Formation tracking of target moving on natural surfaces

Housseyne Nadour, Nicolas Marchand, Lionel Reveret, Pierre Legreneur

Formation tracking of targets moving on natural surfaces*

H. Nadour, N. Marchand, L. Reveret and P. Legreneur

Abstract— **In this paper, we focus on formation control of a swarm of UAVs tracking a moving target (typically a climber) on a non regular mountainous surface. The main objective is to show how to dispatch the whole swarm of the quadrotors around the target with respect to the natural environment (mountain) taking advantage as much as possible from the free space around the climber. The formation shape adapts with the neighbour's environment topology, so that it avoids collision with the environment and the occlusion of the target from the UAV's cameras sights, and ensures as well the formation compactness and its closure to the target respecting the agent-agent and the agent-target security distances.**

Based on a free and safe hovering space (SHS), the proposed work proposes two algorithms to determine for each agent its desired position within the SHS according to some maximum occupation criterium. The main contribution resides in handling complex obstacle constraints while maintaining visibility of the target.

Keywords: UAVs control, formation control, target tracking, surveillance, film-making, 3D reconstruction, monitoring.

I. INTRODUCTION

The collaboration between multiple UAVs is vital in many complex tasks. Its efficiency is demonstrated in the literature [1] and deployed for various objectives, including but not limited to collaborative reconnaissance, target tracking, identification and collaborative combat as well in the process of executing combat missions [2]. Most of the collaborative tasks studies seek to maintain some desirable geometric configuration. Deep-space interferometry is an example for geometric keeping [3], where a fleet of networked air-crafts are required to perform a sequence of formation manoeuvres while maintaining a relative attitudes accurately. [4] is another good example of keeping the formation persistent and rigid in the course of its moving using distributed cohesive motion control. In [5], adaptive leader-following approach is deployed combined with artificial potential field (APF) in order to control a triangular formation a with collision avoidance using repulsive potential. However, the above mentioned works, among others, deals with a finite number of simple-shaped obstacles, for which it is intuitive to seek for rigid formation with respect to the leader or the target for the most cases. Furthermore, these works focuses mainly on tracking the target/leader, demonstrating the coordination between the agents, without discussing what reason compels the formation to take some shape or another one. In this paper, we study the possibility of having a formation that is not static and may depend upon the obstacles and the tasks of the formation.

By contrast, Our work needs the formation shape to be adapted with environment topology in which the team of UAVs is dispatching. so the formation must be adjusted automatically and continuously depending on the free space between the natural surface on which the target is moving, in order to ensure clear vision for each UAV's camera without occlusion, and no-collision, achieving some other criteria as well.

Beside target tracking and formation control, this paper subject interests with other fields like surveillance and monitoring of moving objects. It is notable that most of surveillance studies [6]–[10], deal with urban environments, where the common objective is to monitor a static or moving target between urban buildings by means of single or multiple UAVs with on-board cameras. [8] proposed a novel occlusion-aware surveillance algorithm based on decomposition of the surveillance problem into a variant of the 3D art gallery problem and an instance of travelling salesman problem for Dubins vehicles, the algorithm yields a covering vantage points, each of which must be travelled by at least one UAV in a shortest path without need for coordination between agents. It is similar for [6] where he tried to keep a close line of sight from the UAV to the target in a densely populated area, his focus was to find optimal flight paths for UAVs to minimize the chance of losing the moving target under mild assumptions. In order to minimise the occlusion, the UAVs turn on circle path with the minimum turn radius and the maximum allowed altitude, the one thing to be studied is the centre of the circle for which the occlusion is minimised furthermore. Whereas [9] addressed the same problems together with other objectives like predicting the target states based on extended Kalman filter combined with probability estimation, considering the line of sight occlusion of buildings, as well as the energy consumption.

These proposed solutions, among many others, are dedicated for urban environments where it is assumed the surface consists only of the base plane and quadrangular prisms lying on the base plane (i.e the target moves in planar path between buildings), and all UAVs fly at some altitude over the buildings which makes them free from natural obstacles. In addition, as the agents are so far from the target and the terrain, these facts make the targets always under the camera sight, i.e the relative speed between the target and the UAV's eyes is low, for which, in most cases the target can be even considered as static, the same points are noticed for [10] where the author did not even need to consider the occlusion and the collision avoidance, meanwhile our issue compels the UAVs to be very close to the target (5-20 m) to monitor a

*This work was supported in part AirCap Regional project funded by French ARA Region.

H. Nadour and N. Marchand are with Grenoble-Alpes Univ., Grenoble-INP, GIPSA-lab, UMR5216, France. housseyne.nadour, nicolas.marchand@gipsa-lab.fr

P. Legreneur is with University Lyon I, LIBM, France. Pierre.legreneur@univ-lyon1.fr

L. Reveret is with INRIA Rhône-Alpes - Morpheo Team, lionel.reveret@inria.fr

target moving on natural no-planar surface (mountainous surface), this fact increases mainly the probability of occlusion of the line of sight, and the collision with the mountainous surface, which push us to exclude these solutions adapted for the usual surveillance tasks where the agents hover out of the terrain folds, especially in urban environment.

One of the current applications of UAVs, that is under study and development is the tracking of a climber moving on a surface by a swarm of mobile cameras [11]. Where the objective behind this tracking is to capture his motion while climbing the surface. Thanks to some 3D reconstruction algorithms the set of the synchronised videos are processed at the end of the experiment in order to derive the digital model of his motion that is used to study some bio-mechanical characteristics of his movements, like velocity, acceleration, his locomotion, as well as his 3D trajectory using a standard DLT (Direct Linear Transform) reconstruction [11]. The biochemical study of the climber motion can be used for multiple purposes, like enhancing his movements, or stimulating the climber movement on real robots. The process can be expanded later to capture animals movements in naturals environments.

Our objective in this paper is to show how to dispatch the whole swarm of the monitors (UAVs) around a climber target 1 with respect to the natural environment (mountainous surface) taking advantage as much as possible from the free space around the climber, for which it is necessary to take in consideration the whole geometry of the surface without neglecting its curvatures and folds by dividing the environment into framework of small entities like Octomap [12], or framing it within simple shapes like convex polyhedra [13], [14].

Approximating the environment to simple shapes is convenient with the usual objectives of path planning: (i) minimising the length of the overall path and (ii) maximising the margin of safety from obstacles [15]. By contrast, we seek in this paper, in addition to avoiding collisions, to avoid the occlusion, as well as to ensure a maximum safe view range between the two extreme agents in order to allow capturing the motion from multiple perspectives dispersed as much as possible, so that the 3D reconstruction is enhanced and the most out of bio-mechanical characteristics are extracted.

II. Problem Statement and Assumptions

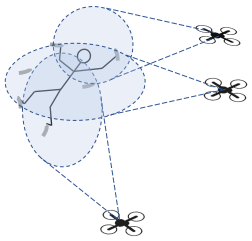

Fig. 1. Schematic view of the climber following task

The problem, that we focus on in this paper, is the tracking of some moving target evolving non regular

Fig. 2. The typical surface of the tracking, [16]

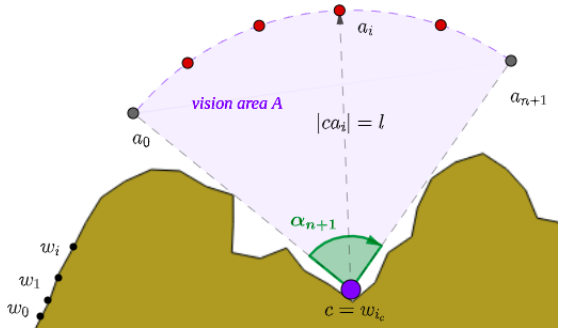

Fig. 3. free vision area

continuous surface. As illustrated in Fig.1, the typical case of study in this paper will be capturing a climber on a wall that may contain ridge, chimneys and off-width due to fissures and cracks in the rock Fig.2. The tracking is performed using embedded cameras on the UAVs and therefore the two main objectives are: avoiding obstacles and keeping the target in the field of view. We proceeded at first to make some assumptions and objectives. Consider a swarm S of m agents $\{a_0, ..., a_{n+1}\}$ moving in $\Re^2$ space . We want them to be well spread around the climber while he is on the wall, The wall $\mathscr{W}$ in this example is a random continuous line, and the point c (climber), moves on the line as indicated in Fig.3.

1) We suppose at first that the best distribution for the inter agents is when they are spread evenly around the climber, i.e they have equal inter-angles, this is an arbitrary choice. Otherwise, the choice is subjected to a cost function that must be minimised in order to maximise the amount of extracted bio-mechanical information.
2) It is desired to maximise the range of view, i.e to have a maximum possible angle α_{n+1} for the view area $\mathscr{A}$ in order to take advantage as much as possible of the surrounding space and therefore to monitor the target from diverse perspectives.
3) Each agent i must hover freely at some distance d_i far away from the climber, d_i is the trade off between the following points:
   - the target must be safe i.e d_i must be no lesser then the safety distance d_c.
   - the swarm formation compactness and its closure to the target must be guaranteed.
   - avoid collision between agents by respecting the minimum separating agent-agent distance d_a.
4) The swarm is delimited by a free region $\mathscr{A}$ that protects the swarm from colliding with the surface

and the occlusion of vision. Accordingly, we introduce two extreme fictive agents a_0 and a_{n+1} that are meant to delimit the boundaries of $\mathscr{A}$.

From these points, it turns out that the region $\mathscr{A}$ we are looking for is a sector, i.e a region between an arc and two radii which its vertices that are the climber c, and the two fictive agents a_0 and a_{n+1}, see Fig.3.

III. Determining the safe hovering space

The cartography is a digital model, which is an ordered list of points $\mathscr{W} = \left(w_0, ..., w_i, ...w_m\right)^t$, we suppose the wall is continuous, the continuity of a digital model here will mean that the distance between any tow adjacent points is inferior then some very tiny distance e, i.e $|w_i - w_{i+1}| < e$. Smoothness property can be added to avoid discontinuity phenomena.

As discussed previously, the desired area $\mathscr{A}$ to be found, is delimited by a sector centreed at c and has a maximum possible view angle between a_0 and a_{n+1}, for which the space is free from possible collision or occlusion of sight, theses constraints define ca_0 and ca_{n+1}, see Fig. 4. If it is the case, then the safe hovering space is defined as:

$$SHS = \{p \in \mathscr{A} \ , \ |p| > d_c\} \tag{1}$$

The two extreme points a_0 and a_{n+1} can be determined based on the viewshed computing [17], the viewshed $\mathscr{V}$ of a point c on a surface $\mathscr{W}$ is defined as:

$$\mathscr{V}(c) = \{v \in \mathscr{W} \ , \ v \text{ is visible from } c\}$$

Considering just the area surrounding the target c, within a circle of radius l, then :

$$\mathscr{V}(c) = \{v \in \mathscr{W} \ , \ |c - v| \leqslant l \text{ and } v \text{ is visible from } c\}$$

That is to say the two lines ca_0 and ca_{n+1}, of length l, must pass through the two points $v_r \in \mathscr{V}$ and $v_l \in \mathscr{V}$ respecting equation (2), each of which corresponds to the maximum sight angles β_r and β_l resp Eq. 3.

$$\begin{cases} a_0 & = c + \frac{l}{|v_r - c|} \cdot (v_r - c) \\ a_{n+1} & = c + \frac{l}{|v_l - c|} \cdot (v_l - c) \end{cases} \tag{2}$$

Let the two viewshed subsets $\mathscr{V}_r$ and $\mathscr{V}_l$ be defined as:

$$\mathscr{V}_{r/l} = \{v \in \mathscr{V} \ , \ v \text{ is on the right (resp left) of } c\}$$

And let $\beta(v)$ define the sight angle associated with the point v, then:

$$\begin{cases} \beta(v_r) \geqslant \beta(v) \ ; \ \forall v \in \mathscr{V}_r \\ \beta(v_l) \geqslant \beta(v) \ ; \ \forall v \in \mathscr{V}_l \end{cases} \tag{3}$$

The angle β is defined by the cross product and the dot product of $\vec{u}_t$, the unit vectors tangent at c, with the sight unit vector $\vec{cv} = \frac{1}{|v-c|} \cdot (v - c)$:

$$\begin{cases} \cos(\beta) = \vec{u}_t \cdot \vec{cv} \\ \sin(\beta) = (\vec{u}_t \times \vec{cv}) \cdot \begin{pmatrix} 0 & 0 & \delta \end{pmatrix}^t \\ \beta = \arctan(\sin(\beta), \cos(\beta)) \end{cases} \tag{4}$$

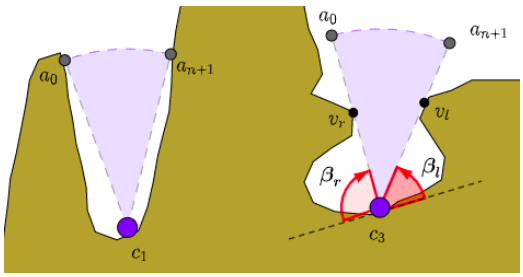

Fig. 4. v_r and v_l are defined by the collision and the occlusion facts.

Such that $\delta = -1$ if $v \in \mathscr{V}_r$ otherwise $\delta = 1$. The vectors $\vec{u}_t$ and $\vec{cv}$ must be extended to 3D ones with equal z-components in order to apply the cross-product.
Notice that as arctan function, we mean the one returning values in the interval $[-\pi, \pi]$, while the point v can make angles out of this range. Nevertheless, the angle β can be calculated using numerical integration $\beta = \sum \delta\beta$ during the algorithm iterations.

IV. Formation shape within SHS

Let ca_0 and ca_{n+1} be two vectors defined as:

$$ca_0 = a_0 - c \text{ and } ca_{n+1} = a_{n+1} - c \tag{5}$$

Each agent must be at the following position:

$$a_i = c + d_i.F_i.\frac{ca_0}{|ca_0|} \tag{6}$$

Where a_i is the position of the agent i, with F_i is a transformation matrix defined as below:

$$F_i = \begin{pmatrix} \cos(\alpha_i) & \sin(\alpha_i) & 0 \\ -\sin(\alpha_i) & \cos(\alpha_i) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{7}$$

while:

$$\alpha_i = \frac{i}{n+1}.\alpha_{n+1} \tag{8}$$

The range (angle) of $\mathscr{A}$ is defined by the cross-product and the dot-product of ca_0 and ca_{n+1}:

$$\alpha_{n+1} = \arctan\left((ca_0 \times ca_{n+1}) \cdot (0,0,1)^t, ca_0 \cdot ca_{n+1}\right) \tag{9}$$

The term d_i is the agent-target distance, and must ensure the closure of the agents to the target (compact formation), that is to say the average must be minimized (equation (10)) respecting both the safety agent-target distance d_c (equation (11)) and the safety agent-agent distance d_a (equation (12)) by Al-Kashi Theorem (Fig. 6(b)), as well as the boundaries limits on the left and the right (equations (13)-(14), Fig. 6(c)) :

$$\min_{d_1 \ldots d_n} J = \sum_{i=1}^{n} d_i \tag{10}$$

$$d_i \geqslant d_c \quad , \quad \forall i \in \{1 \ldots n\} \tag{11}$$

$$d_i^2 + d_j^2 - 2.d_i.d_j.\cos\left(\frac{|i-j|}{n+1}.\alpha_{n+1}\right) \geqslant d_a^2 \tag{12}$$
$$\forall i, j \in \{1 \ldots n\}, i \neq j$$

$$d_i \geqslant \frac{2}{a.\sin(\alpha_i)} \ , \ \forall i \in \{1 \ldots n\} \tag{13}$$

$$d_i \geqslant \frac{2}{a.\sin(\alpha_{n+1} - \alpha_i)} \ , \ \forall i \in \{1 \ldots n\} \tag{14}$$

This is a nonlinear optimisation problem that can be resolved using Non Linear Programming solvers like YALMIP [18]. However the computation of non linear programming is very slow which conducts us to propose another algorithm that yields a sub-optimal solution and guaranties the formation compactness and its closure to the target respecting the agent-agent and the agent-target security distances d_a and d_c, as well as the boundaries limits.

The algorithms proceeds in two steps, the first one aims to rearrange the agents in a list P_r that determines for each agent its degree of priority to be closer to the target Fig. 5, in other words, P_r maps form a given degree of priority dg_i to the number of its corresponding agent: $P_r : dg_i \mapsto i$. The second step computes, orderly and for each agent in P_r, the maximum possible constrained distance among all possible constrained distances Fig. 6(a), the two steps are explained below:

1) Step 1: In this step, the group of the whole agents are separated successively multiple times until we get N groups of one agent; Firstly, the swarm (array of N elements) is divided into two new groups (arrays) based on the parity of agents positions (index), the even positions group is prior. Then, each new group itself is separated again into two new groups based on the parity of the position, this process is reproduced until we get an array P_r of N groups of one agent, with ordered priorities, Fig. 5 illustrates the process of selections for five agents that results $P_r = (4,\ 2,\ 3,\ 5,\ 1)^t$. Notice here that this step can be executed once and offline, because P_r does not depend on any variable but only the invariable number of agents.

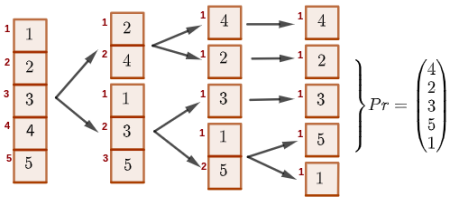

Fig. 5. Successive separations to determine the priority list; a black number i refers to the agent number that corresponds to the angle α_i, a red number refers to an agent position in a group.

2) Step 2: This process part encompasses all agents in P_r, starting orderly from the first prior one $P_r(1)$ until the last one $P_r(N)$. For each agent $i = P_r(dg_i)$ the algorithm computes:

- All possible distances d_i^j, constrained by prior agents $j = P_r(dg_j < dg_i)$; Each d_i^j is computed via Al-Kashi theorem (Fig .6(b)):

$$\begin{cases} d_i^2 + d_j^2 - 2.d_i.d_j.\cos(\delta\alpha) = d_a^2 \\ \text{with } \delta\alpha = \frac{\alpha_{n+1}}{n+1}.|i-j| \end{cases} \tag{15}$$

  Obviously the chosen solution is that that verifies $a_i \geqslant a_j$. If the discriminant $\Delta = d_a^2 - d_j^2 \sin^2(\delta\alpha)$ is negative we assign any value $\leqslant d_c$.
- The two distances d_i^l and d_i^r constrained by the left and the right boundaries; when an agent i is on contact with left or right limit then the distances are given in the two right triangles:

$$\begin{cases} d_i^r = \frac{a}{2.\sin(\alpha_i)} \\ d_i^l = \frac{a}{2.\sin(\overline{\alpha_i})} \end{cases} ;\ \overline{\alpha_i} = \alpha_{n+1} - \alpha \tag{16}$$

Then it picks up the maximum distance among all constrained distances: $d_i = \max\left\{d_i^j, d_i^l, d_i^r, d_c\right\}$.

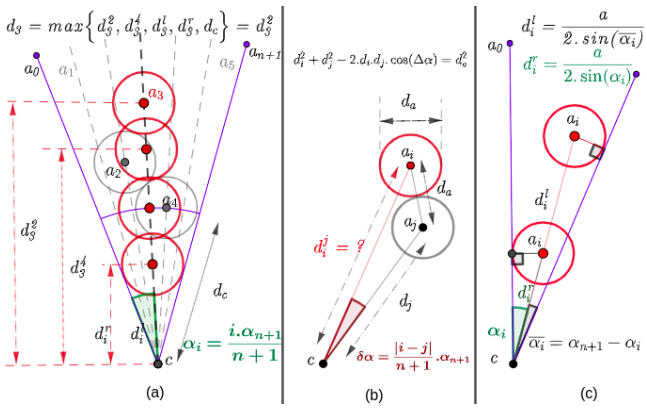

Fig. 6. (a): Different possible distances for a_3 constrained by a_2, a_4, d_c, and left and right boundaries (d_i^r, d_i^l), with: $P_r = (4,\ 2,\ 3,\ 5,\ 1)^t$, (b): The triangle (ca_ia_j) on which Al-Kashi theorem is applied to calculate d_i^j, (c): shows how to calculate d_i^r and d_i^l .

A. The camera set angles

Each agent holds a camera on the axis x_3, it is clear then x_3 must shooting toward the target, let be Ψ_i the angle made between the vector $a_ic = c - a_i$ and x_0 of the world frame R_0 around z_0, then the desired yaw for each agent is

$$\psi_i^* = \Psi_i \tag{17}$$

V. Control, tracking

A. The dynamic model of each agent

In this section, we design low-level controllers for the individual agents. We first describe the dynamic modelling of quadrotor vehicles and then design control laws for trajectory tracking, which will be used to track the trajectories generated by Eq.6. Each agent is a quadrotor vehicle pushed by four intrinsic forces $\vec{f}_{i\in\{1\ldots4\}}$ generated by four rotors Fig.7. The different combinations of the forces gives four degrees of freedom. The body frame R_3

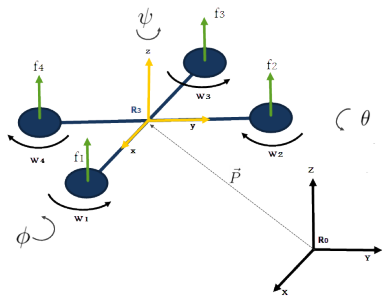

Fig. 7. Quadrotor model in the space

rotates around the intermediate frame R_2 by an angle ψ, which itself rotates around another intermediate frame R_1 by an angle θ, the last one turns around R_0 by an angle

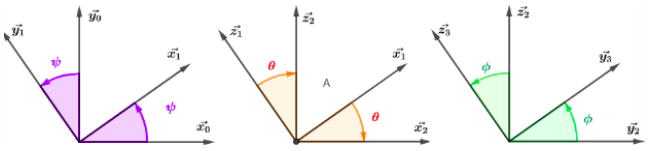

Fig. 8. The quadrotor rotations in the space

ψ as described in Fig. 8. The quad-rotor areal vehicles dynamic is represented in the literature by quaternion form model [19], or by the matrix form model [20], the last one takes the following from:

$$m\ddot{p} = z_3 . \sum f_i - m.g.z_0 \tag{18}$$

$$\dot{\omega} = I^{-1}\left(-\omega \times (I\omega)\right) + I^{-1}\tau + I^{-1}\tau_g \tag{19}$$

where: p is the vehicle centre position in the inertial frame, f_i is the thrust generated by the rotor i, m is the body mass, g is the gravity acceleration, ω is the angular velocity expressed in the body frame R_3 by $(\omega_x, \omega_y, \omega_z)$, $I = \text{diag}\,(I_{xx}, I_{yy}, I_{zz})$ is the inertia matrix assuming that the intrinsic axis are symmetric axis (I is diagonal), $\tau = (\tau_x, \tau_y, \tau_z)^t$ is the intrinsic applied torque, τ_g is the gyroscopic torque:

$$\tau_g = I_r.\omega \times W = (\omega_y.w_g, -\omega_x.w_g, 0)^t \tag{20}$$

with $w_g = w_1 - w_2 + w_3 - w_4$, $W = w_g z_3$ and the w_is are the rotation speed of the blades.The inputs of the system are:

$$\begin{cases} u_1 = \sum f_i \\ \begin{pmatrix} u2 \\ u3 \\ u4 \end{pmatrix} = \tau = \begin{pmatrix} d(f_2 - f_4) \\ d(f_3 - f_1) \\ -k_d\left(w_1^2 + w_3^2 - w_2^2 - w_4^2\right) \end{pmatrix} \end{cases}$$

where $f_i = k_l.w_i^2$, with k_l is the lift coefficient, and w_i is the rotor i angular velocity, k_d is the drag coefficient. Equations (18)-19 can be developed as below:

$$\begin{cases} \ddot{x} = (\cos(\phi)\sin(\theta)\cos(\psi) - \sin(\phi)\sin(\psi))\,\frac{u1}{m} \\ \ddot{y} = (\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi))\,\frac{u1}{m} \\ \ddot{z} = \cos(\phi)\cos(\theta)\,\frac{u1}{m} - g \end{cases} \tag{21}$$

$$\begin{cases} \dot{\omega}_x = -\frac{I_{zz}-I_{yy}}{I_{xx}}\omega_y\omega_z + (I_{xx}I_r)^{-1}\omega_y w_g + I_{xx}^{-1}\tau_x \\ \dot{\omega}_y = -\frac{I_{xx}-I_{zz}}{I_{yy}}\omega_z\omega_x - (I_{yy}I_r)^{-1}\omega_x w_g + I_{yy}^{-1}\tau_y \\ \dot{\omega}_z = -\frac{I_{yy}-I_{xx}}{I_{zz}}\omega_x\omega_y + I_{zz}^{-1}\tau_z \end{cases} \tag{22}$$

Bouabdallah *et al.* approximated in [21] the attitude dynamic around the equilibrium point, i.e for $\theta \approx 0$ and $\phi \approx 0$, as below:

$$\begin{cases} \ddot{\phi} = \frac{I_{yy}-I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} + \frac{1}{I_r.I_{xx}}\dot{\theta}w_g + \frac{d}{I_{xx}}u_2 \\ \ddot{\theta} = \frac{I_{zz}-I_{xx}}{I_{yy}}\dot{\psi}\dot{\phi} - \frac{1}{I_r.I_{yy}}\dot{\phi}w_g + \frac{d}{I_{yy}}u_3 \\ \ddot{\psi} = \frac{I_{xx}-I_{yy}}{I_{zz}}\dot{\phi}\dot{\theta} + \frac{d}{I_{zz}}u_4 \end{cases} \tag{23}$$

The inputs are decoupled, and the model is easy to be handled, for that reason this approximation has been used widely in the literature [22]–[24]. The demonstration in appendix shows that the following equations are a better approximation:

$$\begin{cases} \ddot{\phi} = \left(1 + \frac{I_{yy}-I_{zz}}{I_{xx}}\right)\dot{\psi}\dot{\theta} + \frac{1}{I_r I_{xx}}\cdot\dot{\theta}w_g + \frac{d}{I_{xx}}u_2 \\ \ddot{\theta} = \left(-1 + \frac{I_{zz}-I_{xx}}{I_{yy}}\right)\dot{\phi}\dot{\psi} - \frac{1}{I_r I_{yy}}\dot{\phi}w_g + \frac{d}{I_{yy}}u_3 \\ \ddot{\psi} = \left(1 + \frac{I_{xx}-I_{yy}}{I_{zz}}\right)\dot{\theta}\dot{\phi} + \frac{d}{I_{zz}}u_4 \end{cases} \tag{24}$$

B. Individual control

Each agent has two dynamics, the dynamic of the attitude whose the inputs are u_2, u_3 and u_4 (equation (19)), and the dynamic of the translation (equation (18)) whose the inputs are u_1, ϕ and θ. It is convenient to deploy cascade control that involves two controllers, one of which nestling inside the other, such that the outer controller (position controller) feeds the inner one (attitude controller) with the set references (θ^* and ϕ^*), Fig.9. It's a cascade control. Such a system can give an improved response to disturbances [25].

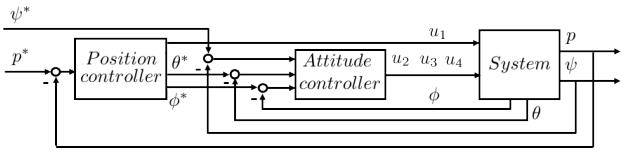

Fig. 9. synoptic scheme of the controller

1) Controlling the attitude: let be w_ϕ,w_θ,w_ψ the new inputs, such that:

$$\begin{cases} u_2 = \frac{I_{xx}}{d}\left(w_\phi - \frac{I_{yy}-I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} + b_\phi.\dot{\theta}\cdot w_g\right) \\ u_3 = \frac{I_{yy}}{d}\left(w_\theta - \frac{I_{zz}-I_{xx}}{I_{yy}}\dot{\psi}\dot{\phi} - b_\theta.\dot{\phi}\cdot w_g\right) \\ u_4 = \frac{I_{zz}}{d}\left(w_\psi - \frac{I_{xx}-I_{yy}}{I_{zz}}\dot{\phi}\dot{\theta}\right) \end{cases} \tag{25}$$

We get a decoupled linear system with respect to the new inputs, composed of three 2nd order systems :

$$\begin{cases} \ddot{\phi} = w_\phi \\ \ddot{\theta} = w_\theta \\ \ddot{\psi} = w_\psi \end{cases} \tag{26}$$

We refer by ϕ^* to the set point of ϕ, so if we take:

$$w_\phi = k_{\phi_1}e_\phi + k_{\phi_2}\dot{e}_\phi \text{ such that : } e_\phi = \phi^* - \phi \tag{27}$$

then the characteristic polynomial is :

$$s^2 + k_{\phi_2}.s + k_{\phi_1} \tag{28}$$

where the parameters k_1 and k_2 are tuned so that the system is stable. The same strategy is applied on θ and ψ.

2) Controlling the position: As previous, let $A_x = \ddot{x}$, $A_y = \ddot{y}$, and $A_z = \ddot{z}$ be the new inputs of the translation dynamic. A PID controller is applied on each of these linear subsystems, i.e:

$$\begin{cases} A_x = k_{x_1}e_x + k_{x_2}\dot{e}_x \\ A_y = k_{y_1}e_y + k_{y_2}\dot{e}_y \\ A_z = k_{z_1}e_z + k_{z_2}\dot{e}_z \end{cases} \tag{29}$$

where $e_x = x^* - x$, $e_y = y^* - y$, $e_z = z^* - z$. with x^*, y^*, and z^* are the set points. It should be noticed that the inner controller must be faster than the controllers of x and y.

These controllers give the necessary acceleration A (or the necessary force $\vec{F} = A.m$) that must be applied on the body frame.

3) Extracting u_1, θ^* and ϕ^* from the position PID-controllers outputs: On one hand, we have z_3 and z_0 expressed in the frame R_1 as following:

$$z_3 = \begin{pmatrix} \cos(\phi)\sin(\theta) \\ -\sin(\phi) \\ \cos(\phi)\cos(\theta) \end{pmatrix}_{R_1} ; z_0 = z_1 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}_{R_1} \quad (30)$$

which allows to rewrite equation (18) in the frame R_1 as below:

$$\ddot{p} = \frac{u_1}{m} \begin{pmatrix} \cos(\phi)\sin(\theta) \\ -\sin(\phi) \\ \cos(\phi)\cos(\theta) \end{pmatrix}_{R_1} - g. \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}_{R_1} \quad (31)$$

On the other hand, the position controller gives the required acceleration expressed in R_0:

$$\ddot{p} = A \quad (32)$$

If $(A_x, A_y, A_z)^T$ is the coordinates of A in R_0, then its coordinates in R_1 is

$$A = \begin{pmatrix} A_x.\cos(\psi) + A_y.\sin(\psi) \\ -A_x.\sin(\psi) + A_y.\cos(\psi) \\ A_z \end{pmatrix}_{R_1} \quad (33)$$

Which yields according to equation (31)-33:

$$\begin{pmatrix} \cos(\phi)\sin(\theta) \\ -\sin(\phi) \\ \cos(\phi)\cos(\theta) \end{pmatrix} \frac{u_1}{m} = \underbrace{\begin{pmatrix} A_x.\cos(\psi) + A_y.\sin(\psi) \\ -A_x.\sin(\psi) + A_y.\cos(\psi) \\ A_z + g \end{pmatrix}}_{Ag=(Ag_x, Ag_y, Ag_z)^t}$$

Then we can conclude the set points for the attitude controller:

$$\begin{cases} \theta^* & = \arctan\left(Ag_x, Ag_z\right) \\ & = \arctan\left(A_x.\cos\left(\psi\right) + A_y.\sin\left(\psi\right), A_z + g\right) \\ \phi^* & = \arctan\left(-Ag_y, \frac{Ag_z}{\cos(\theta^*)}\right) \\ & = \arctan\left(A_x.\sin\left(\psi\right) - A_y.\cos\left(\psi\right), \frac{A_z+g}{\cos(\theta^*)}\right) \end{cases}$$

The input u_1 is preferred to be concluded from the third line $\cos(\phi)\cos(\theta) = Ag_z$ in order, firstly, to avoid singularity caused by sin function around the equilibrium points $\phi \approx 0$ and $\psi \approx 0$, and secondly, to involve directly θ and ϕ instead of θ^* and ϕ^* in the following equation:

$$u_1 = \frac{(A_z + g).m}{\cos\left(\theta\right).\cos\left(\phi\right)} \quad (34)$$

This makes the control of z independent from the inner controller (attitude controller), and as result the control of z is more performing.

VI. Simulation

The algorithms are validated using ROS together with SimuLink. The swarm consists of 5 agents trying to capture the movement of the climber where he is moving horizontally on 2D surface at a desired constant height $z = 20$ Fig .10.

The PID controllers parameters are tuned so that each vehicle followed its assigned position $a_i(t)$ (equation (6)) as quickly as possible 11-12, the swarm respected the desired specifications and was able to maintain its desire form i.e. to be inside SHS, within compact formation while respecting the security distances. As one might expect, the curvature discontinuity increases greatly the discontinuity of the set trajectory a_i assigned to the agent i, and consequently:

- the rotational movements θ and ϕ are increased and become so dynamic which effects the camera shooting quality on the target, it appears in form of noises.
- in some cases, some agents are outside of SHS while they are trying to track it and be within it, this fact is natural when the swarm has a dynamic and cannot move instantly contrary to SHS.

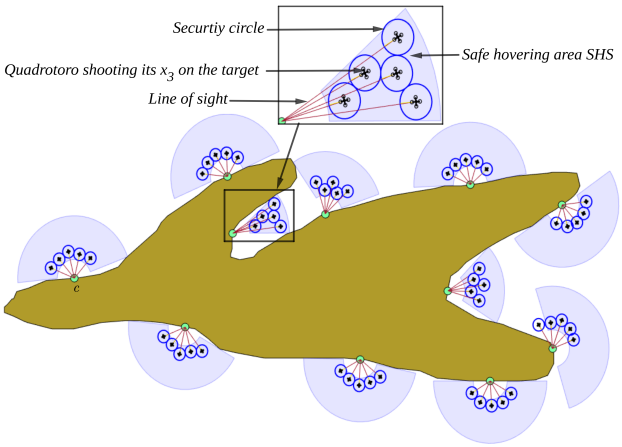

Fig. 10. The formation while capturing the climber motion

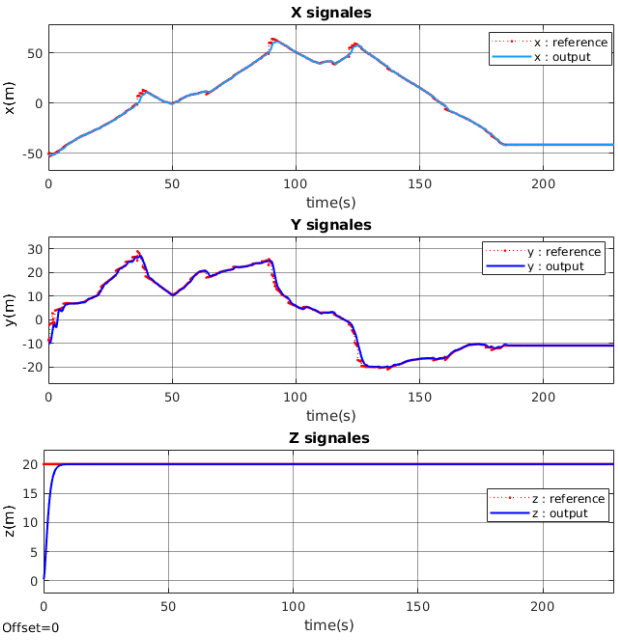

Fig. 11. UAV_1: Tracking simulation results for x, y, and z

VII. Conclusions and Perspectives

We presented the basic idea for controlling a formation of UAVs monitoring a target moving on a non regular mountainous surface. we showed mainly how to determine the safe hovering space SHS where the whole agents can be dispatched around the target (climber) taking advantage as much as possible from the free neighbour space around the climber, so that the formation shape

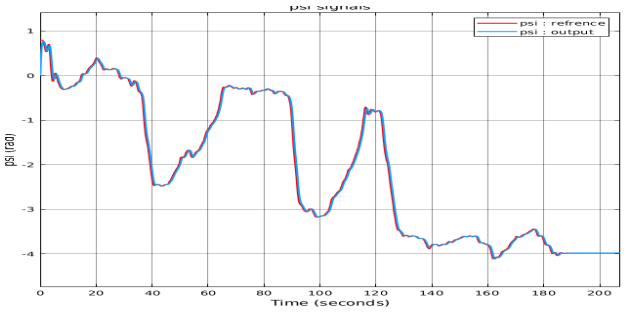

Fig. 12. UAV_1: Tracking result for ψ

adapts with the neighbour environment topology, in order to avoid both collision with the wall and occlusion of the line of sight. Besides, In order to ensure the formation compactness respecting the the limits of $\mathscr{A}$ together with the agent-agent and the agent-target security distances, a sub-solution was proposed and tested with sufficiency, instead of using Non Linear Programming that might not be practical. This work represents one ring among many others that will be put together to accomplish the final protocol to monitor the target autonomously, for instance:

- The strategy will be extended for 3D environment.
- An intermediate safe corridor spaces between free regions $\mathscr{A}$ are necessary to remedy some problem due to the curvature discontinuity of the wall.
- The safe hovering space $\mathscr{A}$ could be in some cases smaller then the minim required surface to store the whole formation, which compels to add a complementary protocol to full this gap.
- A protocol that guides each agent to the target neighbourhood within the SHS is necessary before assigning for each agent its suitable position (α_i and d_i) within SHS.
- The path given to each agent should be redressed (optimised), in order to reduce some effects due the curvature discontinuity like high perturbation on the attitude of the agents which effects the quality of monitoring the climber.
- Replacing the hypothesis of predefined cartography by sensors, this point will change the algorithm to compute SHS and holds technical difficulties.
- Integrating a bio-mechanical cost-function that disperses the agents on the angular coordinates instead of taking arbitrary choices equation (8).
- Generalising the idea of computing SHS for discontinuous surfaces.

REFERENCES

[1] S. Pirbhulal, W. Wu, G. Li, and A. K. Sangaiah, "Medical information security for wearable body sensor networks in smart healthcare," *IEEE Consumer Electronics Magazine*, vol. 8, no. 5, pp. 37–41, 2019.

[2] J. Zhang, J. Yan, and P. Zhang, "Multi-uav formation control based on a novel back-stepping approach," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 2437–2448, 2020.

[3] W. Ren, "Formation keeping and attitude alignment for multiple spacecraft through local interactions," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 2, pp. 633–638, 2007.

[4] I. Bayezit and B. Fidan, "Distributed cohesive motion control of flight vehicle formations," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 12, pp. 5763–5772, 2012.

[5] Q. Shi, T. Li, J. Li, C. P. Chen, Y. Xiao, and Q. Shan, "Adaptive leader-following formation control with collision avoidance for a class of second-order nonlinear multi-agent systems," *Neurocomputing*, vol. 350, pp. 282–290, 2019.

[6] J. Kim and Y. Kim, "Moving ground target tracking in dense obstacle areas using uavs," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 8552–8557, 2008.

[7] P. Yao, H. Wang, and H. Ji, "Multi-uavs tracking target in urban environment by model predictive control and improved grey wolf optimizer," *Aerospace Science and Technology*, vol. 55, pp. 131–143, 2016.

[8] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, "Autonomous uav surveillance in complex urban environments," in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2. IEEE, 2009, pp. 82–85.

[9] C. Hu, Z. Zhang, N. Yang, H.-S. Shin, and A. Tsourdos, "Fuzzy multiobjective cooperative surveillance of multiple uavs based on distributed predictive control for unknown ground moving target in urban environment," *Aerospace Science and Technology*, vol. 84, pp. 329–338, 2019.

[10] H. Wang, J. Liu, and J. Han, "Rs-cps: A distributed architecture of robotic surveillance cyber-physical system in the nature environment," in *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*. IEEE, 2015, pp. 1287–1292.

[11] L. Reveret, S. Chapelle, F. Quaine, and P. Legreneur, "3D motion analysis of speed climbing performance," in *4th International Rock Climbing Research Congress*, Chamonix, France, Jul. 2018. [Online]. Available: https://hal.inria.fr/hal-01963008

[12] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems," in *In Proceedings of the ICRA Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, vol. 2, 2010.

[13] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[14] D. Mellinger, A. Kushleyev, and V. Kumar, "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 477–483.

[15] S. Mittal and K. Deb, "Three-dimensional offline path planning for uavs using multiobjective evolutionary algorithms," in *2007 IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 3195–3202.

[16] Tristan Higbee – CC BY 2.0, "Tips and techniques on how to climb chimneys," Sep 15, 2017, [accessed July, 2020]. [Online]. Available: https://www.outdoorrevival.com/adventure/tips-techniques-climb-chimneys.html

[17] M. A. Magalhaes, S. V. Magalhaes, M. V. A. Andrade, and J. Lisboa Filho, "An efficient algorithm to compute the viewshed on dem terrains stored in the external memory." in *IX Brazilian Symposium on GeoInformatics*, 2007, pp. 183–194.

[18] J. Löfberg, "Yalmip : A toolbox for modeling and optimization in matlab," in *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[19] E. Fresk and G. Nikolakopoulos, "Full quaternion based attitude control for a quadrotor," in *2013 European Control Conference (ECC)*, 2013, pp. 3864–3869.

[20] A. Mokhtari and A. Benallegue, "Dynamic feedback controller of euler angles and wind parameters estimation for a quadrotor unmanned aerial vehicle," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 3, 2004, pp. 2359–2366 Vol.3.

[21] S. Bouabdallah, A. Noth, and R. Siegwart, "Pid vs lq control techniques applied to an indoor micro quadrotor," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, 2004, pp. 2451–2456.

[22] H. Mo and G. Farid, "Nonlinear and adaptive intelligent control techniques for quadrotor uav–a survey," *Asian Journal of Control*, vol. 21, no. 2, pp. 989–1008, 2019.

[23] Y. Wang, J. Sun, H. He, and C. Sun, "Deterministic policy gradient with integral compensator for robust quadrotor control," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–13, 2019.

[24] H. Bouadi, M. Bouchoucha, and M. Tadjine, "Sliding mode control based on backstepping approach for an uav type-quadrotor," *World Academy of Science, Engineering and Technology*, vol. 26, no. 5, pp. 22–27, 2007.
[25] W. Bolton, *Instrumentation and Control Systems (Second Edition)*. Newnes, 2015, ch. 13.5. Cascade Control, pp. 281–302.

APPENDIX

THE PROOF OF THE APPROXIMATE MODEL

The body has three rotations around z_0,y_1 and x_2:

$$\omega = \underbrace{z_0.\dot{\psi}}_{\omega_{z_0}} + \underbrace{y_1.\dot{\theta}}_{\omega_{y_1}} + \underbrace{x_2.\dot{\phi}}_{\omega_{x_2}} \tag{35}$$

When the pitch and the roll (Fig.8) are almost null, we get:

$$\begin{cases} x_3 \underset{*_1}{=} x_2 \underset{*_2}{\approx} x_1 \quad ; (*_1 : \forall\phi \quad) ; (*_2 : \theta \approx 0) \\ y_3 \underset{*_3}{\approx} y_2 \underset{*_4}{=} y_1 \quad ; (*_3 : \phi \approx 0) ; (*_4 : \forall\theta) \\ z_3 \underset{*_3}{\approx} z_2 \underset{*_2}{\approx} z_1 \underset{*_5}{=} z_0 \quad ; (*_5 : \forall\psi) \end{cases} \tag{36}$$

Equations (35) and (36) give the approximation of $\vec{\omega}$ in R_3:

$$\omega \approx z_3.\dot{\psi} + y_3.\dot{\theta} + x_3.\dot{\phi} \approx \begin{pmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{pmatrix}^t_{R_3} \tag{37}$$

Next, the derivation of $\vec{\omega}$ is:

$$\dot{\omega} = z_0.\ddot{\psi} + y_1.\ddot{\theta} + x_2.\ddot{\phi} + \dot{z_0} \cdot \dot{\psi} + \dot{y_1} \cdot \dot{\theta} + \dot{x_2} \cdot \dot{\phi} \tag{38}$$

Given the velocity of the rotation axis:

$$\begin{cases} \dot{z_0} = 0 \\ \dot{y_1} = \omega_{z_0} \times y_1 = \dot{\psi} \cdot z_0 \times y_1 \\ \dot{x_2} = (\omega_{z_0} + \omega_{y_1}) \times x_2 = \left(\dot{\psi} \cdot z_0 + \dot{\theta} \cdot y_1\right) \times x_2 \end{cases} \tag{39}$$

Then equation (39) and the approximations (36) yield:

$$\begin{cases} \dot{z_0} = 0 \\ \dot{y_1} \approx \dot{\psi} \cdot z_3 \times y_3 = -\dot{\psi} \cdot x_3 \\ \dot{x_2} \approx \left(\dot{\psi} \cdot z_3 + \dot{\theta} \cdot y_3\right) \times x_3 = \dot{\psi} \cdot y_3 - \dot{\theta} \cdot z_3 \end{cases} \tag{40}$$

Then we get the angular acceleration approximation in R_3

$$\begin{aligned} \dot{\omega} &\approx z_3.\ddot{\psi} + y_3.\ddot{\theta} + x_3.\ddot{\phi} - x_3.\dot{\psi} \cdot \dot{\theta} + \dot{\phi} \cdot \left(\dot{\psi} \cdot y_3 - \dot{\theta} \cdot z_3\right) \\ &\approx \begin{pmatrix} \ddot{\psi} - \dot{\theta} \cdot \dot{\psi} \\ \ddot{\theta} + \dot{\psi} \cdot \dot{\phi} \\ \ddot{\psi} - \dot{\phi} \cdot \dot{\theta} \end{pmatrix}_{R_3} \end{aligned} \tag{41}$$

Finally, by substituting equations (37) and (41) into (22), we get (24).

Validation of the approximate model (24)*:*

The two models (23) and (24) are tested in this section using the matrix form inverse dynamic model defined as:

$$\begin{cases} R = f(\varepsilon) \\ \omega_\times = R^{-1} \cdot \dot{R} \\ \tau = \omega \times I.\omega + I.\dot{\omega} - \tau_g \end{cases} \tag{42}$$

With $\varepsilon = (\psi, \theta, \phi)$, while f is a function that maps from ε to its corresponding rotation matrix R (in matlab $f = eul2rotm$). The idea behind this validation is to generate a rotational movements ε^* so that at some moment $t = t_0$ we have:

$$\begin{cases} \phi^*(t_0) = 0 \text{ and } \theta^*(t_0) = 0 \\ \dot{\psi}^*(t_0).\dot{\theta}^*(t_0).\dot{\phi}^*(t_0) \neq 0 \end{cases} \tag{43}$$

The following angular movement is sufficient:

$$\varepsilon^* = \begin{pmatrix} 1, 1, 1 \end{pmatrix}^t \frac{1}{10}. \arctan\left(150. \left(t - 0.5\right)\right)$$

The inverse dynamic model Eq.42 is introduced to calculate the reference torque τ^* which is necessary to regenerate the desired rotational movements using the approximated direct dynamic models Eq.23 and Eq.24. So each approximated model is fed with the Eular angular velocities $\dot{\varepsilon}^*$ and the torque τ^* as it is illustrated in Fig.13. The comparison is done between the outputs

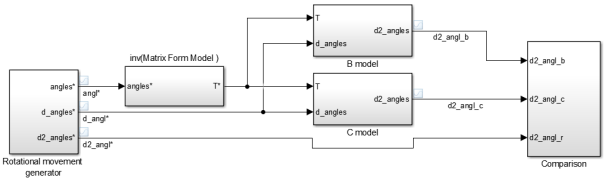

Fig. 13. Simulation schema; (B) model = Eq.23, (C) model= Eq.24, $angles^*$= ε^*; $dangles^*$ = $\dot{\varepsilon}^*$, $d2angles^*$= $\ddot{\varepsilon}^*$

(i.e the regenerated Euler angular acceleration $\ddot{\varepsilon}$) of the approximated direct dynamic models and the reference Euler angular acceleration vector $\ddot{\varepsilon}^*$, the validity of each model is confirmed if the following errors:
$E^\psi = \left(\ddot{\psi}^* - \ddot{\psi}\right)^2$, $E^\theta = \left(\ddot{\theta}^* - \ddot{\theta}\right)^2$, $E^\phi = \left(\ddot{\phi}^* - \ddot{\phi}\right)^2$

- are null for $t = 0.5$
- they are closed to be null at the neighbourhood of $t = 0.5$.

The errors $E^{\psi,\theta,\phi}_{b,c}$ are plotted in Fig.14. The errors of the model Eq.24 are null for $t_0 = 0.5$ and closed to be null at the neighbourhood, contrary to the model Eq.23. Hence the validity of the established model in this paper.

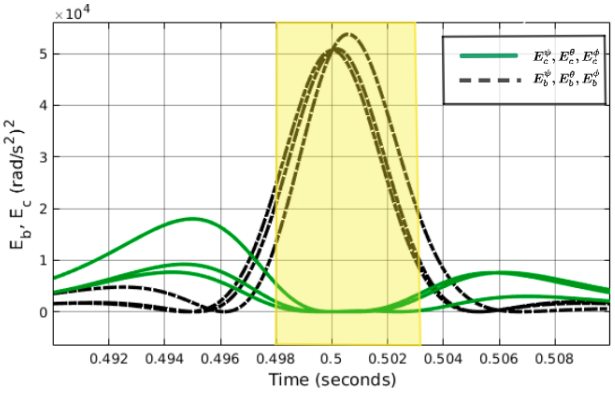

Fig. 14. E_c and E_b represent resp the models Eq.24 -23 errors.