accounted for during the initialization phase of the experimental tests, as the satellites always pointed towards each other before beginning the docking approach. The work in [23] eliminates this constraint by developing specific docking phases coupled with new control and autonomy algorithms for the SPHERES to dock to complex tumbling satellites.

• Using any pre-computed trajectory requires chaser to have a precise tracking control and modeling of the testing conditions. The computed trajectory has many desirable characteristics, such as being fuel-minimal, collision-free, and ensures a safe passive abort in case of a thruster shutoff. These characteristics are enforced only if the modeled dynamics match the testing conditions and the chaser follows the trajectories’ position and velocities precisely. Otherwise, fuel-cost increases and more importantly, there is a higher potential for a collision with the target. Also, as the final position of the pre-computed trajectory lies to be within the targets’ docking port defined by the terminal box, any large enough deviation from final position would result in either a collision or an unsuccessful capture. Due to these concerns, the SPHERES team recently developed several high-performance tracking controllers in addition to an online path planner with obstacle avoidance [23].

• An active abort with a collision avoidance maneuver in the glideslope case for a fixed target showed higher deviation in the drift from the target than the passive abort maneuver by the safe trajectory. It is also observed that it is easier to implement an abort for docking to a tumbling spacecraft in either of the two algorithms. These aborts consisted of a complete thruster shut-off and letting the tangential velocity drift the chaser away from the target. Also, the formulation used in Ref. [4] allows for safety over multiple orbits, while the precomputed CAM performed in the glideslope approach only allows immediate collision avoidance.

VI. Current and Future Work

The SPHERES team is currently addressing the topic of adding more autonomy to the spacecraft and improving its tracking con-
controllers. The autonomy improvement is made by implementing an online path planner that computes an energy sub-optimal trajectory with obstacle avoidance of the target spacecraft. The latest docking test demonstrating the current improvements is of a fixed target spacecraft that is facing backwards to the chaser. This requires the chaser to plan a trajectory online to maneuver around the target and get in front of its docking port for capture. This test demonstrated the first use of an online path planner by a spacecraft on-orbit. A detailed description of the new algorithms and experimental results is discussed in Ref. [23].

The future work for docking is to continue building on the current GN&C architecture to achieve consistent docking to complex tumbling target spacecraft with the presence of anomalies. The plan to achieve this is by upgrading the algorithms to handle topics such: planning under uncertainty, robust trajectory tracking, calculating the inevitable collision space, and autonomous abort maneuvers.

VII. Summary

This paper presented results and comparisons between two algorithms used for experimental tests in micro-gravity of autonomous docking to a fixed and rotational tumbling satellite in the presence and without anomalies. The first, glideslope algorithm, commands the chaser to reduce its approach velocity with the distance-to-go along a straight path. This can be extended to tumbling target by having other controllers maintain tangential alignment. The second involves a pre-planned safe trajectory that is fuel-optimal, collision-free, and ensures a passive abort in case of a detected failure.

These algorithms were tested experimentally using the SPHERES testbed aboard the ISS. The experiments demonstrated the flexibility of the GN&C architecture presented in Figure 4. In this case, the simple replacement of a GN&C module with another one permitted the testing of exotic docking scenarios (safe trajectory docking). The results demonstrated the algorithms to successfully account for simulated failures and safely escape collision. Both algorithm have their advantages and disadvantages. The glideslope algorithm is computationally efficient and previously demonstrated in-space while the safe trajectory optimization shows desirable characteristics for spacecraft docking. All tests exhibited new challenges in each algorithm that requires further research and validation to reach consistent docking to tumbling satellites in the presence of anomalies.

VIII. Acknowledgments

The authors wish to acknowledge the complete SPHERES team at MIT and Aurora Flight Sciences Corporation, as well as the DoD and its Space Test Programs office, for their funding, launch opportunities, and continued operational support.

IX. Bibliography


[17] Edmund M. C. Kong, Mark O. Hilstad, Simon Nolet, and David W. Miller. Development and verification of algorithms for spacecraft formation flight using the SPHERES testbed: application to TPF. In *Wesley A.


