# GUIDANCE FOR AUTONOMOUS RENDEZVOUS AND DOCKING WITH ENVISAT USING HARDWARE-IN-THE-LOOP SIMULATIONS

**S. Vromen[*], and F.J. de Bruijn[†], E. Mooij[‡]**

In this paper a convex guidance and control algorithm is developed to enable rendezvous and docking with the non-operational, rotating satellite Envisat. The algorithm employs the concept of model predictive control to allow for an unconstrained time until the docking has to be achieved, while still maintaining a guidance algorithm that is computationally efficient enough to be applied in real-time. Tests are performed to evaluate the performance of the MPC, with and without feedback control. These test cases are evaluated using a functional simulator. This is complemented by hardware-in-the-loop tests using the flat-floor test facility, TEAMS. Both the functional and the real-time simulations results show that the developed algorithms enable a performance with sufficient accuracy to complete the operations. It is also demonstrated that a higher accuracy and propellant-efficiency is obtained when model predictive control, without feedback control, is employed.

## INTRODUCTION

Since the beginning of space flight the collision hazard has increased substantially due to the large growth of artificial objects in space. At present, less than ten percent of the Earth-orbiting objects that are tracked are operational satellites. The remainder of the objects is considered to be space debris. Studies have shown that the so-called Kessler effect predicts that collisions between orbiting objects can create a cascade of debris particles and further collisions, effectively rendering an orbital belt unusable. This scenario is not as remote as one might think, which was underlined by recent events, such as the collision of the operational Iridium 33 and the inactive Kosmos 2251 satellites on February 10, 2009.[1]

To address the space-debris problem, many studies are being done on active debris removal. This method uses a dedicated spacecraft to de-orbit debris objects or put them in a graveyard orbit. The focus is on regions with the highest object population density, such as the low-Earth and geostationary orbital regions. Studies have shown that the low-Earth orbit environment could be stabilized if five debris objects with the highest product of collision probability and mass are actively removed every year.[2] The study also shows that large space-debris objects up to an orbital height of 1600 km should be targeted first. The majority of these objects is in near-circular, highly inclined orbits. Envisat, a non-operational, ESA-owned satellite fits this profile and is designated by ESA as a

[*]Master student, Astrodynamics and Space Missions, Delft University of Technology, Kluyverweg 1, 2629HS Delft, The Netherlands.

[†]PhD student, Institute of Space Systems, Deutsches Zentrum für Luft-und Raumfahrt e.V.,Robert-Hooke-Str. 7, 28359 Bremen, Germany.

[‡]Assistant professor, Astrodynamics and Space Missions, Delft University of Technology, Kluyverweg 1, 2629HS Delft, The Netherlands.

high-profile target for such a mission.[3] This satellite is therefore chosen as the target object of this study.

Envisat is a rotating and uncooperative target, which poses many challenges. Successfully docking to a target with these characteristics has never before been accomplished in space. To increase the robustness and safety of the system and to loosen the constraints on communication it is highly desirable to perform these rendezvous and docking operations autonomously. In this study a convex guidance and control algorithm is developed to enable this. The main advantages of guidance based on convex optimization are that convex optimization theory proves that a well-posed convex problem is guaranteed to converge and the obtained solution will be the global optimum. The design will be evaluated on accuracy and the required thrust force to complete the operation.

A key ingredient of the guidance-system development is the capability to test and evaluate the guidance algorithms. Functional simulations are often the first step in this process, therefore a functional simulator is developed to test the performance of the designed guidance and control algorithm. These simulations can be complemented by HIL (Hardware-In-the-Loop) simulations. There are multiple methods that exist for HIL testing of GNC (Guidance, Navigation, and Control) algorithms. To reproduce the six DOFs (Degrees-of-Freedom) kinematics and vehicle dynamics of a spacecraft in a micro-gravity environment to a high degree, neutral buoyancy facilities can be used.[4] These provide an adequate representation of the frictionless environment in orbit.[5] The kinematics of the six DOFs of the relative motion of two spacecraft, can be reproduced by using robotic simulators or cranes.[6] The three DOFs attitude kinematics and torque-free motion can be reproduced by suspending the spacecraft simulator on a hemispherical air bearing, where the center of mass coincides with the center of rotation of the bearing.[7] The method used in this research employs another option that enables the reproduction of the kinematics and vehicle dynamics for three DOFs (one rotational and two translational). The TEAMS (Test Environment for Applications of Multiple Spacecraft) facility of DLR, the German aerospace center, in Bremen, emulates the force and momentum-free dynamics of satellites in orbit, i.e., it reproduces the weightlessness and frictionless environment.[8] The experiments are performed using two free-floating, air-cushion vehicles that move over a surface, which consists of two granite tables with a total test area of 5 m by 4 m. The vehicle dynamics, however, are reduced from orbit dynamics in the functional simulator to a double integrator on TEAMS. But a large advantage that is obtained by tests performed with TEAMS is that it captures the interaction of the GNC algorithms with actual sensors and actuators. Another benefit is that data transmission is also fully reproduced.

The remainder of this paper provides a more detailed discussion on the reference scenario, followed by a description of the applied equations of motion in this scenario. The convex guidance and control algorithms developed to execute the mission are then discussed in-depth. These algorithms form the basis of the constructed functional simulator and the simulator developed to execute tests on TEAMS. A detailed discussion on the development of both simulators and the corresponding test campaign will be provided, after which the obtained results are analyzed. Finally, conclusions are presented in the last section.

## REFERENCE SCENARIO

The reference mission for this research is based on the e.deorbit mission.[3] The e.deorbit concurrent design facility study was the first system level study on active debris removal by ESA. The mission objective of e.deorbit is to safely de-orbit the target object Envisat, an ESA owned object (non-operational satellite). Envisat suffered a major anomaly on 8 April 2012 resulting in a loss of

communication links. It is in a near-polar, near-circular orbit at an altitude of approximately 770 km. The reference mission consists of a chaser satellite that performs a safe and propellant-efficient rendezvous and docking with the target satellite, and de-orbits it. The capturing mechanism will grasp and/or clamp Envisat on the upper side, near the center of mass, where no instruments are situated. This is defined by ESA as the most suitable location to make a connection.

### Assumptions

Radar measurements of Envisat performed at the end of 2013 showed that the main motion of Envisat is a rotation of approximately 3.5 deg/s around its orbital angular momentum vector.[9] On top of this rotation, Envisat is slightly tumbling around its other body axes. There is uncertainty on the future evolution of Envisat's motion and currently ESA is analysing Envisat to enable better predictions.[10] Because of this uncertainty several assumptions have to be made for this research concerning the attitude of the target. It is assumed to be rotating with its spin axis under an angle of 30 deg offset from its orbital angular momentum vector. The spin axis itself is also precessing with a rotational rate of 0.2 deg/s around the orbital angular momentum vector.

The reference scenario is concerned with the final part of the rendezvous operation. The chaser starts at 50 m from the target's center of mass and is aligned with the spin axis. During the approach the alignment with the spin axis shall be maintained. The scenario ends at 3 m from the target's center of mass. It is assumed that the clamping/grasping mechanism will be deployed at this point.

The parameters of the chaser and target satellite that are important for the developed algorithms which will be presented in the next sections are given in Table 1.

**Table 1**: **Important parameters of the target and chaser satellite**

| Parameter | Target | Chaser |
|---|---|---|
| Mass (kg) | 7828 | 1444 |
| Average frontal area ($m^2$) | 38.14 | 4.5 |
| Reflectance factor (-) | 0.3 | 0.3 |
| Drag coefficient (-) | 2.2 | 2.2 |
| Initial semi-major axis (km) | 7146 | - |
| Maximum thrust (N) | - | 44 |

## EQUATIONS OF MOTION

### Reference frames

For the reference scenario two types of reference frames are of interest, the ECI frame, i.e., the Earth-centered J2000 inertial frame, which is used to describe the absolute motion of the satellites. To describe the motion of the chaser relative to the target, the rotating Hill frame is used. The origin of this frame is fixed to the center of mass of the target spacecraft. The +X-axis is aligned with the radius vector of the target, in the direction away from the Earth (R-bar). The +Y-axis lies in the orbital plane, in the direction of the velocity (V-bar) and the +Z-axis completes the right-handed system and lies in the direction of the orbital angular momentum (H-bar).

### Absolute dynamics

The absolute motion of the satellites includes the most important relative perturbations. For this scenario these are the $J_2$-effect of the Earth's gravity field, atmospheric drag based on the NRLMSISE-00 model[11] (constant density assumption) and solar-radiation pressure.

Furthermore, only translational motion is considered, and thus the three rotational degrees of freedom are ignored. This is a common assumption in the development of guidance laws, mainly caused by the fact that the rotational dynamics are much faster. It is thus assumed that the commanded attitude is realized instantaneously.

### Relative dynamics

The guidance laws developed for the mission reference scenario use the Clohessy-Wiltshire equations as the basis for the relative motion model, which are given by:

$$\begin{aligned} \ddot{x} - 2n\dot{y} - 3n^2 x &= a_x \\ \ddot{y} + 2n\dot{x} &= a_y \\ \ddot{z} + n^2 z &= a_z \end{aligned} \tag{1}$$

The applied thrust accelerations in x, y, and z direction of the rotating Hill frame are given by $a_x, a_y$, and $a_z$. Finally, $n$ is the mean motion of the target satellite in the ECI frame.

In state-space form $\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$ this set of equations translates into:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3n^2 & 0 & 0 & 0 & 2n & 0 \\ 0 & 0 & 0 & -2n & 0 & 0 \\ 0 & 0 & -n^2 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1/m_c & 0 & 0 \\ 0 & 1/m_c & 0 \\ 0 & 0 & 1/m_c \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{2}$$

where $T_x, T_y$, and $T_z$ are the thrust forces applied in the x, y, and z direction, respectively, and $m_c$ is the mass of the chaser satellite.

## CONVEX GUIDANCE AND CONTROL

The computational power of on-board computers has increased significantly over recent years and there have been promising developments in optimization theory. This has led to new guidance algorithms rooted in numerical optimization. Convex optimization is capable of delivering globally-optimal solutions, while at the same time constraining the trajectory.[12] The main advantages of guidance based on convex optimization are that 1) convex optimization theory proves that a well-posed convex problem is guaranteed to converge, 2) the obtained solution will be the global optimum, 3) there is a number of different, very efficient solvers for this kind of problem, and 4) constraints and penalties can be imposed. For these reasons convex guidance has been selected as the foundation for the path-planning algorithm developed in this study.

### Problem formulation

The trajectory will be minimized for a weighed combination of propellant use and the error with respect to the reference trajectory. This is translated into the following minimization problem,

$$\begin{aligned} &\text{minimize } \int_0^{t_f} \|\mathbf{T}\|_2 + \gamma\|\mathbf{s}\|_2 \, dt \\ &\text{subject to} \\ &Eq.(1) \\ &\mathbf{x}_{ref} = \mathbf{x} + \mathbf{s} \\ &\|\mathbf{T}\|_2 \leq T_{max} \end{aligned} \tag{3}$$

where $\mathbf{x} = \left(\mathbf{r}^T \, \dot{\mathbf{r}}^T\right)^T$ is the trajectory of the chaser in the rotating Hill frame, $\mathbf{T}$ is the thrust of the chaser, $T_{max}$ is the maximum thrust that can be applied in any direction, $t_f$ is the fixed final time, and $\mathbf{x}_{ref} = \left(\mathbf{r_{ref}}^T \, \dot{\mathbf{r}}_{\mathbf{ref}}^T\right)^T$ is the reference trajectory. The slack variable $\mathbf{s}$ denotes the error with respect to the reference trajectory, which is weighed using $\gamma$ to obtain a solution which is close to the reference trajectory.

The objective function and inequality constraint are norm functions which are proven to be convex functions.[12] The Clohessy-Wiltshire equations are a set of linear, time-invariant differential equations. The resulting problem is thus convex.

## DISCRETIZED PROBLEM FORMULATION

Having established a convex optimization problem, it now has to be discretized to obtain a problem that can be solved using convex programming. The given time interval $t \in [t_0, t_f]$ is divided into a number of steps $N$. Let each time step be denoted by $\Delta t$, then the time at node $k$ is simply described by:

$$t_k = k\Delta t + t_0, \quad k = 0, 1, ...., N \tag{4}$$

The relative motion model is discretized using the first order forward method and is given by Equation (5).

$$\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d\mathbf{u}(k) \tag{5}$$

The discrete matrices $A_d$ and $B_d$ are related to the continuous state-space matrices $A$ and $B$ by:

$$A_d = I_{6\times6} + A\Delta t, \; B_d = B\Delta t \tag{6}$$

which is a simple Euler integration.

The dynamics for subsequent time steps depend on the initial state and the control inputs at all subsequent time steps.[13] This is illustrated in Equation (7).

$$\begin{aligned}\mathbf{x}(k+1) &= A\mathbf{x}(k) + B\mathbf{u}(k)\\ \mathbf{x}(k+2) &= A\mathbf{x}(k+1) + B\mathbf{u}(k+1)\\ &= A^2\mathbf{x}(k) + AB\mathbf{u}(k) + B\mathbf{u}(k+1)\\ \mathbf{x}(k+N) &= A^N\mathbf{x}(k) + A^{N-1}B\mathbf{u}(k) + ... + B\mathbf{u}(k+N-1)\end{aligned} \tag{7}$$

These equations can be combined into one matrix equation:

$$\mathbf{Z}(k) = F\mathbf{x_0} + H\mathbf{U}(k) \tag{8}$$

where $\mathbf{x_k}$ has been denoted by $\mathbf{x_0}$ is the state at the current time step and

$$\mathbf{Z}(k) = \begin{bmatrix}\mathbf{x}(k+1)\\ \mathbf{x}(k+2)\\ \vdots\\ \mathbf{x}(k+N)\end{bmatrix}, \quad \mathbf{U}(k) = \begin{bmatrix}\mathbf{u}(k)\\ \mathbf{u}(k+1)\\ \vdots\\ \mathbf{u}(k+N-1)\end{bmatrix}, \quad F = \begin{bmatrix}A\\ A^2\\ \vdots\\ A^N\end{bmatrix}$$

$$H = \begin{bmatrix}B & 0 & & \\ AB & B & 0 & \\ \vdots & \vdots & \ddots & \\ A^{N-1}B & A^{N-2}B & \cdots & B\end{bmatrix}$$

**Differently sized time steps**

As stated before, the Euler method is used in the discretization. To reduce the error induced by applying the Euler method, it is therefore desirable to use a small time step $\Delta t$. A setback with such a small time step is that it significantly reduces the planning horizon, i.e., $N\Delta t$, for a fixed value of $N$. To compensate for this, two different sizes for $\Delta t$ are implemented. The first several time steps of the planning horizon will be equal to $\Delta t_1$ and the remaining time steps are equal to $\Delta t_2$. The F and H matrix in Equation (8) are then adjusted according to:

$$F = \begin{bmatrix}A_1\\ {A_1}^2\\ \vdots\\ A^{Ns}\\ A_2A_1^{Ns}\\ \vdots\\ A_2^{N-Ns}A_1^{Ns}\end{bmatrix}$$

$$H = \begin{bmatrix} B_1 & 0 & & & \\ A_1 B_1 & B_1 & 0 & & \\ \vdots & \vdots & \ddots & & \\ {A_1}^{Ns-1} B_1 & {A_1}^{Ns-2} B_1 & \cdots & B_1 & 0 \\ A_2 {A_1}^{Ns-1} B_1 & A_2 A_1^{Ns-2} B_1 & \cdots & B_2 & 0 \\ \vdots & \vdots & \vdots & \ddots & \\ {A_2}^{N-Ns} {A_1}^{Ns-1} B_1 & {A_2}^{N-Ns} A_1^{Ns-2} B_1 & {A_2}^{N-Ns} {A_1}^{Ns-3} B_1 & \cdots & B_2 \end{bmatrix}$$

where the number of time steps with size $\Delta t_1$ is denoted by $Ns$ and the $A_1$ and $A_2$ matrices pertain to the updates with time steps $\Delta t_1$ and $\Delta t_2$, respectively. The same holds for the $B_1$ and $B_2$ matrices.

In the optimization a weight matrix $Q$ is introduced that is multiplied with the thrust vector to account for the two different time steps applied in the discretization. The smaller time steps receive a weight proportional to $\frac{\Delta t_1}{\Delta t_2}$.

Having discretized the optimization problem it is now of the format:

$$\begin{aligned} &\text{minimize } \int_0^{t_f} Q\|\mathbf{T}\|_2 + \gamma\|\mathbf{s}\|_2 \, dt \\ &\text{subject to} \\ &\mathbf{Z}(k) = F\mathbf{x_0} + H\mathbf{T}(k) \\ &\mathbf{Z}_{ref} = \mathbf{Z} + \mathbf{s} \\ &\|\mathbf{T}\|_2 \leq T_{max} \end{aligned} \tag{9}$$

where $\mathbf{Z}_{ref}$ is the discrete reference trajectory and $\mathbf{U}(k)$ is substituted by $\mathbf{T}(k)$.

**Implementation and problem solution**

Convex-optimization solvers have reached a high level of maturity over the last decade.[12] It is therefore decided to make use of an existing solver. To solve the guidance problem numerically it has to be transcribed into a format that can be read by a solver. To avoid having to go through the cumbersome process of doing this by hand a modeling language is used for this.

For the functional simulator the modeling language CVX is used. This is a Matlab-based modeling system for convex optimization.[14] CVX turns Matlab into a modelling language, which allows constraints and objectives to be specified using standard Matlab expression syntax. The default solver in CVX is SDPT3. Its performance is sufficient to obtain satisfactory results in this research. SDPT3 is a Matlab implementation of infeasible primal-dual path-following algorithms.[15]

A large disadvantage of using a modeling language like CVX is that it decreases the performance of the optimization. Therefore, the real-time application on TEAMS makes use of CVXGEN. CVXGEN is a software tool that takes a high level description of a convex optimization problem, and automatically generates custom C code that compiles into a reliable, high speed solver for the problem.[16]

## SIMULATION AND TEST CAMPAIGN

### High-level architecture

The simulation programs for both the functional simulator and the TEAMS simulator have been developed using a combination of Matlab and Simulink. The generic high-level architecture of these simulators is portrayed in Figure 1. In the TEAMS simulator the sensors provide information on the state of the vehicle, which is then processed by a navigation filter (Linear Kalman filter). The filter uses this noisy input data to produce a statistically optimal estimate of the state of the target/chaser vehicle. No sensor models or navigation filters are included in the functional simulator. The guidance system then produces a commanded state based on the estimated state provided by the navigation filter (TEAMS) or the state computed by the system dynamics (functional simulator), and an analytically produced reference trajectory, which is included in the guidance algorithm. The guidance system also provides a set of optimized feedforward force commands that correspond to the commanded state. For the functional simulator, control only encompasses tracking (translational motion). The tracking system computes the error between the commanded state provided by the guidance system and the estimated state and provides a set of feedback commands to eliminate this error. The TEAMS simulator also includes attitude control to steer the rotational motion.

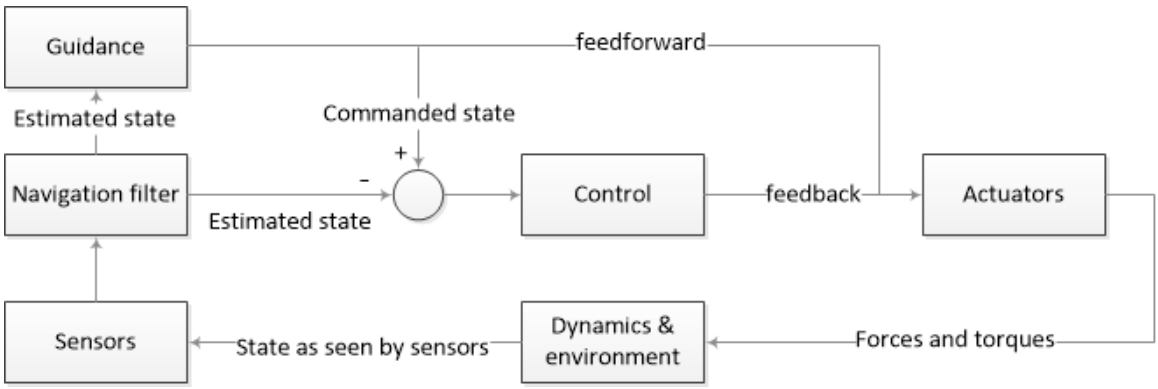


**Figure 1**: **High-level architecture of the simulators.**

### HIL test facility

On TEAMS two 3D air cushion vehicles represent the target and chaser spacecraft, portrayed in Figure 2. These vehicles are able to emulate three degrees of freedom, two translational degrees and one rotational. Beneath each vehicle are three air cushion pads, which create a thin air film on which the vehicles can frictionlessly float. Each vehicle is equipped with a docking adapter, shown in Figure 3, which is mounted so that the two pins of the one can slide into the two holes of the other docking adapter, resulting in a tight fit.

The inertial reference frame of the test area used to represent the dynamics has its origin at the center of the table. The y-axis is directed along the short side of the table and the x-axis along the long side. The dynamics of the vehicles are represented by:

$$\ddot{x} = \frac{T_x}{m},\ \ddot{y} = \frac{T_y}{m},\ \ddot{\theta} = \frac{T_z}{J} \tag{10}$$

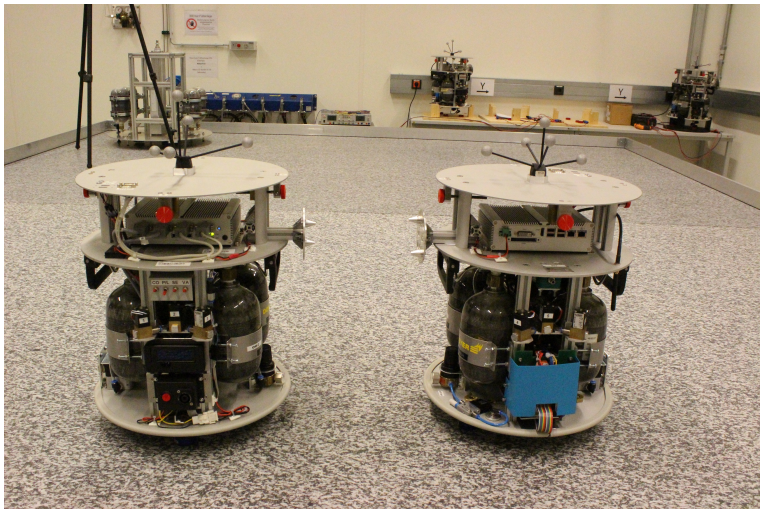

**Figure 2**: **TEAMS facility with two TEAMS 3D air cushion vehicles.**

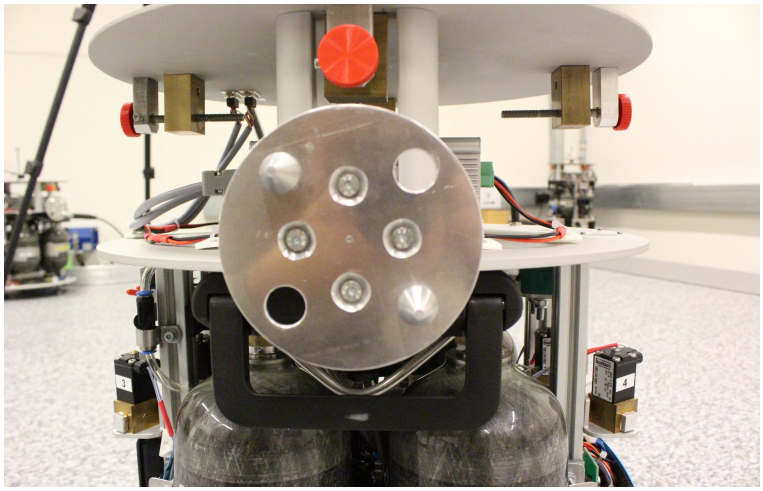

**Figure 3**: **Docking adapter of a TEAMS 3D vehicle.**

where $T_x$ and $T_y$ are the thrust forces in x, and y direction, respectively, $m$ is the mass of the vehicle, $T_z$ is the torque exerted on the vehicle, and $J$ is the mass moment of inertia of the vehicle.

A DTrack infrared tracking system is the main sensor for position and attitude measurements. To control the position and attitude of the vehicles, they are each equipped with proportional cold gas thrusters supported by 6-8 bar pressurized air. The maximum thrust of the chaser vehicle is about 33 mN. The onboard computer runs the QNX real-time operating system. Software can be uploaded and realtime data can be downloaded, displayed, or saved via a WLAN connection. The GNC algorithms designed in this paper are developed using Matlab/Simulink together with Simulink coder for automatic generation of C code, whereas the optimization problem is solved onboard in real-time using code generated by CVXGEN.

**Guidance and control algorithms**

*Reference trajectory generation* As discussed in the reference scenario description, Envisat is an uncooperative, rotating target. The reference trajectory for both simulators is constructed such that during the final approach the docking axis is aligned with the spin axis. The construction of these reference trajectories is different for the functional and TEAMS simulators due to the three DOFs limitation of representing the reference scenario on TEAMS.

The reference trajectory for the approach of the chaser towards the target assumes a straight trajectory along the rotating spin axis with a constant velocity until it is at distance of 3 m from the target. Here it will no longer move towards the target but it will keep tracking the spin axis, this is the point where the clamping mechanism could be deployed. The reference trajectory is therefore produced in a reference frame where the Z-axis is always aligned with the spin axis. This reference frame will from now on be referred to as the spin axis reference frame. The CW equations are defined in the rotating Hill frame and therefore the transformation matrix in Equation (11) is constructed to convert the trajectory from the spin axis frame to the rotating Hill frame.

$$T_{sh} = \begin{bmatrix} \cos(\beta)\cos(\delta(t)) & -\sin(\delta(t)) & -\sin(\beta)\cos(\delta(t)) \\ \cos(\beta)\sin(\delta(t)) & \cos(\delta(t)) & -\sin(\beta)\sin(\delta(t)) \\ \sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \tag{11}$$

Here the angle $\beta$ is the constant offset of the spin axis from the H-bar and $\delta$ is the precession angle of the spin axis, which is equal to $\delta = \omega t$.

The reference trajectory state $\mathbf{x_{ref_{rh}}} = [\mathbf{r}_{rh}\dot{\mathbf{r}}_{rh}]^T$ in the rotating Hill frame is then given by:

$$\mathbf{r}_{rh} = T_{sh}\mathbf{r}_{sa}, \quad \dot{\mathbf{r}}_{rh} = T_{sh}\dot{\mathbf{r}}_{sa} + \boldsymbol{\omega} \times \mathbf{r}_{rh} \tag{12}$$

where $\boldsymbol{\omega} = [0\ 0\ \omega]^T$, and $\mathbf{x_{ref_{sa}}} = [\mathbf{r}_{sa}\dot{\mathbf{r}}_{sa}]^T$ is the reference trajectory state in the spin axis frame. The reference trajectory in the rotating Hill frame is depicted in Figure 4.

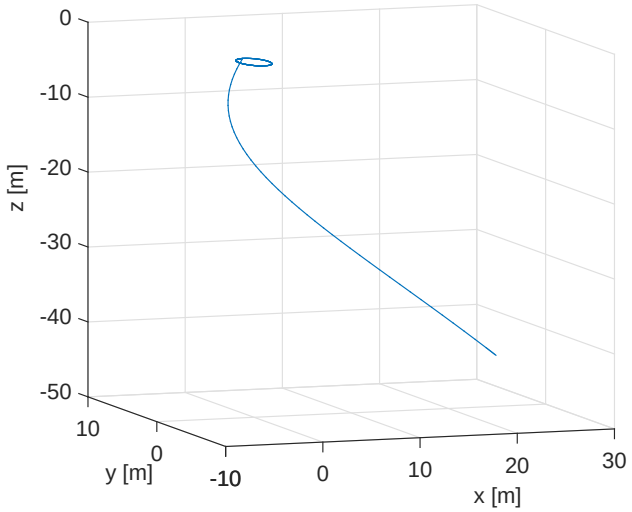


**Figure 4**: **Reference trajectory in the rotating Hill frame.**

The reference trajectory that was defined for the functional simulator has to be adapted to accommodate the 2D environment of TEAMS. The inertial reference frame axes of TEAMS correspond to the axes in the rotating Hill frame as follows: the x-axis and y-axis of the TEAMS frame represent the R-bar and H-bar in the reference scenario. The movement of the spin axis in the R-bar-H-bar plane can be represented by an arc over which the spin axis keeps moving back and forth, representing the precession of the spin axis. The movement of the spin axis is thus translated into a simple rotation of the fixed target vehicle of 0.2 deg/s around its center of mass.

The chaser starts at approximately 1.0 m from the target vehicle on the y-axis of the table. The reference trajectory for the chaser is computed using dynamics relative to the target. The docking adapter of the target is initially aligned with the y-axis of the table.

The construction of the reference trajectory is separated into two phases:

In *phase 1* the chaser moves towards a point at a specific radial distance from the target. The attitude will be altered to ensure alignment of the docking adapters before the second phase commences. This trajectory is depicted in Figure 5.

In *phase 2* the chaser moves in a straight line, in the target-centered, rotating reference frame, towards the target, with a constant velocity, until docking is achieved (left-side of Figure 6). In the target-centered, non-rotating reference frame this will be a trajectory that spirals inward towards the target (right-side of Figure 6).

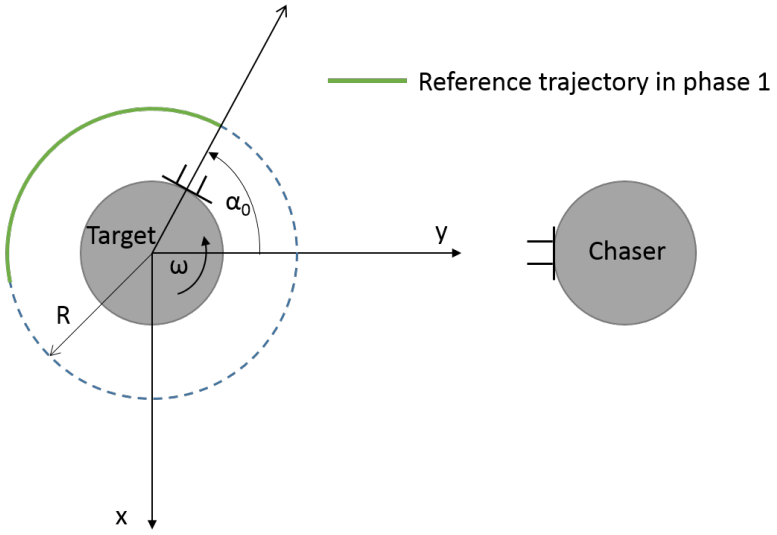


**Figure 5**: **Reference trajectory of the chaser in phase 1 of the approach.**

The reference state trajectory in the first phase is given by:

$$\mathbf{r}_{ref} = \begin{bmatrix} \mathbf{x}_{ref} \\ \mathbf{y}_{ref} \end{bmatrix}, \quad \dot{\mathbf{r}}_{ref} = \begin{bmatrix} -\omega\ \mathbf{y}_{ref} \\ \omega\ \mathbf{x}_{ref} \end{bmatrix} \tag{13}$$

where $\omega$ is the rotational velocity. The rotational velocity vector $\boldsymbol{\alpha}$ and the reference state in x and y direction, denoted by $\mathbf{x}_{ref}$ and $\mathbf{y}_{ref}$ , respectively, are given by:

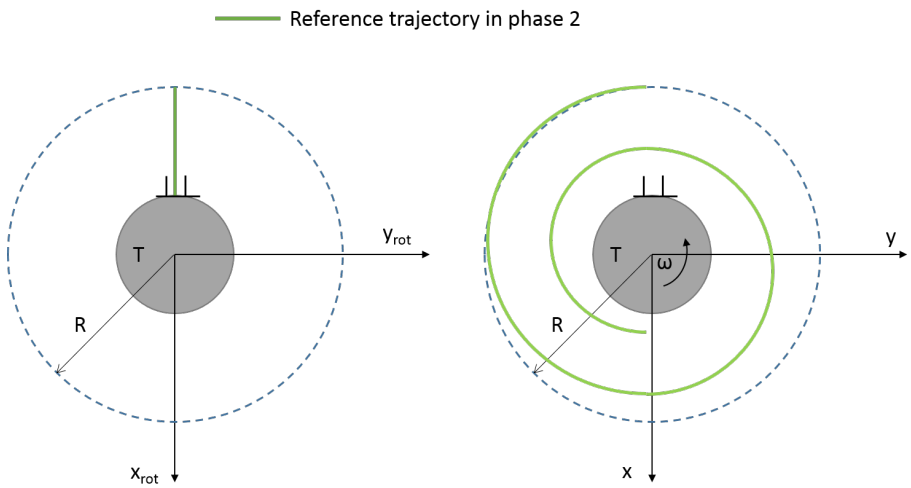


**Figure 6**: **Reference trajectory of the chaser in phase 2 of the approach. Left: target-centered, rotating reference frame. Right:target-centered, non-rotating reference frame.**

$$\begin{aligned} \boldsymbol{\alpha} &= \alpha_0 + \omega\ \mathbf{t} \\ \mathbf{x}_{ref} &= -R\ \sin(\boldsymbol{\alpha}) \\ \mathbf{y}_{ref} &= R\ \cos(\boldsymbol{\alpha}) \end{aligned} \tag{14}$$

The radial distance $R$ and the angle $\alpha_0$ representing the attitude are depicted in Figure 5.

Phase 2 will commence if the the following two conditions are met:

$$\begin{aligned} (x_{ref_0} - x_{ch})^2 + (y_{ref_0} - y_{ch})^2 <= r_{ph2} \\ (\dot{x}_{ref_0} - \dot{x}_{ch})^2 + (\dot{y}_{ref_0} - \dot{y}_{ch})^2 <= \dot{r}_{ph2} \end{aligned} \tag{15}$$

where $x_{ref_0}$ and $y_{ref_0}$ are the coordinates of the starting point of the reference trajectory and $\dot{x}_{ref_0}$ and $\dot{y}_{ref_0}$ the velocities in x and y direction at this point. This means that at every guidance update step it is checked if 1) the chaser is within a radius of $r_{ph2}$ m to the starting point of the reference trajectory of that update step and 2) if the relative velocity of the chaser to the target is less than $\dot{r}_{ph2}$ m/s.

In the rotating, target-centered reference frame the docking adapter is always aligned with the y-axis. The reference trajectory in phase 2 is thus a straight line along this axis. The conversion of the reference trajectory in the rotating reference frame to the non-rotating frame is given by:

$$\begin{aligned} \mathbf{r}_{ref_{inert}} &= \begin{bmatrix} \mathbf{x}_{ref_{inert}} \\ \mathbf{y}_{ref_{inert}} \end{bmatrix} \\ \mathbf{r}_{ref_{inert}} &= T\ \mathbf{r}_{ref_{rot}} \\ \dot{\mathbf{r}}_{ref_{inert}} &= T\ \dot{\mathbf{r}}_{ref_{rot}} + \begin{bmatrix} -\omega\ \mathbf{y}_{ref_{inert}} \\ \omega\ \mathbf{x}_{ref_{inert}} \end{bmatrix} \end{aligned} \tag{16}$$

where the transformation matrix T is given by:

$$T = \begin{bmatrix} \cos(\boldsymbol{\alpha}) & -\sin(\boldsymbol{\alpha}) \\ \sin(\boldsymbol{\alpha}) & \cos(\boldsymbol{\alpha}) \end{bmatrix} \tag{17}$$

Throughout both phases the attitude control of the chaser is given by:

$$\begin{aligned} \alpha &= \alpha_{target} + \pi \\ \omega &= \omega_{target} \end{aligned} \tag{18}$$

where $\alpha_{target}$ and $\omega_{target}$ are the attitude and angular velocity of the target, respectively.

*Generation of the commanded state* The obtained reference trajectory in the rotating Hill frame for the functional simulator is fed into the CVX solver together with the estimated state, which then solves the optimization problem. For the TEAMS simulator the reference trajectory in the non-rotating, target-centered reference frame is fed into the CVXGEN solver together with the estimated state, which then solves the optimization problem in real-time.

To allow for an unconstrained time until the docking has to be achieved while still maintaining a guidance algorithm that is computationally efficient enough to apply in real-time the concept of model predictive control is implemented. The MPC (model predictive controller) uses information on the current state and the relative dynamics to predict how future control accelerations change the system state.[13] Using this information a trajectory will be planned for a certain specified time, i.e., the planning horizon. This planned trajectory will deviate from the actual trajectory due to modeling errors and unmodeled perturbations. To reduce the effect of these errors only a fraction of the plan, i.e, the feedfoward commands are executed, thereafter the guidance function re-initiates the planning algorithm and the process repeats itself. This process is therefore also known as receding horizon control. The MPC is implemented in both simulators.

### Test cases

Depending on the sampling time of the guidance system, the used guidance algorithms, and the accuracy of the dynamical representation of the system, the feedforward commands produced by the MPC can be sufficient to achieve the commanded state without using feedback control. To test this two test cases are defined for both simulators, shown in Table 2. One test case solely uses feedforward commands, whilst the other test case also includes feedback control. Both cases uses sampling rates that are relatively high to decrease the effect of unmodeled perturbations and modeling errors.

**Table 2**: **Test cases to evaluate the performance of the MPC with and without feedback control.**

| **Parameter** | **Functional simulator** | | **TEAMS simulator** | |
|---|---|---|---|---|
| | Case 1 | Case 2 | Case 1 | Case 2 |
| **Feedforward sampling rate [Hz]** | 1 | 1 | 1 | 10 |
| **Feedback sampling rate [Hz]** | 1 | - | 10 | - |

For the TEAMS simulation scenario the rates are increased to the maximum level the real-time system can function at to ensure sufficient performance. This is unnecessary for the functional simulator, for which no extra errors are included.

## ANALYSIS OF RESULTS

The two cases corresponding to the functional simulator have been tested, for which the results are presented in this section. The same holds for the test cases corresponding to the TEAMS simulator. The cases for both the functional simulations and the TEAMS simulations are evaluated on the resulting accuracy and the required thrust. The results obtained with both simulators can naturally not be compared directly to each other, due to the different dynamics and errors involved. The difference in performance between case 1 and case 2 for both simulators can however provide insight into the effectiveness of the two different guidance and control philosophies.

### Functional simulations

Simulations are performed to evaluate the performance of the designed guidance and control algorithm for the two described test cases. The corresponding general simulation parameters can be found in Table 3.

**Table 3**: **General functional simulation parameters, used for both test cases.**

| $\Delta t_1$ (s) | $\Delta t_2$(s) | $N$(-) | $Ns$ (-) | $\beta$ (deg) | $\delta_0$ (deg) |
|---|---|---|---|---|---|
| 1 | 10 | 50 | 10 | 30 | 0 |

Figure 7 shows the actual path and the commanded path of the chaser in the rotating Hill frame for case 1. The figure shows that during the first stage the chaser is attempting to align with the spin axis. The deviant behavior from the reference trajectory shown in Figure 7 is the result from the initial velocity difference between the actual and commanded state, i.e., after the simulation starts the chaser immediately lags behind the spin axis, which has to be corrected for. Once this is accomplished the chaser spirals inwards towards the target, ending in a circular motion in the x-y-plane, in the rotating Hill reference frame, that tracks the spin axis. The chaser displays the same behavior for case 2. The applied $\Delta$V to realize the trajectories up to the final circular motion are given in Table 4, which shows that the total required $\Delta$V is 6.8$\%$ less for case 2.

**Table 4**: **$\Delta$V to actualize the trajectories for both case 1 and case 2.**

| **Parameter** | **Case 1** | **Case 2** |
|---|---|---|
| $\Delta$V feedback (m/s) | 2.25 | - |
| $\Delta$V total (m/s) | 9.15 | 8.53 |

The error between the actual and commanded path for both case 1 and 2 is shown in Figure 8. The first peak in this figure corresponds to the chaser lagging behind the spin axis. The second peak corresponds to the point where the chaser starts the circular motion at 3 m distance in z-direction in the spin axis reference frame. Both peaks are slightly smaller for case 2, indicating less overshoot. After the chaser settles into this motion the error in all directions is for both cases reduced to values in the order of $10^{-5}$m. Given the $\Delta$V results and the lower peaks in Figure 7 it can be concluded that the case that solely applies MPC has a better overall performance than the case that includes feedback control.

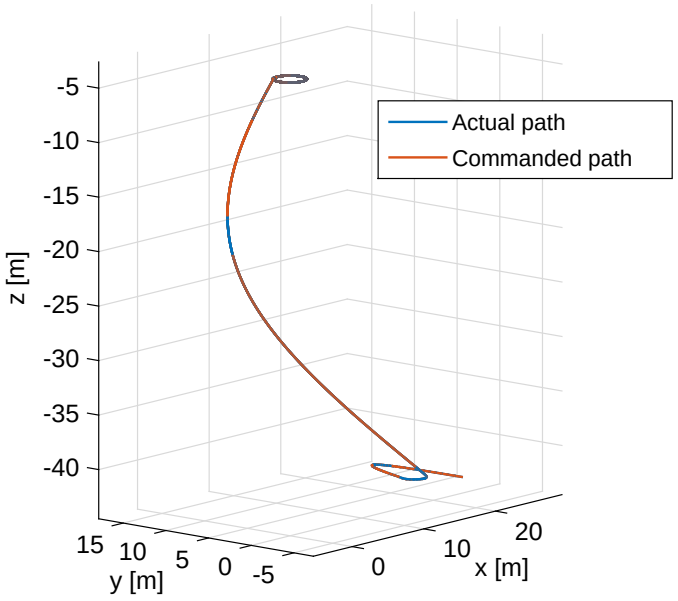


**Figure 7**: **Commanded and actual trajectory of the chaser in the rotating Hill frame for case 1.**

### TEAMS results

The general simulation parameters that are used to perform the tests of the two cases on TEAMS are given in Table 5. The number of steps in the discretization is decreased significantly to decrease the full optimization problem size to enable the use of CVXGEN, which has a limited problem size capability.

**Table 5**: **General TEAMS simulation parameters, used for both test cases.**

| $\Delta t_1$ (s) | $\Delta t_2$ (s) | $N$ (-) | $Ns$ (-) | $\dot{r}_{ph2}$ (m) | $\dot{r}_{ph2}$ (m/s) | $R$ (m) | $\omega$ (deg/s) |
|---|---|---|---|---|---|---|---|
| 1 | 10 | 15 | 2 | 0.05 | 0.01 | 0.75 | 0.2 |

Several reference trajectories that are planned at different times for case 1 and case 2 are shown in Figure 11. They are computed at 1 Hz and 10 Hz (case 1 and case 2), but for clarity only a subset is shown in this figure. Recall that the trajectory depends on the measured rotational velocity of the target vehicle, which is shown for case 1 in Figure 9. This figure shows that the estimator output has a maximum error of about 0.1 rad/s. For case 2 the guidance and control employed by the target is equal to that of case 1 and thus the rotational motion shows similar behavior. A higher measured rotational velocity results in a reference trajectory that encompasses a longer path in total length and it thus assumes that docking will occur at a later point in time. Case 2 shows at $t = 74.9$s that the computed reference trajectory even lies in the opposite direction, which is a result from a measured rotational velocity that has a negative value. To improve this, the estimator output of the target should be tuned further.

The commanded path and the actual path followed are plotted in Figure 12, for both case 1 and

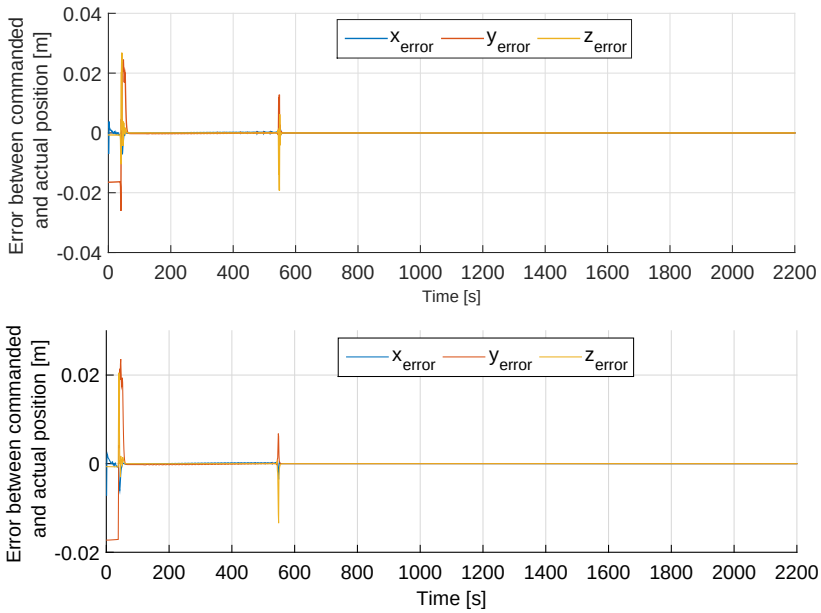


**Figure 8**: **Error between the commanded and actual trajectory of the chaser in the rotating Hill frame for case 1 (top) and case 2 (bottom).**

case 2. These figures also show the path that the center of mass of the chaser vehicle should follow when it is docked, which in both cases coincides with that of the chaser vehicle, thus indicating a successful docking maneuver. From this figure it can also be seen that the error between the commanded path and actual path is for both cases in the order of mm, which is sufficiently small. The figure also shows that a slightly better result is obtained for case 2.

The alignment error during the rendezvous and docking operation is shown in Figure 10 for both case 1 and 2. It is clear from this figure that for case 2 the alignment in phase 2 is better, but in both cases the attitude error in phase 2 was small enough to ensure a successful docking operation.

To gain more insight into the robustness of the guidance and control algorithms three tests are performed for each test case. Each test resulted in a successful docking operation. Table 6 shows the attitude error at docking, the total magnitude of the applied forces, and the time until docking

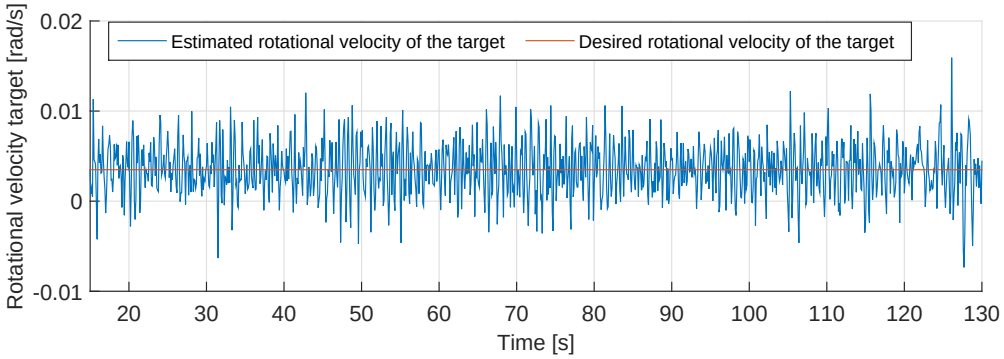


**Figure 9**: **The estimated and commanded rotational velocity of the target vehicle for case 1.**

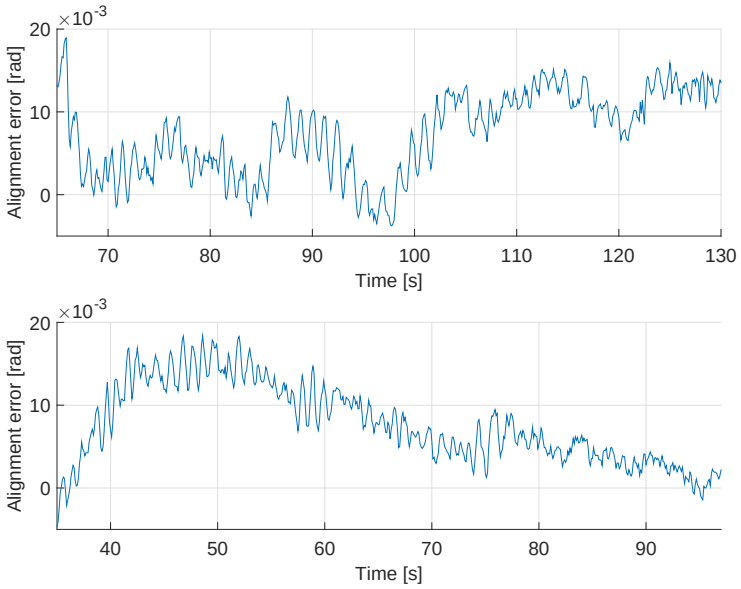


**Figure 10**: **Alignment error for case 1 (top) and case 2 (bottom) in phase 2.**

is achieved, for each test. The initial states of the vehicles vary significantly from each other for each test (order of 0.1 m) and therefore the time until docking and the applied thrust forces can not be compared directly to one another. The multiple tests do, however, provide some insight into the difference in performance between case 1 and case 2. It shows, for example, that case 2 consistently requires less thrust force and time to achieve docking than case 1. To verify that this is always the case more tests should be performed for which the initial conditions are equal.

**Table 6**: **Performance analysis of both cases at docking. Each cases is tested three times.**

| **Parameter** | **Case 1** | | | **Case 2** | | |
|---|---|---|---|---|---|---|
| | Test 1 | Test 2 | Test 3 | Test 1 | Test 2 | Test 3 |
| $att_{error}$ (rad) | 0.0137 | 0.0062 | 0.0083 | 0.00094 | 0.0051 | -0.0084 |
| $T_{fb_x}$ (Ns) | 1.65 | 2.36 | 1.78 | - | - | - |
| $T_{fb_y}$ (Ns) | 1.66 | 1.75 | 1.58 | - | - | - |
| $T_{tot_x}$ (Ns) | 3.55 | 4.53 | 3.58 | 2.50 | 2.63 | 2.43 |
| $T_{tot_y}$ (Ns) | 3.30 | 3.64 | 2.96 | 2.25 | 2.19 | 2.17 |
| $t$ [s] | 131.3 | 162.8 | 127.3 | 97.4 | 107.3 | 93.8 |

## CONCLUSIONS

In this paper a convex guidance and control algorithm was developed to enable rendezvous and docking with a rotating target, which was the non-operational satellite Envisat in the reference scenario. The main advantages of guidance based on convex optimization are that convex optimization theory proves that a well-posed convex problem is guaranteed to converge and the obtained solution will be the global optimum. To allow for an unconstrained time until the docking has to be achieved while still maintaining a guidance algorithm that is computationally efficient enough to apply in

real-time, the concept of model predictive control was also implemented. This is also desirable to increase the safety of the system with respect to unmodeled perturbations and modeling errors.

Depending on the sampling time of the guidance system, the used guidance algorithms, and the accuracy of the dynamical representation of the system, the feedforward commands produced by the MPC can be sufficient to achieve the commanded state without using feedback control. To test this one test case was defined that solely uses feedforward commands, whilst the other test case also includes feedback control. The guidance and control algorithms for the reference scenario were evaluated for these test cases using a functional simulator. The reference scenario was then adapted to enable HIL tests with the flat-floor testing facility TEAMS.

The design was evaluated on accuracy and the required thrust force to complete the operation. The functional simulation results showed that the MPC without feedback control delivered a slightly higher accuracy, while using 6.8% less $\Delta$V. For both cases the reference scenario was completed successfully. The results obtained with TEAMS also show a better performance for the MPC without feedback control, while completing the docking operation in less time. However, it should be noted that the initial states of the vehicles vary significantly from each other for each test on TEAMS (order of 0.1 m) and therefore, the time until docking and the applied thrust forces can not be compared directly to one another. More tests that have equal initial conditions are needed to verify the consistency of the obtained results.

## REFERENCES


[1] N. O. D. Q. News, "Satellite Collision Leaves Significant Debris Clouds," `http://orbitaldebris.jsc.nasa.gov/newsletter/pdfs/ODQNv13i2.pdf`, 2009. Accessed: October 2, 2014.

[2] J. Liou, N. Johnson, and N. Hill, "Controlling the growth of future LEO debris populations with active debris removal," *Acta Astronautica*, Vol. 66, No. 56, 2010, pp. 648 – 653.

[3] ESA, "e.Deorbit Assessment," 2012. CDF Study Report: CDF-135(C).

[4] M. Romano, D. Friedman, and T. Shay, "Laboratory experimentation of autonomous spacecraft approach and docking to a collaborative target," *Journal of Spacecraft and Rockets*, Vol. 44, No. 1, 2007, pp. 164–173.

[5] C. R. Carignan and D. L. Akin, "The reaction stabilization of on-orbit robots," *Control Systems, IEEE*, Vol. 20, No. 6, 2000, pp. 19–33.

[6] H. Benninghoff, T. Boge, and T. Tzschichholz, "Hardware-in-the-loop rendezvous simulation involving an autonomous guidance, navigation and control system," *Advances in the Astronautical Sciences*, Vol. 145, 2012, pp. 953–972.

[7] J. L. Schwartz, M. A. Peck, and C. D. Hall, "Historical review of air-bearing spacecraft simulators," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 4, 2003, pp. 513–522.

[8] M. Schlotterer and S. Theil, "Testbed for on-orbit servicing and formation flying dynamics emulation," *AIAA, editor,"AIAA Guidance, Navigation and Control Conference and Exhibit," AIAA-2010-8108*, 2010.

[9] B. Bastida Virgili, "Investigation on Envisat attitude motion," *e.deorbit symposium*, May 2014.

[10] J. Deloo, "Analysis of the Rendezvous Phase of e. deorbit: Guidance, Communication and Illumination," 2015. MSc. thesis.

[11] C. I. R. Atmosphere, "Models of the Earth's Upper Atmoshpere," 2012. CIRA-2012.

[12] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2009.

[13] F. d. Bruijn and E. Gill, "Influence of sensor and actuator errors on impulsive satellite formation control methods," *Acta Astronautica*, Vol. 94, No. 2, 2014, pp. 608–618.

[14] M. Grant, S. Boyd, and Y. Ye, "CVX: Matlab software for disciplined convex programming," 2008.

[15] R. H. Ttnc, K. C. Toh, and M. J. Todd, "Solving semidefinite-quadratic-linear programs using SDPT3," *MATHEMATICAL PROGRAMMING*, Vol. 95, 2003, pp. 189–217.

[16] J. Mattingley and S. Boyd, "CVXGEN: a code generator for embedded convex optimization," *Optimization and Engineering*, Vol. 13, No. 1, 2012, pp. 1–27.

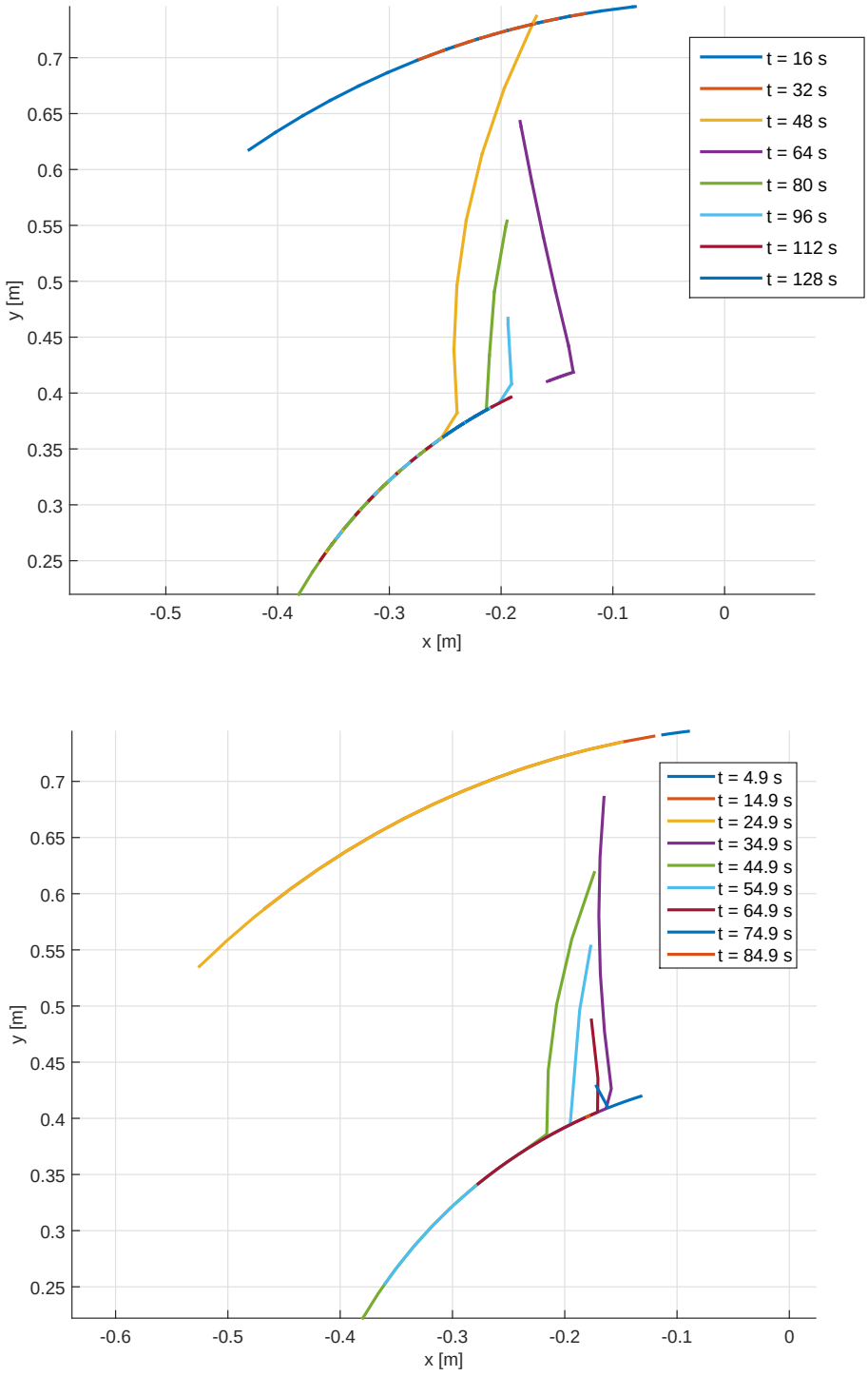


**Figure 11**: **Chaser reference trajectories planned at different time intervals for case 1 (top) and case 2 (bottom).**

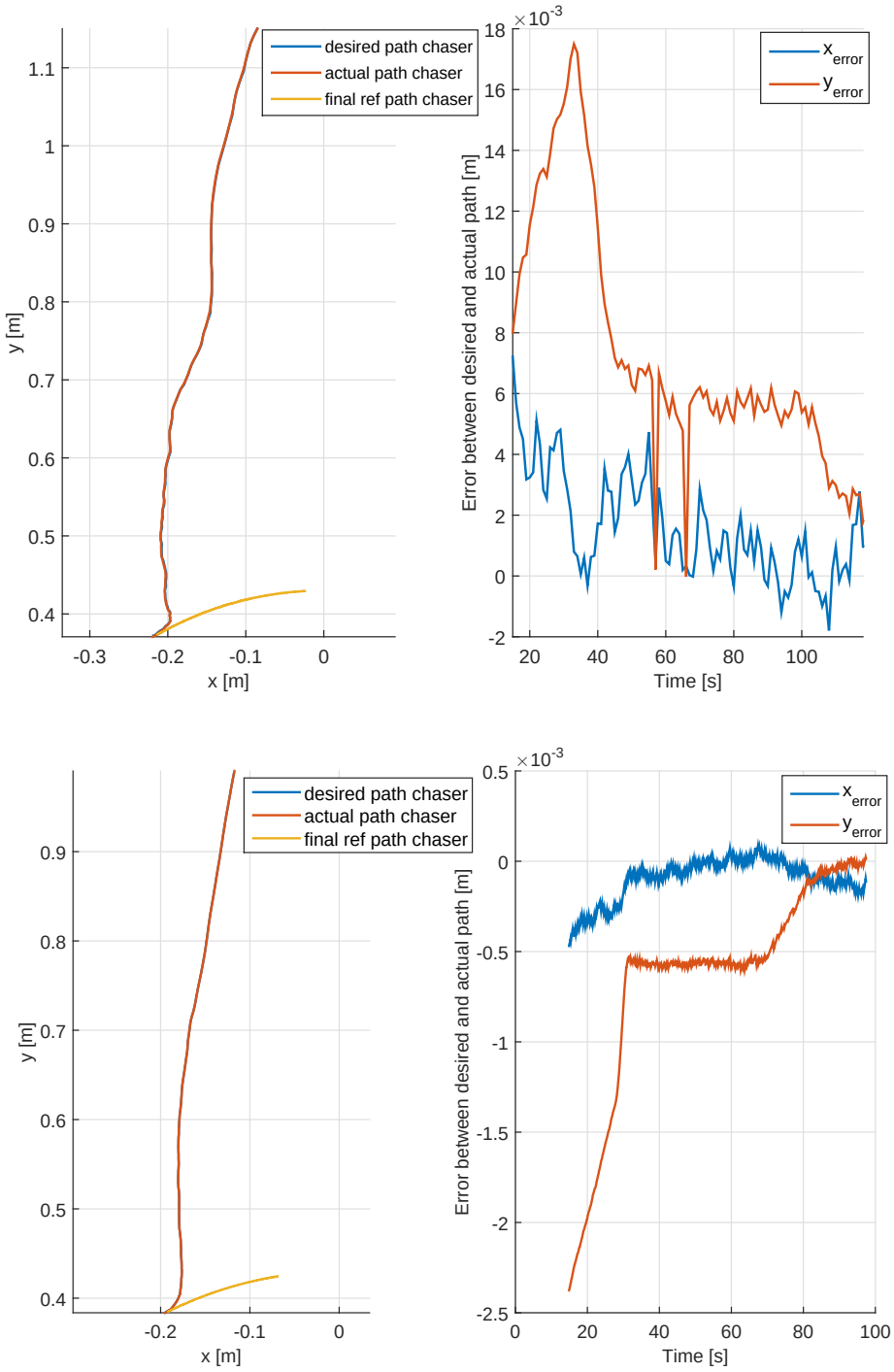


**Figure 12**: **Left: Commanded and actual path of the chaser vehicle and the final path of the reference trajectory of the chaser for case 1 (top) and case 2 (bottom). Right: Error between commanded and actual path of the chaser vehicle in x and y direction for case 1 (top) and case 2 (bottom).**