INTERNSHIP REPORT

Control Barrier Functions

Author:
Lucas Palazzolo

Supervisors:
Jean-Baptiste Pomet
Lamberto Dell'Elce
Jean-Baptiste Caillau

August 20, 2022

Contents

Introduction

The internship topic is based on the paper "High Relative Degree Control Barrier Functions Under Input Constraints" by Joseph Breeden and Dimitra Panagou which deals with optimal control applied to space mechanics. As this course is a continuation of a project, an entire part of it will be the project document with some error corrections.

The problem described in this paper is the analysis of the deployment of large constellations of satellites, so-called "mega constellations", via low thrust propulsion. For this purpose, we are interested in a control system that models the movement of satellites while respecting certain constraints such as keeping them in a safe zone. For this, the use of control barrier functions will be very important. Control barrier functions allow controllers to be designed to ensure safe manoeuvres, i.e. to stay within a safe zone.

The objective of the project was to understand and assimilate the article in particular on the use of control barrier functions to ensure safe manoeuvres. Part of the project was devoted to the numerical implementation of these techniques. For this purpose, a simplified problem was considered, namely a dual integrator with bounds on the maximum control action. Here, a spacecraft that has to modify its trajectory in order to avoid a spherical object has been studied. Finally, using Lyapunov control functions, the study of a satellite approaching a target fixed point while avoiding collision with the sphere was carried out.

For the intership, an improvement of the project was carried out. Indeed, the study was no longer based on a target fixed point but on a target trajectory with the addition of a gravitational term which we considered as null in the project in order to simplify the study. Numerous corrections were also made compared to the draft, such as the modification of the Lyapunov control function or some calculations that were not homogeneous. It is therefore necessary to pay attention to the changes made in relation to the project document. The main objective being the anti-collision of satellites in orbit around a planet or an asteroid, it was necessary to understand how relative motion works in contrast to absolute motion. For this, the reference [4] is very complete. Thus, it only remained to apply the previous work to two satellites orbiting a planet.

1 Introduction to the problem

1.1 Notation

Throughout this work, we will use the same notations as the article. For that, let us define some notations which will be useful for the continuation. Note:

- $\mathcal{C}^r$ the set of functions r times continuously differentiable.
- ∂S the boundary of a set S.
- $\mathcal{K}$ the set of functions where $\alpha : \mathbb{R} \to \mathbb{R} \in \mathcal{K}$ if it is continuous, strictly increasing and $\alpha(0) = 0$.
- $\mathcal{L}_f h(x) = \frac{\partial h}{\partial x} f(x)$ the Lie derivative of an application $h : \mathbb{R}^n \to \mathbb{R}$ in the direction $f : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ at point x.

Remark 1.1.1. *Lie derivatives are essentially used to generalize the notion of directional derivative on manifold. In our case, we place ourselves in $\mathbb{R}^n$.*

If $h : \mathbb{R}^n \to \mathbb{R}$ and $f : \mathbb{R}^n \to \mathbb{R}^n$, we can thus reformulate this derivative as

$$\mathcal{L}_f h(x) = \langle \nabla h(x), f(x) \rangle .$$

If $h : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^{n \times m}$, we reformulate this derivative as

$$\mathcal{L}_g h(x) = \begin{bmatrix} \langle \nabla h(x), g_1(x) \rangle \\ \vdots \\ \langle \nabla h(x), g_m(x) \rangle \end{bmatrix}$$

where $g(x) = (g_1(x), ..., g_m(x))$ and $g_i : \mathbb{R}^n \to \mathbb{R}^n$.

1.2 Problem formulation

Throughout this project, the following control-affine system will be considered

$$\dot{x}(t) = f\left(x(t)\right) + g\left(x(t)\right) u\left(x(t)\right) \tag{1.1}$$

$$\text{where} \begin{cases} x \in \mathbb{R}^n, \\ u \in U \quad \text{compact}, \\ f : \mathbb{R}^n \to \mathbb{R}^n \in \mathcal{C}^r, \\ g : \mathbb{R}^n \to \mathbb{R}^{n \times m} \in \mathcal{C}^r. \end{cases}$$

In the second part, we will reformulate this system to adapt it to our simplified problem. Let $h : \mathbb{R}^n \to \mathbb{R}$, we define the safe set S such that

$$S = \{x \in \mathbb{R}^n \ \mid \ h(x) \leq 0\}. \tag{1.2}$$

We call h a constraint function for the set S. This function allows us to link the dynamic system we are studying to a safe zone in which our machine must be throughout its navigation.

Definition 1.2.1. *For the system* (1.1)*, a function of class $\mathcal{C}^1$ $h : \mathbb{R}^n \to \mathbb{R}$ is a zeroing control barrier function (ZCBF) on S in* (1.2) *if*

$$\exists \alpha \in \mathcal{K} \ \ \textit{such that} \ \inf_{u \in U} \left[\dot{h}(x,u) - \alpha(-h(x))\right] \leq 0 \ \ \forall x \in S \tag{1.3}$$

Here, we define the time derivative by a point above our application. We can start by noticing that $\dot{h}(x,u) = \mathcal{L}_f h(x) + \mathcal{L}_g h(x) u$. Indeed, using the chain rule formula, we get:

$$\begin{aligned} \dot{h}(x,u) &= \frac{\partial h}{\partial x}\dot{x} \\ &= \frac{\partial h}{\partial x}\left[f(x) + g(x)u(x)\right] \\ &= \mathcal{L}_f h(x) + \mathcal{L}_g h(x) u. \end{aligned}$$

Thus, the ZCBF means that we can make the derivative at worst zero on the frontier, when $h(x) = 0$. The fact that the inequality remains true in S ensures that trajectories that start in S remain in S. We allow h to be increasing inside S.

Moreover, we obtain the following lemma which ensures the forward invariance of S, i.e. that all solutions starting from S remain in S. To verify this, it suffices to show that all solutions starting on the boundary fit into S.

Lemma 1.2.1. *If h is a ZCBF on S under* (1.1) *then for any Lipschitzian controller $u \in U$ such that*

$$\dot{h}(x,u) \leq \alpha(-h(x)) \ \ \forall \ \ x \in S$$

will make the set S forward invariant.

Definition 1.2.2. *$h : \mathbb{R}^n \to \mathbb{R}$ is said to be of relative degree r with respect to the dynamics* (1.1) *if*

i) $h \in \mathcal{C}^r$ on an open set O, *ii)* $\mathcal{L}_g\mathcal{L}_f^k h(x) = 0 \quad \forall x \in O \quad \forall k \in [\![0, r-2]\!]$

iii) $\exists C \subseteq \mathbb{R}^n$ *such that* $\mathcal{L}_g\mathcal{L}_f^{r-1} h(x) \neq 0 \quad \forall x \in C$.

Let $\mathcal{G}^r$ be the space of functions of relative degree r satisfying the given system.

Remark 1.2.1. *According to the second point of the definition of a function of relative degree r, a ZCBF is necessarily of relative degree 1. Indeed, we need the first derivative of the ZCBF to depend on the control. We will see later that in the case we are studying, the control will not come out with the first derivative of h but with its second derivative. Thus, we will use a solution proposed by the article and presented in the next section.*

1.3 Other notations

To try to keep it as clear as possible, we change the notations. To do this, we use the following system

$$\begin{cases} \dot{y}(t) = f\left(y(t)\right) + g\left(y(t)\right)u\left(y(t)\right) = F\left(t, y(t)\right) \\ y(0) = x \end{cases} \tag{1.4}$$

and we note the operator $(t, x) \mapsto \varphi_u(t, x)$ as being the value of $y(t)$ resulting from the above system with x as initial value and the control u.

2 General case

In this section, we will describe the general approach to determine a control to approach a target located on an object while avoiding collision. The aim is to apply these results to more concrete cases and thus to be able to carry out numerical simulations.

2.1 The function H

Let's give ourselves a known control that we note u^*. A new function is introduced

$$H(x) = \sup_{t\geq 0} h\left(\varphi_{u^*}(t,x)\right) \tag{2.1}$$

which we seek to make negative or equal to zero. To calculate H numerically, it is therefore necessary to integrate a trajectory for a given initial state and take the maximum of h along this same trajectory.

Remark 2.1.1. *For example, if we take as an initial condition $y(0) = x_0$, a control u and suppose that we have $H(x_0) > 0$ then there exists a t such that $h\left(y(t)\right) > 0$ under u, i.e. u does not make the system safe for x_0.*

If H is reached for a finite time t, we designate the set of times which maximizes $h\left(\varphi_{u^*}\right)$ as

$$t_c(x) = \left\{\operatorname*{argmax}_{t\geq 0} h\left(\varphi_{u^*}(t,x)\right)\right\}$$

which allows us to write in an equivalent way

$$H(x) = h\left(\varphi_{u^*}(t_{c,o},x)\right) \quad \forall t_{c,o} \in t_c(x).$$

Remark 2.1.2. *As in the article, we will assume that H is differentiable.*

Lets

$$S_H = \{x \in \mathbb{R}^n \ \mid \ H(x) \leq 0\} \tag{2.2}$$

which is clearly a subset of S.

Theorem 2.1.1. *The function H defined in* (2.1) *is a ZCBF for the set S_H in* (2.2) *for the control set U provided that $S_H \neq \emptyset$.*

The proof given will be an outline of the proof. A simple case will be considered, where the maximum is unique.

Proof. Let $x \in S_H$, that is $H(x) \leq 0$. By uniqueness of the maximum, we can therefore write

$$t_c(x) = \{t_{c,0}\}$$

and we suppose moreover that h is at least of relative degree 2, i.e. that $h \in \mathcal{G}^r$ with $r \geq 2$. Let us note

$$\tilde{x} : t \mapsto \varphi_{u^*}(t, x).$$

and then we have

$$H(x) = \sup_{t \geq 0} h\left(\varphi_{u^*}(t, x)\right) = \sup_{t \geq 0} h\left(\tilde{x}(t)\right).$$

Let us first consider the case where $t_{c,0} = 0$. We then have

$$\dot{h}(x) \leq 0$$

otherwise the maximum would not be reached in 0 (this is valid for all u in U because $\dot{h}$ is independent of u by relative degree greater than or equal to 2). We have

$$H(x) = h\left(\tilde{x}(0)\right) = h(x)$$

and therefore

$$\dot{H}(x, u) = \dot{h}(x) \leq 0$$

for all u in U and thus in particular for $u = u^*$.

Now consider the case where $t_{c,0} > 0$. It is clear that $\forall \tau \in [0, t_{c,0}]$

$$H\left(\tilde{x}(\tau)\right) = h\left(\tilde{x}(t_{c,0})\right).$$

Since H is differentiable and $Ho\tilde{x}$ is constant on the interval $[0, t_{c,0}]$ then

$$\begin{aligned}\frac{\mathrm{d}H\left(\tilde{x}(\tau)\right)}{\mathrm{d}\tau} &= 0 \\ &= \frac{\partial H}{\partial \tilde{x}} \frac{\partial \tilde{x}}{\partial \tau} \\ &= \dot{H}\left(\tilde{x}(\tau), u^*\right).\end{aligned}$$

This is valid for all τ in $[0, t_{c,0}]$ so in particular for $\tau = 0$. Finally, we have

$$\dot{H}(x, u^*) = 0.$$

We have shown in both cases that there is always a control u which makes $\dot{H}(x, u) = 0$. Moreover, we have $H(x) \leq 0$ and thus for any function $\alpha \in \mathcal{K}$ we obtain

$$\dot{H}(x, u) \leq 0 \leq \alpha\left(-H(x)\right).$$

We have therefore shown that H is a ZCBF. □

The philosophy behind the choice to work with H is that we are looking for a better control than u^* but which appear at the output of a first derivative in order to be able to make the set safe forward invariant. But for this, we will need a previously defined control allowing us to "predict the future". In this paragraph, we will work with

$$\begin{cases} \dot{y}(t) = f\left(y(t)\right) + g\left(y(t)\right) u^*\left(y(t)\right) = F\left(t, y(t)\right) \\ y(0) = x \end{cases}$$

We need to apply the implicit function theorem in order to calculate the Jacobian of H. We note

$$\begin{cases} \varphi(t,x) = y(t) \\ \varphi(0,x) = x \end{cases}$$

and then we have

$$\begin{cases} \frac{\partial \varphi}{\partial t}(t,x) = \dot{y}(t) = F\left(t, y(t)\right) = F\left(t, \varphi(t,x)\right) \\ \varphi(0,x) = x \\ \\ \frac{\partial \varphi}{\partial x}(t,x) = Y(t) \in M_n(\mathbb{R}) \\ \frac{\partial \varphi}{\partial x}(0,x) = I_n = Y(0). \end{cases}$$

On top of that, we have the following differential equation

$$\dot{Y}(t) = \frac{\partial^2 \varphi}{\partial t \partial x}(t,x) = \frac{\partial}{\partial x} F\left(t, \varphi(t,x)\right) = \frac{\partial F}{\partial x}\left(t, \varphi(t,x)\right) \frac{\partial \varphi}{\partial x}(t,x) = \frac{\partial F}{\partial x}\left(t, \varphi(t,x)\right) Y(t)$$

and we obtain the final system

$$\begin{cases} \dot{Y}(t) = \frac{\partial F}{\partial x}\left(t, y(t)\right) Y(t) \\ Y(0) = I_n \\ \\ \dot{y}(t) = f\left(y(t)\right) + g\left(y(t)\right) u^*\left(y(t)\right) = F\left(t, y(t)\right) \\ y(0) = x \end{cases} \tag{2.3}$$

where $\varphi(t,x)$ is replaced by $y(t)$ to simplify.

Moreover, the Jacobian of H is well defined because we assume that H is differentiable. We have the following Jacobian

$$
\begin{aligned}
J_H(x) &= \begin{pmatrix} \frac{\partial H}{\partial r}(x) & \frac{\partial H}{\partial v}(x) \end{pmatrix}^T \\
&= \max_{t \in t_c(x)} \left(\frac{\partial h(y(t))}{\partial x} \right) \\
&= \max_{t \in t_c(x)} \left(\frac{\partial h}{\partial y} \frac{\partial y(t)}{\partial x} \right) \\
&= \max_{t \in t_c(x)} \left(\frac{\partial h}{\partial y} Y(t) \right) \qquad (2.4)
\end{aligned}
$$

where Y is the solution of the system (2.3).

2.2 Control Lyapunov Function

A Lyapunov control function [5] is a function for checking whether a system is asymptotically stabilisable, i.e. whether for any state there is a control such that the system can be brought to the steady state by applying this control. In our case, we will simply try to approach a state.

Definition 2.2.1. *A control-Lyapunov function is a function $V : D \to \mathbb{R}$ that is continuously differentiable, positive-definite (that is $V(x)$ is positive except at $x = 0$ where it is zero), and such that*

$$\forall x \neq 0, \exists u \qquad \dot{V}(x, u) < 0.$$

The important point, according to the definition, is that we are looking for a control u that makes the derivative of V strictly negative and thus decreases V.

Remark 2.2.1. *In the definition, a CLF is presented that allows the state $x = 0$ to be approached by means of feedback. In our case of study, it will be a trajectory that we want to approach and therefore V will depend on time.*

2.3 The desired control

The objective is to be able to find a minimal control $\tilde{u}$ at each time t for a given state x such that one avoids the collision thanks to H while approaching a target thanks to V. Indeed, we have seen from the theorem (2.1.1) that H is a ZCBF. Moreover, if we are able to satisfy the constraint in H, we satisfy the inequality described in the lemma (1.2.1) and thus we make the set S_H forward invariant. As for the constraint on V, it allows us to find a control which makes the derivative of V negative as explained in the previous paragraph. The control function we are looking for here is defined by the following expression:

$$\tilde{u}(x,t) = \underset{(u,\delta)\in C}{\operatorname{argmin}}\, u^T u + k\delta^2 \tag{2.5}$$

such that C is the following set

$$C = \left\{ (u,\delta) \in U \times \mathbb{R} \;\middle|\; \begin{array}{c} \dot{H}(x,u) \leq -H(x) \\ \dot{V}(x,u) + \delta \leq -k_3 V(x,t) \end{array} \right\}.$$

where k et k_3 are constants. If we take x in S_H then by the theorem (2.1.1) there exists a control u satisfying the first inequality because H is a ZCBF. We add a parameter δ in order to try to make the set C non empty. It is necessary to pay attention that the calculation of H is done with the help of the predefined u^* control. This is an tricky important point to understand.

3 Special cases

We will use the theory developed in the article and detailed in the first part on CBFs to apply it to low-gravity spacecraft, in which the spacecraft must navigate around an object under the use of command inputs via a ZCBF.

We note $y = (r, v) \in \mathbb{R}^6$ the state of the satellite where r defines its position and v its speed. According to the second law of Newton, we obtain the following system:

$$\dot{y}(t) = \begin{bmatrix} \dot{r}(t) \\ \dot{v}(t) \end{bmatrix} = \begin{bmatrix} v(t) \\ f_\mu(r(t)) + u(t) \end{bmatrix} \tag{3.1}$$

where f_μ is the local gravitational force and u a control function.

3.1 Spherical object

In our case, we suppose $f_\mu = 0$. So we obtain the following system:

$$\dot{y}(t) = \begin{bmatrix} \dot{r}(t) \\ \dot{v}(t) \end{bmatrix} = \begin{bmatrix} v(t) \\ u(t) \end{bmatrix} = \underbrace{\begin{bmatrix} v(t) \\ 0_3 \end{bmatrix}}_{f(y(t))} + \underbrace{\begin{bmatrix} 0_{3\times 3} \\ I_{3\times 3} \end{bmatrix}}_{g(y(t))} u(t) \tag{3.2}$$

Thus, we come back to the generalized system (1.1) and we are reduced to a double integrator. We start by considering a spherical object of radius ρ_a fixed at the position r_a. Maintaining a distance ρ_s from the sphere is equivalent to maintaining a relative-degree 2 constraint function h less than or equal to 0 :

$$h(y(t)) = \rho - \|r_a - r(t)\|_2 \tag{3.3}$$

where $\rho = \rho_a + \rho_s$.

3.1.1 h is a relative-degree 2 function

Let's prove that the problem is equivalent to satisfying $h(y) \leq 0$ with h a relative-degree 2 constraint function, i.e $h \in \mathcal{G}^2$.

So we have a sphere of center $r_a \in \mathbb{R}^3$ and radius $\rho_a \in \mathbb{R}$ and moreover, we want our spacecraft to be at a distance $\rho_s > 0$ from the sphere at least. Finally, we obtain the inequation:

$$\rho_s \leq \|r_a - r\|_2 - \rho_a.$$

Thus, by putting the h application defined previously, we obtain the following equivalence

$$h(y) \leq 0 \Leftrightarrow \rho \leq \|r_a - r\|_2 \Leftrightarrow \rho_s \leq \|r_a - r\|_2 - \rho_a$$

Now let's prove that $h \in \mathcal{G}^2$.

Let's start by showing that $h \in \mathcal{C}^2$ on an open. The application $x \mapsto \|x\|$ is $\mathcal{C}^\infty$ on $\mathbb{R}^n \backslash \{0\}$. So we deduce that h is $\mathcal{C}^2$ on $\mathbb{R}^6 \backslash \{r_a\}$. Here, r_a is an abuse of notation. Indeed, $r_a \in \mathbb{R}^3$, we should write instead the derivatives of h on $\mathbb{R}^6$ deprived of $\begin{bmatrix} r_a & v \end{bmatrix}^T$ for all $v \in \mathbb{R}^3$.

From the previous paragraph, we can define the gradient of h on the open defined above. So we have $\frac{\partial h}{\partial r} = \frac{2(r_a - r)}{2\|r_a - r\|} = \frac{r_a - r}{\|r_a - r\|}$ and $\frac{\partial h}{\partial v} = 0$. Thus

$$\nabla h(y) = \begin{bmatrix} \frac{\partial h}{\partial r} \\ \frac{\partial h}{\partial v} \end{bmatrix} = \begin{bmatrix} r_a - r \\ 0_3 \end{bmatrix} \frac{1}{\|r_a - r\|}. \tag{3.4}$$

Now we must show that $\mathcal{L}_g \mathcal{L}_f^0 h(y) = \mathcal{L}_g h(y) = 0$ forall y in $\mathbb{R}^6$. According to the remark 1.1.1 and (3.4), we have

$$\mathcal{L}_g h(y) = \begin{bmatrix} \langle \nabla h(y), g_1(y) \rangle \\ \langle \nabla h(y), g_2(y) \rangle \\ \langle \nabla h(y), g_3(y) \rangle \end{bmatrix} = \frac{\partial h}{\partial v} = 0_3$$

where $g(y) = (g_1(y), g_2(y), g_3(y))$.

Finally, it only remains to show that there exists C in $\mathbb{R}^6$ such that $\mathcal{L}_g \mathcal{L}_f^1 h(y) \neq 0$ for all y in C. Let's start by writing $\mathcal{L}_f h$ and its gradient.

$$\begin{aligned} \mathcal{L}_f h(y) &= \langle \nabla h(y), f(y) \rangle \\ &= \frac{1}{\|r_a - r\|} \left[(r_{a_x} - r_x) v_x + (r_{a_y} - r_y) v_y + (r_{a_z} - r_z) v_z \right] \end{aligned}$$

This gives the following derivatives $\frac{\partial \mathcal{L}_f h}{\partial r} = -\frac{v}{\|r_a - r\|^2} \left[\|r_a - r\| + \frac{(r_a - r)^2}{\|r_a - r\|} \right]$ and $\frac{\partial \mathcal{L}_f h}{\partial v} = \frac{r_a - r}{\|r_a - r\|}$ and we obtain

$$\nabla \mathcal{L}_f h = \begin{bmatrix} \frac{\partial \mathcal{L}_f h}{\partial r} \\ \frac{\partial \mathcal{L}_f h}{\partial v} \end{bmatrix}.$$

We can now calculate the Lie derivative of $\mathcal{L}_f h$ in the g direction:

$$\begin{aligned}
\mathcal{L}_g\mathcal{L}_f h(y) &= \begin{bmatrix} \langle \nabla \mathcal{L}_f h(y), g_1(y)\rangle \\ \langle \nabla \mathcal{L}_f h(y), g_2(y)\rangle \\ \langle \nabla \mathcal{L}_f h(y), g_3(y)\rangle \end{bmatrix} = \frac{\partial \mathcal{L}_f h}{\partial v} \\
&= \begin{bmatrix} r_{a_x} - r_x \\ r_{a_y} - r_y \\ r_{a_z} - r_z \end{bmatrix} \frac{1}{\|r_a - r\|} \neq 0 \quad \forall y \in C
\end{aligned}$$

where $C = \{[r_a, v]^T \mid v \in \mathbb{R}^3\}$.

Finally, we have shown that the h application defined in our problem about the spherical object is indeed relative degree 2 with respect to the dynamics (3.2).

3.1.2 Some sets and independence of $\dot{h}$

The following safe area can therefore be defined

$$S = \{y \in \mathbb{R}^6 \;\mid\; h(y) \leq 0\}.$$

In the rest of the paragraph, we will consider

$$U = \{u \in \mathbb{R}^3 \;\mid\; \|u\|_2 \leq u_{max}\}$$

where u_{max} will characterize the maximum acceleration of our satellite.

However, we cannot apply the lemma 1.2.1 because $\dot{h}$ does not depend on the control u. Indeed, we have

$$\begin{aligned}
\dot{h}(y,u) &= \mathcal{L}_f h(y) + \mathcal{L}_g h(y) u \\
&= \mathcal{L}_f h(y) \quad \text{because} \quad h \in \mathcal{G}^2
\end{aligned}$$

and therefore $\dot{h}(y,u) = \dot{h}(y)$.

3.1.3 Control u_{ball}

As proposed in the article, we will try to minimize the second derivative of h which depends on u. Let us calculate this second derivative

$$\begin{aligned}
\ddot{h}(y,u) &= \frac{\partial \dot{h}}{\partial y}\frac{\partial y}{\partial t} \\
&= \mathcal{L}_f\mathcal{L}_f h(y) + \mathcal{L}_g\mathcal{L}_f h(y) u(y).
\end{aligned}$$

We can therefore define the following control

$$u_{ball}(y) = \underset{u \in U}{\operatorname{argmin}} \, \mathcal{L}_g \mathcal{L}_f h(y) u \tag{3.5}$$

which minimises $\ddot{h}$. It is possible to determine the explicit expression of u_{ball} :

Let $a = \frac{\partial \mathcal{L}_f h}{\partial v}$ and $\|a\|^2 = \frac{1}{\|r_a - r\|^2} \left[(r_{a_x} - r_x)^2 + (r_{a_y} - r_y)^2 + (r_{a_z} - r_z)^2 \right] = 1$ Therefore, we have

$$u_{ball}(y) = \underset{u \in U}{\operatorname{argmin}} \, \mathcal{L}_g \mathcal{L}_f h(y) u = \underset{u \in U}{\operatorname{argmin}} \begin{bmatrix} \frac{\partial \mathcal{L}_f h}{\partial v_x} u_x \\ \frac{\partial \mathcal{L}_f h}{\partial v_y} u_y \\ \frac{\partial \mathcal{L}_f h}{\partial v_z} u_z \end{bmatrix}$$

Thus, it would be necessary to take $\tilde{u} = -a$ to minimize the preceding quantity. However $\tilde{u}$ does not belong to U which is a ball of radius u_{max}. To respect this condition, we take

$$u_{ball} = -a \frac{u_{max}}{\|a\|} = -a \times u_{max} = -\frac{u_{max}}{\|r_a - r\|} \begin{bmatrix} r_{a_x} - r_x \\ r_{a_y} - r_y \\ r_{a_z} - r_z \end{bmatrix}. \tag{3.6}$$

Now that we have determined our control, we will apply it for different initial conditions. It can be seen that the chosen control amounts to pushing the satellite "perpendicular" to the sphere. It is said in the paper that for this particular combination of dynamics (3.2) and control law (3.5), there is always a unique maximizer time. There is one case that must be taken into account when finding the numerical solution: the case where h is strictly decreasing from the start. In this case, the satellite is already moving away from the obstacle. There is therefore no point in modelling its trajectory and we can simply state that it is in the safe zone.

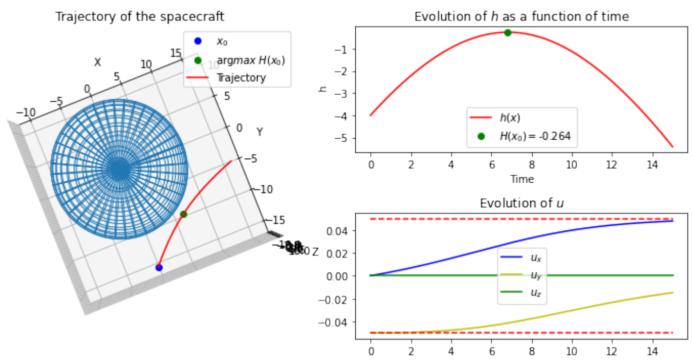

Figure 2: $v_0 = [0.9, 1, 0]$

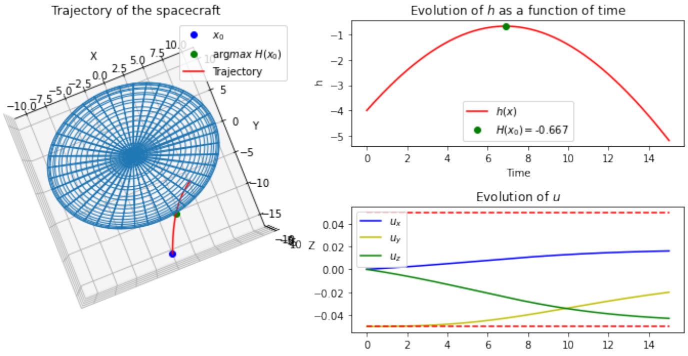

Figure 3: $v_0 = [0.3, 0.9, -0.8]$

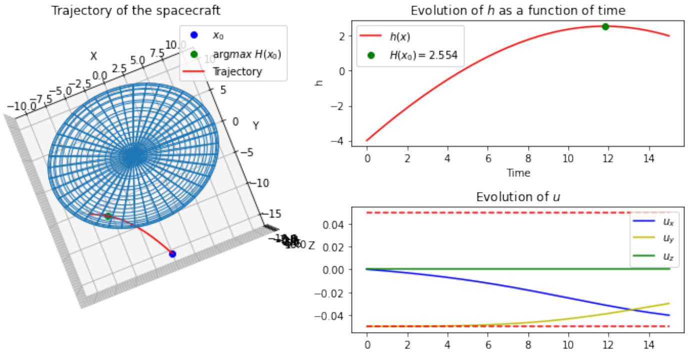

Figure 4: $v_0 = [-0.4, 1, 0]$

Beware, whether on these simulations or the next ones, the graphs of the u control are distorted. Indeed, the `solve_ivp` solver of python does not necessarily go through the time in an increasing order and can go backwards. Thus, by constructing u, one obtains discontinuities which are not expected especially for the case of u_{ball}.

3.1.4 Sketch of H

Let us place ourselves at a fixed state (t, x) where $t \in \mathbb{R}_+$ and $x = (r, v) \in \mathbb{R}^6$. The idea behind the application H is that in the case where we move away from the obstacle, r is the closest point to the obstacle so we take

$$H(x) = h(r).$$

In the other case, we get closer to the obstacle and we have to solve the double integrator (1.4) with u^* to know the closest point to the obstacle which we call

r_{min}. We therefore take here

$$H(x) = h(r_{min}).$$

The figure below illustrates this approach.

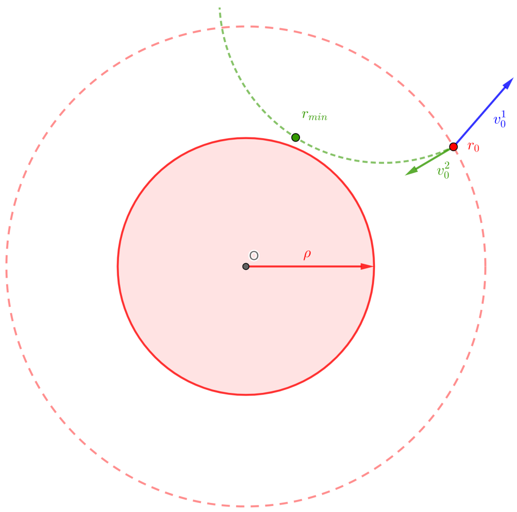

Figure 5: Illustration of H

To visualise the behaviour of H, we can draw its level lines. Let $c \in \mathbb{R}_-$, let us seek the set of $x = (r_0, v_0)$ such that $H(x) = c$. Let us start in the 1D case and place ourselves in the region where $r_0 \geq 0$. The reasoning is identical in the region where $r_0 \neq 0$. If we move away from the obstacle, i.e. if $v_0 \geq 0$, then $H(x) = h(r_0) = c$ and therefore

$$r_0 = \rho - c.$$

Now we approach the obstacle, i.e. $v_0 \leq 0$. In 1D, we can easily solve the double integrator. Indeed, we have

$$\begin{cases} \dot{r} = v \\ \dot{v} = u_{ball}(r) = u_{max}\frac{r}{|r|} = u_{max} \end{cases}$$

because $r > 0$. So

$$r(t)) = r_0 + v_0 t + \frac{1}{2}u_{max}t^2$$

with $t_{min} = -\frac{v}{u_{max}}$. We obtain

$$r_{min} - r_0 + \frac{1}{2u_{max}}v_0^2$$

which corresponds to the equation of a parabola. We can therefore easily draw a sketch of a contour line of H as in the first picture.

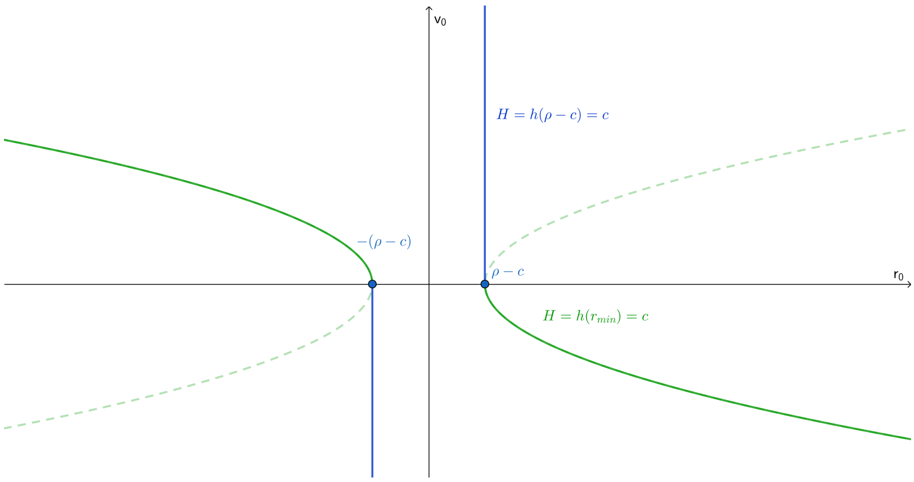

Figure 6: Sketch of a contour line

Furthermore, without difficulty, it is possible to display different contour lines of H using python as shown in the second graph with $H = 0$ in red.

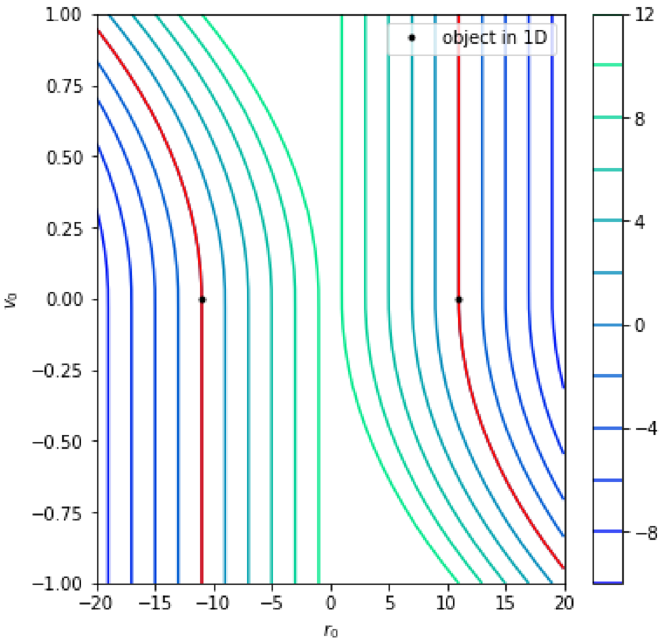

Figure 7: Contour lines with python

Let's look at the 2D case. To do this, we take a section where $z = 0$ and we set a velocity $v_0 = (v_x, v_y)$ where $v_x < 0$ and $v_y > 0$. The reasoning is exactly the same for choosing another v_0. If we move away from the obstacle, then as before,

$$H(x) = h(r_0).$$

If we get closer then the case becomes much more difficult to explain. However, it is easy to interpret. The more the velocity "points" towards the centre of the sphere, the further away we have to go to stop. Thus, we obtain the following figure.

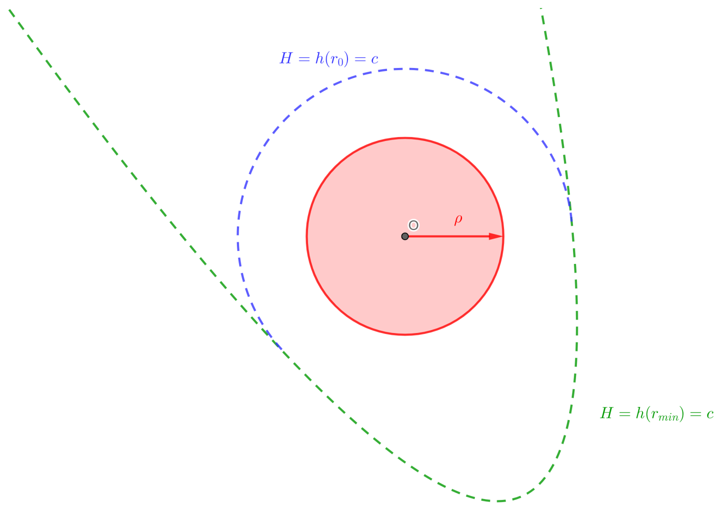

Figure 8: Sketch of a contour line

Similarly, without too much difficulty one can draw the different level lines of H using python. The level line $H = 0$ is drawn in red.

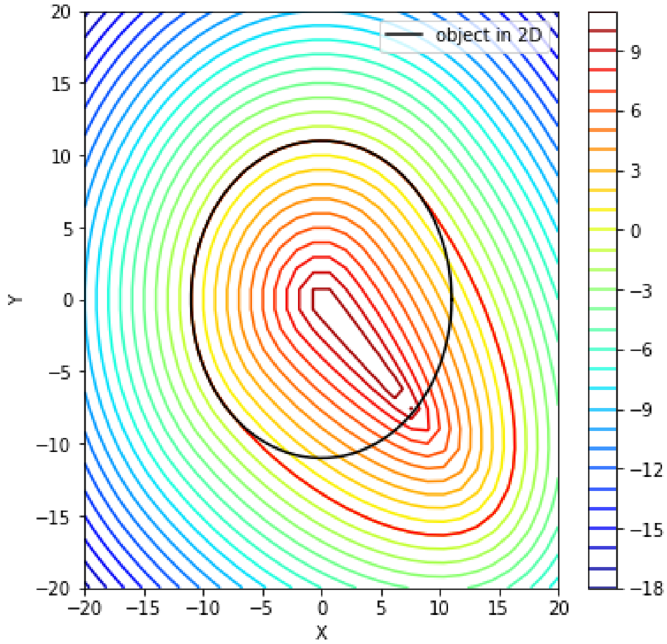

Figure 9: Contour lines with python

3.2 Spherical object with point to be observed

In this paragraph, take the same obstacle as before and therefore the same function h. However, the problem is different here. Indeed, we want to approach a particular point on the spherical obstacle, which we note r_p, while keeping a certain safety distance. In order to be able to act on the speed of our satellite according to the speed of the trajectory to follow, it is necessary to add the term $\dot{r}_p$ to the CLF proposed in the article. We thus use

$$V(y,t) = \frac{1}{2}\|r - r_p(t)\|_2^2 + \frac{1}{2}k_2\|v - \dot{r}_p(t) - k_1(r - r_p(t))\|_2^2. \tag{3.7}$$

with k_1 and k_2 as constants. Here, as r_p corresponds to a static point, it does not depend on time.

Let us start by showing that (3.7) is indeed a CLF. We have that V is $\mathcal{C}^1$ by the continuous differentiability of $\| \ . \ \|^2$ and that V is positive definite. It only remains to show the last point. We have

$$
\begin{aligned}
\dot{V}(y,t,u) =& \frac{\partial V}{\partial t} = \frac{\partial V}{\partial y}\dot{y} + \frac{\partial V}{\partial r_p}\dot{r}_p + \frac{\partial V}{\partial \dot{r}_p}\ddot{r}_p \\
=& \mathcal{L}_f V + \mathcal{L}_g V u + \langle -(r-r_p) + k_2 k_1 \left(v - \dot{r}_p - k_1(r-r_p)\right), \dot{r}_p \rangle \\
& - \langle k_2(v - \dot{r}_p - k_1(r-r_p)), \ddot{r}_p \rangle \\
=& \langle r - r_p, v - \dot{r}_p \rangle - k_1 k_2 \langle v - \dot{r}_p - k_1(r-r_p)), v - \dot{r}_p \rangle + k_2 \langle v - \dot{r}_p - k_1(r-r_p)), u - \ddot{r}_p \rangle
\end{aligned}
$$

For all y different from the trajectory r_p, we take

$$
u = k_1(v - r_p) - \frac{1}{k_2}(r - r_p) + k_1\left(v - \dot{r}_p - k_1(r-r_p)\right) + \ddot{r}_p
$$

By injecting it into $\dot{V}$, we obtain

$$
\dot{\tilde{V}}(y,t) = k_1\|r - r_p\|^2 + k_1 k_2 \|v - \dot{r}_p - k_1(r-r_p)\|^2 < 0.
$$

where $k_1 < 0$ and $k_2 > 0$. So we have that V is a CLF.

Since here the target is a fixed point, we have $\dot{r}p = 0$ and so we take

$$
V(y,t) = \frac{1}{2}\|r - r_p(t)\|_2^2 + \frac{1}{2}k_2\|v - k_1(r - r_p(t))\|_2^2.
$$

The control function we are looking for here is defined by the following expression:

$$
\begin{aligned}
\tilde{u}(x,t) &= \underset{(u,\delta)\in C}{\operatorname{argmin}}\, u^T u + k\delta^2 \\
&= \underset{(u,\delta)\in C}{\operatorname{argmin}}\, \|(u, \sqrt{k}\delta) - (0_3, 0)\|_2^2
\end{aligned}
\tag{3.8}
$$

such that C is the following set

$$
C = \left\{ (u,\delta) \in \mathbb{R}^3 \times \mathbb{R} \;\middle|\; \begin{array}{c} \mathcal{L}_f H(x) + \mathcal{L}_g H(x) u + H(x) \leq 0 \\ \mathcal{L}_f V(x,t) + \mathcal{L}_g V(x,t) u + \delta + k_3 V(x,t) \leq 0 \\ \|u\|_2 - u_{max} \leq 0 \end{array} \right\}.
$$

We can start by noticing that the minimization problem is the same as looking for the projection of the null vector on the set C. It can be described as the intersection between the two lower parts of two hyperplanes and a cylinder. Let's write these 3 applications that define C

$$
\begin{aligned}
G^1_{x,t}(u,\delta) &= \mathcal{L}_f H(x) + \mathcal{L}_g H(x) u + H(x) = \dot{H}(x,u) + H(x) \\
G^2_{x,t}(u,\delta) &= \mathcal{L}_f V(x,t) + \mathcal{L}_g V(x,t) v + \delta k_3 V(x,t) = \dot{V}(x,t,u) + \delta + k_3 V(x,t) \\
G^3_{x,t}(u,\delta) &= \|u\|_2^2 - u_{max}^2
\end{aligned}
$$

and note the constraint function G such that

$$G_{x,t}(u,\delta) = \left(G^1_{x,t}(u,\delta), G^2_{x,t}(u,\delta), G^3_{x,t}(u,\delta)\right).$$

However, this is where it becomes a little less intuitive. The resulting solution is the control to be applied at time t to the spacecraft at position x. Thus, for the first inequality constraint, we seek to "predict the future" by using x as the initial condition in the various calculations where H intervenes. As for the second inequality, we are looking at a kind of distance between our satellite at position x at time t and the point to be approached. The aim is to make the derivative of H and V negative in order to make H and V decreasing.

3.2.1 Existence and uniqueness of the solution

We will show here that there is a unique solution to this minimisation problem.

First, let us show that C is convex. The lower parts of the hyperplanes are convex. Indeed, we can rewrite our inequation as $L\left((u,\delta)^T\right) + c \leq 0$ with L a linear operator and c a constant. We thus clearly obtain convexity. Moreover, the cylinder is also convex.

Thus, C is an intersection of convexes and is therefore also convex.

Now, let us show that C is a closure. We can write C such that

$$C = G^1_x(]-\infty, 0]) \cap G^2_x(]-\infty, 0]) \cap G^3_x(]-\infty, 0])$$

with G^i_x continuous applications. Thus, C is an intersection of closures and is therefore also closed.

We therefore look for the projection of the null vector onto the set C which is a non-empty closed convex in the vector space $\mathbb{R}^4$. Let us suppose that C is non-empty even if we will see it later, according to the constants k_i chosen and the initial conditions, we can find the opposite case. According to the projection theorem, there is a unique solution u in C.

3.2.2 Constraints

In this section, we will explicitly define the various constraints that define C.

3.2.2.1 Hyperplane defined by H

The H hyperplane is one of the trickiest parts of the subject. In the following, we will work with $u^* = u_{ball}$.

According to (2.3) and (2.4), we must solve the following system

$$\begin{cases} \dot{Y}(t) = \frac{\partial F}{\partial x}\,(t, y(t))\,Y(t) \\ Y(0) = I_n \\ \\ \dot{y}(t) = f\,(y(t)) + g\,(y(t))\,u_{ball}\,(y(t)) = F\,(t, y(t)) \\ y(0) = x \end{cases}$$

with

$$J_H(x) = \max_{t \in t_c(x)} \left(\nabla h(y(t)) Y(t)\right).$$

In order to solve our system, we need to determine $\frac{\partial F}{\partial x}$. To begin with, we write F explicitly

$$F(t, y(t)) = f(y(t)) + g(y(t)) u_{ball}(y(t)) = [v, u_{ball}]^T$$

and so we get

$$\frac{\partial F}{\partial x} = \begin{bmatrix} \frac{\partial v}{\partial r} & \frac{\partial v}{\partial v} \\ \frac{\partial u_{ball}}{\partial r} & \frac{\partial u_{ball}}{\partial v} \end{bmatrix} = \begin{bmatrix} 0_3 & I_3 \\ u_{max}\left(\frac{1}{\|r_a - r\|} I_3 - \frac{1}{\|r_a - r\|^3}(r_a - r)(r_a - r)^T\right) & 0_3 \end{bmatrix}.$$

The final result is thus

$$\begin{aligned} \dot{H}(x, u) &= \mathcal{L}_f H(x) + \mathcal{L}_g H(x) u \\ &= \langle J_H(x), f(x) \rangle + \left\langle \frac{\partial H}{\partial v}(x), u \right\rangle \end{aligned}$$

which allows us to characterize the lower part of the hyperplane defined by H

$$G^1_{x,t}(u, \delta) = \mathcal{L}_f H(x) + \mathcal{L}_g H(x) u + H(x) \leq 0.$$

3.2.2.2 Hyperplane defined by V

Now that you've done the hard part, the rest becomes rather trivial. To define the convex defined by V, it is enough to calculate its partial derivatives

$$\begin{aligned} \frac{\partial V}{\partial r} &= (r - r_p) - k_2 k_1 \left(v - k_1 (r - r_p)\right) \\ \frac{\partial V}{\partial v} &= k_2 \left(v - k_1 (r - r_p)\right) \end{aligned}$$

in order to obtain its gradient

$$\nabla V = \begin{bmatrix} \frac{\partial V}{\partial r} \\ \frac{\partial V}{\partial v} \end{bmatrix}.$$

We can therefore characterise the lower part of the hyperplane defined by V

$$G^2_{x,t}(u, \delta) = \mathcal{L}_f V(x, t) + \mathcal{L}_g V(x, t)v + \delta + k_3 V(x, t) \leq 0$$

3.2.2.3 Cylinder

It only remains to study the simplest convex of C, the cylinder. We call it a cylinder in comparison with the cylinder in dimension three.

Indeed, here on the first three coordinates (u), we try to stay in a $\mathbb{R}^3$ ball while the last coordinate (δ) belongs to a vector line. In 3D, we would have the first two coordinates that should remain in a $\mathbb{R}^2$ ball and the third would belong to a straight line.

3.2.3 Solving the optimisation problem

Now that we have described our optimisation problem, we need to solve it.

3.2.3.1 Cvxpy

For that, we will use the cvxpy [1] library on python allowing to solve convex optimization problems. Its use is very intuitive as shown below.

```
# The variable and its dimension
U = cp.Variable(4)

#The cost function
cost = cp.#function(U)

#The objective to be achieved
obj = cp.Minimize(cost)

#The set of constraints
constr = [...]

#The problem and its resolution
prob = cp.Problem(obj, constr)
opt_val = prob.solve()
```

Listing 1: Cvxpy

Moreover, this solver has very big advantages like returning errors in cases where our problem is not convex or if there is no solution! And this is exactly what we need to check if our set C is empty or not.

3.2.3.2 Dykstra

Another way to do this is to use an algorithm to calculate the projection of our null vector onto a convex intersection. The projections onto individual convexes are easily explained. One of the best known algorithms is the alternating projection. However, it has one big flaw: it does not guarantee that we get the projected value of our initial vector but only a point belonging to this intersection. We must therefore turn to an algorithm that is a little less known but extremely efficient and which is perfectly suited to our problem: Dykstra [2]. This algorithm is an adaptation of the alternating projection except that it guarantees to obtain the projected of our vector. Its strong point is that the starting point of the algorithm is the point from which the projection is sought!

The algorithm proposed in [2] works for a projection on an intersection of two convexes. Dykstra's algorithm finds for each r the only $\bar{x} \in C \cap D$ such that:

$$\|\bar{x} - r\|^2 \leq \|x - r\|^2, \text{for all } x \in C \cap D$$

where C, D are convex sets. This problem is equivalent to finding the projection of r onto the set $C \cap D$. To use Dykstra's algorithm, one must know how to project onto the sets C and D separately.

Start with $x_0 = r, p_0 = q_0 = 0$ and update by

$$\begin{aligned} y_k &= \mathcal{P}_D(x_k + p_k) \\ p_{k+1} &= x_k + p_k - y_k \\ x_{k+1} &= \mathcal{P}_C(y_k + q_k) \\ q_{k+1} &= y_k + q_k - x_{k+1}. \end{aligned}$$

We will therefore try to adapt this algorithm to work with several convexes. We create a function `Dykstra` which takes as argument a list of projections `List_proj` corresponding to the projections on each convex and the initial point `x0`.

```
def Dykstra(List_proj , x0 , eps = 1e-7, iter_max = 5_000) :

    n = len(List_proj)
    coeff = np.zeros((n,len(x0)))
    x = x0
    x_inter = x0
    List_iter = [x]
    error = 1
    iter_ = 0

    while (error > eps) and (iter_ < iter_max) :
```

```python

        for i in range(n) :
            proj = List_proj[i]
            x_inter = x
            x = proj(x_inter +coeff[i,:])
            coeff[i,:] = x_inter + coeff[i,:] - x
            error = np.linalg.norm(x - List_iter[-1])
            iter_+=1
            List_iter.append(x)

    return List_iter[-1]
```

Listing 2: Dykstra

Some tests have been done to verify the generalized algorithm for several convexes. However, these tests do not in any way justify the safe convergence to the projected one. One should therefore be careful when using it.

First, we want to know how to calculate the projection on the lower part of a hyperplane. Let x be the vector we want to project and H the hyperplane such that $H = \{a_1x_1 + ... + a_nx_n + d \mid (a_1, ..., a_n + d) \in \mathbb{R}^{n+1}\}$. Let $n = (a_1, ..., a_n)$ be the normal vector of H. We can therefore write

$$x = v + w$$

where $v \in H$ and $w \in H^\top$. We therefore obtain the projection onto H.

$$p_H(x) = x - w = x - tn$$

with $t = \frac{\langle x,n\rangle + d}{\|n\|^2}n$. Here, we study the projection on the lower part of the hyperplane H, so we finally have

$$p(x) = \begin{cases} x & \text{if } x \text{ is in the lower part of } H \\ p_H(x) & \text{otherwise.} \end{cases}$$

For projection on a cylinder it is sufficient to adapt the projection on a ball. Let C be a cylinder of radius R. We can write it as $C = B(0, R) \times \mathbb{R}$. The projection onto the ball $B(0, R)$ is written

$$p_B(x) = \begin{cases} x & \text{if } \|x\| \leq R \\ \frac{xR}{\|x\|} & \text{otherwise.} \end{cases}$$

Let $x = (\tilde{x}, x')$ be the vector we are trying to project. We have

$$p(x) = \begin{cases} x & \text{if } \tilde{x} \text{ is in the ball} \\ (p_B(\tilde{x}), x') & \text{otherwise.} \end{cases}$$

3.2.4 Resolution of our system

Recall that we want to solve the following differential equation

$$\dot{y}(t) = \begin{bmatrix} \dot{r}(t) \\ \dot{v}(t) \end{bmatrix} = \begin{bmatrix} v(t) \\ \tilde{u}(y(t), t) \end{bmatrix}$$

where $\tilde{u}$ is the solution of the following problem

$$\tilde{u}(x, t) = \underset{(u,\delta)\in C}{\operatorname{argmin}} \, u^T u + k\delta^2$$

and where we also have to do an integration to define our constraint depending on H because we have to calculate the future trajectory according to the previously defined control, here u_{ball}.

Thus, to solve our final integration for that solve, inside, another integration and an optimization problem. The storage of the control during the resolution requires that you have to create your own RK4 function for example. Indeed, for RK4 we calculate four constants k_i for different values of t and x. To compute these values, we have to apply the system which requires to solve the optimisation problem each time. Thus, it is necessary to be able to store the control find only once and not four times for each time t. Moreover, having to solve the optimisation problem several times slows down the solution. To solve this problem, we have to make a compromise between computation time and accuracy. We will therefore reduce the precision of our integrator.

3.2.4.1 Influence of V

In this part, we will deactivate the constraint defined by H in order to see the influence of V on the satellite trajectory. To do this, we will test different sets of constants in order to understand their influence and to decide which combination to keep for the future. For all simulations in this paragraph, we take $x_0 = (0, -15, 0, -0.2, 0.3, 0)$.

Let us start by studying the influence of k_1. We fix $k_2 = 0.1$, $k_3 = 0.1$ and $k = 0.2$.

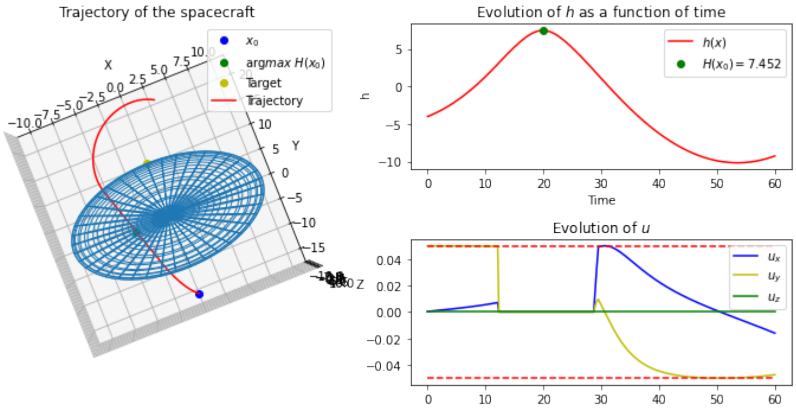

Figure 10: $k_1 = -1$

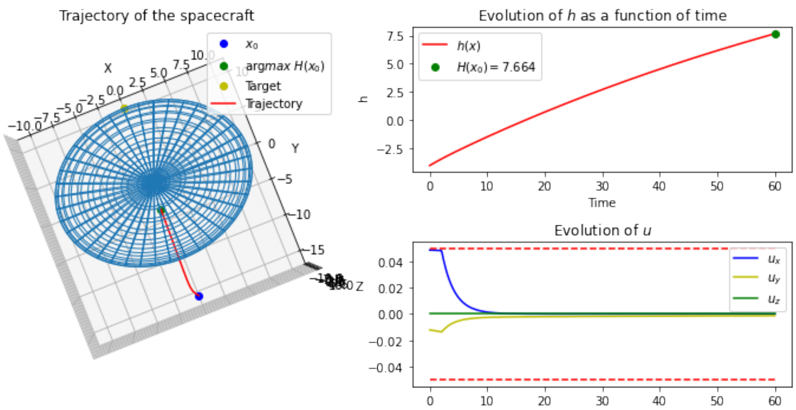

Figure 11: $k_1 = -0.01$

It can be seen that the smaller k_1 is, the greater the attraction of the satellite and the greater its acceleration. However, this makes it less manoeuvrable and therefore requires several manoeuvres to realign with the target.

Let's move on to the influence of k_2. We fix $k_1 = -0.1$, $k_3 = 0.1$ and $k = 0.2$.

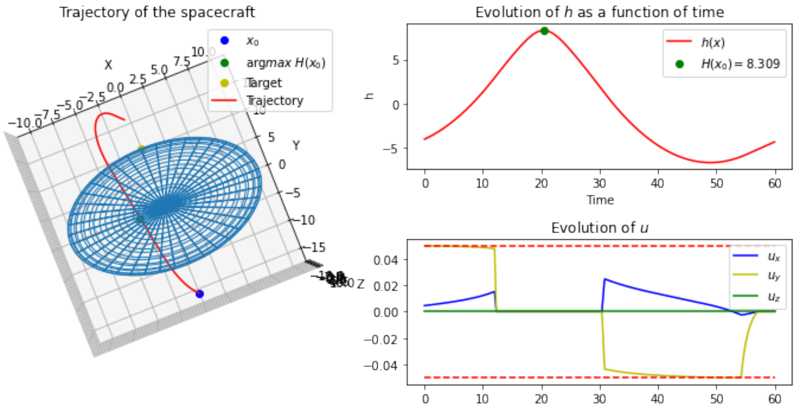

Figure 12: $k_2 = 1$

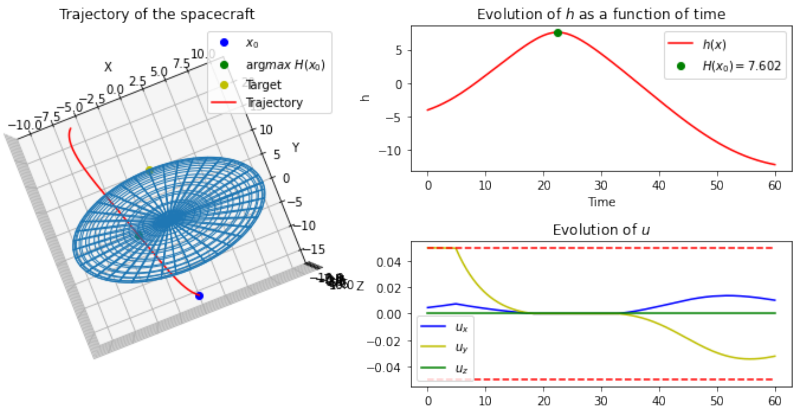

Figure 13: $k_2 = 0.01$

We can see that the larger k_2 is, the more attractive the constraint on V is.

Let us turn to the constant k_3. We fix $k_1 = -0.1$, $k_2 = 0.5$ and $k = 0.2$.

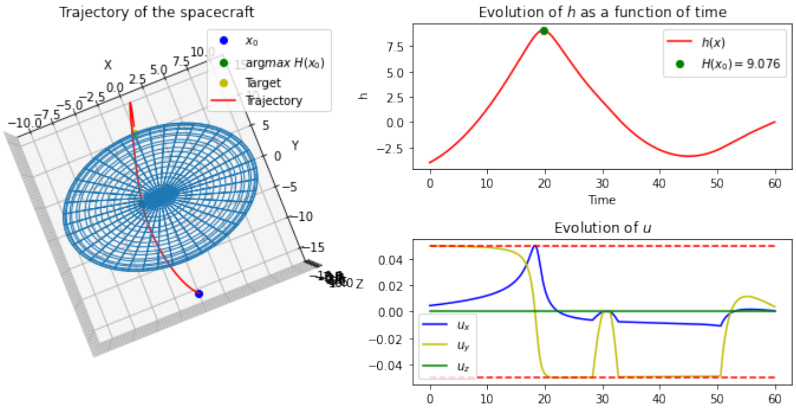

Figure 14: $k_3 = 1$

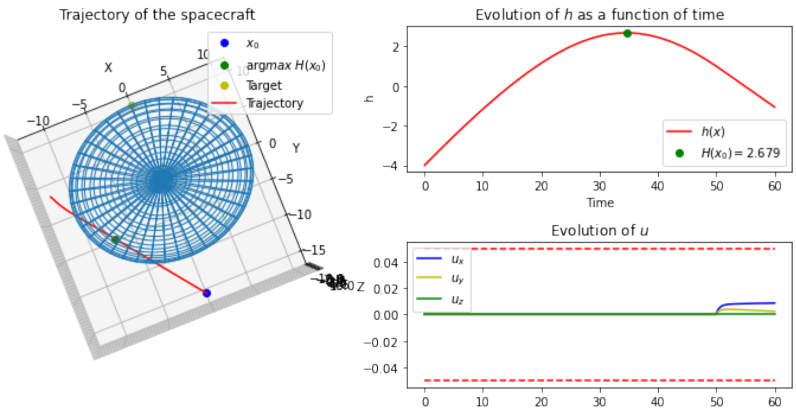

Figure 15: $k_3 = 0.01$

The larger k_3 is, the more the satellite is attracted to the target. One can even notice that k_3 has a greater attractive power than the other constants seen before. Indeed, in the formulation of the constraints, the larger k_3 is, the more one forces the derivative of V to become negative and thus the faster V decreases. This explains this attractive phenomenon.

Let us study the influence of the last constant, k. We fix $k_1 = -0.1$, $k_2 = 0.5$ and $k3 = 0.1$.

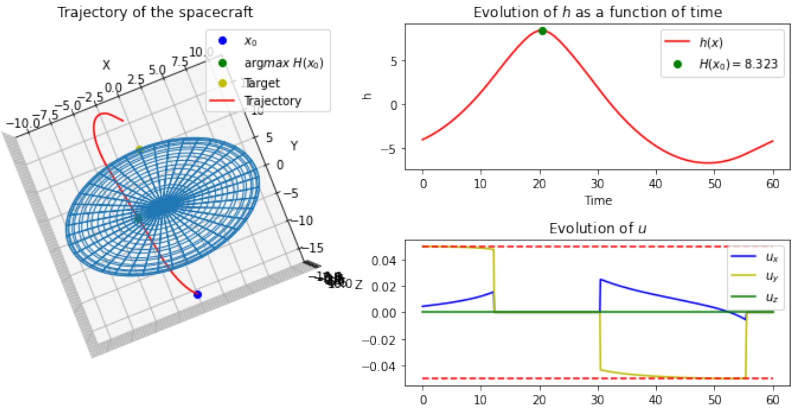

Figure 16: $k = 10$

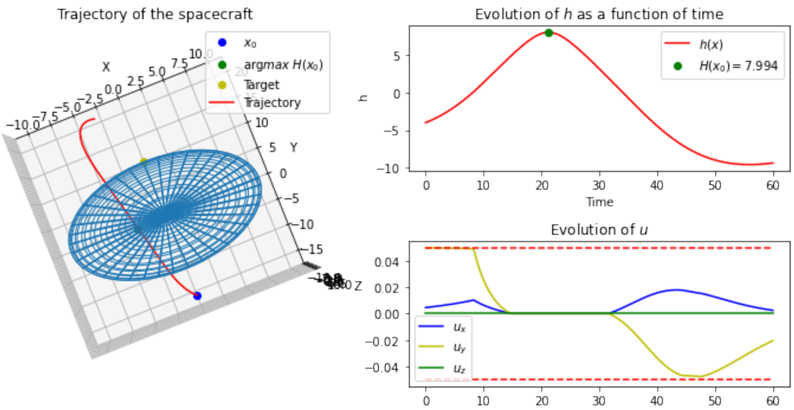

Figure 17: $k = 0.01$

We see a slightly larger attraction when k is larger. But the most important thing is that the bigger k is, the more the satellite seems to be able to take bigger and jerkier accelerations. This can be deduced from the fact that the larger k is, the more we will try to decrease δ and therefore the less $\|u\|^2$.

Based on the various observations, the following values will be used in the following

$$k_1 = -0.1 \qquad k_2 = 0.5 \qquad k_3 = 0.1 \qquad k = 10$$

which seem to be a good compromise.

3.2.4.2 Influence of H

In this part, we will deactivate the constraint defined by V in order to see the influence of H on the satellite trajectory. The problem is that the constraint on H requires additional integrations to be added over time and therefore slows the program down considerably.

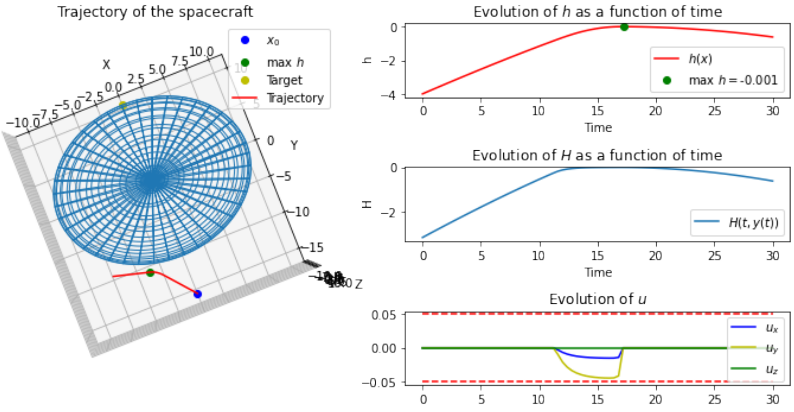

Figure 18: $v_0 = (-0.2, 0.3, 0)$

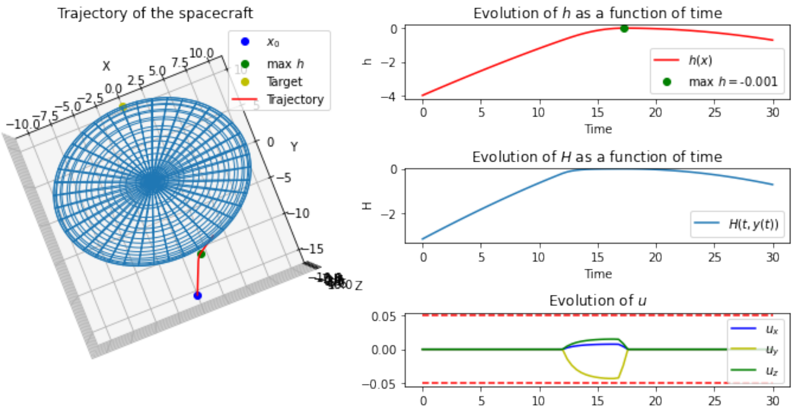

Figure 19: $v_0 = (0.1, 0.3, 0.2)$

In both cases, the obstacle can be circumvented. However, we can see that we avoid the obstacle with rather weak controls compared to u_{ball} and the attraction exerted by V.

3.2.4.3 All constraints activated

Now, we will perform a series of simulations with all the constraints. Let's start by comparing the result obtained with Cvxpy and Dykstra. Here we take $v_0 = (-0.2, 0, 0)$.

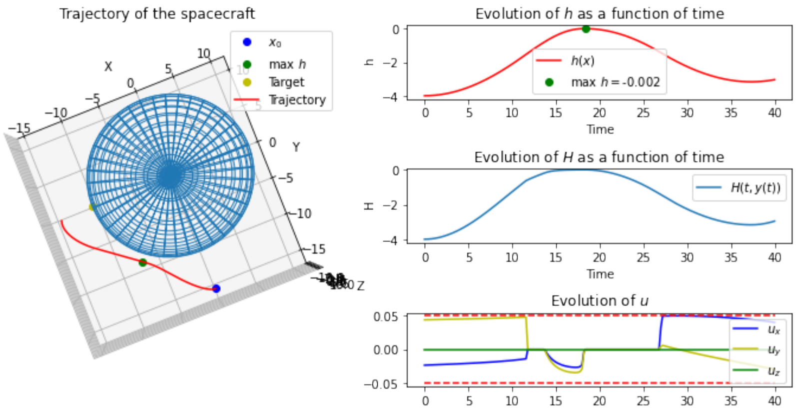

Figure 20: *Cvxpy*

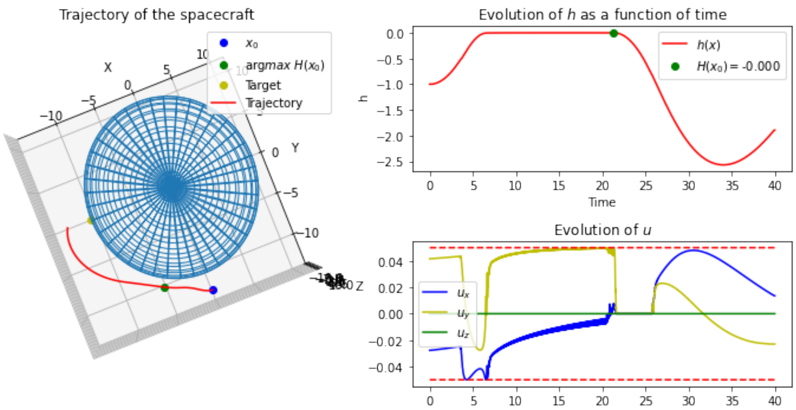

Figure 21: *Dykstra*

Almost the same result is obtained between Cvxpy and Dykstra. However, one should not forget that with Dykstra, first of all, a second stopping condition is used which is the maximum number of iterations and, secondly, convergence to the projected one has not been demonstrated. For the rest, we will use Cvxpy which will return an error in case the optimisation problem has no solution compared to Dykstra which will still return a value.

Let's do a final test by taking another target on the sphere but this time starting with zero speed. The attraction provided by V should give energy to the spacecraft in order to start it. We take $r_p = (7.29, -3.6, -5.82)$ which belongs to the sphere and $x_0 = (0, -12, 0, 0, 0, 0)$.

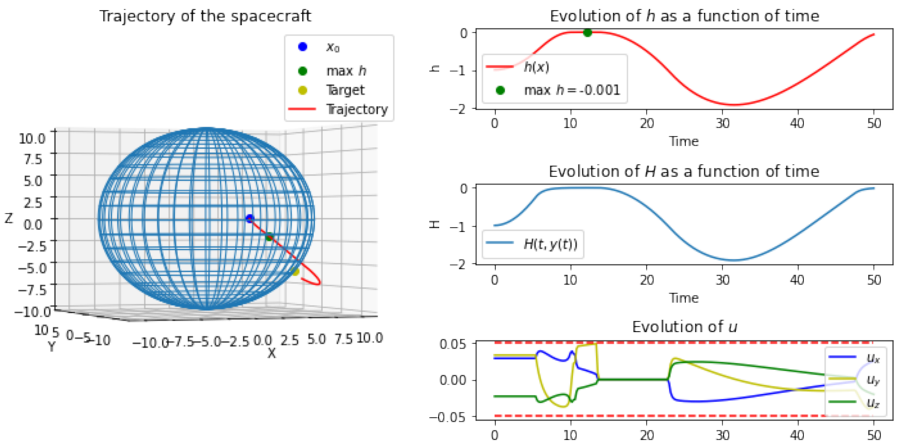

Figure 22: *Cvxpy*

As can be seen, the device approached the target point well while avoiding hitting the sphere. This is exactly what we wanted to do!

3.3 Spherical object with a target path

Now, we want to approach a target trajectory on the surface of the obstacle and therefore we only consider $\dot{r}_p = 0$. Thus, we must modify the derivative of V calculated previously. Indeed

$$\begin{aligned}\dot{V}(y,t,u) = \frac{\partial V}{\partial t} &= \frac{\partial V}{\partial y}\dot{y} + \frac{\partial V}{\partial r_p}\dot{r}_p + \frac{\partial V}{\partial r_p}\ddot{r}_p \\ &= \mathcal{L}_f V + \mathcal{L}_g V u + \frac{\partial V}{\partial r_p}\dot{r}_p + \frac{\partial V}{\partial \dot{r}_p}\ddot{r}_p\end{aligned}$$

and so the constraint on V is now written

$$G^2_{x,t}(u,\delta) = \mathcal{L}_f V(x,t) + \mathcal{L}_g V(x,t)u + \frac{\partial V}{\partial r_p}\dot{r}_p + \frac{\partial V}{\partial \dot{r}_p}\ddot{r}_p + \delta + k_3 V(x,t) \leq 0.$$

On top of that, we can easily calculate the partial derivatives of V, we obtain

$$\begin{aligned}\frac{\partial V}{\partial r_p} &= -(r - r_p) + k_2 k_1 \left(v - \dot{r}_p - k_1(r - r_p)\right) \\ \frac{\partial V}{\partial \dot{r}_p} &= -k_2 \left(v - \dot{r}_p - k_1(r - r_p)\right)\end{aligned}$$

3.3.1 Simple target

We start by studying a trajectory on the surface of the sphere. As with the other simulations, a small conflict is created. Indeed, the trajectory is located on the sphere but inside the safety bubble. Thus, the CLF will try to get as close as possible to the target, but the CBF will interfere when we are almost outside the safe zone. Hence the conflict.

We can try to cover the whole sphere. To do this, we will take a target trajectory on the surface of the sphere that follows the z axis with a certain speed and rotates around it with another higher speed in order to see this spinning motion correctly. To obtain a continuously differentiable trajectory, we express it in spherical coordinates and thus obtain

$$r_p(t) = \rho_a \begin{bmatrix} \sin(\phi(t))\cos(\theta(t)) \\ \sin(\phi(t))\sin(\theta(t)) \\ \cos(\phi(t)) \end{bmatrix}.$$

Since r_p is clearly continuously differentiable, we can calculate $\dot{r}p$

$$\dot{r}_p = \rho_a \begin{bmatrix} \dot{\phi}\cos(\phi)\cos(\theta) - \sin(\phi)\sin(\theta)\,\dot{\theta} \\ \dot{\phi}\cos(\phi)\sin(\theta) + \sin(\phi)\cos(\theta)\,\dot{\theta} \\ -\dot{\phi}\sin(\phi) \end{bmatrix}.$$

For the same reason, we can calculate $\ddot{r}p$ and obtain the following result directly using the Leibnitz formula

$$\ddot{r}_p = \rho_a \begin{bmatrix} \left(\cos(\phi)\,\ddot{\phi} - \sin(\phi)\,\dot{\phi}^2\right)\cos(\theta) - 2\cos(\phi)\,\dot{\phi}\sin(\theta)\,\dot{\theta} - \sin(\phi)\left(\cos(\theta)\,\dot{\theta}^2 + \sin(\theta)\,\ddot{\theta}\right) \\ \left(\cos(\phi)\,\ddot{\phi} - \sin(\phi)\,\dot{\phi}^2\right)\sin(\theta) + 2\cos(\phi)\,\dot{\phi}\cos(\theta)\,\dot{\theta} + \sin(\phi)\left(\cos(\theta)\,\ddot{\theta} - \sin(\theta)\,\dot{\theta}^2\right) \\ -\cos(\phi)\,\dot{\phi}^2 - \sin(\phi)\,\ddot{\phi} \end{bmatrix}.$$

We place ourselves in a very simple case, we take θ and ϕ of the linear functions and thus we have $\ddot{\theta} = \ddot{\phi} = 0$ which enables us to obtain a formula a little simpler for $\ddot{r}_p$,

$$\ddot{r}_p = \rho_a \begin{bmatrix} -\sin(\phi)\,\dot{\phi}^2\cos(\theta) - 2\cos(\phi)\,\dot{\phi}\sin(\theta)\,\dot{\theta} - \sin(\phi)\cos(\theta)\,\dot{\theta}^2 \\ -\sin(\phi)\,\dot{\phi}^2\sin(\theta) + 2\cos(\phi)\,\dot{\phi}\cos(\theta)\,\dot{\theta} - \sin(\phi)\sin(\theta)\,\dot{\theta}^2 \\ -\cos(\phi)\,\dot{\phi}^2 \end{bmatrix}.$$

To obtain the desired result, it is of course necessary to take a trajectory that is not too fast. Indeed, we could fall into the case where the spacecraft would be in phase with the rotation and therefore would go along a certain longitude without rotating around the sphere. For this, we have taken arbitrary values which are

$$\theta(t) = C_1 t \qquad \phi(t) = C_2 t$$

with

$$C_1 = \frac{2\pi}{60} \qquad C_2 = \frac{C_1}{6}.$$

for with turn much faster than what we travel along the z axis. Taking this trajectory and the initial conditions

$$y_0 = \begin{bmatrix} 0 \\ 0 \\ \rho + 2 \\ 0_3 \end{bmatrix}$$

we obtain this simulation

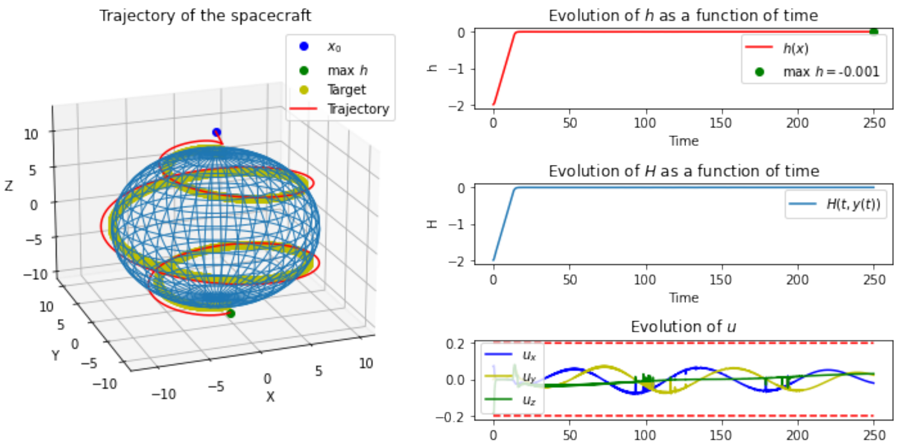

Figure 23: Beautiful trajectory

which greatly satisfies our expectations!

3.3.2 Trajectory with conflict

In this paragraph we will study the limitations of our method. We can ask ourselves how to deal with the case where the target is very deep inside the sphere and therefore causes a larger conflict. For this, we take a very simple case where the target trajectory makes a circular orbit inside the sphere for different orbital radiuses. We therefore have

$$r_p(t) = \rho_o \begin{bmatrix} \cos(\theta(t)) \\ \sin(\theta(t)) \\ 0 \end{bmatrix}$$

with θ a linear function and ρ_o the orbital radius. Let us start with an orbital radius slightly smaller than the radius of the sphere.

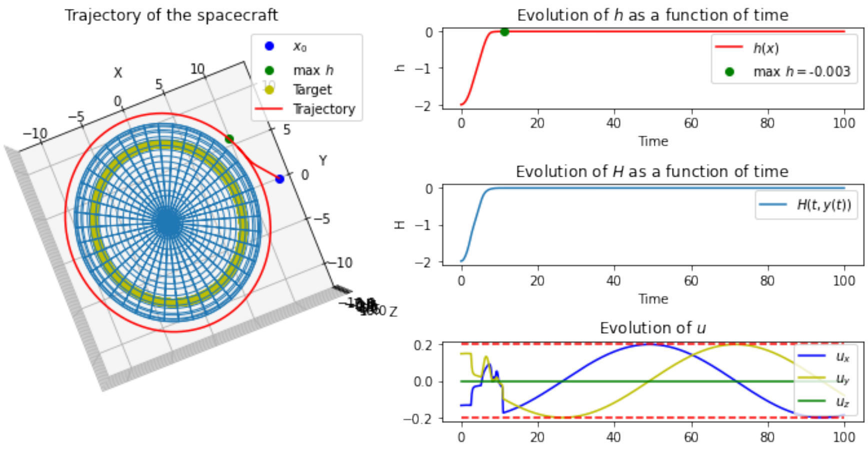

Figure 24: Small conflict

In our simulation, we have $\rho_a = 10$, $\rho_s = 1$ and $\rho_o = 8$. If we take this time $\rho_o = 5$, our algorithm returns an error because our optimisation problem at a certain time t has no solution because the set of constraints C is empty. Thus, if the conflict is too strong, we may find ourselves in a situation where there is no solution that remains in the safe zone. In this case, we would have to play on the constants k_i to make V less attractive and thus avoid having C empty. Moreover, if we take as an initial condition a point closer to the sphere then even for an orbital radius equal to 8, we may find no solution to our problem.

3.4 Addition of gravitation

Now we add the gravitational potential, which we had previously assumed to be zero. Thus, we must modify our function f and we obtain

$$f(y) = \begin{bmatrix} v \\ f_\mu(r) \end{bmatrix}.$$

In the same way it is necessary to modify u_{ball} and $\frac{\partial F}{\partial x}$. Now, to calculate u_{ball}, we must calculate the scalar product between ∇h and f. The last three components of ∇h being zero, we do not need to modify u_{ball}. Finally, all that remains is to adapt $\frac{\partial F}{\partial x}$. We thus have

$$\begin{aligned}\frac{\partial F}{\partial x} &= \begin{bmatrix} \frac{\partial v}{\partial r} & \frac{\partial v}{\partial v} \\ \frac{\partial f_\mu}{\partial r} + \frac{\partial u_{ball}}{\partial r} & \frac{\partial f_\mu}{\partial v} + \frac{\partial u_{ball}}{\partial v} \end{bmatrix} \\ &= \begin{bmatrix} 0_3 & I_3 \\ u_{max}\left(\frac{1}{\|r_a - r\|} I_3 - \frac{1}{\|r_a - r\|^3}(r_a - r)(r_a - r)^T\right) + \frac{\partial f_\mu}{\partial r} & 0_3 \end{bmatrix}.\end{aligned}$$

After a simple second law of Newton, we get

$$f_\mu(r) = \mu \frac{r_a - r}{\|r_a - r\|^3}$$

where $\mu = \mathcal{G}M$ with G the gravitational constant and M the mass of the object. And we also have

$$\begin{aligned}\frac{\partial f_\mu}{\partial r} &= \mu \left[-\frac{I_3}{\|r_a - r\|^3} + (r_a - r) \frac{1}{\|r_a - r\|^6} 3\|r_a - r\|(r_a - r)^T \right] \\ &= -\mu \left[\frac{I_3}{\|r_a - r\|^3} - \frac{3(r_a - r)(r_a - r)^T}{\|r_a - r\|^5} \right].\end{aligned}$$

To study the impact of gravity on control and trajectory, initial conditions and a target trajectory are given. Several simulations are carried out for different values of the standard gravitational parameter μ. Let us start with $\mu = 0$, i.e. we place ourselves in the case where we add no gravitation.

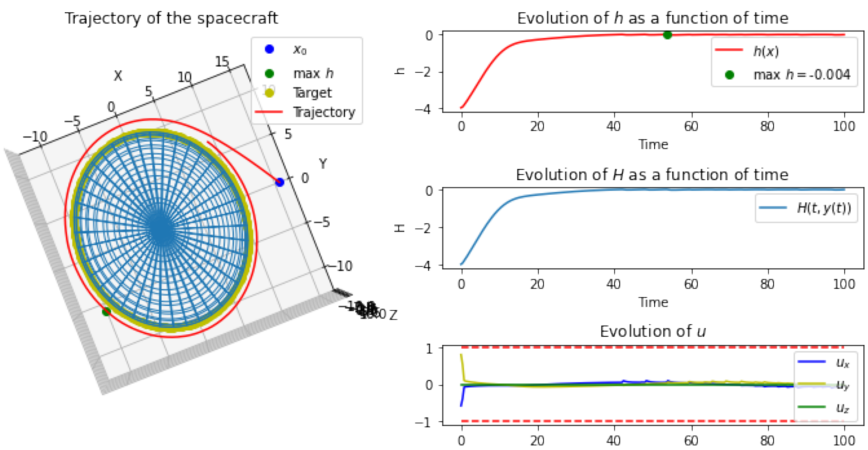

Figure 25: $\mu = 0$

Now we take a small value of gravitation, for example $\mu = 5$. We notice that here the spacecraft needs to brake to avoid colliding with the sphere because of the gravitational attraction.

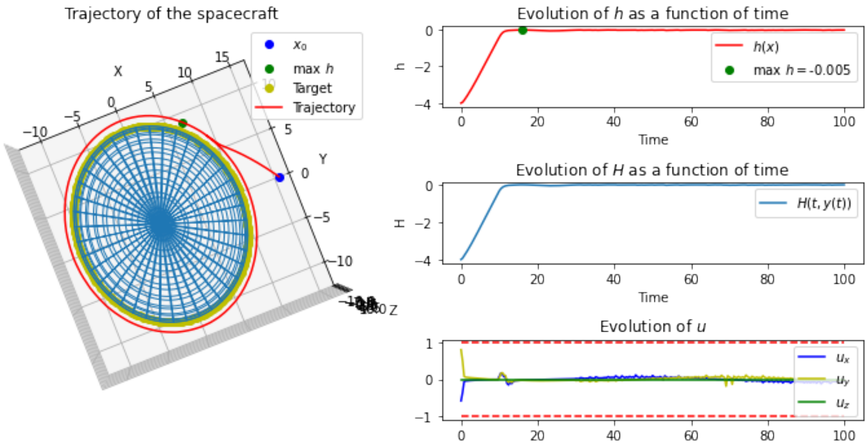

Figure 26: $\mu = 5$

Finally, we take a large gravitation, μ =50. As can be seen from the trajectory, the spacecraft is much more attracted to the sphere. Thus the acceleration needed to avoid the collision is much higher.

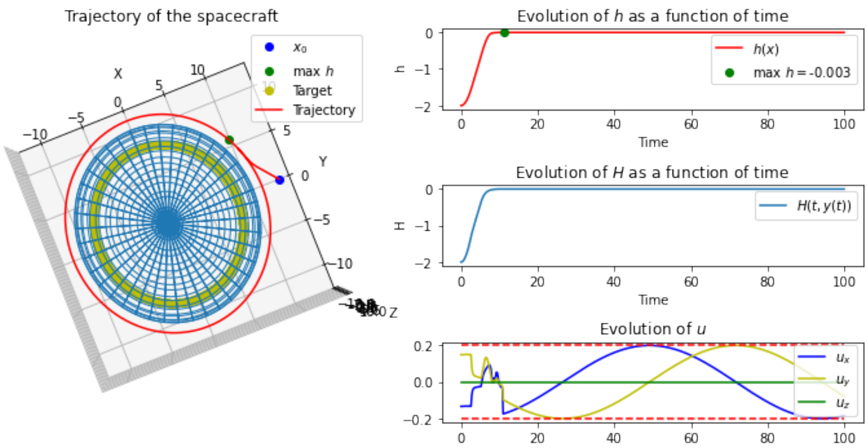

Figure 27: $\mu = 50$

4 Relative motion

We will study the relative motion of two satellites orbiting a planet [4]. Here, we will have a first target satellite S_1 which will be in orbit around the planet. The second chaser satellite, S_2, will aim to approach the target satellite. The objective is to study the movement of S_2 with respect to S_1. To do this, we place a first fixed reference frame at the level of the planet and a second moving frame attached to S_1. It will thus be necessary to study the motion of S_2 with respect to the moving frame located on S_1. Note R the position of S_1 and V its velocity and r the position of S_2.

We place ourselves in the configuration where

$$\hat{\imath} = \frac{R}{\|R\|} \qquad \hat{\jmath} = \frac{V}{\|V\|} \qquad \hat{k} = \hat{\imath} \wedge \hat{\jmath}$$

with

$$\begin{aligned}
\hat{\imath} &= l_x \hat{I} + m_x \hat{J} + n_x \hat{K} \\
\hat{\jmath} &= l_y \hat{I} + m_y \hat{J} + n_y \hat{K} \\
\hat{k} &= l_z \hat{I} + m_z \hat{J} + n_z \hat{K}.
\end{aligned}$$

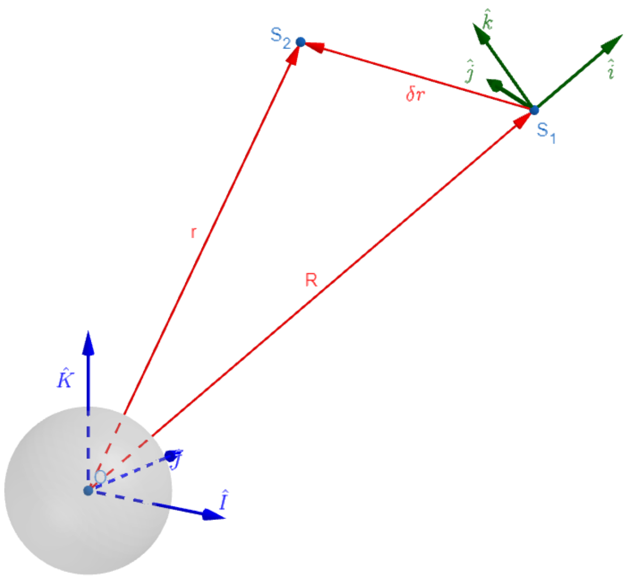

Figure 28: Configuration

We note Q_{Xx} the matrix of passage of the base $(\hat{I}, \hat{J}, \hat{K})$ to the base $(\hat{\imath}, \hat{\jmath}, \hat{k})$ which is written thus

$$Q_{Xx} = \begin{bmatrix} l_x & m_x & n_x \\ l_y & m_y & n_y \\ l_z & m_z & n_z \end{bmatrix}.$$

4.1 Linearization of the equations of relative motion in orbit

We note R and r the vectors associated respectively with the target S_1 and the chaser S_2. We can therefore define the vector δr, the position of the chaser with respect to the motion frame located on the target. We therefore have

$$r = R + \delta r.$$

For the following, we will assume that

$$\frac{\|\delta r\|}{\|R\|} << 1,$$

i.e. that $\|\delta r\|$ is negligible before $\|R\|$.

After two quick seconds of Newton's Law in the inertial frame, we get the following two equations (as the masses of the two statellites are very small, the gravitational forces between them can be neglected)

$$\ddot{r} = -\mu \frac{r}{\|r\|^3} \quad \text{and} \quad \ddot{R} = -\mu \frac{R}{\|R\|^3}$$

where $\mu = \mathcal{G}M$ with $\mathcal{G}$ the gravitational constant and M the mass of "the planet" around which S_1 orbits. It is easy to deduce that

$$\begin{aligned} \ddot{\delta r} &= -\ddot{R} + \ddot{r} \\ &= \mu \left(\frac{R}{\|R\|^3} - \frac{r}{\|r\|^3} \right) \\ &= \mu \left(\frac{R}{\|R\|^3} - \frac{R + \delta r}{\|R + \delta r\|^3} \right) \end{aligned}$$

The aim is to linearise this differential equation around the equilibrium point $\delta r = 0$. To do this, we will use the fact that $\|\delta r\|$ is negligible in front of $\|R\|$. Note

$$\begin{array}{rcl} f : \mathbb{R}_+^* & \to & \mathbb{R} \\ x & \mapsto & \frac{1}{x^{3/2}} \end{array} \quad \text{and} \quad \begin{array}{rcl} g : \mathbb{R}^3 & \to & \mathbb{R}_+ \\ X & \mapsto & \|X\|^2 \end{array}$$

We can therefore define the application h such that

$$\begin{array}{rcl} \alpha : \mathbb{R}^3 & \to & \mathbb{R}_+ \\ X & \mapsto & fog(X) = \frac{1}{\|X\|^3} \end{array}$$

The applications f and g are clearly differentiable on $\mathbb{R}_+^*$ respectively on $\mathbb{R}^3$ and therefore α is differentiable on $\mathbb{R}^3 \setminus \{0\}$. Let $h \in \mathbb{R}$ and $H \in \mathbb{R}^3$, we have

$$D_x f(h) = -\frac{3h}{2x^{5/2}} \qquad \text{and} \qquad D_X g(H) = 2\langle X \mid H \rangle,$$

and so by compound differentiation we obtain

$$D_X \alpha(H) = (D_{g(X)} f) o (D_X g)(H) = -3\frac{\langle X \mid H \rangle}{\|X\|^5}.$$

In our case, as $||deltar\|$ is negligible before $|R\|$, we have

$$\begin{aligned} \frac{1}{\|R + \delta r\|^3} &= \alpha(R + \delta r) \\ &= \alpha(R) + D_R \alpha(\delta r) + o(\|\delta r\|) \\ &= \frac{1}{\|R\|^3} - 3\frac{\langle R \mid \delta r \rangle}{\|R\|^5} + o(\|\delta r\|). \end{aligned}$$

Neglecting the higher order terms in $\|\delta r\|$, we obtain the linearized differential equation in δr

$$\ddot{\delta r} = -\frac{\mu}{\|R\|^3}\left[\delta r - \frac{3}{\|R\|^2}\langle R \mid \delta r \rangle R\right].$$

It is possible to greatly simplify this differential equation. For that, it is necessary to express our vectors in the base $\left(\hat{\imath}, \hat{\jmath}, \hat{k}\right)$. We have

$$\begin{aligned} R &= \|R\|\hat{\imath}, \\ \delta r &= \delta x\hat{\imath} + \delta y\hat{\jmath} + \delta z\hat{k}, \\ \delta v &= \delta\dot{x}\hat{\imath} + \delta\dot{y}\hat{\jmath} + \delta\dot{z}\hat{k}, \\ \delta a &= \delta\ddot{x}\hat{\imath} + \delta\ddot{y}\hat{\jmath} + \delta\ddot{z}\hat{k}, \end{aligned}$$

and so

$$\ddot{\delta r} = -\frac{\mu}{\|R\|^3}\left(-2\delta x\hat{\imath} + \delta y\hat{\jmath} + \delta z\hat{k}\right).$$

We use $_{rel}$ to indicate the vectors located in the moving frame and $_0$ to indicate the vectors associated with the moving frame in the fixed frame. We thus have the following relation

$$a = a_0 + a_{rel} + \dot{\Omega} \wedge r_{rel} + \Omega \wedge (\Omega \wedge r_{rel}) + 2\Omega \wedge v_{rel}$$

where $\Omega = \frac{h}{\|r_0\|}$ and $\dot{\Omega} = -2h\frac{\langle v_0 \mid r_0 \rangle}{\|r_0\|^4}$ with $h = v_0 \wedge r_0$. That is,

$$\delta\ddot{r} = \frac{d^2}{dt^2}\delta r = \frac{d^2}{dt^2}(r_{S_2} - r_{S_1}) = a_{S_2} - a_{S_1}$$

$\delta\ddot{r}$ is not to be confused with δa, the relative acceleration measured in the comoving frame. In our case, we have $a_{rel} = \delta a$, $r_{rel} = \delta r$, $v_{rel} = \delta v$, $a = a_{S_2}$ and $a_0 = a_{S_1}$ so

$$\delta a = \delta\ddot{r} - \dot{\Omega} \wedge \delta r - \Omega \wedge (\Omega \wedge \delta r) - 2\Omega \wedge \delta v$$

with $\Omega = \frac{\|h\|}{\|R\|^2}\hat{k}$ and $\dot{\Omega} = -2\|h\|\frac{\langle V \mid R \rangle}{\|R\|^4}\hat{k}$. We can therefore explicitly calculate the different terms that appear and obtain the following equations

$$\begin{aligned}
\dot{\Omega} \wedge \delta r &= 2\|h\|\frac{\langle V \mid R \rangle}{\|R\|^4}\left(\delta y\hat{\imath} - \delta x\hat{\jmath}\right), \\
\Omega \wedge (\Omega \wedge \delta r) &= \frac{\|h\|^2}{\|R\|^4}\left(\delta x\hat{\imath} + \delta y\hat{\jmath}\right), \\
2\Omega \wedge \delta v &= 2\frac{\|h\|}{\|R\|^2}\left(\delta\dot{x}\hat{\jmath} - \delta\dot{y}\hat{k}\right).
\end{aligned}$$

We therefore obtain the following equation

$$\begin{aligned}
\delta a = &- \frac{\mu}{\|R\|^3}\left(-2\delta x\hat{\imath} + \delta y\hat{\jmath} + \delta z\hat{k}\right) - 2\|h\|\frac{\langle V \mid R \rangle}{\|R\|^4}\left(\delta y\hat{\imath} - \delta x\hat{\jmath}\right) - \frac{\|h\|^2}{\|R\|^4}\left(\delta x\hat{\imath} + \delta y\hat{\jmath}\right) \\
&- 2\frac{\|h\|}{\|R\|^2}\left(\delta\dot{x}\hat{\jmath} - \delta\dot{y}\hat{\imath}\right)
\end{aligned}$$

which allows us to obtain a differential equation

$$\begin{cases}
\delta\ddot{x} - \left(\frac{2\mu}{\|R\|^3} + \frac{\|h\|^2}{\|R\|^4}\right)\delta x + 2\|h\|\frac{\langle V \mid R \rangle}{\|R\|^4}\delta y - 2\frac{\|h\|}{\|R\|^2}\delta\dot{y} = 0 \\
\delta\ddot{y} + \left(\frac{\mu}{\|R\|^3} - \frac{\|h\|^2}{\|R\|^4}\right)\delta y - 2\|h\|\frac{\langle V \mid R \rangle}{\|R\|^4}\delta x + 2\frac{\|h\|}{\|R\|^2}\delta\dot{x} = 0 \\
\delta\ddot{z} + \frac{\mu}{\|R\|^3}\delta z = 0
\end{cases} \tag{4.1}$$

The first two equations are coupled since δx and δy appear in each of them. δz appears alone in the last equation and nowhere else, which means that the relative motion in the z direction is independent of that in the other two directions. It is important to note that R and V are time dependent. They must therefore be included in the resolution, i.e.

$$\begin{cases} \dot{R} = V \\ \dot{V} = -\mu\frac{R}{\|R\|^3} \\ \delta\dot{r} = \delta v \\ \delta\ddot{x} = \left(\frac{2\mu}{\|R\|^3} + \frac{\|h\|^2}{\|R\|^4}\right)\delta x - 2\|h\|\frac{\langle V \mid R\rangle}{\|R\|^4}\delta y + 2\frac{\|h\|}{\|R\|^2}\delta\dot{y} \\ \delta\ddot{y} = -\left(\frac{\mu}{\|R\|^3} - \frac{\|h\|^2}{\|R\|^4}\right)\delta y + 2\|h\|\frac{\langle V \mid R\rangle}{\|R\|^4}\delta x - 2\frac{\|h\|}{\|R\|^2}\delta\dot{x} \\ \delta\ddot{z} = -\frac{\mu}{\|R\|^3}\delta z \end{cases}$$

We can easily notice that the set of equilibrium points of (4.1) are the set of points in $\{(0, \lambda, 0, 0, 0, 0)^T \mid \lambda \in \mathbb{R}\}$. Indeed, thanks to the linearisation, if we lie on the $\hat{j}$ axis, the spacecraft S_2 will lie on the same orbit as S_1. This is what we can see just below.

For the rest of the simulations, we take a concrete example. We are going to simulate 2 satellites in orbit around the planet Earth. All measurements will be made in km and we will study the trajectories obtained on 2 orbital periods. Let us begin by considering the case where the orbit of the satellite S_1 is a circular orbit. We therefore have $\|R\|$ which will be constant over time and $\|V\| = \mu/\|R\|$.

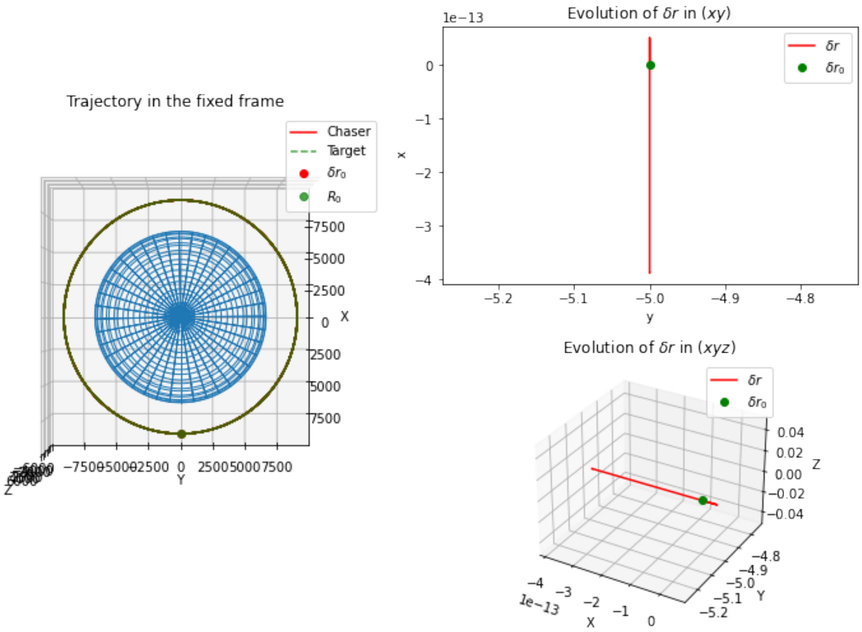

Figure 29: Circular orbit and $\delta r_0 = (0, -5, 0)$

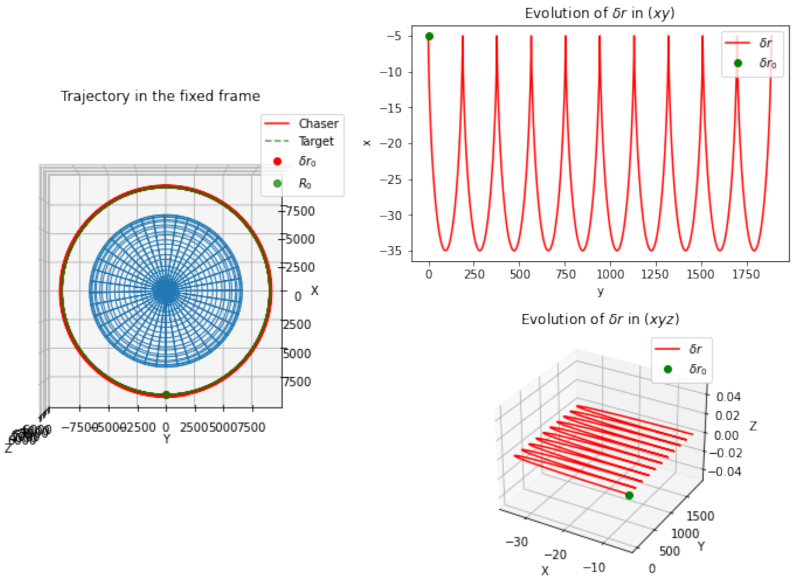

Figure 30: Circular orbit and $\delta r_0 = (-5, 0, 0)$

We can confirm what we said above, if we take an equilibrium point we remain on the same orbit. On the contrary, if we do not take an equilibrium point as an initial condition, then we leave the orbit little by little. It should be noted that when we take an equilibrium point, the position in the reference frame (x, y), i.e. $(\hat{\imath}, \hat{\jmath})$ is constant. Indeed, we obtain a variation of δx of order 10^{-13} which we can consider as zero. The same applies to the elliptical trajectory.

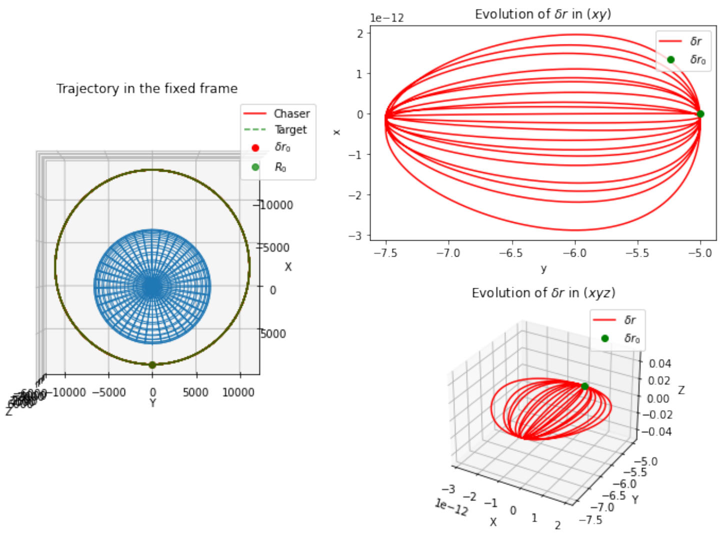

Figure 31: Elliptical orbit and $\delta r_0 = (0, -5, 0)$

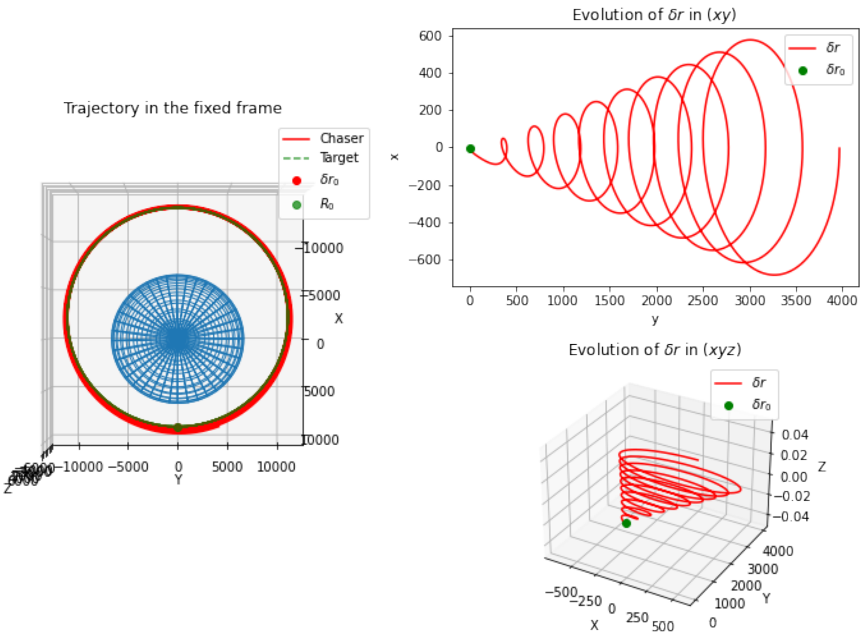

Figure 32: Elliptical orbit and $\delta r_0 = (-5, 0, 0)$

4.2 Hill-Clohessy–Wiltshire equations

The Hill-Clohessy-Wiltshire equations describe a simplified model of relative orbital motion, in which the target bearing the relative coordinate system is in a circular orbit and the chaser spacecraft in an elliptical or circular orbit. This model gives a first order approximation of the relative motion of the hunter. However, this model is very limited, especially because of the approximations that lead to larger and larger errors over time.

4.2.1 The classic case

Here we have a special case: we take a circular orbit for the spacecraft S_1. We therefore have

$$\begin{cases} \langle V \mid R \rangle = 0, \\ \|h\| = \sqrt{\mu \|R\|} \end{cases}$$

because $\|V\| = \sqrt{\mu / \|R\|}$. The proof is not very complicated but a bit long. For that, it is necessary to express the position of the spacecraft with respect to the angle formed in the course of time compared to the perigee. It only remains to consider the case where $e = 0$ which is the case of a circular orbit. All this is very well described in the reference.

We then obtain the differential equation

$$\begin{cases} \delta\ddot{x} - 3\frac{\mu}{\|R\|^3}\delta x - 2\sqrt{\frac{\mu}{\|R\|^3}}\delta\dot{y} = 0 \\ \delta\ddot{y} + +2\sqrt{\frac{\mu}{\|R\|^3}}\delta\dot{x} = 0 \\ \delta\ddot{z} + \frac{\mu}{\|R\|^3}\delta z = 0. \end{cases}$$

Taking as notation $n = \frac{\|V\|}{\|R\|} = \sqrt{\frac{\mu}{\|R\|^3}}$, we finally have

$$\begin{cases} \delta\ddot{x} - 3n^2\delta x - 2n^2\delta\dot{y} = 0 \\ \delta\ddot{y} + 2n\delta\dot{x} = 0 \\ \delta\ddot{z} + n^2\delta z = 0 \end{cases}$$

In this case the coefficients are constant, the solution can be calculated analytically. Let's start by looking at the second equation, which translates into

$$\frac{d}{dt}\left(\delta\dot{y} + 2n\delta x\right) = 0$$

and therefore there exists a constancy C_1 such that

$$\delta\dot{y} + 2n\delta x = C_1.$$

By replacing $\delta\dot{y}$ by its value in the first equation, we obtain

$$\delta\ddot{x} + n\delta x = 2nC_1.$$

It is easy to solve a differential equation under this and we obtain

$$\begin{aligned} \delta x &= \frac{2}{n}C_1 + C_2\sin(nt) + C_3\cos(nt) \\ \delta\dot{x} &= C_2 n\cos(nt) - C_3 n\sin(nt) \end{aligned}$$

and

$$\begin{aligned} \delta\dot{y} &= -3C_1 - 2C_2 n\sin(nt) - 2C_3 n\cos(nt) \\ \delta y &= -3C_1 t - 2C_2\cos(nt) - 2C_3\sin(nt) + C_4. \end{aligned}$$

All that remains is to make the constants explicit. For that, we take as initial condition, at $t = 0$,

$$\delta x(t=0) = \delta x_0 \quad \delta y(t=0) = \delta y_0 \quad \delta\dot{x}(t=0) = \delta\dot{x}_0 \quad \delta\dot{y}(t=0) = \delta\dot{y}_0.$$

We therefore have the following system

$$\begin{cases} \frac{2}{n}C_1 + C_3 = \delta x_0 \\ C_2 n = \delta \dot{x}_0 \\ -3C_1 - 2C_3 n = \delta \dot{y}_0 \\ 2C_2 + C_4 = \delta y_0 \end{cases} = \begin{cases} C_1 = 2n\delta x_0 + \delta \dot{y}_0 \\ C_2 = \frac{1}{n}\delta \dot{x}_0 \\ C_3 = -3\delta x_0 - \frac{2}{n}\delta \dot{y}_0 \\ C_4 = -\frac{2}{n}\delta \dot{x}_0 + \delta y_0 \end{cases}$$

All that remains is to deal with the case of z. Similarly, we can easily calculate the solution of this differential equation

$$\begin{aligned} \delta z &= C_5 \sin(nt) + C_6 \cos(nt) \\ \delta \dot{z} &= C_5 n \cos(nt) - C_6 n \sin(nt) \end{aligned}$$

where $C_5 = \frac{\delta \dot{z}_0}{n}$ and $C_6 = \delta z_0$ with $\delta z(t = 0) = \delta z_0$ and $\delta \dot{z}(t = 0) = \delta \dot{z}_0$.

This gives the solution

$$\begin{cases} \delta x = 4\delta x_0 + \frac{2}{n}\delta \dot{y}_0 + \frac{\delta \dot{x}_0}{n}\sin(nt) - (3\delta x_0 + \frac{2}{n}\delta \dot{y}_0)\cos(nt) \\ \delta y = \delta y_0 - \frac{2}{n}\delta \dot{x}_0 - 3\left(2n\delta x_0 + \delta \dot{y}_0\right)t + 2\left(3\delta x_0 + \frac{2}{n}\delta \dot{y}_0\right)\sin(nt) + \frac{2}{n}\delta \dot{x}_0 \cos(nt) \\ \delta z = \frac{1}{n}\delta \dot{z}_0 \sin(nt) + \delta z_0 \cos(nt) \end{cases}$$

and

$$\begin{cases} \delta \dot{x} = 3n\sin(nt)\delta x_0 + \cos(nt)\delta \dot{x}_0 + 2\sin(nt)\delta \dot{y}_0 \\ \delta \dot{y} = 6n\left(\cos(nt) - 1\right)\delta x_0 - 2\sin(nt)\delta \dot{x}_0 + \left(4\cos(nt) - 3\right)\delta \dot{y}_0 \\ \delta z = -n\sin(nt)\delta z_0 + \cos(nt)\delta \dot{z}_0 \end{cases}$$

which can be written as a linear system

$$\delta r(t) = \Phi_{rr}(t)\delta r_0 + \Phi_{rv}(t)\delta v_0$$

and

$$\delta v(t) = \Phi_{vr}(t)\delta r_0 + \Phi_{vv}(t)\delta v_0$$

where

$$\Phi_{rr}(t) = \begin{bmatrix} 4-3\cos(nt) & 0 & 0 \\ 6\left(\sin(nt)-nt\right) & 1 & 0 \\ 0 & 0 & \cos(nt) \end{bmatrix}$$

$$\Phi_{rv}(t) = \begin{bmatrix} \frac{1}{n}\sin(nt) & \frac{2}{n}\left(1-\cos(nt)\right) & 0 \\ \frac{2}{n}\left(\cos(nt)-1\right) & \frac{1}{4}\left(4\sin(nt)-3nt\right) & 0 \\ 0 & 0 & \frac{1}{n}\sin(nt) \end{bmatrix}$$

$$\Phi_{vr}(t) = \begin{bmatrix} 3n\sin(nt) & 0 & 0 \\ 6n\left(\cos(nt)-1\right) & 0 & 0 \\ 0 & 0 & -n\sin(nt) \end{bmatrix}$$

$$\Phi_{vv}(t) = \begin{bmatrix} \cos(nt) & 2\sin(nt) & 0 \\ -2\sin(nt) & 4\cos(nt)-3 & 0 \\ 0 & 0 & \cos(nt) \end{bmatrix}.$$

Taking the same initial conditions as before, we obtain the same trajectories as in the general case. We can therefore conclude that our results are consistent.

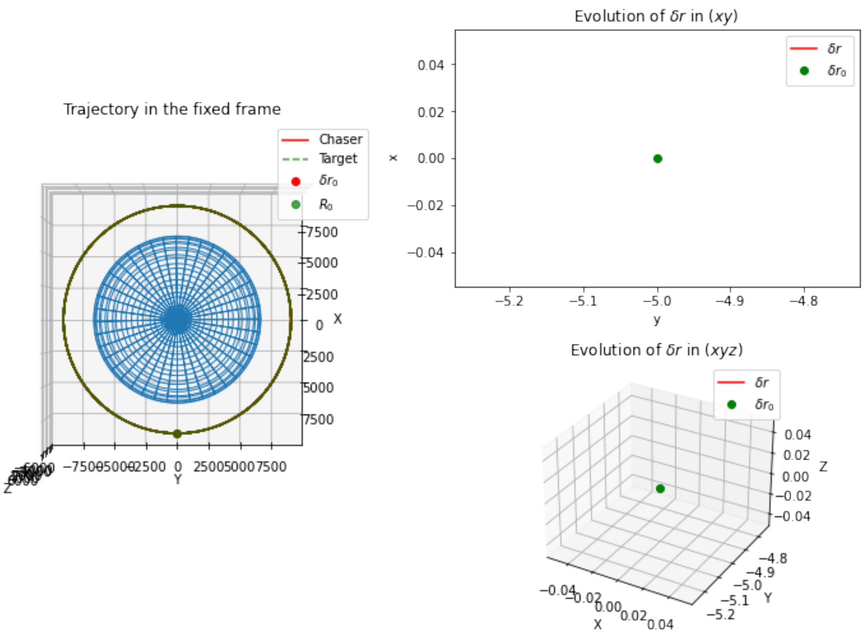

Figure 33: $\delta r_0 = (0, -5, 0)$

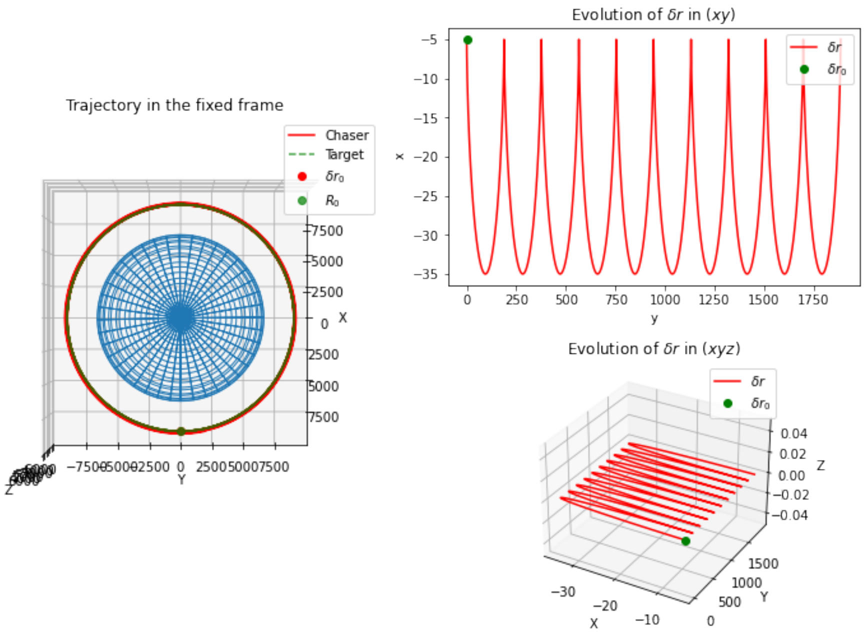

Figure 34: $\delta r_0 = (-5, 0, 0)$

4.2.2 With control

The objective of this work is to avoid the collision of two satellites in orbit around a planet, an asteroid. The cases studied in this paragraph do not allow us to prevent collisions. This is where we see the interest of the work done so far. We will therefore build on the work done above in order to avoid a collision and approach a point, a target trajectory. To do this, we will define a safety sphere around the satellite in circular orbit S_1, allowing us to deviate the trajectory of the second satellite if necessary.

Let us take the differential equation obtained previously with the Hill-Clohessy-Wiltshire equations and add a control u

$$\begin{cases} \delta\ddot{x} = 3n^2\delta x + 2n\delta\dot{y} + u_x \\ \delta\ddot{y} = -2n\delta\dot{x} + u_y \\ \delta\ddot{z} = -n^2\delta z + u_z \end{cases}$$

and let us note f_n the following application

$$f_n(\delta r, \delta v) = \begin{bmatrix} 3n^2\delta x + 2n\delta\dot{y} \\ -2n\delta\dot{x} \\ -n^2\delta z \end{bmatrix}.$$

We finally obtain the following differential system

$$\dot{y} = \begin{bmatrix} \delta\dot{r} \\ \delta\dot{v} \end{bmatrix} = \begin{bmatrix} \delta v \\ f_n\left(\delta r, \delta v\right) + u \end{bmatrix} \tag{4.2}$$

and we can return to the generalized system seen in the paragraph on general resolution

$$\dot{y} = \begin{bmatrix} \delta\dot{r} \\ \delta\dot{v} \end{bmatrix} = \begin{bmatrix} \delta v \\ f_n(\delta r, \delta v) + u \end{bmatrix} = \underbrace{\begin{bmatrix} \delta v \\ 3n^2\delta x + 2n\delta\dot{y} \\ -2n\delta\dot{x} \\ -n^2\delta z \end{bmatrix}}_{f(y)} + \underbrace{\begin{bmatrix} 0_{3\times 3} \\ I_{3\times 3} \end{bmatrix}}_{g(y)} u. \tag{4.3}$$

Remark 4.2.1. *In the literature, a linear system like the one we have is written as*

$$\dot{y} = Ay + Bu$$

with $A \in M_6(\mathbb{R})$ and $B \in M_{6\times 3}(\mathbb{R})$. In our case, we have

$$A = \begin{bmatrix} & 0_3 & & & I_3 & \\ 3n^2 & 0 & 0 & 0 & 2n & 0 \\ 0 & 0 & 0 & -2n & 0 & 0 \\ 0 & 0 & -n^2 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0_3 \\ I_3 \end{bmatrix}$$

The aim is to avoid collision with the safe area, so we look for a function h such that it is always negative in the safe area. So let

$$h(y) = \rho - \|\delta r\|$$

where ρ is the radius of the safe area. The satellite S_1 is considered to have a sufficiently negligible size not to be included in h. Our function h is relative of degree 2, the demonstration being exactly the same as (3.3).

The function h being the same one as for the cases of before, one deduces easily from it that our predefined control is exactly the same one, i.e. that $\dot{h}$ does not depend on u and thus one seeks to minimize $\ddot{h}$ and one obtains in the same way

$$u^* = u_{ball}.$$

Thus, it suffices to solve (3.8) with the exception that $\dot{H}$ is slightly different. Indeed, since our function f is different, we must modify $\frac{\partial F}{\partial x}$ in order to adapt it to our current problem. We have

$$F(t, y(t)) = [\delta v,\ f_n\left(\delta r(t), \delta v(t)\right) + u_{ball}]^T$$

and so

$$\frac{\partial F}{\partial y} = \begin{bmatrix} \frac{\partial \delta v}{\partial \delta r} & \frac{\partial \delta v}{\partial \delta v} \\ \frac{\partial f_n}{\partial \delta r} + \frac{\partial u_{ball}}{\partial \delta r} & \frac{\partial f_n}{\partial \delta v} + \frac{\partial u_{ball}}{\partial \delta v} \end{bmatrix} = \begin{bmatrix} 0_3 & I_3 \\ \frac{\partial f_n}{\partial \delta r} + u_{max}\left(\frac{1}{\|r_a - r\|} I_3 - \frac{1}{\|r_a - r\|^3}(r_a - r)(r_a - r)^T\right) & \frac{\partial f_n}{\partial \delta v} \end{bmatrix}$$

where

$$\frac{\partial f_n}{\partial \delta r} = \begin{bmatrix} 3n^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -n^2 \end{bmatrix}$$

$$\frac{\partial f_n}{\partial \delta v} = \begin{bmatrix} 0 & 2n & 0 \\ -2n & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

We therefore seek to solve the following system

$$\dot{y}(t) = \begin{bmatrix} \dot{\delta r}(t) \\ \dot{\delta v}(t) \end{bmatrix} = \begin{bmatrix} v(t) \\ \tilde{u}(y(t), t) \end{bmatrix}$$

where $\tilde{u}$ is the solution of the following problem

$$\tilde{u}(x, t) = \underset{(u,\delta) \in C}{\operatorname{argmin}}\, u^T u + k\delta^2.$$

The objective is for the spacecraft S_2 to bypass S_1 to position itself in the same orbit, in a stable position, but on the other side. Thus, we take r_p a target point on the $\hat{\jmath}$ axis, i.e. a point of equilibrium.

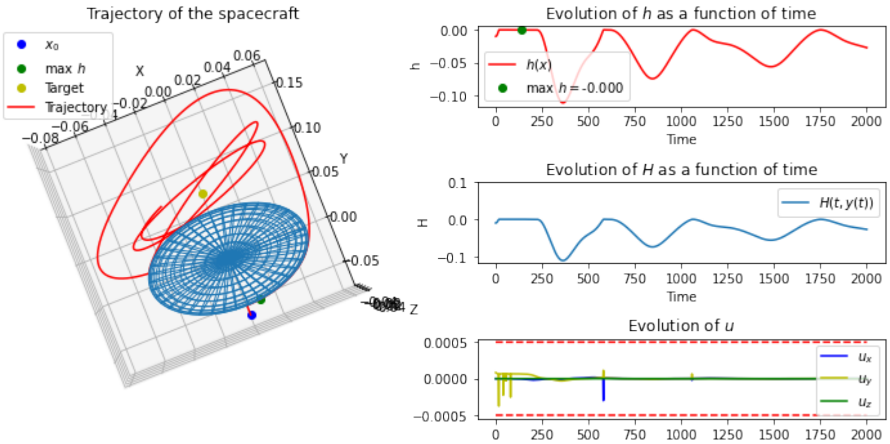

Figure 35: With H

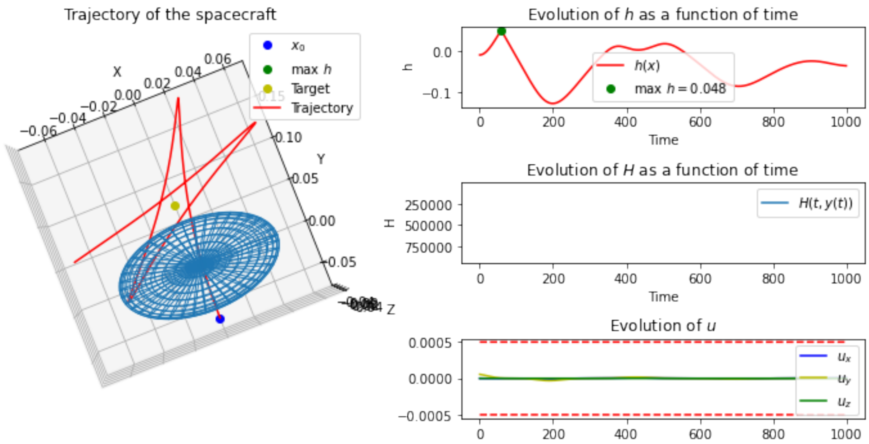

Figure 36: Without H

However, as can be seen above by deactivating the constraint in H, the satellite S_2 does not manage to fix itself on the orbit of S_1 and "rotates" around the fixed point. To check if the problem comes from the CLF or from somewhere else, we replace our optimisation problem (without constraint on H) by

$$\tilde{u} = \underset{\|u\| \leq u_{max}}{\text{argmin}} \ \mathcal{L}_f V + \mathcal{L}_g V u. \tag{4.4}$$

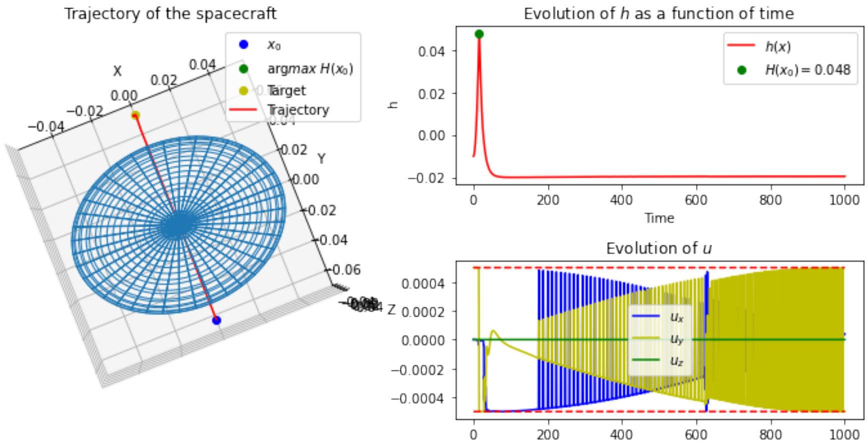

Figure 37: With (4.4)

Now we are able to reach the target point. We can therefore deduce that the problem does not come from how our CLF is defined. Let us return to our initial optimisation problem without the constraint in H. The idea would be to play on the constants k_i in order to have the existence of a solution. Indeed, we have

$$V(y,t) = \frac{1}{2}\|r - r_p(t)\|_2^2 + \frac{1}{2}k_2\|v - k_1(r - r_p(t))\|_2^2.$$

and for a certain control u

$$\dot{\tilde{V}}(y,t) = k_1\|r - r_p\|^2 + k_1k_2\|v - \dot{r}_p - k_1(r - r_p)\|^2 < 0.$$

Taking $0 < k3 \leq -2k1$ and the control u to obtain $\dot{\tilde{V}}$, we have

$$\begin{aligned}\dot{V} + k_3V + \delta &= \dot{\tilde{V}} + k_3V + \delta \\ &= \dot{\tilde{V}}\left(1 + \frac{k_3}{2k_1}\right) \leq 0\end{aligned}$$

because $\dot{\tilde{V}} \leq 0$ and $\frac{k_3}{2k_1} \geq -1$. It has therefore been shown that, given a certain set of parameters, there is a control that satisfies the constraint on V. However, even with these parameters, we still do not obtain the desired result.

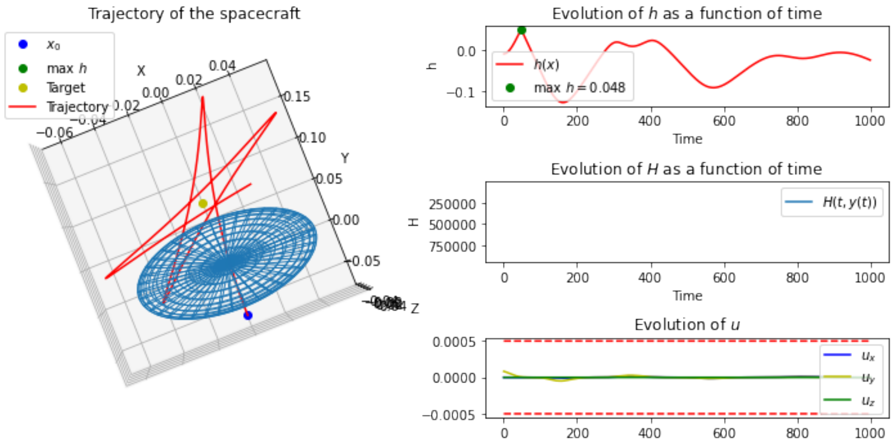

Figure 38: Without H and $k_3 = -1.5k_1$

For the first simulations, we took target points and target trajectories on the surface of the sphere to avoid. So we never had to stabilise on them. The method proposed in the article does not allow stabilisation on a point outside the sphere to be avoided. According to the results obtained, it should be possible to exchange the optimisation problem between (3.8) and (4.4) when the risk of collision is small enough.

For that we add a positive constant m. If we have $H \leq -m$ then we use the (3.8) optimisation problem, otherwise we use the usual optimisation problem, i.e. (4.4). In the two figures below, we can see that the target is reached with different values of m.

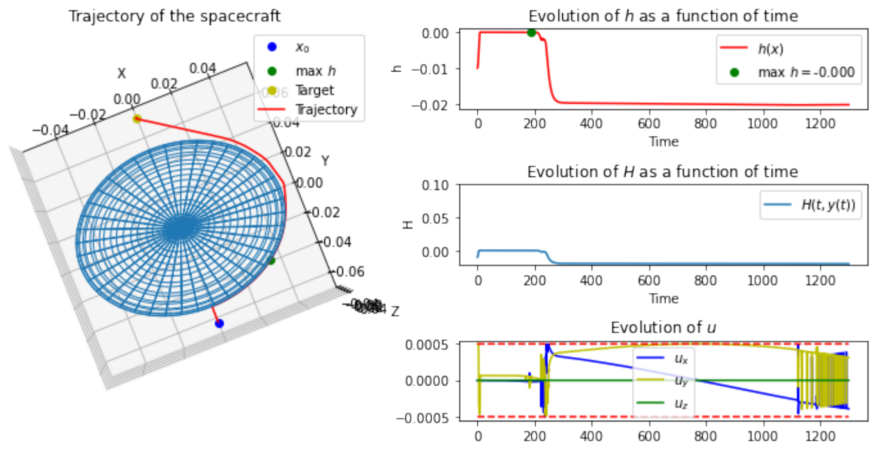

Figure 39: $m = 0.002$

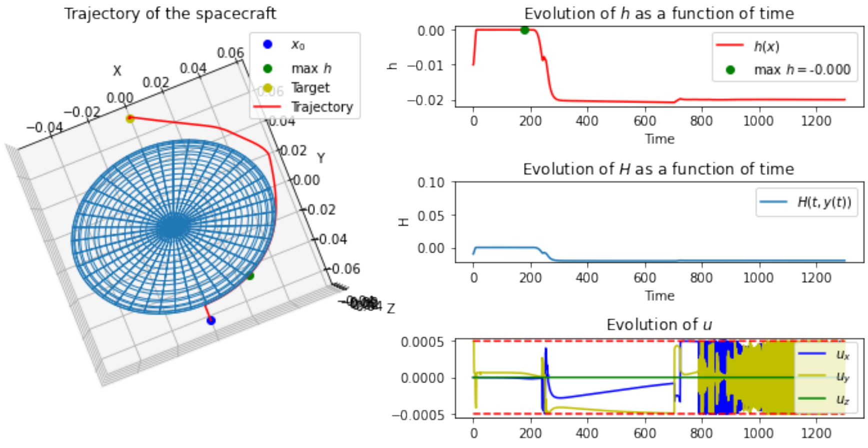

Figure 40: $m = 0.008$

However, the proposed solution is not at all satisfactory. Indeed, if we take a closer look at the evolution over time, we observe a lack of stability in a neighbourhood of the target. The control keeps oscillating while we would like it to stabilise at 0 because the spacecraft is on the same circular orbit as the moving frame.

5 Conclusion

Throughout the project and the internship, we have relied on the formulation of ZCBFs introduced by the [3] paper. This allowed us to develop a strategy to make our safety zone forward invariant. As the article was not very precise in some places, we had to reason from time to time with intuition. In the last case studied during the project, we can notice a small conflict in the simulations which can be problematic. The further away the target point is, the more conflict there will be between the attraction emitted by V and the repulsion emitted by H. As a consequence, the computation time is significantly higher. Thus, the ideal way to reach a target point at the other end of the satellite would be to set intermediate targets over time.

This is how the target trajectory comes into play. However, it was necessary to modify the Lyapunov function stated in the paper. Indeed, the velocity of the target trajectory was missing and without this term, it was very difficult to approach our goal. Finally, we adapted the previous work to our main objective, that of avoiding the collision of two satellites in orbit around a planet or an asteroid. Nevertheless, the case studied in the article has some limitations and cannot be directly applied to our case. The strategy developed in [3] allows to approach a target without reaching it because this same target is located in the zone that we try to avoid. However, in our case, we want to move a satellite in relation to another so that it reaches a target allowing it to remain in a stable position. Thus, we had to slightly modify our optimisation problem to overcome the previous problem. Nevertheless, the proposed solution has some limitations.

The code in **.ipython** and **.py** format can be found on the following GitHub repository: *https://github.com/master-csmi/2022-m1-cbf* https://github.com/master-csmi/2022-m1-cbf.

References

[1] CVXPY: A Python-embedded modeling language for convex optimization, 2022.

[2] H. Bauschke and J. Borwein. Dykstra 's alternating projection algorithm for two sets. *Journal of Approximation Theory*, 79(3):418–443, 1994.

[3] J. Breeden and D. Panagou. High relative degree control barrier functions under input constraints. *Proceedings of 60th IEEE Conference on Decision and Control*, 2021.

[4] H. D. Curtis. Orbital mechanics for engineering students. *Embry-Riddle Aeronautical University Daytona Beach, Florida*, 2010.

[5] E. D. Sontag. Feedback stabilization of nonlinear systems. In M. A. Kaashoek, J. H. van Schuppen, and A. Ran, editors, *Robust control of linear systems and nonlinear control (vol. 2 of proceedings of MTNS'89)*, pages 61–81, Basel-Boston, 1990. Birkhäuser.