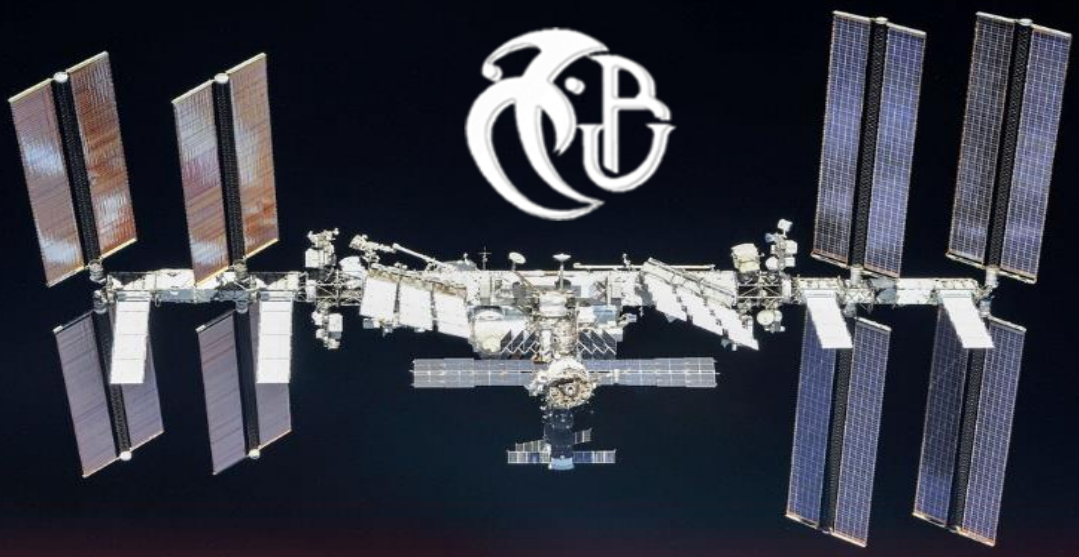
People's Democratic Republic Of Algeria

Ministry Of Higher Education And Research

University Of Saad Dahleb, Blida 1

The Institute of aeronautic and space studies



Memory submitted

Linear multivariable compensator design for flexible structure satellite

Major of Avionics

Presented by :

Belmahdjoub Sid Ali

Anane Ishak

Before The Jury Composed of :

Mr.BEKHITI Belkacem	Dr	USDB	Supervisor
Mr.DILMI Smail	Dr	USDB	CO-Supervisor
Mr.		USDB	Examiner
Mr.		USDB	Examiner
Mr.		USDB	Examiner

Acadimic Year 2019-2020

إهداء

بسم الله الرحمن الرحيم

قال الله تعالى: ﴿ وقضى ربك ألا تعبدوا إلا إياه و بالوالدين احساناْ ﴾

أهدي هذا العمل المتواضع إلى الوالدين الكريمين حفظهما الله

إلى إخوتي

إلى أحمد و خليل و أسماء

إلى الأ خ الصغير عبد الرحمان

حفظهم الله جميعا وأجزل لهم المثوبة والله أسأل أن يتولاهم في الدنيا و الآخرة وان يجعلهم ممن

إذا أعطي شكر وإذ

إبتلي صبر وإذا أذنب استغفر وأن يسبغ عليهم نعمه ظاهرة و باطنه

إلى كل الأصدقاء و النفوس الطيبة

إلى كل أساتذتي إلى كل من درسني وعلمني وأعانني من قريب أو من بعيد

إلى من وسعتهم ذاكرتي و لم تسعهم مذكرتي .

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to the following individuals who have helped me accomplish my dissertation for the master degree.
At the forefront, I would like to thank my dissertation supervisor, ***Dr.Bekhiti belkacem*** *for his continuous support and guidance throughout the dissertation stages. He has given his knowledge and put in the effort at all times for the benefit of this dissertation. would also like to acknowledge the immense impact and support that my family has given me, including my father and mother whose encouragement was beyond description.*

Table of Contents

List of Figures

Nomenclature

Q : value of the point mass.
L : distance of the point mass from the satellite center of mass.
L_p : moment of inertia of the movable material frame with respect to the x-axis.
L_y : moment of inertia of the rigid body.
L_z : moment of inertia of the rigid body.
τ : time constant.
C_z : the damping factor with respect to the z-axis.
C_y : the damping factor with respect to the y-axis.
K_y : the spring constant with respect to the y-axis.
K_z : the spring constant with respect to the z-axis.
ω_n : natural frequency of the appendages.
ξ : damping factor.
$I_{\acute{x}}$: the moments of inertia of the undeformed satellite with respect to the x-axis.
$I_{\acute{x}}$: the moments of inertia of the undeformed satellite with respect to the y-axis.
$I_{\acute{x}}$: the moments of inertia of the undeformed satellite with respect to the z-axis.
H : momentum with respect to the body axes.
G :is a gain.
ϕ, ψ, θ: Euler angles.
ϕ : pitch, θ : roll, ψ : yaw.
s : Laplace variable.
A : state matrix.
B : input matrix.
C : output matrix.
D : input-output matrix.
$[A1A2A3A4A5A6A7A8A9A10A11A12A13A14A15A16A17]$: state coefficient.
$[B1B2B3B4]$: input coefficient.
$\vec{a}_G$: absolute linear acceleration vector of **B** in R_G.
$\vec{a}_p$:absolute linear acceleration vector of **B** at **p**.
$\vec{\omega}$: absolute angular velocity vector of R_p w.r.t R_i .
$\vec{F}_{ext}$: external forces vector applied to the hub **B**.
$\vec{T}_{ext}$: external torque vector applied to the hub **B**.
D_G^B:the static-dynamic model of **B** at point **G**.
D_G^A:the static-dynamic model of **A** at point **G**.
$\mathbb{T}_{GP}$:transport the direct dynamic model of a hub body **B** from a point **G** to a point **P**.
$\vec{r}_{GP}$:vector position of point **P** in frame R_G
$\vec{r}_{cp}$:vector position of point **C** in frame R_P
$\mathbb{J}_G^B$: moment of inertia tensor of the main body with respect to **G** written in R_G.
$\vec{V}_G$: absolute linear velocity vector of **B** in R_G.
$\vec{V}_P$: absolute linear velocity vector of **B** in P.
$\vec{F}_P$: reaction forces vector between hub/appendage.
$\vec{T}_P$: reaction torque vector between hub/appendage.
Q_P: is called generalized force.
D_P^A:the static-dynamic model of **A** at point **P**.
D_P^B:the static-dynamic model of **B** at point **P**.
$\mathbb{T}_{CP}$:transport the direct dynamic model of an appendage **A** from a point **C** to a point **P**

$[r_{GP}] \in R^{3\times3}$: outer product of vector $\vec{r}_{GP}$.
$[r_{CP}] \in R^{3\times3}$: outer product of vector $\vec{r}_{CP}$.
$\mathbb{J}_C^A$: moment of inertia tensor of the appendage with respect to **C** written in R_P.
$\mathbb{I}_P$=$\mathbb{J}_C^A$-$m_A[r_{GP}]^2$: moment of inertia tensor of the appendage about **P**.
$I_{3\times3}$: is the 3×3 identity matrix.
D=diag($2\xi_i\omega_i$): : flexible mode damping ratios.
K=diag($\omega_i{}^2$): : flexible stiffness coefficients.
$C_m(i)$: is the 3 drive torques applied at the pivot joint between an appendage and the hub.
$0_{3\times3}$: is the 3×3 zero matrix.
ξ_P: position vector of flexible modal coordinates.
L_P: the modal contributions of beams at point **P**.
$M_P^A(s)$: direct transfer matrix of the appendage.
ω_i: frequency of the flexible mode.
ξ_i: damping ratio of the flexible mode.
s : laplace variable.
k : number of flexible mode.
θ_{A1}, θ_{A2}: angle of rotation of solar panels.
θ_{ar}: pointing angle of an antenna reflector.
m_A: mass of antenna main reflector.
m_{sa}: mass of each solar panel (Solar Array).
m : mass of satellite main body.
TM: Direction cosine matrix of the rotation from frame **RG** to frame **RP**.

Acronyms

DOF: Degree Of Freedom.
AOCS: Attitude and Orbit Control System.
FMS: Flexible Multibody Systems.
TITOP: Different Two-Input Two-Output Port.
PSO: particle swarm optimization.
ACO: ant colony optimization.
MIMO: multi input multi output.
SISO: single input single output.

Abstract

The kinematic model of a body is a system of equations that define the position in space of each particle of the body from the values of a set of variables. The time variation of these variables describes the motion of the body. After, we review the different ways to describe the rigid motion and how to express all the kinematic variables and their time derivatives that we need later to formulate the dynamics. Finally, the kinematics of the flexible body are described by adding a set of deformation functions on to the rigid body kinematics, Many multibody dynamics software are available to build such kind of models but they address the nonlinear behavior and they are too much loud to be handled at the early prototyping phase. in the present work we starts by defining the metaheuristic optimization algorithm used later in PID tuning and after that we have two linear aproche the direct state space methode with three degree of freedom and free hybrid model with six degree of freedom after that in each model we apply the diffirent control thecnique proposed, this fact leads clearly to an improvement of the pointing accuracy. finaly a comparative stady between the two aproaches

Keywords: Structures flexibles, Dynamic Model, Linear system, Pivot Joint, Modélisation multi-corps, Cantilever Hybrid Model, Decoupling Control, MIMO-PID, PSO algorithm, ACO algorithm

General introduction

introduction

DESIGNING satellites with various missions such as climatic, military, geology and astrological missions will cause the payloads being increased and as a result, the increase of their dimensions, weight and consumed power. So, for more energy absorption, the effective section surface of satellite should be increased to installing more solar panels. On the other side, the existent limitations on satellites launch will cause the restriction on their volume and weight[8]. To decrease the volume of satellites, they are designed as a concentrated structure with some supplemental parts which are fastened before launch and are opened after settlement in orbit, and to decrease the weight of satellites, the light materials are used in designing structures. The whole of these factors, means light weight, low volume and large section surface will cause the flexibility of satellites structure. In this case, preserving correct direction of main body and flexible parts encounter with many challenges. According to these realities, many theoretical researches have been done to identify and control flexible structures [30, 31]. In the past three decades, the flexible satellites which are known as big spatial structures in some articles are considered a lot. In some NASA reports, the effect of satellites flexibility in attitude control system, as unusual acts, has been mentioned [25]. More researches on this issue have specified that the reason of this strange act is the flexibility of the structure which will be intensified in some cases by attitude control system [26]. Before 70's decade, the attitude control and stabilization systems of satellites were designed according to dynamic modeling of rigid bodies and Single-Input SingleOutput (SISO) controllers. Along with the development of spatial sciences in the late 70's, big satellites which have flexible parts and include many sensors and actuators were considered. So, the need for using complex control laws and Multi-Input Multi-Output (MIMO) control systems for satellites were found to be essential[27] . For high-accuracy performance in pointing, three-axis attitude control will be used for satellites which lead to a MIMO control system.

Chapter 1

Dynamics and Control of Flexible Space Vehicles

1.1 Introduction

SPACECRAFT are very complex mechanical multi-body systems including flexible and/or rotating appendages. The design of the AOCS requires a linear model taking into account all the rigid and flexible couplings between the hub (where the AOCS acts) and the various appendages. Note that the linear assumption is quite realistic for such systems since perturbations and so motions are very small (except for very dexterous observation satellites). This linear assumption is furthermore valid in the field of future missions for deep space exploration involving formation flying of several spacecraft. For this kind of formation flying mission, it is more and more accepted that the 3 rotation D.O.F. and the 3 translation D.O.F. must be treated all together (Gaulocher et al. 2005). Therefore, a 6 D.O.F. model including couplings between rotations and translations must be developed. Lots of multibody softwares are available to build such kind of models but they address the nonlinear behavior and they are too much loud to be handled at the early prototyping phase. So a tool is required to develop quickly the dynamic model and to prototype the AOCS or to analyze and to optimize the main dynamic parameters of the mechanical structure or AOCS and finally to assess the global performance of the system. (Khalid H.M. 2008)

1.2 Spacecraft Attitude Actuators:

We can conceptually divide actuator types into two general classes, passive and active. Passive actuators operate more or less open loop. In other words, after the spacecraft is in the desired attitude, passive actuators will keep it there with little or no additional torques needed. Active actuators, on the other hand, require continuous feedback and adjustment.

1.2.1 Dampers:

A damper is another actuator usually used in combination with others for a complete system. Generally speaking, a damper is a device that changes angular momentum by absorbing energy. We know momentum is constant only as long as energy stays constant. If we add or take away energy, then a momentum will change. As a spacecraft attitude actuator, dampers absorb unwanted momentum.

1.2.2 Thrusters:

are perhaps the simplest types of active actuator to visualize. Thrusters are simply rockets that rely on "brute force" to rotate a spacecraft. By applying a balanced force with a pair of rockets on opposite sides of a spacecraft, we can produce a torque, as shown in Figure below. By varying which thruster pair we use and how much force we apply, we can rotate a spacecraft in any direction. Placing thrusters as far from the satellite's center of mass as possible gives them a larger moment arm and allows them to exert a greater torque for a given force. This is evident from the important concept we saw earlier. The greater the distance over which a force is applied, the more torque is delivered from the same force. However, as we learned earlier, because of precession, when a spacecraft is already spinning, any applied torque in a direction other than the spin axis causes the spacecraft to rotate at constant velocity about an axis perpendicular to the torque direction. The biggest advantage of using thrusters is that they can produce a well-defined "torque on demand," allowing the spacecraft to slew quickly from one attitude to another.

1.2.3 Momentum-control Devices:

The most common actuator for spacecraft attitude control is a family of systems that all rely on angular momentum. These momentum-control devices actively vary the angular momentum of small, rotating masses within a spacecraft to change its attitude. Typically, an attitude control system uses at least three separate reaction wheels, oriented at right angles to each other, as seen in Figure.

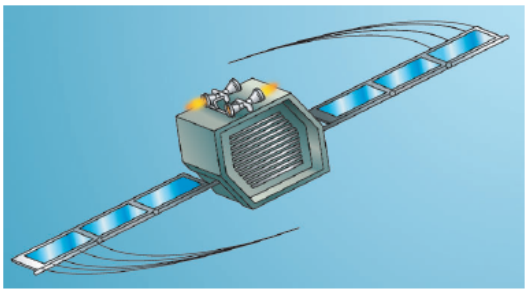


Figure Thrusters. Thrusters are rockets that apply a force some distance away from the center of mass, causing a torque that rotates the spacecraft.

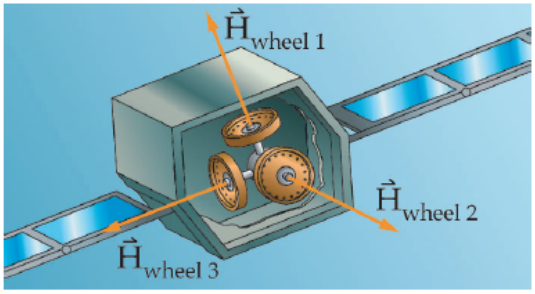


Figure Reaction Wheels. Reaction wheels are part of a zero-bias system that uses three independent wheels, one along each axis, normally with zero or nearly zero momentum. To rotate the spacecraft or absorb disturbance torques, one or more wheels begin to spin. Often, designers add a fourth wheel, skewed with respect to the other three, for redundancy.

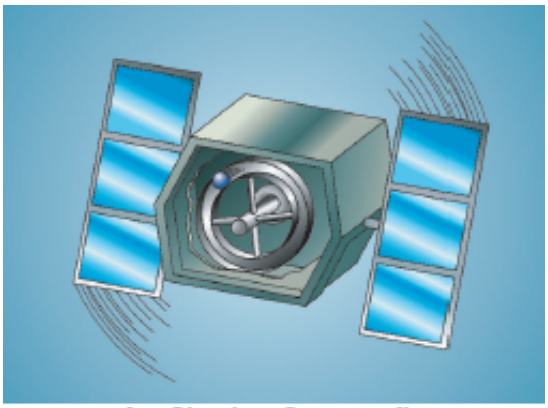


Figure A Simple Spacecraft Damper. Dampers "absorb" unwanted angular momentum by converting the energy into friction, in much the same way as the brakes in a car turn linear momentum into heat through friction. A ball inside a circular tube filled with a viscous fluid is one type of damper. As the spacecraft rotates, the ball moves through the fluid. The resistance produces heat, dissipating the angular motion.

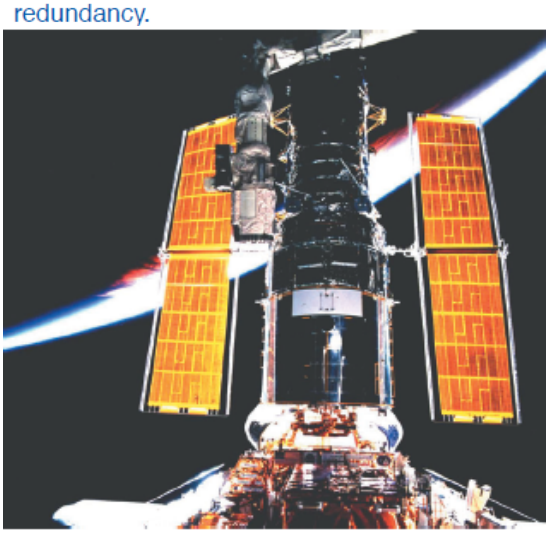


Figure 4.3.1-35. Reaction Wheels for Accurate Pointing. The Hubble Space Telescope observes many interstellar objects at such long distances that it must point very accurately. To be this accurate, it relies on very accurate reaction wheels. *(Courtesy of NASA/Johnson Space Center)*

1.3 Sensors Technologies:

This section shows different types of sensors used in attitude determination algorithms. The different types of sensors (gyro, sun sensor, earth sensor, star tracker) discussed in this section are commercially available. For all of them different suppliers exist capable to offer products based on the same technology and a wide offer in terms of accuracy [9] , field of view (FOV), mass, power consumption and cost.

1.3.1 Gyroscopes:

Gyroscopes (gyros) are inertial sensorsmeasuring the angular rate or the incremental angular rotation about its measurement axis. Two basic types of gyro exist: the rate gyro and the rate integrated gyro. first the classic mechanical gyro technology is based

on a rapidly spinning mass that senses changes in the inertial orientation of the gyro spin axis. Any satellite rotation about the gyro's input axis realizes a torque on the gyro output axis that causes an angular rotation or another physical reaction which is measured in output. second the rate gyro sensors (RG)measure the spacecraft rotation rate. They are relatively inexpensive and they can be used for the attitude stabilization (e.g. in a PID controller) or the rate damping modes (see [9] Chap. 4). All these sensors are affected by different error sources like angular drift errors that can be estimated and then calibrated using optical attitude sensor measurements such as those provided by Earth-Sun sensors or star trackers. The rate integrating gyros (RIG) measure the angular displacement,They aremore expensive than rate gyro sensors and are used in attitude determination algorithms.

1.3.2 Rate Integrating Sensor (Fiber Optic Gyro):

The first optical gyroscope was described by Sagnac [44] in 1913, under the form of a ring interferometer. the most precise sensors among any other types of gyroscopes[42] Sagnac an optical interferometer enclosing a surface located on a rotating support will detect a phase difference of the optical signal, which is proportional to the angular rate and to the surface enclosed by the optical path. A ring interferometer rotation detector (gyroscope) using optical fiber waveguide is designed. The sensitivity of the device is enhanced via multiple traverses of counterrotating beams around an area, but restrictions on the optimum fiber length are imposed by the photon noise limit. A well-defined wavefront and efficient coupling of light into the fiber are required. Laser light divided by a beam splitter is focused on the terminations of single-mode fibers. After exiting, the light is returned through the fibers in the opposite direction and the beams are recombined at the beam splitter, with the image magnified and displayed. Displacement of one end of the fiber from the focal point of the converging lens produces growing fringes as an error signal. The system is sufficiently sensitive for navigation applications, is free from pulling and lock-in characterizing ring laser systems (and hence can detect very low angular velocities), but is inferior to the ring laser in ultimate theoretical sensitivities ,Fiber optic gyro (FOG) technology (i.e. continuously operating for a long period of time, typically 15 years) for space applications can provide very good performances in all the main requirements [9] .

1.3.3 Sun Sensor:

The Sun sensors are the most commonly used attitude sensors in safe modes (see [9] Chap. 4 the SAM-EM mode). When the mission is not very demanding in terms of accuracy the sun sensors can be selected also for the Normal Pointing Mode attitude determination.The Sun sensor measures the incident sunlight direction with respect to the sensor optical reference plane. The design is generally simple and reliable and its algorithms are robust, simple and inexpensive. There are two main types of Sun sensors: the analog and the digital sensors. First the analog sun sensor typically has an analog output signal that is a function of the Sun incidence angle. Analog sensors are based on photocells that have a current output proportional to the cosine of the angle between the Sun's direction and the perpendicular to the detectors.Second The digital Sun sensors are based on different types of detectors. The most common detectors are Charged-Coupled Device or Active Pixel Sensor. The sensor is typically equipped with a measurement unit designed around digital devices like Application Specific Integrated Circuit or Field

Programmable Gate Array, able to calculate the Sun's incidence angle relative to the perpendicular of the sensor mounting surface when the Sun is in the sensor's field of view (FOV) and to provide this information to the AOCS. Moreover, the sensor provides other information like Sun presence, detector temperature, secondary voltage and other housekeeping information that can be used by the AOCS to determine the attitude of the spacecraft and to detect sensor failures. The digital Sun sensor is more expensive than the analog Sun sensor but ensures a more accurate performance (i.e. a few hundredths of degree instead of a few tenths of degree).

Solar panels can also be used as sun sensors. The currents from the different solar panels are monitored. Accuracies of 1° are obtainable ([24]) . As the sun is the only reference used, only the pointing direction toward the sun can be determined.

1.3.4 Star Tracker:

Star sensors represent today themost accurate attitude sensors for demanding satellite missions. The basic principle of a star sensor is based on the accurate measurement of the star directions in Body Reference Frame that is compared with the known accurate star reference direction in Earth Centered Inertial frame available in the on board catalog. Using both measured and known star directions an attitude estimation algorithm like QUEST(see [23],[24])can be used to obtain a three axis measurement of the satellite attitude in Earth Centered Inertial frame. Star sensors are divided in two major categories: scanners and fixed head star trackers. Scanners, popular 20–30 years ago, can be used when the satellites are spinning. In this case, the sensor uses the spin of the spacecraft for the searching and scanning function when stars pass through multiple slits in the scanner's field of view. Today the more used star sensors are the fixed-head star trackers (single or multiple heads) that, thanks to the more powerful processors available in today space application, are able to electronically implement a search and track function over a limited field of view. These sensors are able to acquire, select and recognize stars from a lost-in-space attitude, with relatively high tumbling rates in few seconds with an high value of probability of success. After the attitude acquisition, the star trackers are today able to maintain autonomously the tracking of stars (up to 15 stars or more for each head) and on this basis are able to provide a very accurate attitude information.

Chapter 2

Metaheuristic Optimization Algorithm

2.4 Introduction:

METAHEURISTIC algorithms become an important part of modern optimization. A wide range of metaheuristic algorithms have emerged over the past two decades, and many metaheuristics algorithms is becoming more and more popular.optimization is an important subject with many important application[28], and algorithms for optimization are diverse with a wide range of successful applications [29, 16] Among these optimization algorithms, modern metaheuristics are becoming increasingly popular, leading to a new branch of optimization,called metaheuristic optimization, most metaheuristic algorithms are nature-inspired [17, 18, 34], from simulated annealing [34] to ant colony optimization [17], and from particle swarm optimization [32] to cuckoo search . Since the appearance of swarm intelligence algorithms such as PSO in the 1990s, more than a dozen new metaheuristic algorithms have been developed and these algorithms have been applied to almost all areas of optimization, design, scheduling and planning, data mining, machine intelligence, and many others. Thousands of research papers and dozens of books have been published

2.5 Particle swarm optimization(PSO):

2.5.1 Introduction:

PARTICLE swarm optimization algorithms were introduced in 1995 byKennedy and Eberhart as an alternative to standard genetic algorithms. These algorithms are inspired by swarms of insects (or schools of fish or flocks of birds) and their coordinated movements, in fact, just as these animals move in groups to find food or avoid predators, particle swarm algorithms seek solutions for an optimization problem. The individuals in the algorithm are called particles and the population is called a swarm.[5], Swarm-based algorithms emerged as a powerful family of optimization techniques, inspired by the collective behavior of social animals. In particle swarm optimization (PSO) the set of candidate solutions to the optimization problem is defined as a swarm of particles which may flow through the parameter space defining trajectories which are driven by their own and neighbors best performances. (Federico Marini). In this algorithm, a particle decides its next movement based on its own experience, which in this case is the memory of the best position it has encountered, and based on its best neighbor. The new speed and direction of the particle will be defined according to three trends: the propensity to follow its own path, its tendency to return to its best position reached and its tendency to go to its best neighbor.

2.5.2 Principle:

The principle of the PSO developed by Kennedy and Eberhart [32] is based on the behavior of flocks of birds. Thus, PSO was fundamentally developed through the simulation of the behavior of flocks of birds in two-dimensional space. The position of each agent is represented by its coordinates along the two axes X and Y, to which the speeds expressed by Vx (speed along the X axis) and Vy (speed along the Y axis) are associated. The modification of the behavior of each agent is based on the position and speed information. At each iteration the agent proceeds via an objective function to the evaluation of its best value until the (best) and its positive along the two axes X and Y. This information is obtained from an analysis of the personal experiences of each agent. In addition, each agent knows the best overall value of the group (gbest) among the (pbest). This information represents the value around which other agents are performing. Thus, each agent tries to modify his position based on the following information:

- Current position (x, y),
- Current speed (V_x, V_y),
- Distance between current position and pbest
- Distance between current position and gbest

2.5.2.1 Concept of speed:

The modified speed of each agent will be written as follows [14] :

$$V_I^{k+1} = WV_I^k + C_1 rand_1 * (pbest_i - S_i^k) + C_2 rand_2 * (gbest_i - S_i^k) \quad (1)$$

Where:
V_I^k: the speed of agent I at iteration K.
W: agent speed at iteration.
C_j: weighting factor.
rand: random number between 1 and 0.
S_i^k: current position of agent i at iteration k.
pbest: best agent position i.
gbest: best overall position of the group.

2.5.2.2 Weighting function(W):

Usually used in the equation(1),and called inertia weight approch(IWA).

$$W = W_{max} - \frac{W_{max} - W_{min}}{iter_{max}} * iter \tag{2}$$

W_{max}: final weight,
W_{min}: initial weight,
$iter_{max}$: maximal itertion number,
$iter$: curant iteration number.

2.5.2.3 The current position:

looking for the point in the Solution space is modified according to the equation below:

$$s_i^{k+1} = s_i^k + V_i^{k+1} \tag{3}$$

s_i^k: current agent position,
s_i^{k+1}: agent's changed position,
V_i^{k+1}: agent's changed speed.

2.5.2.4 Algorithme pso:

For each particle $i = 1, \ldots..s$ do
Randomly initialize x_i
Randomly initialize v_i(or just set v_i to zero)
Set $y_i = x_i$
End for
Repeat
For each particle $i = 1, \ldots s$ do
Evaluate the fitness of particle (x_i)
Update y_i using equation
Update y using equation
For each dimension $j = 1, \ldots..nd$ do
Apply velocity update using equation
End loop
Apply position using equation
End loop Until some convergence criteria is satisfied

using what we have defined we can build the following flowchart.

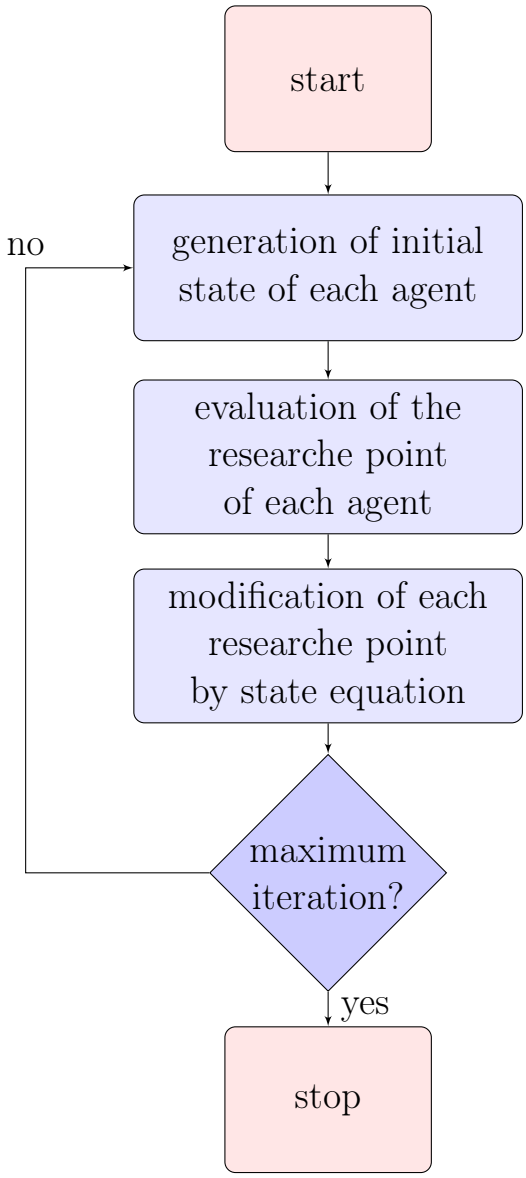


Figure 1: the general Flot chart organization of (PSO)

2.5.3 Conclusion:

This Algorithm is particularly simple to implement. As we can see, the solution space is explored by multiple particles, the best areas discovered by a particle being communicated to a given neighborhood in order to pass on the information. However, in general the neighborhood is not complete, which prevents the algorithm from falling into local optima.

2.6 Ant Colony Optimization (ACO):

2.6.1 Introduction:

THE Ant Colony Optimization (ACO) is a metaheuristic inspired by the behavior of real ants in their search for the shortest paths to food sources. It has recently attracted a lot of attention and has been successfully applied to a number of different optimization problems. Due to the importance of the feature selection problem and the potential of ACO, is a population-based metaheuristic that can be used to find approximate solutions to difficult optimization problems.[15]
In ACO, a set of software agents called artificial ants search for good solutions to a given optimization problem. To apply ACO, the optimization problem is transformed into the problem of finding the best path on a weighted graph. The artificial ants (hereafter ants) incrementally build solutions by moving on the graph. The solution construction process is stochastic and is biased by a pheromone model, that is, a set of parameters associated with graph components (either nodes or edges) whose values are modified at runtime by the ants.[33]

2.6.2 Principle:

Let's say the number of cities is n, the number of ants is m, the distance between city i and j is d_{ij},$(i, j = 1, 2....., n)$,and the concentration of pheromone in city (i, j) at time t is $\tau_{ij}(t)$. At the initial time, the pheromone concentration $\tau_{ij}(t)$ between cities is equal to $\tau_{ij}(0)$=C (C is a constant), and the probability of its choice is expressed by p_{ij}^k, and the formula is as follows:

$$p_{ij}^k = \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}(t)]^\beta}{\sum_{s \in allowed_k} [\tau_{is}(t)]^\alpha [\eta_{is}(t)]^\beta} \tag{4}$$

$\eta_{ij}(t)$ is the heuristic function, which indicates the degree of expectation of ants from city i to city j. $allowed_k(k = 1, 2, ..., m)$ indicates that ant k is to visit the urban set. β is the important factor of heuristics function, and α is the important factor of pheromone. When all ants complete a cycle, they update the pheromone according to formula (5):

$$\begin{cases} \tau_{ij}(t+n) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij} \\ \Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^k \end{cases} \tag{5}$$

ρ is the pheromone volatilization coefficient, $(1 - \rho)$ is called the pheromone residual factor, and $\Delta\tau_{ij}$ is the pheromone concentration released by the ant of k on the path of (i, j). In the basic ACO, only the positive feedback pheromone concentration is usually updated. the model of formula (6) is used to update the pheromone concentration in the search process.

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{L_k} \\ 0 \end{cases} \tag{6}$$

Q is a constant that represents the total amount of pheromone released once by an ant . L_k indicates that the ant of k passes the length of path (i, j) [11] .

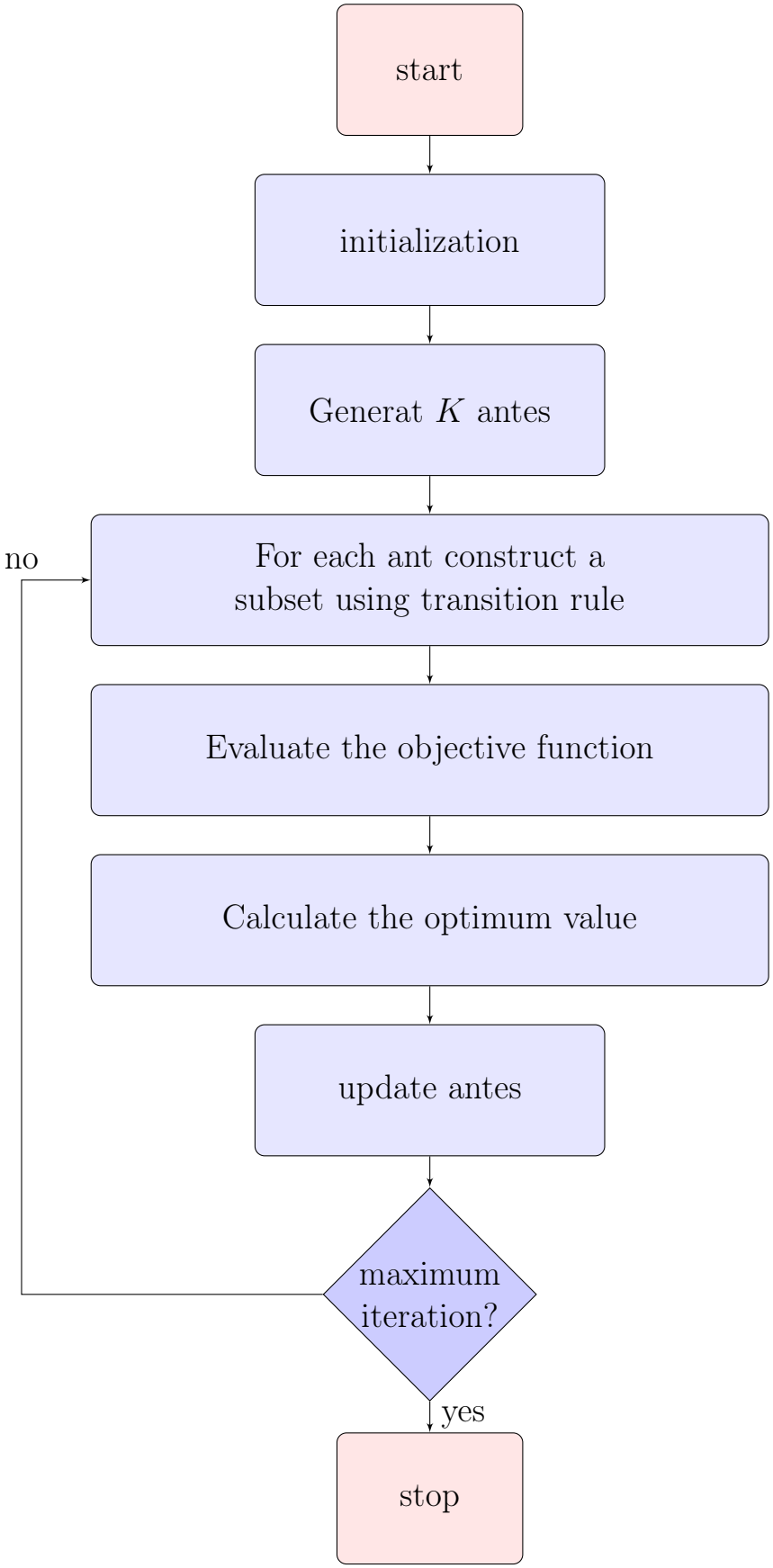


Figure 2: the general Flot chart organization of (aco)

2.6.3 Algorithm ACO:

Start.
Initialize the pheromone-trails and parameters.
Generate a random population of m ants (solution).
For every individual ant ascertain the best position according to the objective Function.
Get the best ant in search space.
Restore the pheromone-trail.
Verify if the termination is true.
End.

2.6.4 Conclution:

Artificial ants implement a randomized construction heuristic, which makes probabilistic decisions. The accumulated search experience is taken into account by the adaptation of the pheromone trail. In ACO Local search is extremely important to obtain good results.

Chapter 3

Direct state space model for Satellites with Flexible Appendages Attitude Control

3.1 Introduction:

THE interaction between the attitude control system and flexible appendages manifests itself on the overall system by degradation of the pointing accuracy, instability may also arise, and, eventually, in appendage deflections which are too large[2]. The problem of avoiding excessively large appendage deflections by means of auxiliary control loops has been discussed in [48].Although system instability may be generally avoided by means of a suitable control design, such as limited loop gains [46, 47] and opportune control system bandwidth [48], flexibility may lead to large attitude errors [45], particularly when the attitude control system is designed using a rigid model of the satellite. For these reasons, the necessity of designing a controller specifically for the flexible satellite has been pointed out by many authors. An interesting approach was proposed by Likins [46]. The method calls for an accurate model of the flexible satellite and consists of a three-stage process for the control system design, utilizing classical frequency domain techniques. A direct application of a multi-loop synthesis method, in the frequency domain, has been also recently proposed in [49].

3.2 Modling of the satellite:

The very simple satellite configuration considered is shown in Figure(3)

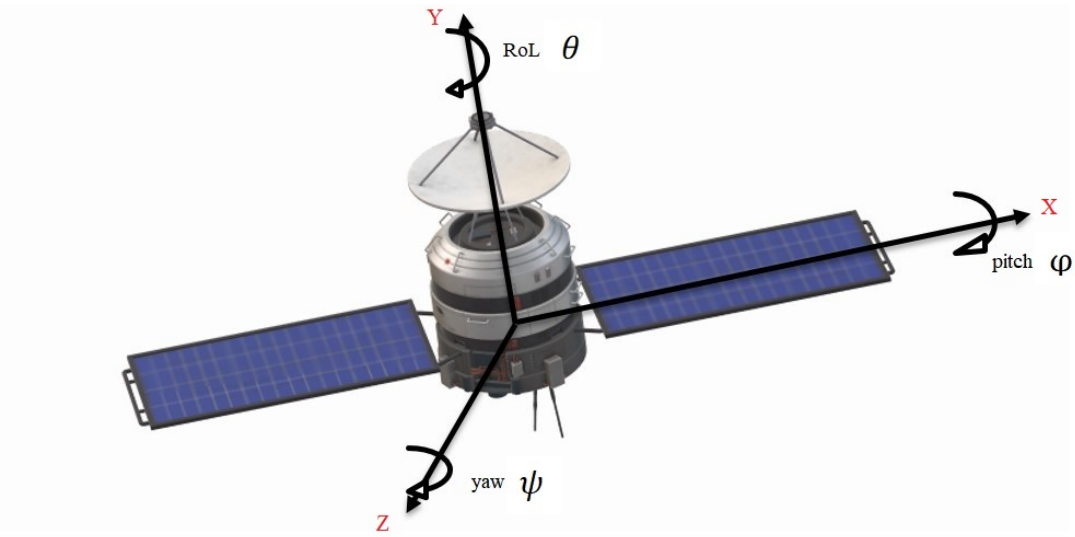
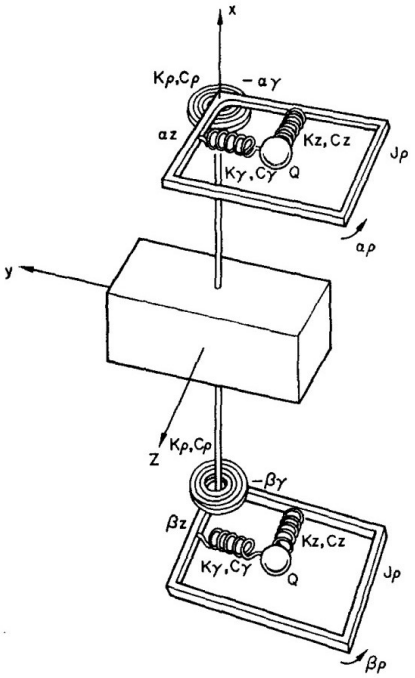


Figure 3: satellite configuration

x, y, z : principal axes of inertia of the undeformed
ϕ, θ, ψ : Euler angles, pitch, roll and yaw,satellite,

the flexible solar array are represented here as shown in two point masses having two degrees-of-freedom mass,spring is the very simple mold of a satellite with main body and flexible appendage for simplicity raison which can only rotate with respect to the x axis. All the movements are harmonic with some damping, assumed, for simplicity, to be proportional to the velocity. K_y ,K_z, K_p are the spring constants and C_y ,C_z, C_p the damping factors.are also shown the coordinates α_y,α_z,β_y,β_z (deflection) and α_p,β_p (rotations), which describe the position of the movable parts with respect to the main body.



where:
Q=value of the point mass.
L=distance of the point mass from the satellite centrer of mass.
I_p=moment of inertia of the movable material frame with respect to the x axis.

the moments of inertia of the undeformed satellite

are

$$\begin{cases} I_{\acute{x}} = I_x + I_p \\ I_{\acute{y}} = I_y + 2QL^2 \\ I_{\acute{z}} = I_z + 2QL^2 \end{cases} \tag{7}$$

where I_x,I_y,I_z are the principal moments of inertia of the rigid body.

3.2.1 Equation of movement:

The dynamics of the satellite, without any momentum exchange devices acting, can be obtained, in this simple case, by utilizing Lagrange's equations of motion the same result is obviously obtained using other approaches, like the 'hybrid coordinates' one.the following equation result:

$$\begin{cases} M_x = (\ddot{\alpha}_p + \ddot{\beta}_p)I_p + (I_{\acute{y}} - I_{\acute{z}})rp + I_{\acute{x}}\dot{p} \\ M_y = (\ddot{\alpha}_p + \ddot{\beta}_p)QL + (I_{\acute{z}} - I_{\acute{x}})rp + I_{\acute{y}}\dot{q} \\ M_z = (\ddot{\alpha}_p + \ddot{\beta}_p)QL + (I_{\acute{x}} - I_{\acute{y}})pq + I_{\acute{z}}\dot{r} \end{cases} \tag{8}$$

where M=(M_x, M_y, M_z) are the torques applied to the satellite main body.
and:

$$\dot{r}\text{=-}\frac{1}{L}(\ddot{\alpha_y} + \frac{C_y}{Q}\dot{\alpha_y} + \frac{K_y}{Q}\alpha_y) \qquad \dot{r}\text{=}\frac{1}{L}(\ddot{\beta_y} + \frac{C_y}{Q}\dot{\beta_y} + \frac{K_y}{Q}\beta_y)$$

$$\dot{q}\text{=-}\frac{1}{L}(\ddot{\beta_z} + \frac{C_z}{Q}\dot{\beta_z} + \frac{K_z}{Q}\beta_z) \qquad \dot{q}\text{=}\frac{1}{L}(\ddot{\alpha_z} + \frac{C_z}{Q}\dot{\alpha_z} + \frac{K_z}{Q}\alpha_z)$$

$$\dot{p}\text{=}\frac{1}{L}(\ddot{\alpha_p} + \frac{C_p}{i_p}\dot{\alpha_p} + \frac{K_p}{i_p}\alpha_p) \qquad \dot{p}\text{=}\frac{1}{L}(\ddot{\beta_p} + \frac{C_p}{i_p}\dot{\beta_p} + \frac{K_p}{i_p}\beta_p)$$

3.2.2 Liniarization of equation of movement:

the consuption of the satellite attitude controler are based on non linear control theory or linear control theory this part permits the linearisation of equation(8) ,assuming as usual, p, q and r very small, therefore neglecting their products. In this case:

$$p \approx \dot{\phi}, q \approx \dot{\theta}, r \approx \dot{\psi} \tag{9}$$

and if :

$$\xi = \alpha_p + \beta_p, \quad \zeta = \alpha_y - \beta_y, \quad \eta = \beta z - \alpha_z \tag{10}$$

the dynamics is described by the decoupled set of equations

$$\begin{cases} M_x = \ddot{\xi}I_p + \ddot{\phi}I_{\acute{x}} \\ -2I_p\ddot{\phi} = I_p\ddot{\xi} + C_p\dot{\xi} + K_p\xi \\ \\ M_y = \ddot{\eta}\phi L + \ddot{\theta}I_{\acute{y}} \\ -2QL\ddot{\theta} = Q\ddot{\eta} + C_z\dot{\eta} + K_z\eta \\ \\ M_z = \ddot{\psi}I_{\acute{z}} + \ddot{\zeta}QL \\ -2QL\ddot{\psi} = Q\ddot{\zeta} + C_y\dot{\zeta} + K_y\zeta \end{cases} \tag{11}$$

It can be concluded that the dynamics around each body axis is described by a set of equations of the type:

$$\begin{Vmatrix} M = A\ddot{\sigma} + B\ddot{\gamma} \\ F\ddot{\sigma} = C\ddot{\gamma} + D\dot{\gamma} + E\gamma \end{Vmatrix} \tag{12}$$

where σ is the angular displacement of the main body with respect to an inertial reference and γ is the angular displacement of the flexible arrays with. respect to the main body; $A, ..., F$ are suitable coefficients and M is the control torques applied to the main body Calling :

$$x_1 = \sigma, \quad x_2 = \gamma, \quad x_3 = \dot{\sigma}, \quad x_4 = \dot{\gamma} \tag{13}$$

system (12) my be rewritten as:

$$\begin{aligned} \dot{X}_1 &= X_3 \\ \dot{X}_2 &= X_4 \\ \dot{X}_3 &= A_1X_4 + A_2X_2 + A_3M \\ \dot{X}_4 &= A_4X_4 + A_5X_2 + A_6M \end{aligned} \tag{14}$$

where (A_1,....,A_6) A are suitable coefficients. It may be verified, by means of a continuous controllability test, that this linearized system is controllable. Particularly does this mean that it is possible to find a time history for M, the torque applied to the satellite main body, which is able, starting from arbitrary values of displacements and vibrations, to force the system to an established position in space, with zero rate and with vibrations completely damped out.

3.2.3 Control with momentum exchange devices:

The most common actuator for spacecraft attitude control is a family of systems that all rely on angular momentum. These momentum-control devices actively vary the angular momentum of small, rotating masses within a spacecraft to change its attitude. Typically, an attitude control system uses at least three separate reaction wheels, oriented at right angles to each other,When an actuator is used to provide a momentum H with respect to the body axes, the forces exerted on the satellite are

$$\begin{cases} M_x = -\dot{H}_x + rH_y - qH_z \\ M_y = -\dot{H}_y + pH_z - rH_x \\ M_z = -\dot{H}_z + qH_x - pH_y \end{cases} \tag{15}$$

where H_x, H_y and H_z are the components of H along the body axes.
Usually the actuator configuration employed provides a momentum which is much larger along one axis (say x) than along the others; as p, q and r are considered small, it is possible to rewrite (15) using (9) as follows

$$\begin{cases} M_x = -\dot{H}_x \\ M_y = -\dot{H}_y - rH_0 \approx -\dot{H}_y - \dot{\psi}H_0 \\ M_z = -\dot{H}_z + qH_0 \approx -\dot{H}_z - \dot{\theta}H_0 \end{cases} \tag{16}$$

where H_0 is the nominal value of H_x, the largest component of H. If (16) is substituted into (11), the set of equations describing the flexible satellite may be written

$$\begin{cases} -\dot{H}_x = \ddot{\xi} I_p + \ddot{\phi} I_{\acute{x}} \\ I_p \ddot{\xi} + C_p \dot{\xi} + K_p \xi + 2 I_p \ddot{\phi} = 0 \end{cases} \tag{17}$$

$$\begin{cases} -\dot{H}_y = \ddot{\eta} Q L + \ddot{\theta} I_{\acute{y}} + H_0 \dot{\psi} \\ Q \ddot{\eta} + C_z \dot{\eta} + K_z \eta + 2 Q L \ddot{\theta} = 0 \\ -\dot{H}_z = \ddot{\zeta} Q L + \ddot{\psi} I_{\acute{z}} + H_0 \ddot{\theta} \\ Q \ddot{\zeta} + C_y \dot{\zeta} + K_y \zeta + 2 Q L \ddot{\psi} = 0 \end{cases} \tag{18}$$

In such a way the system breaks into two linear ones, the first describing only the behaviour around the x axis, and the second describing the behaviour of the coupled y and z axes. As the problem of designing a control loop for the decoupled x axis, described by (17), is quite simple and classical, it will not be considered here. A controller for system (18), describing the coupled motion of the y and z axes, will be designed. In this case interest is not in the control of the variables η and ζ, which refer to the flexible appendages, but in the control of the attitude angles of the rigid body, θ and ψ. This means, for instance, correct pointing of the satellite antennae, placed on the rigid body, independently from the oscillation of the large flexible solar arrays. System (18) is therefore considered, with the addi- tion of the attitude angle feedback through sensors with a given time constant τ_s. Then the effective inputs (the reference attitude angles θ_R and ψ_R), are put in evidence. The additional equations are:

$$\begin{cases} \dot{H}_y = G(U_{sy} - \theta_R) \\ \dot{H}_z = G(U_{sz} - \psi_R) \end{cases} \tag{19}$$

where G is a gain and U_{sy}, U_{sz} are the output signals from the sensors, that is:

$$\begin{cases} U_{sy} = (\dfrac{1}{\tau_s})\theta_s \\ U_{sz} = (\dfrac{1}{\tau_s})\psi_s \\ \dot{\theta}_s = \theta - (\dfrac{1}{\tau_s})\theta_s \\ \dot{\psi}_s = \psi - (\dfrac{1}{\tau_s})\psi_s \end{cases} \tag{20}$$

From the above considerations, it is possible to write the system equations in the conventional form.

$$\begin{cases} \dot{X} = [A]X + [B]U \\ Y = [C]X \end{cases} \tag{21}$$

with vectors and matrices defined as follows, where the actual satellite attitude is considered as an output vector.

state vector: $X = [\theta, \eta, \dot{\theta}, \dot{\eta}, \psi, \zeta, \dot{\psi}, \dot{\zeta}, \dot{\theta}_s, \dot{\psi}_s]^T$.

input vector: $U = \begin{bmatrix} \theta d \\ \psi d \end{bmatrix}$.

output vector: $Y = \begin{bmatrix} \theta \\ \psi \end{bmatrix}$.

$$[A] = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & A_4 & 0 & A_3 & 0 & 0 & -A_2 & 0 & -A_1 & 0 \\ 0 & -A_8 & 0 & -A_7 & 0 & 0 & A_6 & 0 & A_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & A_{10} & 0 & 0 & A_{12} & 0 & A_{11} & 0 & A_9 \\ 0 & 0 & -A_{14} & 0 & 0 & -A_{16} & 0 & -A_{15} & 0 & A_{13} \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -A_{17} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -A_{17} \end{bmatrix}$$

$$[B] = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ B_1 & 0 \\ -B_2 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & B_3 \\ 0 & -B_4 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad [C] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

the resulting coefficients and nimurical values used are:

numerical values has been used	
I_y	$1500Kg.m^2$
L	$10m$
H_0	$100Nms$
I_z	$1800Kg.m^2$
Q	$70Kg$
G	$100Nms/rad$
τ_s	$2sec$
$C_y = C_z$	$175 \times 10^2 m/sec$
$K_y = K_z$	$240 \times 10^2 m/sec$
ω_n	$0.2rad/sec$
ξ_n	0.062

$$A_1 = \frac{G}{I_y \tau s} \qquad A_2 = \frac{H_0}{I_y} \qquad A_3 = \frac{C_z L}{I_y}$$

$$A_4 = \frac{K_z L}{I_y} \qquad A_5 = \frac{2GL}{I_y \tau s} \qquad A_6 = \frac{2LH_0}{I_y}$$

$$A_7 = \frac{C_z I_{\acute{y}}}{QI_y} \qquad A_8 = \frac{K_z I_{\acute{y}}}{QI_y} \qquad A_9 = \frac{G}{I_z \tau s}$$

$$A_{10} = \frac{H_0}{I_z} \qquad A_{11} = \frac{C_y L}{I_z} \qquad A_{12} = \frac{K_y L}{I_z}$$

$$A_{13} = \frac{2GL}{I_z \tau s} \qquad A_{14} = \frac{2LH_0}{I_z} \qquad A_{15} = \frac{C_y I_{\acute{z}}}{QI_z}$$

$$A_{16} = \frac{K_y I_{\acute{z}}}{QI_z} \qquad A_{17} = \frac{1}{\tau s} \qquad B_1 = \frac{G}{I_y}$$

$$B_2 = \frac{2GL}{I_y} \qquad B_3 = \frac{G}{I_z} \qquad B_4 = \frac{2GL}{I_z}$$

3.3 Control theory and application:

3.3.1 Input/output decupled control:

The problem of input-output decoupling of a system may be stated as follows. assume that it has the same number of inputs and outputs, i.e., assume that $p = m$. Determine a pair of matrices K and F of the state feedback law $\{u(t) = -Kx(t) + Fr(t)\}$, such that every input of the closed-loop system influences only one of the systems outputs, and vice-versa, every output of the closed-loop system is influenced by only one of its inputs. More precisely, in an input-output decoupled system the following relation must hold $\{y_i(s) = g(s)_{ii} \times u_i(s)\}$ for $(i = 1, ..., m)$. The basic motivation for inputoutput decoupling of a system is that by making each output of the system depend only upon one input and vice versa, we convert a MIMO system to m single-input-single-output (SISO) systems. This fact significantly simplifies and facilitates the control of the closedloop system, since one has to deal with m scalar systems rather than a MIMO system. For these reasons the problem of input-output decoupling is of great practical importance. It should be pointed out that dynamic decoupling is very demanding. A signal applied to input u_i must control output y_i and have no whatsoever effect on the other outputs. In many cases this requires a complex and highly sensitive control law, and in other cases it cannot be achieved at all (without additional compensation). We will concentrate on dynamic decoupling and then we will apply the decupled control according to falb-wolovich[4]

3.3.1.1 Dynamic decoupling:

Let we consider the linear time invariant system described by $\{sx(t) = Ax(t) + Bu(t)\}$ and $\{y(t) = Cu(t)\}$ where ($A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{m\times n}$) are matrix and $y = [y_1, y_2, ..., x_n]^T$, $u = [u_1, u_2, ..., u_m]^T$, $x = [x_1, x_2, ..., u_m]^T$ The number of output derivatives until the input appears with nonzero coefficient is of main interest, and a key assumption is that this number not changes with time.

$$\frac{d}{dt}y = C\dot{x}(t) = CAx(t) + CBu(t) \tag{22}$$

$$\text{let } C = \begin{bmatrix} C1 \\ C2 \\ . \\ . \\ . \\ Cm \end{bmatrix} \begin{matrix} \}\mathbb{R}^{1*n} \\ \}\mathbb{R}^{1*n} \\ . \\ . \\ . \\ \}\mathbb{R}^{1*n} \end{matrix} \rightsquigarrow CA = \begin{bmatrix} C1A \\ C2A \\ . \\ . \\ . \\ CmA \end{bmatrix} \text{ and } CB = \begin{bmatrix} C1B \\ C2B \\ . \\ . \\ . \\ CmB \end{bmatrix}$$

let now define a operator that map $(1 \times n)$ time function, into $(1 \times n)$ time functions :

$L_A[C_i] = C_iA$

$L_A^{j+1}[C_i] = L_A[L_A^j[C_i]]$

$L_A^0[C_i] = C_i$

$L_{A+BK}[C_i] = C_i[A + BK]$

Definition:(relative degree of a system). the relative degree k, $m \leqslant k \leqslant n$ of a system of the order n with m outputs y_i, $(i = 1, 2, ..., m)$, is $\{k = \Sigma_{i=1}^{m} k_i\}$

Note: The relative degrees are invariant under the state feedback control and the number of poles is fixed by the relative degree k_i of the output y_i. Unfortunately, not every system can be decoupled by state feedback.

If $CB \neq 0$ then the relative degree is one, otherwise we continue derivation until $u(t)$ appears in the output derivatives. In continuing this calculation the coefficient of $u(t)$ in the j^{th} derivative is:

$$L_A^{j-1}[C_i]B = A^{j-1}B \tag{23}$$

take the k_i^{th}derivatives of the output y_i we get the following equation

$$\begin{bmatrix} y_1^{k_1} \\ \vdots \\ y_m^{k_m} \end{bmatrix} = \Gamma_1 x(t) + \Gamma_2 u(t) \tag{24}$$

where:

$$\Gamma_1 = \begin{bmatrix} L_A^{k_1}[C_1] \\ L_A^{k_2}[C_2] \\ \vdots \\ L_A^{k_m}[C_m] \end{bmatrix} = \begin{bmatrix} C_1 A^{k_1} \\ C_2 A^{k_2} \\ \vdots \\ C_m A^{k_m} \end{bmatrix}, and\ \Gamma_2 = \begin{bmatrix} L_A^{k_1-1}[C_1]B \\ L_A^{k_2-1}[C_2]B \\ \vdots \\ L_A^{k_m-1}[C_m]B \end{bmatrix} = \begin{bmatrix} C_1 A^{k_1-1}B \\ C_2 A^{k_2-1}B \\ \vdots \\ C_m A^{k_m-1}B \end{bmatrix} \tag{25}$$

If the state feedback control u(t) is given by $u(t) = -Kx(t) + Nv(t)$ then

$$\begin{bmatrix} y_1^{k_1} \\ \vdots \\ y_m^{k_m} \end{bmatrix} = (\Gamma_1 - \Gamma_2 K)x(t) + \Gamma_2 N \begin{bmatrix} u_1^{k_1} \\ \vdots \\ u_m^{k_m} \end{bmatrix} \tag{26}$$

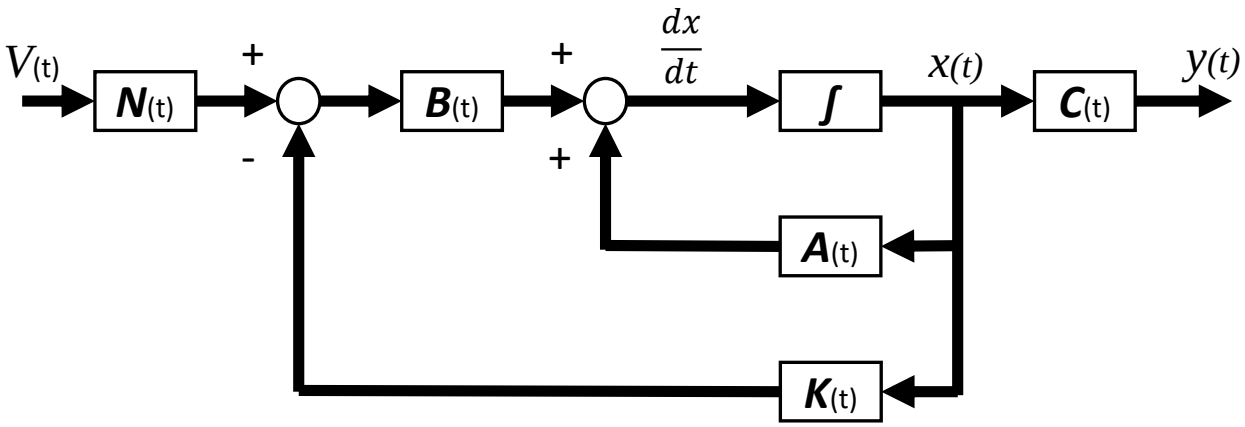


Figure 4: block diagram of the state feedback

Theorem: (Wang, Q.-G. et.al 2003) There exists a feedback control law of the form $\{u(t) = -Kx(t) + Nv(t)\}$ to decouple the system if and only if the matrix Γ_2 is non-singular. If this is the case, by choosing $\left\{K = {\Gamma_2}^{-1}\Gamma_1\right\}$ and $N = {\Gamma_2}^{-1}$, the resultant feedback system has the transfer function matrix: $H(s) = diag\left\{s^{-k_1}, s^{-k_2}, ..., s^{-k_m}\right\}$

3.3.1.2 Decoupled Control According to Falb-Wolovich:

It is seen that the dynamic decoupling system presented previously can be used only to decouple the multivariable system into integrator types of diagonal system, which makes the closed-loop design difficult. If one can still assume that the decoupling law is state feedback which can be written as $\{u(t) = -Kx(t) + Nv(t)\}$, one may expect to have a decoupled system in the form $H_{cl}(s) = diag\{H_i(s)\}$,$i = \{1, 2, ..., m\}$ and H_{cl} is the trensfer function H in closed-loop whith

$$H_i(s) = \left\{\frac{K_i}{a_{i,0}s^{k_i} + a_{i,1}s^{k_{i-1}} + ... + a_{i,k_{i-1}}s + a_{i,k_i}}\right\}, a_{i,0} = 1 \tag{27}$$

The parameters $a_{i,\mu}$ and k_i with $\mu =(\ 1,\ 2,\ ...\ ,\ k_i)$ in each of the polynomials $H_i\ (s)$. can be assigned by the pole placement method. The expected polynomial can be defined as the standard transfer function. The standard transfer function of an n^{th} order system, with the integral of time-multiplied absolute value of error (ITAE) optimal criterion, is defined in (Dorf, R.C. and Bishop, R.H. et.al 2001).
Define a matrix Γ_2, where each row is ${}^2\Gamma_i = C_iA^{k_i-1}B$, and each row ${}^1\Gamma_i$ in another matrix Γ_1 can be defined as $\left\{{}^1\Gamma_i = C_i(A^{k_i} + a_{i,1}A^{k_{i-1}} + ... + a_{i,k_{i-1}}I)\right\}$ The state feedback matrix

$$K = \begin{bmatrix} C_1A^{k_1-1}B \\ C_2A^{k_2-1}B \\ \vdots \\ C_mA^{k_m-1}B \end{bmatrix}^{-1} \begin{bmatrix} C_1\sum_{\mu=0}^{k_1} a_{1,\mu}A^{k_1-\mu} \\ C_2\sum_{\mu=0}^{k_2} a_{2,\mu}A^{k_2-\mu} \\ \vdots \\ C_m\sum_{\mu=0}^{k_m} a_{m,\mu}A^{k_m-\mu} \end{bmatrix}, N = \begin{bmatrix} C_1A^{k_1-1}B \\ C_2A^{k_2-1}B \\ \vdots \\ C_mA^{k_m-1}B \end{bmatrix}^{-1} \begin{pmatrix} K_1 & & \circ \\ & \ddots & \\ \circ & & K_m \end{pmatrix} \tag{28}$$

The still free parameters $a_{i,\mu}$ can then be determined separately for each of these systems, e.g. by pole placement, and one can impose additionally $k_i = a_i$ in order to guarantee the static accuracy of each of them.

$$\begin{cases} \dot{x} = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \tag{29}$$

Where:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0187 & 0 & 0.0117 & 0 & 0 & -0.0667 & 0 & -0.0333 & 0 \\ 0 & -0.4133 & 0 & -0.2583 & 0 & 0 & 1.3333 & 0 & 0.6667 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.0556 & 0 & 0 & 0.0156 & 0 & 0.0097 & 0 & 0.0278 \\ 0 & 0 & -1.1111 & 0 & 0 & -0.3511 & 0 & -0.2194 & 0 & 0.5556 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.5000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.5000 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.0667 & 0 \\ -1.3333 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0.0556 \\ 0 & -1.1111 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \text{ and } C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The order differences (relative degrees) of the two outputs, k_1=2 and k_2=2 , are both equal to 2, since:

$$C_1 \times A \times B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \times A \times B = \begin{bmatrix} 0.0667 & 0 \end{bmatrix} \neq 0$$

$$C_2 \times A \times B = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \times A \times B = \begin{bmatrix} 0 & 0.0556 \end{bmatrix} \neq 0$$

The total order difference of the plant amounts therefore to $(k = k_1 + k_2)$, the matrix Γ_2 has the forme

$$\Gamma_2 = \begin{bmatrix} C_1AB \\ C_2AB \end{bmatrix} = \begin{bmatrix} 0.0667 & 0 \\ 0 & 0.0556 \end{bmatrix} \tag{30}$$

Examining the determinant of the matrix $\Gamma_2 \neq 0$ the system under control can be decoupled using the feedback law $\{u(t) = -Kx(t) + Nv(t)\}$. Let us use the eigenvalues $\lambda_1 = 8$ and $\lambda_2 = 7$ means that $H_1(s) = \left[\frac{8}{s+8}\right] and, H_2(s) = \left[\frac{7}{s+7}\right]$ To determine the matrices K and N according to relation (28), we must first compute the matrix Γ_1.
We have:

$$\Gamma_1 = \begin{bmatrix} C_1(A + 8I) \\ C_2(A + 7I) \end{bmatrix} = \begin{bmatrix} 8 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Hence, the matrices K and N are given by

$$K = \Gamma_2^{-1}\Gamma_1 = \begin{bmatrix} 120 & 0 & 15 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 126 & 0 & 18 & 0 & 0 & 0 \end{bmatrix}$$

$$N = \Gamma_2^{-1} \begin{bmatrix} 8 & 0 \\ 0 & 7 \end{bmatrix} = \begin{bmatrix} 120 & 0 \\ 0 & 126 \end{bmatrix}$$

The decoupled closed-loop system is the following

$$\dot{x} = Ax + Bu \tag{31}$$

Where

$$A = \begin{bmatrix} 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 \\ -8.0000 & 0.0187 & -1.0000 & 0.0117 & 0 & 0 & -0.0667 & 0 & -0.0333 & 0 \\ 160.0000 & -0.4133 & 20.0000 & -0.2583 & 0 & 0 & 1.3333 & 0 & 0.6667 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0.0556 & 0 & -7.0000 & 0.0156 & -1.0000 & 0.0097 & 0 & 0.0278 \\ 0 & 0 & -1.1111 & 0 & 140.0000 & -0.3511 & 20.000 & -0.2194 & 0 & 0.5556 \\ 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.5000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.5000 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 8 & 0 \\ -160 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 7 \\ 0 & -140 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

The transfer function matrix $H_c l(s)$ of the closed-loop system is given by:

$$H_c l(s) = C(sI - A + BK)^{-1}BN$$

the m-file code is given by:

```
close all
clear all
clc

%-------------------------------------------------------------------------%
%                                DATA
%-------------------------------------------------------------------------%
s=tf('s');
Iy=1500;%kgm^2
Iz=1800;%kgm^2
L=10;%m
H=100;%nms
Q=70;%kg
G=100;%nms/rad
Tos=2;%sec
Cq=2.5*10^-2;%m/s
Kq=4*10^-2;%m/s
Cz=Cq*Q;
Cy=Cz;
Kz=Kq*Q;
Ky=Kz;
Wn=0.2;%rad/s
Psi=0.062;
Iyd=Iy+2*Q*(L^2);
```

```
Izd=Iz+2*Q*(L^2);

%-------------------------------------------------------------------------%
%                       state variable
%-------------------------------------------------------------------------%

A1=G/(Iy*Tos);
A2=H/Iy;
A3=(Cz*L)/Iy;
A4=(Kz*L)/(Iy);
A5=(2*G*L)/(Iy*Tos);
A6=(2*L*H)/Iy;
A7=(Cz/Q)*(Iyd/Iy);
A8=(Kz*Iyd)/(Q*Iy);
A9=G/(Iz*Tos);
A10=H/Iz;
A11=(Cy*L)/Iz;
A12=(Ky*L)/Iz;
A13=(2*G*L)/(Iz*Tos);
A14=(2*L*H)/Iz;
A15=(Cy/Q)*(Izd/Iz);
A16=(Ky/Q)*(Izd/Iz);
A17=1/Tos;
B1=G/Iy;
B2=(2*G*L)/Iy;
B3=G/Iz;
B4=(2*G*L)/Iz;

%-------------------------------------------------------------------------%
%                         state space
%-------------------------------------------------------------------------%

A=[0 0 1 0 0 0 0 0 0 0;0 0 0 1 0 0 0 0 0 0;0 A4 0 A3 0 0 -A2 0 -
    A1 0;...
0 -A8 0 -A7 0 0 A6 0 A5 0;0 0 0 0 0 0 1 0 0 0;0 0 0 0 0 0 0 1 0
    0;...
0 0 A10 0 0 A12 0 A11 0 A9;0 0 -A14 0 0 -A16 0 -A15 0 A13;...
1 0 0 0 0 0 0 0 -A17 0;0 0 0 0 0 0 0 0 -A17];

B=[0 0;0 0;B1 0;-B2 0;0 0;0 0;0 B3;0 -B4;0 0;0 0];

C=[1 0 0 0 0 0 0 0 0 0;0 0 0 0 1 0 0 0 0 0];
D=[0 0;0 0];
H1=tf(ss(A,B,C,D));%transfer function
%-------------------------------------------------------------------------%
%    Decoupled Control According to Falb-Wolovich
%-------------------------------------------------------------------------%
C1=C(1,:);%the first line of C matrix
```

```
71  C2=C(2,:);%the second line of C matrix
72  L2=[C1*A*B;C2*A*B];%
73  L1=[C1*(8*eye(10,10)+A);C2*(7*eye(10,10)+A)];
74  K=inv(L2)*L1;
75  N=inv(L2)*diag([8  7]);
76
77  Anew=A-(B*K);%state matrix A of the new system
78  Bnew=B*N;%state matrix B of the new system
79  Cnew=C;%state matrix C of the new system
80  Dnew=D;%state matrix D of the new system
81  Gs=Cnew*inv(s*eye(10,10)-Anew)*Bnew+Dnew;%transfer function of
       the closed loop sys
82  Gs=minreal(Gs);
83  H=Gs;
84  %-------------------------------------%
85  %     plot part
86  %-------------------------------------%
87  T=0.01;
88  t=[0:T:20]';%time vector
89  n=length(t);%size of t
90  u1=0.1*ones(length(t),1);%step signal with 0.1 initial/final
       value and 0 step time
91  u2=0.08*ones(length(t),1);%step signal with 0.08 initial/final
       value and 0 step time
92  Filterr=tf(100,[1 14 100]);%filter use to the step signal 'see
       input signal interpritation'
93  u1=lsim(Filterr,u1,t);
94  u2=lsim(Filterr,u2,t);
95  u=[u1 u2];
96  y =lsim(H,u,t);
97  y1=y(:,1);
98  y2=y(:,2);
99  u1=u(:,1);
100 u2=u(:,2);
```

3.3.2 MIMO PID controler:

3.3.2.1 Introduction:

THE PID controller is the most common form of feedback , and considerably used in industrial processes [19] , because their structure consisting on only three parameters is very simple to implement We will start by summarizing the key features of the PID controller. The "textbook" version of the PID algorithm is described by [20] .

$$u(t) = (K_p e(t) + \frac{1}{K_i}\int_0^t e(\tau)d\tau + K_d\frac{de(t)}{dt}) \tag{32}$$

where $u(t)$ is the control signal and $e(t)$ is the control error . The reference variable is often called the set point. The control signal is thus a sum of three terms: the P-term (which is proportional to the error) , the I-term (which is proportional to the integral of the error) , and the D-term (which is proportional to the derivative of the error). The controller parameters are proportional gain K_p , integral time K_i , and derivative time K_d and the figure (5) will illustrated PID in parallel configuration

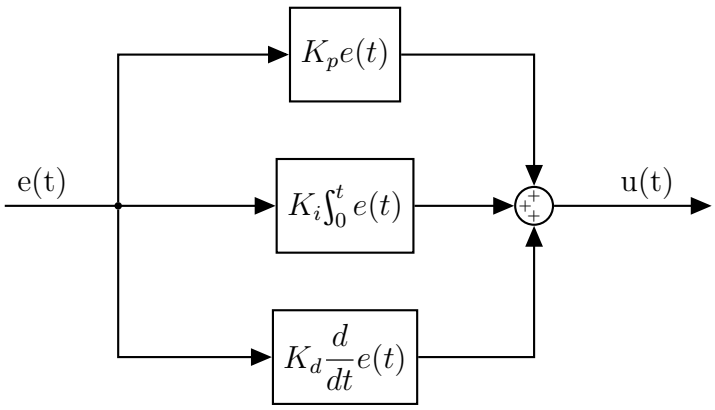


Figure 5: PID in parallel configuration

where:
K_p , K_i and K_d are matrices $(n \times n)$ and n is number of inputs

3.3.2.2 PID Controller tuning methods:

The goal of tuning the PID controller is to determine parameters that meet closed loop system performance specifications, and to improve the robust performance of the control loop over a wide range of operating conditions. Practically, it is often difficult to simultaneously achieve all of these desirable qualities. For example, if the PID controller is adjusted to provide better transient response to set point change, it usually results in a sluggish response when under disturbance conditions. On the other hand, if the control system is made robust to disturbance by choosing conservative values for the PID controller, it may result in a slow closed loop response to a set point change. A number of tuning techniques that take into consideration the nature of the dynamics present within a process control loop have been proposed (see Ziegler and Nichols, 1942; Cohen and Coon, 1953; Åström and Hägglund, 1984; De Paor and O'Malley, 1989; Zhuang and Atherton, 1993; Venkatashankar and Chidambaram, 1994; Poulin and Pomerleau, 1996;

Huang and Chen, 1996). All these methods are based upon the dynamical behavior of the system under either open-loop or closed-loop conditions.

3.3.2.3 Controler tuning MIMO-PID using multiobjective ant colony optimization:

Ant colony optimization algorithms are especially suited for finding solutions to difficult optimization problems .A colony of artificial ants cooperates to find good solutions , which are an emergent property of the ants' cooperative interaction . Based on their similarities with ant colonies in nature , ant algorithms are adaptive and robust and can be applied to different versions of the same problem as well as to different optimization problems [50] .a tuning of PID controllers method using multiobjective ant colony optimization [21] .The design objective was to apply the ant colony algorithm in the aim of tuning the optimum solution of the PID controllers (K_p , K_i , and K_d) by minimizing the multiobjective function .The potential of using multiobjective ant algorithms is to identify the Pareto optimal solution .

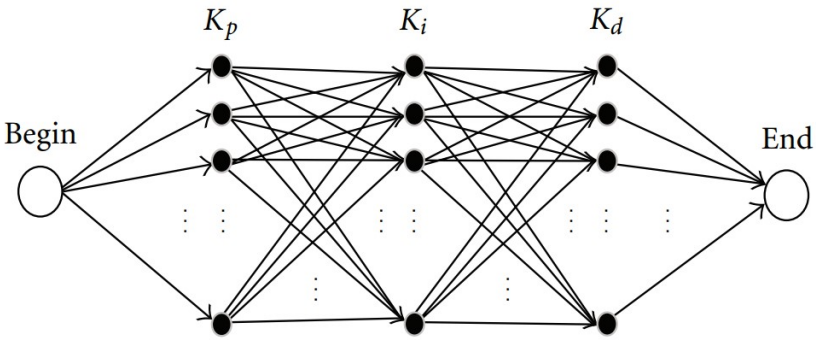


Figure 6: Ant colony optimization graph.

Feedback Loop of Control System A feedback is a common and powerful tool when designing a control system. Feedback loop is the tool which take the system output into consideration and enables the system to adjust its performance to meet a desired result of system. or any kind of disturbance. So one signal is taken from output and is fed back to the input. This signal is compared with reference input and then error signal is generated. This error signal is applied to controller and output is corrected. Such a system is called feedback system. using matlab code. this code contain matlab-file and simulink model working to gether, the matlab-file contain the numerical value and (aco) programe the (aco) programe run simulink model with diffirent gain value trying to get the optimal one for this model

we have intreduced aco algorithm and the m-file code is given by:

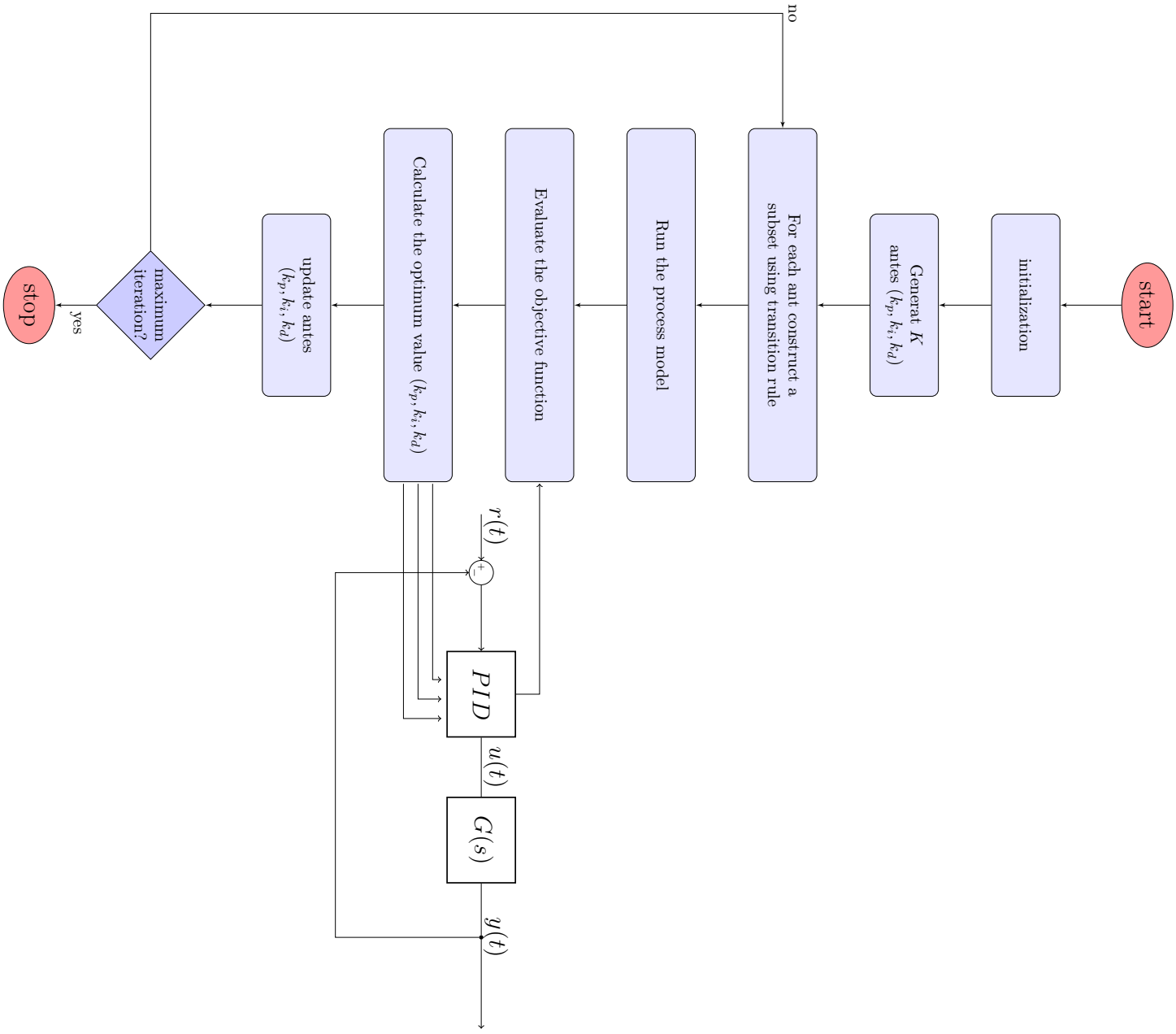


Figure 7: PID control system

```
clear all
clc
format long
%-------------------------------------------------------%
%                      model value
%-------------------------------------------------------%

Iy=1500;%kgm^2
Iz=1800;%kgm^2
L=10;%m
H=100;%nms
Q=70;%kg
G=100;%nms/rad
Tos=2;%sec
Cq=2.5*10^-2;%m/s
Kq=4*10^-2;%m/s
Cz=Cq*Q;
Cy=Cz;
Kz=Kq*Q;
Ky=Kz;
Wn=0.2;%rad/s
Psi=0.062;
Iyd=Iy+2*Q*(L^2);
Izd=Iz+2*Q*(L^2);

%-------------------------------------------------------%
%                    model variable
%-------------------------------------------------------%

A1=G/(Iy*Tos);
A2=H/Iy;
A3=(Cz*L)/Iy;
A4=(Kz*L)/(Iy);
A5=(2*G*L)/(Iy*Tos);
A6=(2*L*H)/Iy;
A7=(Cz/Q)*(Iyd/Iy);
A8=(Kz*Iyd)/(Q*Iy);
A9=G/(Iz*Tos);
A10=H/Iz;
A11=(Cy*L)/Iz;
A12=(Ky*L)/Iz;
A13=(2*G*L)/(Iz*Tos);
A14=(2*L*H)/Iz;
A15=(Cy/Q)*(Izd/Iz);
A16=(Ky/Q)*(Izd/Iz);
A17=1/Tos;
B1=G/Iy;
```

```
B2=(2*G*L)/Iy;
B3=G/Iz;
B4=(2*G*L)/Iz;

%-------------------------------------------------------%
%                      state space
%-------------------------------------------------------%

A=[0 0 1 0 0 0 0 0 0 0;0 0 0 1 0 0 0 0 0 0;0 A4 0 A3 0 0 -A2 0 -
    A1 0;...
0 -A8 0 -A7 0 0 A6 0 A5 0;0 0 0 0 0 0 1 0 0 0;0 0 0 0 0 0 0 1 0
    0;...
0 0 A10 0 0 A12 0 A11 0 A9;0 0 -A14 0 0 -A16 0 -A15 0 A13;...
1 0 0 0 0 0 0 0 -A17 0;0 0 0 0 0 0 0 0 -A17];

B=[0 0;0 0;B1 0;-B2 0;0 0;0 0;0 B3;0 -B4;0 0;0 0];

C=[1 0 0 0 0 0 0 0 0 0;0 0 0 0 1 0 0 0 0 0];

D=[0 0;0 0];

%-------------------------------------------------------%
%                   ant colony optimization
%-------------------------------------------------------%

step=1;
% ----------------starting point----------------------
K11=445.2325 ;
K12=26.6227;
K21=31.5737;
K22=512.1150;

L11=145.0960;
L12=47.6145;
L21=50.6626;
L22=206.2175;

M11=330.5603;
M12=63.3566;
M21=40.9493;
M22=408.3806;
%----initialize starting position of the ant----------
startingptK11=K11;
startingptK12=K12;
startingptK21=K21;
startingptK22=K22;

startingptL11=L11;
```

```
94  startingptL12=L12;
95  startingptL21=L21;
96  startingptL22=L22;
97
98  startingptM11=M11;
99  startingptM12=M12;
100 startingptM21=M21;
101 startingptM22=M22;
102
103 error=2; %initialize error with any random values
104 %errordot=1;
105 prev_error=2e7;  %such that prev_error>>error
106 times=1;
107
108 total_ants=3; %greater the number of ants, more optimum the path
109
110 it=0
111 while error(1)<prev_error && it<15%if error>prev_error it means
112
113 if times>1
114 prev_error=error;    %so that this does not execute for the
115 end                  %first iterstion of while loop
116 it=it+1
117 times=times+1;
118
119 for i=1:1:total_ants
120 startingK11(i)=startingptK11+rand*10.10; %each ant randomly
    takes any
121 startingK12(i)=startingptK12+rand*10.0; % position near its
    starting point
122 startingK21(i)=startingptK21+rand*10.0;
123 startingK22(i)=startingptK22+rand*10.0;
124
125 startingL11(i)=startingptL11+rand*10.0; %each ant randomly takes
     any
126 startingL12(i)=startingptL12+rand*10.0; % position near its
    starting point
127 startingL21(i)=startingptL21+rand*10.0;
128 startingL22(i)=startingptL22+rand*10.0;
129
130 startingM11(i)=startingptM11+rand*10.0; %each ant randomly takes
     any
131 startingM12(i)=startingptM12+rand*10.0; % position near its
    starting point
132 startingM21(i)=startingptM21+rand*10.0;
133 startingM22(i)=startingptM22+rand*10.0;
134
135
```

```
K=[startingK11(i) startingK12(i);%create K gain matrix
startingK21(i) startingK22(i)]
L=[startingL11(i) startingL12(i);%create I gain matrix
startingL21(i) startingL22(i)]
M=[startingM11(i) startingM12(i);%create P gain matrix
startingM21(i) startingM22(i)]

sim('attitude')%run simulation in simulink model

error(i)=(max(max(abs((input-output)))))%the error between in\
   out signal

best=find(error==min(error(i)))%the minimal error

if error(best)<prev_error% reinitialization taking the best
   position of the ant

startingptK11=startingK11(best);
startingptK12=startingK12(best);
startingptK21=startingK21(best);
startingptK22=startingK22(best);

startingptL11=startingL11(best);
startingptL12=startingL12(best);
startingptL21=startingL21(best);
startingptL22=startingL22(best);


startingptM11=startingM11(best);
startingptM12=startingM12(best);
startingptM21=startingM21(best);
startingptM22=startingM22(best);
error=error(best);
end
end
H=[K;L;M];
end
sim('attitude')% run the simulink model with optimal PID gain
plot(tsim,compariason_plot,'linewidth',1.7)%plot the in output
   from the simulink model
grid on
stepinfo(compariason_plot)%signal infomation "Rise time,
   settling time, and other step-response characteristics"
```

3.3.2.4 Controler tuning MIMO-PID using matlab System tuner $Toolbox^{tm}$

Control System $Toolbox^{tm}$ provides algorithms and apps for systematically analyzing, designing, and tuning linear control systems. The toolbox automatically tunes both SISO and MIMO compensators, including PID controllers. Control System Tuner automatically tunes the controller parameters to satisfy the must-have requirements (design constraints) and to best meet the remaining requirements (objectives). The library of tuning goals lets you capture your design requirements in a form suitable for fast automated tuning. Available tuning goals include standard control objectives for reference tracking. in this method we includ the fourth gain in PID configiration how give us the possibility of redusing the value of the three other gain (K_p, K_i, K_d) the new parallel configuration of the PID used is:

$$K_p + K_i\frac{1}{s} + K_d\frac{s}{T_f s + 1} \tag{33}$$

were T_f is an filter.
the interface of the toolbox are very simple to use the next figure illustrate how to use

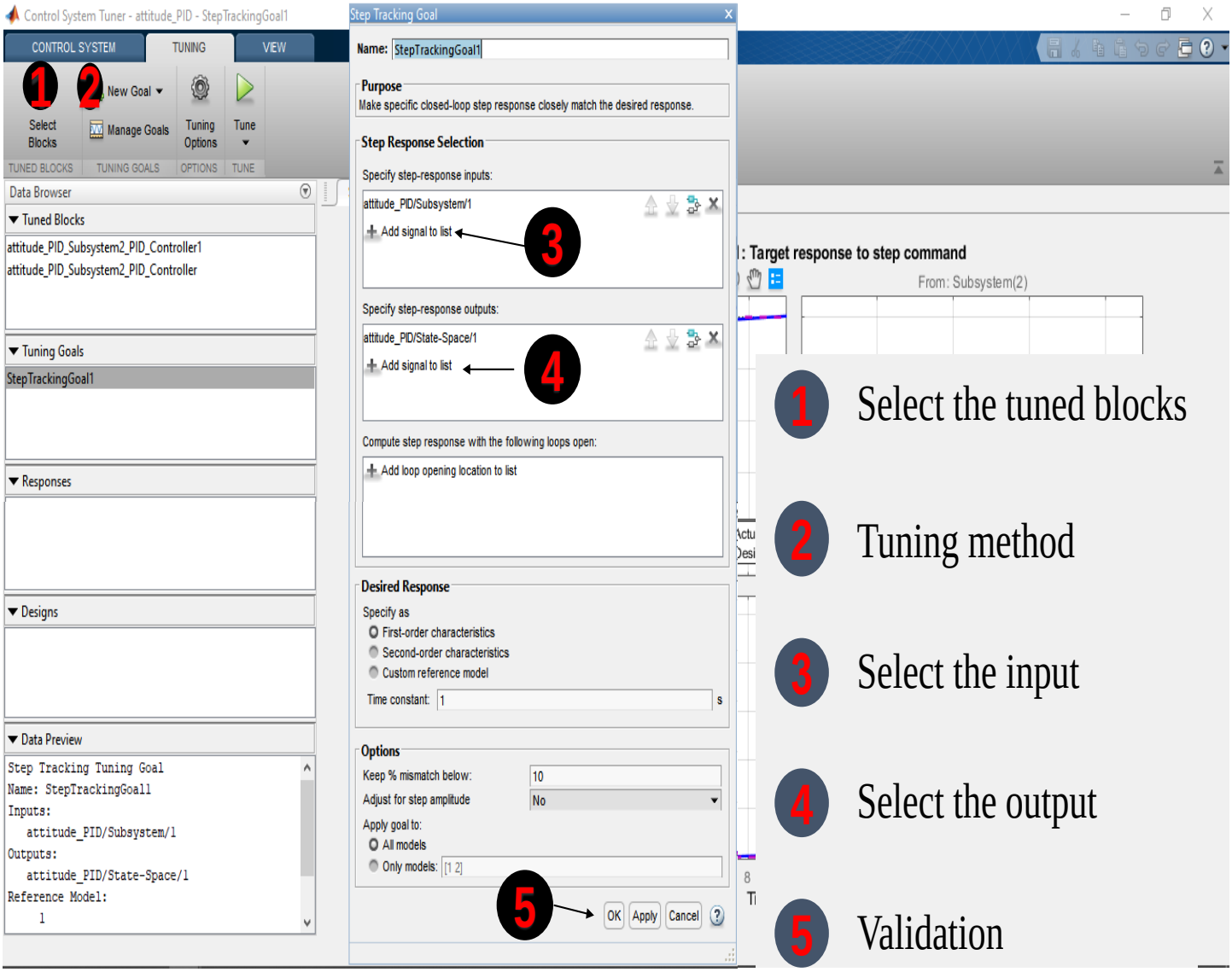


Figure 8: Control System Tuner

3.3.3 Simulation and results:

The validation of the control strategy was obtained both in simulation on the previously described prototype. The used simulation procedures were:
Define the initial position $(\theta d, \psi d = 0)$. use filter to the input step signal. it is known that attitude maneuvering would excite appendages vibration, which is closely related to the attitude angular acceleration. A sudden change in attitude signal, especially in the form of step signal, may cause a serious appendages vibration. Here,in order to achieve high performance of attitude control and attenuate the residual vibration [10] so we use transfer function iter in the input to simulate path planning scheme based on quantic polynomial transition is developed, which is motivated by the previous study of robot trajectory planning in joint space [51])

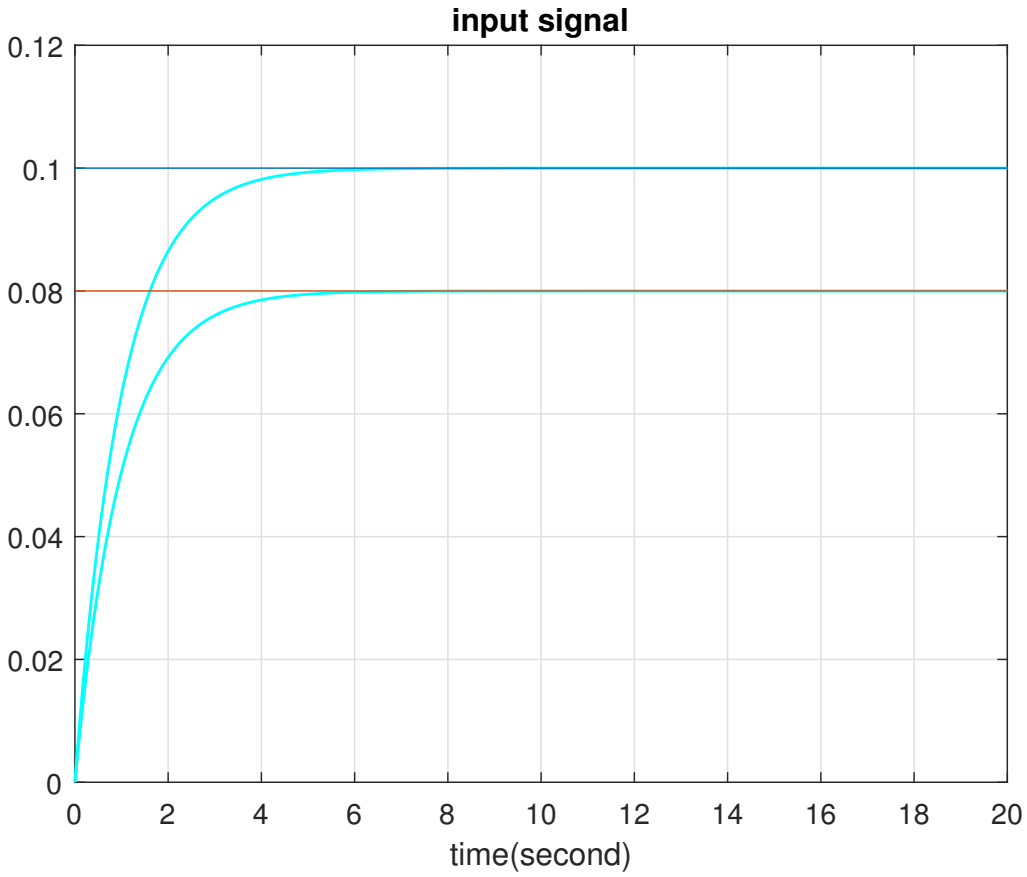


Figure 9: desired input signal

the filter transfer function is given by Tf= $\frac{1}{s+1}$ and input signal charactiristque:

	Risetime	SettlingTime	SettlingMin	SettlingMax	Overshoot	Undershoot	Peak	Peaktime
input1(θd)	2,197	3,912	0,090	0,099	0	0	0,099	20
input2(ψd)	2,197	3,912	0,072	0,079	0	0	0,079	20

Next, by using this parameters in the previous control law we obtain :

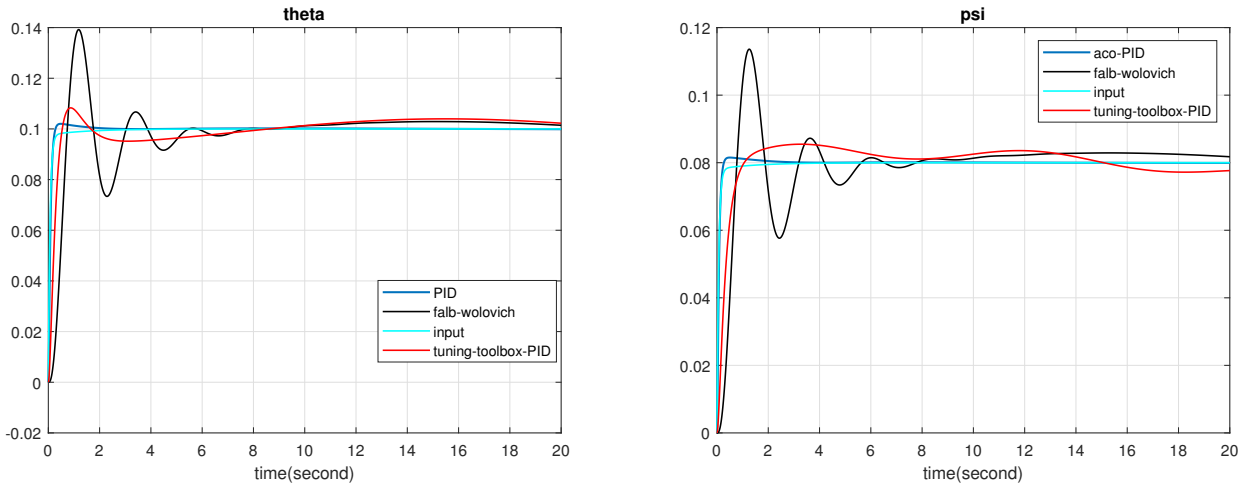


Figure 10: Trajectory tracking control using step input

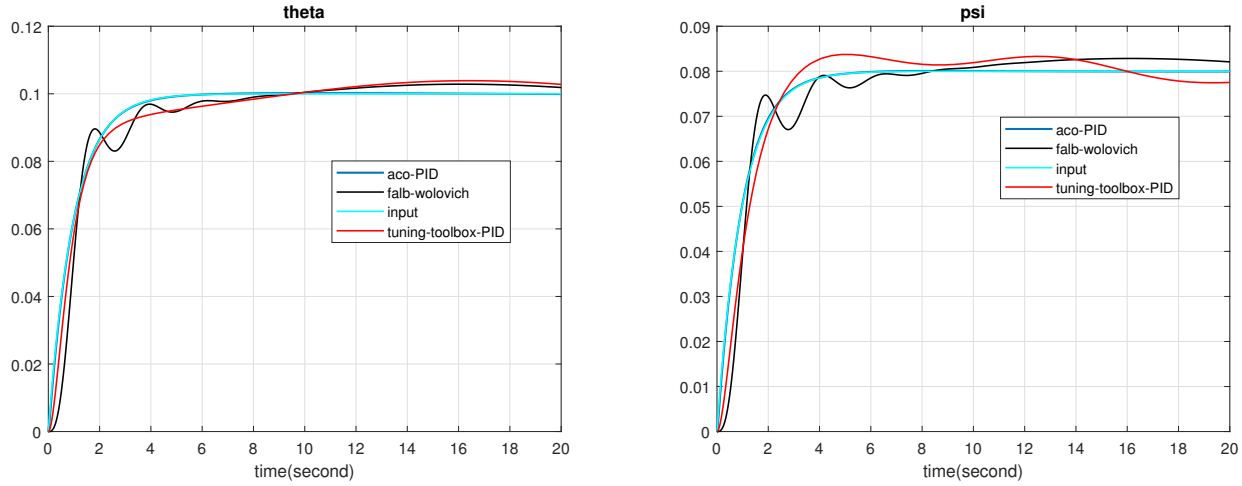


Figure 11: Trajectory tracking control using the desired input

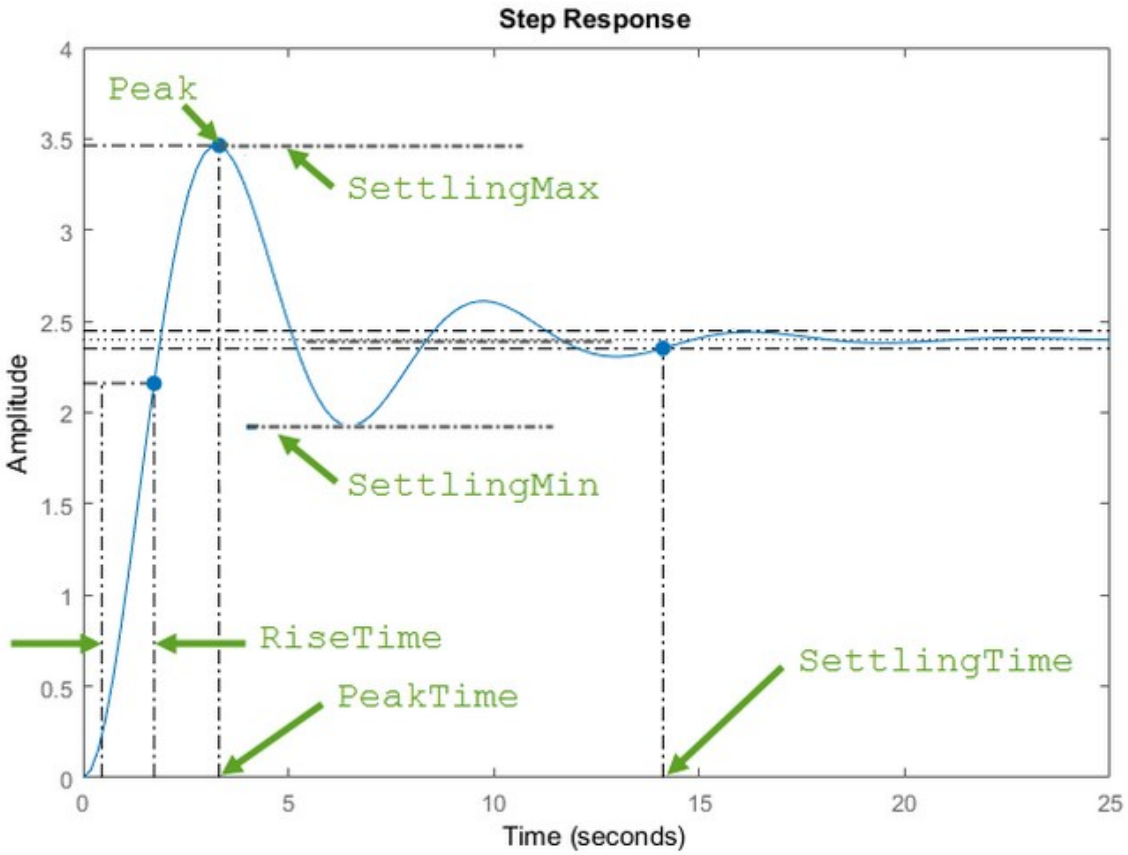
where the $PIDs$ parameters are :

MIMO-PID	K_p	K_i	K_d	T_f
$aco(*10^2)$	$\begin{bmatrix} 6.656 & 2.376 \\ 2.907 & 7.455 \end{bmatrix}$	$\begin{bmatrix} 3.516 & 2.856 \\ 2.786 & 4.361 \end{bmatrix}$	$\begin{bmatrix} 5.395 & 2.872 \\ 2.439 & 6.322 \end{bmatrix}$	
$matlab, tuning$	$\begin{bmatrix} 90.1 & 0 \\ 0 & 9.42 \end{bmatrix}$	$\begin{bmatrix} 1.14 & 0 \\ 0 & 27.9 \end{bmatrix}$	$\begin{bmatrix} 68.8 & 0 \\ 0 & 61.7 \end{bmatrix}$	$\begin{bmatrix} 0.000166 & 0 \\ 0 & 0.000118 \end{bmatrix}$

and the following table present the performance charactiristics of the simulationshown in figure (11)

	Risetime	SettlingTime	SettlingMin	SettlingMax	Overshoot	Undershoot	Peak	Peaktime
valb	2.833	9.311	0.091	0.102	0.961	0	0.102	16.129
wolovich	1.243	8.876	0.067	0.082	0.921	0	0.082	16.109
aco	2.169	3.932	0.089	0.100	0.282	0	0.100	10.385
PID	2.145	3.856	0.071	0.080	0.181	0	0.080	9.088
tuning	3.077	10.345	0.092	0.103	1.063	0	0.103	16.444
PID	1.877	16.693	0.072	0.083	8.010	0	0.083	4.856

By default, the rise time is Time it takes for the response to rise from 10% to 90% of the steady-state response, and SettlingTime Time it takes for the error [$y(t)$ - y_{final}] between the response $y(t)$ and the steady-state response y_{final} to fall to within 2% of y_{final}. the Settling(Min/max) is (Minimum/maximum) value of y(t) once the response has risen.and overshoot,undershoot are respectively the Percentage undershoot, overshoot relative to y_{final}. The following figure illustrates some of these quantities on a typical second-order response.



3.3.3.1 Discussion Of The Results:

the effect of the input on the system behavior it's clear and shown in figure (10) and (11) For aco tuning PID-controller method shows a good traking to the reference input $\theta_d = 0.1$ rad and $\psi_d = 0.08$ rad , is shown in Figure (11),(the input and the response signal are identical) ther is no static error and the rise time and the overshoot value is the favorable in Figure (10) we can eliminate the overshoot by changing the number of iteration in

ACO algorithm but the gain value of PID rise this cause more energy consommation by the controller in our case the satellite is pointed correctly, while, using the falb-wolovich controller large oscillations appear the static error is about tow times greater than the one obtained with the first controller, but for this method you can impose the value of the system unstable poles,and for the PID-tuned with matlab toolbox is remarkable that the gain value are small Comparisons with aco-PID but also we see that the settling time are considerable the satellite take 16 seconds to point to the reference angle,and the overshoot about 1 and 8 per cent of the input

3.4 Conclusion:

The possibility of improving the pointing accuracy of a flexible satellite, by utilizing direct state-space methods and state feedback decoupling for the controller design, has been shown to have a very attractive application. These methods appear powerful and simple to apply, whatever the complexity introduced in the flexible parts model. Simulation results also show that the sensitivity of these controllers is small as, in a case of high flexibility, In the example treated, the actuators' dynamics have been neglected, considering only a unity transfer function between the commanded electrical signal and the momentum supplied by the actuators. This fact is not relevant, as the dynamics of the particular actuator employed can be introduced simply by augmenting the state equations. It should be mentioned that the same results have been obtained, with both synthesis methods, considering or neglecting the sensor time constant in equations (21). In this study, a tuning PID method based on the multiobjective ant colony optimization is developed for getting good performances and tunes the optimal PID parameters. In contrast to the single-objective algorithms, which try to find a single solution of the problem, the multiobjective technique searches for the optimal Pareto set directly. The aim of the multiobjective ACO algorithm is to determine the optimal solutions of the PID controller parameters by minimization the multiobjective function and to identify the Pareto optimal solution. This method is able to ind the optimum solution of the PID controller's parameters ($K_p, K_i, andK_d$) that they allow to guarantee the performance of the system.

Chapter 4

Free hybrid model for Satellites with Flexible Appendages Attitude Control

4.1 Introduction:

We present here a method and some tools developed (Khalid H.M. 2008) to build linear models of multi-body systems for space applications (typically satellites). The multi-body system is composed of a main body (hub) fitted with rigid and flexible appendages (solar panels, antennas, propellant tanks ... etc) and on-board angular momentums (flywheels, control moment gyros). Each appendage can be connected to the hub by a cantilever joint or a pivot joint. More generally, this method can be applied to any open mechanical chain. In our approach, the rigid six degrees of freedom (three translational and three rotational) are treated all together. That is very convenient to build linear models of complex multi-body systems. Then, the dynamics model used to design AOCS, i.e. the model between forces and torques (applied on the hub) and angular and linear position and velocity of the hub, can be derived very easily. This model can be interpreted using block diagram representation.[1] Satellites and other spacecraft are typical multi-body mechanical systems that include both rigid and flexible bodies. Rigid bodies have been shown to be stabilized by applying torques on them [35], complex mechanical systems such as spacecraft are stabilized by the Attitude and Orbit Control System (AOCS) components,The design of the AOCS requires a linear model taking into account all the rigid and flexible couplings between the hub (where the AOCS acts) and the various appendages. The dynamic model, which relates linear and angular accelerations applied on a system, and the resulting the forces and torques are assumed to be linear. This linear assumption is quite realistic for such systems since perturbations and so motions are very small (except for very dexterous observation satellites)[3]. This linear assumption is furthermore valid in the field of future missions for deep space exploration involving formation flying of several spacecraft. For this kind of formation flying mission, it is more and more accepted that the 3 rotational degrees of freedom (d.o.f) and the 3 translational d.o.f's must be treated all together [36]. When deformable bodies make up a part of the multi-body system, then flexibility must be taken into account, many formulations for handling flexible multi-body dynamics have been investigated in [37]. Flexibility in multi-body systems, may be treated by finite element analysis[38, 39]. In this work, the Effective Mass Representation [40, 41] has been used in generating the flexible model of the mechanical elements in which flexibility is present, this use was made possible by applying the Cantilever Hybrid Model [40]. Hence a 6 d.o.f. model including couplings between rotations and translations must be developed [3] .

4.1.1 Mathimatical tools:

The Antisymmetric Matrix:
Before we start building the dynamic model, we will introduce the antisymmetric matrix, which helps us perform numerical calculations in matrix form. Let the angular velocity in the reference frame R_c be expressed as:

$$\vec{\omega}=\omega_x\vec{i}+\omega_y\vec{j}+\omega_z\vec{k}$$

and let the vector from point **G** to point **P** in the same frame of reference be given by:

$$\vec{r}_{GP}=x\vec{i}+y\vec{j}+z\vec{k}$$

Then the vector product of the two vectors expressed in the same reference frame R_c is given by:

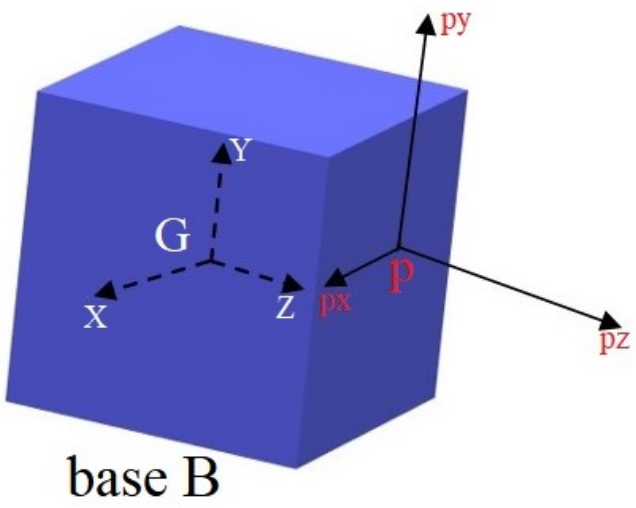
$$\vec{\omega}\wedge\vec{r}_{GP}=(\omega_y z-\omega_z y)\vec{i}+(\omega_z x-\omega_x z)\vec{j}+(\omega_x y-\omega_y x)\vec{k} \tag{34}$$

This vector product can also be written in the matrix form:

$$\vec{\omega}\wedge\vec{r}_{GP}=[\vec{r}_{PG}]\,\vec{\omega}=\begin{bmatrix}0 & z & -y\\ -z & 0 & x\\ y & -x & 0\end{bmatrix}\begin{bmatrix}\omega x\\ \omega y\\ \omega z\end{bmatrix} \tag{35}$$

4.2 Interconnected rigid bodies model:

Let us consider a spacecraft composed of a rigid main body or hub (called here the base **B**) with its center of mass at point **G**, and an appendage cantilevered to the base **B** at points **P** (see Figure). Let us denote $R_G = (G;x;y;z)$ the reference frame rigidly attached to the hub at **G** and $R_P = (P;x;y;z)$) the same frame translated to points **P**. In the sequel the dynamic model of the appendage will be obviously given in the frame R_P. Let us consider the base **B** alone (without appendage),



4.2.1 Newton-Euler equations at the center of mass:

according to the Newton's and Euler's equations, the dynamic model of the base **B** at its center of mass **G** reads as follows:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix} = D_G^B \begin{bmatrix} \vec{a}_G \\ \dot{\vec{\omega}} \end{bmatrix} = \begin{bmatrix} mI_{3\times3} & 0 \\ 0 & \mathbb{J}_G^B \end{bmatrix} \tag{36}$$

were:
$\vec{\omega}$: is the absolute angular velocity vector of the satellite body (i.e. the angular velocity of the frame R_G w.r.t the inertial frame R_i in (rad/s)) .
and $\dot{\vec{\omega}}=\frac{d(\vec{\omega})}{dt}|_{R_G}=\frac{d(\vec{\omega})}{dt}|_{R_i}$ since $\vec{\omega}$ has the same coordinates in R_G and R_i

4.2.2 Newton-Euler equations at any reference point:

In the above equation, the three translational accelerations and the three angular accelerations are considered together. Note that for the rotation dynamics, the relation $\vec{T}_{ext}=\mathbb{J}_G^B \dot{\vec{\omega}}+\vec{\omega}\wedge(\mathbb{J}_G^B\ \vec{\omega})$ were $\wedge$ is the cross product.The nonlinear term $\vec{\omega}\wedge(\mathbb{J}_G^B\ \vec{\omega})$ on the right hand side of the equation above can be neglected if angular velocity $\vec{\omega}$ is small enough (linear assumption). Let us recall that the relation between the velocities at points **P** and **G** is:

$$\vec{V_P} = \vec{V_G} + \vec{\omega} \wedge \vec{r_{GP}} = \vec{V_G} + [r_{PG}]\vec{\omega} \tag{37}$$

where $[r_{PG}]$ is the antisymmetric matrix associated with the vector $\vec{r_{GP}}$, if $[x; y; z]_G^T$ is the coordinate vector of $\vec{r_{SA}}$ projected in any frame R_G then $[r_{PG}]$ reads:

$$[r_{PG}]=\begin{bmatrix} 0 & z & -y \\ -z & 0 & x \\ y & -x & 0 \end{bmatrix} \quad [r_{GP}]=\begin{bmatrix} 0 & z & -y \\ -z & 0 & x \\ y & -x & 0 \end{bmatrix}^T \text{ and } [r_{GP}]=[r_{PG}]^T$$

Rigid Body Kinematics:
Note that velocity equation allows a vector product to be transformed into a matrix-vector product and can be projected in any frame. Then, the six DOF kinematic vectors $\vec{V_G}$ and $\vec{V_P}$ of the satellite body respectively at points **G** and **P** These six-component vectors are related to each other by:

$$\begin{bmatrix} \vec{V_G} \\ \vec{\omega} \end{bmatrix} = \mathbb{T}_{GP} \begin{bmatrix} \vec{V_P} \\ \vec{\omega} \end{bmatrix} = \begin{bmatrix} I_{3\times3} & [r_{GP}] \\ 0 & I_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{V_P} \\ \vec{\omega} \end{bmatrix} \tag{38}$$

where we have introduced the 6x6 kinematic model between the points G and P:

$$\mathbb{T}_{GP} = \begin{bmatrix} I_{3\times3} & [r_{GP}] \\ 0 & I_{3\times3} \end{bmatrix}$$

Now let us consider theinertial acceleration at point **G** and **P** :

$$\vec{a_G} = \left|\frac{d\vec{V_G}}{dt}\right|_{R_i} \qquad and \qquad \vec{a_P} = \left|\frac{d\vec{V_p}}{dt}\right|_{R_i} \tag{39}$$

its well-known that:

$$\vec{a_p} = \vec{a_G} + \dot{\vec{\omega}} \wedge \vec{r}_{GP} + \vec{\omega} \wedge \left[\left| d\frac{d\vec{r}_{GP}}{dt} \right|_{R_G} + \omega \wedge \vec{r}_{GP} \right] \tag{40}$$

for regid body, $\left| \frac{d\vec{r}_{GP}}{dt} \right|_{R_G} = 0$;

and, as explained before, all nonlinear terms can be neglected. The acceleration at point **P** is then deduced from the acceleration at point **G** by the linear relation:

$$\vec{a_p} = \vec{a_G} + \dot{\vec{\omega}} \wedge \vec{r}_{GP} = [\vec{r}_{PG}]\dot{\vec{\omega}} \tag{41}$$

From the last equation one can derive the following kinematic relationship:

$$\begin{bmatrix} \vec{a}_G \\ \dot{\vec{\omega}} \end{bmatrix} = \mathbb{T}_{GP} \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} = \begin{bmatrix} I_{3\times3} & [r_{GP}] \\ 0 & I_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \tag{42}$$

To obtain the relationship between the 6 DOF external force vectors at point **G** and at point **P**, it is interesting to express the external force power computed along a virtual velocity field :

$$\mathbf{power}_{ext} = \begin{bmatrix} \vec{V}_G \\ \vec{\omega} \end{bmatrix}^T \begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RG} = \begin{bmatrix} \vec{V}_P \\ \vec{\omega} \end{bmatrix}^T \begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RP} \tag{43}$$

Combining (38) and (43), one can easily obtain:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RP} = \mathbb{T}_{GP}^T \begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RG} = \mathbb{T}_{GP}^T D_G^B \begin{bmatrix} \vec{a}_G \\ \dot{\vec{\omega}} \end{bmatrix} = \mathbb{T}_{GP}^T D_G^B \mathbb{T}_{GP} \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \tag{44}$$

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RP} = D_p^B \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \tag{45}$$

Thus the transport of the direct dynamic model of a satellite body from a point **G** to a point **P** reads

$$D_P^B = \mathbb{T}_{GP}^T D_G^B \mathbb{T}_{GP} = \begin{bmatrix} mI_{3\times3} & m[\vec{r}_{GP}] \\ -m[\vec{r}_{GP}] & \mathbb{J}_G^B - m[\vec{r}_{GP}]^2 \end{bmatrix} \tag{46}$$

If $\vec{r}_{CP}$ is the vector between **P** and the center of mass **C** of the appendage in the frame R_P, we can also write:

$$D_P^A = \mathbb{T}_{CP}^T D_G^A \mathbb{T}_{CP} = \mathbb{T}_{CP}^T \begin{bmatrix} m_A I_{3\times3} & 0 \\ 0 & \mathbb{J}_C^A \end{bmatrix} \mathbb{T}_{CP} = \begin{bmatrix} m_A I_{3\times3} & m_A[\vec{r}_{CP}] \\ -m_A[\vec{r}_{CP}] & \mathbb{J}_C^A - m_A[\vec{r}_{CP}]^2 \end{bmatrix} \tag{47}$$

4.3 Model of a flexible appendage:

Flexibility of an appendage will be represented by the effective mass approach (Imbert and Mamode [1977]). This representation is very useful when one want to study dynamic couplings between the flexible modes of the appendage and the rigid modes of the whole

system without analysis of internal deformations (or loads) of the appendage. The so called "Cantilever Hybrid Model"' (see Cumer and Chrétien [2001]) will be used: at point **P**, the **static-dynamic** model of the appendage is now governed by the following differential equations:

$$\begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RP} = D_P^A \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} + {L_p}^T \ddot{\xi}_P = \begin{bmatrix} m_A I_{3\times 3} & m_A[\vec{r}_{CP}] \\ -m_A[\vec{r}_{CP}] & \mathbb{J}_C^A - m_A[\vec{r}_{CP}]^2 \end{bmatrix} \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} + \begin{bmatrix} p_p \\ \ddot{..} \\ \ddot{..} \\ \ddot{..} \\ H_p \end{bmatrix} \ddot{\xi}_p \tag{48}$$

$$\ddot{\xi}_p + diag(2\xi_i\omega_i)\dot{\xi}_p + diag({\omega_i}^2)\xi_p = -\begin{bmatrix} {p_p}^T, ..., {H_p}^T \end{bmatrix} \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \tag{49}$$

were $\left\{ L_p = \begin{bmatrix} {p_p}^T, ..., {H_p}^T \end{bmatrix} = \begin{bmatrix} L_p^{1T}, L_p^{2T}, ..., L_p^{kT} \end{bmatrix} \right\}$, ω_i and ξ_i $(i = 1, 2, ..., k)$ are the modal contribution at connection point **P**, the frequency, and the damping ratio of the flexible mode i respectively , for $(i = 1, \ldots, k)$ with (k is the number of flexible modes taken into account). ξ_A is the vector of flexible modal coordinates. ($\mathbb{I}_P$=$\mathbb{J}_C^A$-$m_A[r_{GP}]^2$) the moment of inertia of an appendage. To make the description more compact we denote the following : $(D = diag(2\xi_i\omega_i))$, $(K = diag({\omega_i}^2))$ and (Q_p=-$diag(2\xi_i\omega_i)\dot{\xi}_p$) were D is called flexible mode damping ratios , K is called flexible mode stiffness coefficients and Q_p is called generalized force of solar array . Also as a notable remark we should mention that the modal momentum/angular-momentum coefficients p_p and H_p can be calculated using the mass matrix and mode shape obtained by NASTRAN modal analysis[22]. The direct dynamic model of the appendage can also be described by the state-space representation:

Flexibility in state-space form:

$$\begin{cases} \begin{bmatrix} \dot{\xi}_P \\ \ddot{\xi}_P \end{bmatrix} = \begin{bmatrix} 0_{k\times k} & I_{3\times 3} \\ -K_{k\times k} & -D_{k\times k} \end{bmatrix} \begin{bmatrix} \xi_P \\ \dot{\xi}_P \end{bmatrix} + \begin{bmatrix} 0_{k\times 6} \\ \ddot{..} \\ \ddot{..} \\ \ddot{..} \\ -L_p \end{bmatrix} \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \\ \begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RP} = -\begin{bmatrix} {L_p}^T K_{k\times k} & {L_p}^T D_{k\times k} \end{bmatrix} \begin{bmatrix} \xi_P \\ \dot{\xi}_P \end{bmatrix} + (D_P^A - {L_p}^T L_p) \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \end{cases} \tag{50}$$

This state-space representation allows the direct transfer matrix $M_p^A(s)$ between force and acceleration of the appendage at point **P** (also called dynamic mass matrix) to be computed:

$$\begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RP} = M_p^A(s) \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \tag{51}$$

$$M_p^A(s) = -\left[\begin{array}{cc} {L_p}^T K_{k\times k} & {L_p}^T D_{k\times k} \end{array}\right] \left[\begin{array}{cc} S\, I_{k\times k} & -I_{3\times 3} \\ K_{k\times k} & (S\, I_{k\times k} + D_{k\times k}) \end{array}\right]^{-1} \left[\begin{array}{c} 0_{k\times 6} \\ .. \\ .. \\ -L_p \end{array}\right] + (D_P^A - {L_p}^T L_p) \tag{52}$$

In the case where flexible mode damping ratios are neglected ($D_{k\times k} = 0$), this transfer matrix can be re-arranged in the following way:

$$M_p^A(s) = \mathbf{DC}_{Gain} + \sum_{i=1}^{k} M_i \left(\frac{{\omega_i}^2}{s^2 + {\omega_i}^2}\right) \tag{53}$$

Where:
$\mathbf{DC}_{Gain}$: is called the **DC** gain or residual mass matrix rigidly cantilevered to the base **B** at point **p** and is given by:

$$\mathbf{DC}_{Gain} = M_p^A(\infty) = (D_P^A - {L_p}^T L_p) = (D_P^A - \textstyle\sum_{i=1}^{k} {l_i^p}^T l_i^p)$$

$M_i = \sum_{i=1}^{k} {l_i^p}^T l_i^p$: is rank-1 effective-mass matrix of the i^{th} mode,

Remarque:

All these data (D_P^A, ω_i, ξ_i, and l_P^i) are directly provided by the finite element software used to model such an appendage according to the number N of flexible modes retained in the model and are independent of the main body characteristics.
Another well-known frequency-domain representation of $M_P^A(s)$, also called effective mass/inertia model (Imbert,J.,1991.) can be easily derived from (52) where flexible mode damping ratios are not neglected ($D_{(k\times k)} \neq 0$):

$$M_p^A(s) = \mathbf{DC}_{Gain} + \sum_{i=1}^{k} M_i \left(\frac{2\xi_i\omega_i s + {\omega_i}^2}{s^2 + 2\xi_i\omega_i s + {\omega_i}^2}\right) \tag{54}$$

The appendage dynamic model $M_P^A(s)$ can also be represented by the block-diagram depicted in Figure (12).

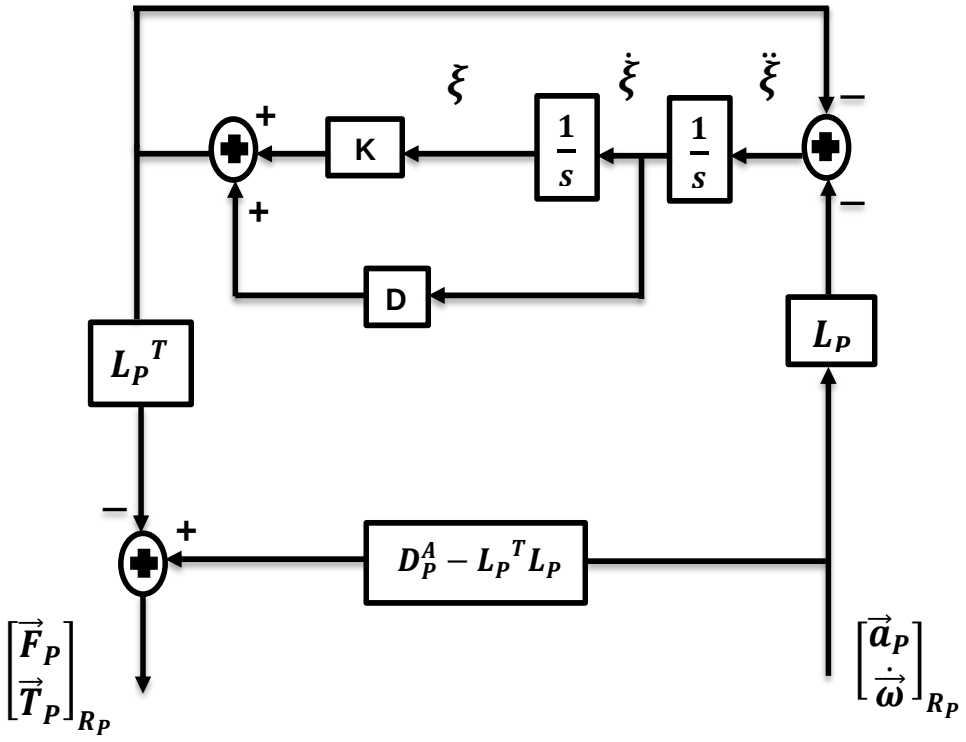


Figure 12: Appendage dynamic model $M_P^A(s)$: block-diagram representation

4.4 Connection of Rigid Body by a Flexible Appendage:

If we consider now that a flexible appendage **A** is cantilevered to the base **B** at point **P**, the reaction forces $\vec{F}_P$ and torque $\vec{T}_P$ at point **P** between the base and the appendage must be taken into account in the dynamic model of the base. Thus equation (45) becomes:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RP} = \begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RP} + D_p^B \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} = \begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RP} + \begin{bmatrix} mI_{3\times 3} & m[\vec{r}_{GP}] \\ -m[\vec{r}_{GP}] & \mathbb{J}_G^B - m[\vec{r}_{GP}]^2 \end{bmatrix} \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \tag{55}$$

The appendage **A** is characterized by its own dynamic model D_P^A at point **P**. If we assume that the only force and torque applied on the appendage **A** are the reaction force and torque with the base **B**, then one can write:

$$\begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RP} = D_P^A \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \quad and \quad \begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RG} = D_G^A \begin{bmatrix} \vec{a}_G \\ \dot{\vec{\omega}} \end{bmatrix} \tag{56}$$

Substituting (56) in (55) we get the equation of motion of the whole system at point **P**:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RP} = (D_P^A + D_P^B) \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} = (D_P^A + \mathbb{T}_{GP}^T D_G^B \mathbb{T}_{GP}) \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} \tag{57}$$

It could be more interesting to express the whole dynamic model at the center of mass **G** of the base **B**, since the external forces and torques will correspond to the Attitude and Orbit Control System **AOCS** (reaction wheel and thrust) which are mounted on the base. Then, From equations (56) and (57) it can be shown that:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RG} = \begin{bmatrix} \vec{F}_P \\ \vec{T}_P \end{bmatrix}_{RG} + D_G^B \begin{bmatrix} \vec{a}_G \\ \dot{\vec{\omega}} \end{bmatrix} \tag{58}$$

The block diagram representation of the dynamic model **H(s)** of the coupled system, presented in the next Fig, shows that the direct appendage dynamic model $M_P^A(s)$ at point **P** interacts as a feedback on the direct main body dynamic model(D_G^B). Consequently, characteristic parameters of each body can be highlighted in such a block-diagram representation.the model of the coupled system can be represented by the block-diagram depicted in Fig.

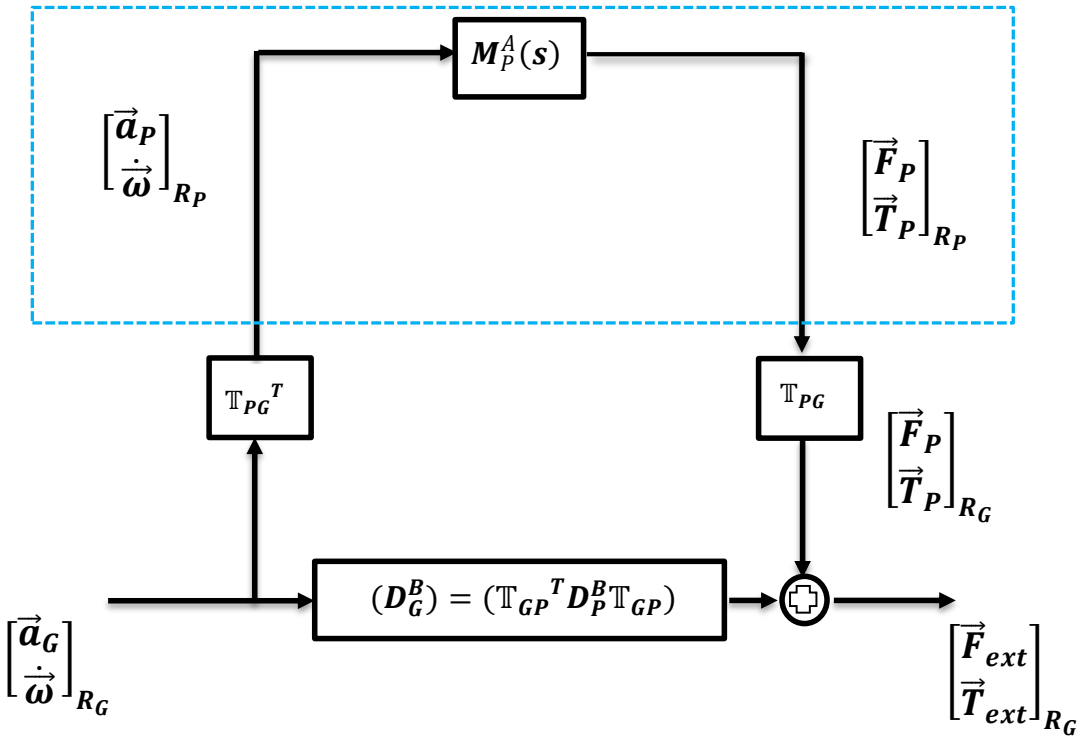


Figure 13: the model of the coupled system: block-diagram representation

4.5 Block diagram representation and simulation in simulink:

4.5.1 Introduction:

In (4.4) we introduced the theoretical background and derived the results that we expect to have. Here we will present that analysis in a block diagram, and simulate it on Simulink (by Mathworks,Inc.)

4.5.2 The Dynamic Model Of a Satellite with one Flexible Appendage to a Rigid Connection with Translation only:

Using the equations we had in (4.4), the block-diagram represented in figure (13) can be described by the following simulink model:

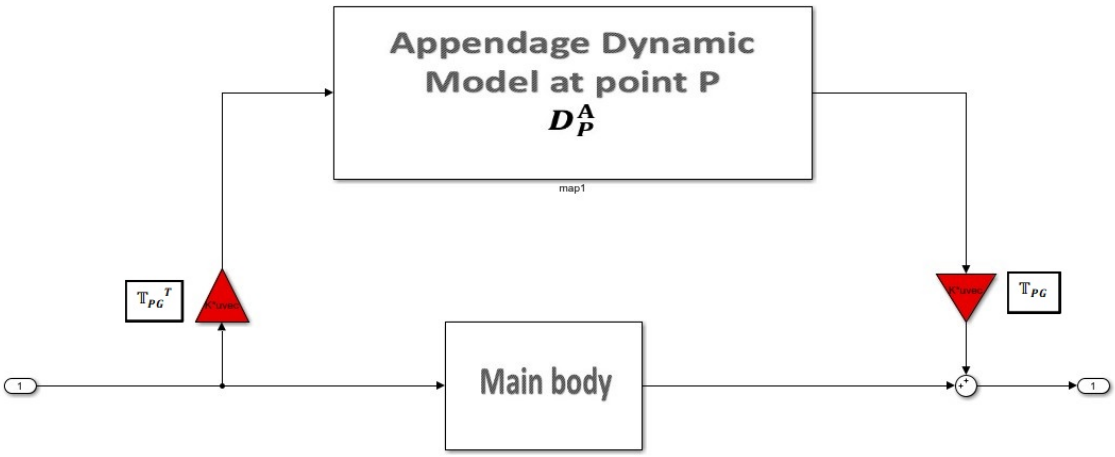


Figure 14: simulink Dynamic Model with Only Translation to a Rigid Connection

main body block is the direct main body dynamic model (D_G^B) . D_P^A block is direct transfer function of the appendage at point **p** the signal entering the positive side of the summation block is the $[F_{ext}; T_{ext}]$ vector at point **P** as illustrated in the figure, thus the force and torque resulting from the appendage can be subtracted directly from that signal because the dynamic model of the appendage is taken at point **P** as well. While in figure, the $[F_{ext}; T_{ext}]$ vector that enters the summation block is that at point **G**, hence, the dynamic model of the appendage has to be translated to point **G** from point **p** before it is subtracted, this is done by pre and post multiplying it by ${\mathbb{T}_{PG}}^T$ and $\mathbb{T}_{PG}$ respectively. Notice that:$\mathbb{T}_{PG}$=${\mathbb{T}_{GP}}^{-1}$

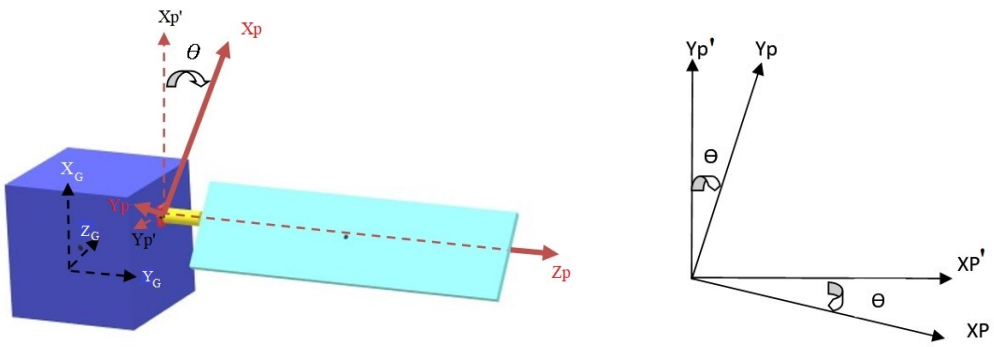
4.5.3 The Dynamic Model Of a Satellite with one Flexible Appendage to a Rigid Connection with Translation and rotation:

In the case where the base **B** and the appendage **A** are linked by a pivot joint (around the z_P axis), the reaction torque about the z_P axis is null.

Then (51) projected in the frame $(P; x_P; y_P; z_P)$ becomes:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix}_{RP} = M_p^A(s) \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} with \ \vec{F_P} = \begin{bmatrix} F_{px} \\ F_{py} \\ F_{pz} \end{bmatrix}, \vec{T_P} = \begin{bmatrix} T_{px} \\ T_{py} \\ 0 \end{bmatrix}, \vec{a_P} = \begin{bmatrix} a_{px} \\ a_{py} \\ a_{pz} \end{bmatrix},$$
$$\vec{\dot{\omega}_p} = \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z + \ddot{\theta} \end{bmatrix} \tag{59}$$

Now to introduce rotation into the system that we have built before it is necessary to know the rotation matrix associated with the axis which rotation is performed [6] . we mean rotating the second reference frame (R_p, x, y, z) by certain degrees about one axis with respect to the original reference frame (R_G, x, y, z) Again we will consider a satellite main body with one appendage attached to it through a rigid connection, but this time the local reference frame of the appendage is located by a certain angle as discussed before. This case was studied by equation the rotation matrix R_3 between the rotationg frame $R_P = (P; x_P; y_P; z_P)$ (in which the dynamic model will be described) and the frame $R_{p'}$=(p, x'_p, y'_p, z'_p) parallel to R_G at point P must be taken into account. That is illustrated in the next Figure, in the special case where the appendage is rotated with an angle θ around z-axis:



$$R_3 = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

While the (6×6) rotation is given by:

$$R_6 = \begin{bmatrix} R_3 & 0_{3\times3} \\ 0_{3\times3} & R_3 \end{bmatrix}$$

Now we need to introduce the transition matrix , it is simply a rotation matrix that describes the transition from the main body reference frame to the new reference frame at point P that was obtained after rotation [3] , i.e. the transition matrix will rotate the frame $R_{p'_1}$=$(p_1, x'_{p1}, y'_{p1}, z'_{p1})$ to be aligned with the frame R_G= (G, x, y, z). in this case :

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(-\frac{\pi}{2}) & \sin(-\frac{\pi}{2}) \\ 0 & -\sin(-\frac{\pi}{2}) & \cos(-\frac{\pi}{2}) \end{bmatrix} \tag{60}$$

and the(6×6) transition matrix :

$$TM = \begin{bmatrix} T & 0 \\ 0 & T \end{bmatrix} \tag{61}$$

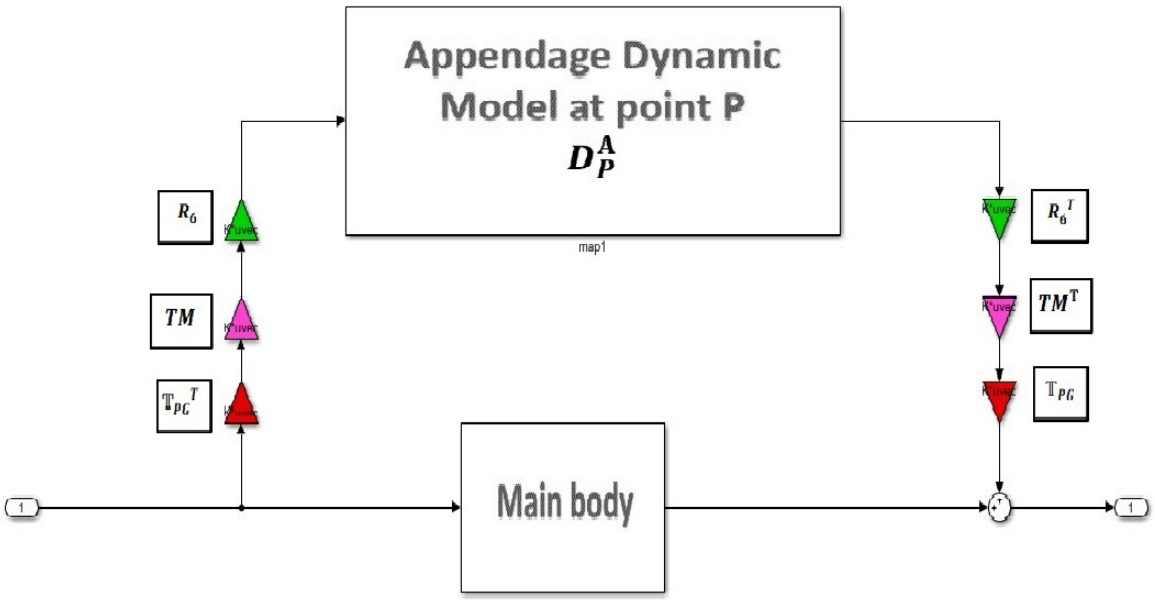


Figure 15: dynamic model of flexible solar array with rotation

the m-file programme is given by :

```
close all
clear all
clc
%  DATA

teta=10*pi/180;%angle of rotation of solar panrl 1 (rad)
alpha=-pi/2%angle of rotation use in transition matrix of solar panrl 1 (rad)
ma=100;%solar pannel 1,2 mass (kg)
m=2000;%main body mass (kg)
lp=[0 0 5 90 0 0;0 0 0 0 14 0;-3 0 0 0 0 119;0 0 4 62 0 0];%the model contribution of beams at point P,P1
wis=2*pi*[0.04 0.111 0.13 0.27];%angular frequencies of the K=4 flexible mode
[rgp]=[0 2 0;-2 0 0;0 0 0];%antisymmetric matrix associated with the vector rgp(distance from G to P)
[rcp]=[0 8 0;-8 0 0;0 0 0];%antisymmetric matrix associated with the vector rcp(distance from p to c)
psis=0.001;%damping ratio of the flexible mode

tt=[1 0 0;0 cos(alpha) sin(alpha);0 -sin(alpha) cos(alpha)]%rotaion matrix use in transition matrix solar panel 1
tm=[tt zeros(3,3);zeros(3,3) tt];%transition matrix solar panel 1

i=eye(3,3);%is the 3*3 identity matrix
jac=[7000 0 0;0 200 0;0 0 10000];%moment of inertia tesor of the appendage with respect to C,C1 written in Rp,Rp1
r3=[cos(teta) sin(teta) 0;-sin(teta) cos(teta) 0;0 0 1];%the 3*3 rotation matrix solar pannel 1
r31=[cos(teta1) sin(teta1) 0;-sin(teta1) cos(teta1) 0;0 0 1];%the 3*3 rotation matrix solar pannel 2
jbg=[2000 100 50;100 8000 80;50 80 8000];%moment of inertia tensor of the main body with respect to G writen in RG
```

```
KS=diag(wis.*wis);%flexible mode ration
DS=diag(2*psis.*wis);%flexible stiffnes coefficient

%pannel 1

[rcp2]=[rcp]*[rcp];%[rcp] power tow
a=ma*i;
b=ma*[rcp];
c=-(ma*[rcp]);
d=jac-(ma*[rcp2]);
dap=[a b;c d];% static-dynamic model of A at point p
tgp=[eye(3,3) [rgp];zeros(3,3) eye(3,3)];%transport
   direct dynamic model of the hub body B from G to P
tpg=inv(tgp);%the invers of tgp
dcgain=dap-(lp'*lp);%residual mass matrix rigidly
   contelivred to the base B at point P
cgp=[r3 zeros(3,3);zeros(3,3) r3];%the 6*6 rotation
   matrix
dag=[ma*i zeros(3,3);zeros(3,3) jac];% static-dynamic
   model of A at point G

%main body

dbg=[m*i zeros(3,3);zeros(3,3) jbg];% static-dynamic
   model of B at point G
```

4.5.4 The Dynamic Model Of a Satellite with [n] Flexible Appendage to a Rigid Connection with Translation and rotation:

now we suppose that the satellite contain tow flexible solar array in this case,it is easy to prove that the model of the main body is[6]:

$$\begin{bmatrix} \vec{F}_{ext} - \sum_{i=1}^{n} \vec{F}_{B/p_i} \\ \vec{T}_{ext} - \sum_{i=1}^{n} \vec{T}_{B/p_i} \end{bmatrix} = D_G^B \begin{bmatrix} \vec{a}_G \\ \dot{\vec{\omega}} \end{bmatrix} \tag{62}$$

and:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext} \end{bmatrix} = \sum_{i=1}^{n} \begin{bmatrix} \vec{F}_{B/p_i} \\ \vec{T}_{B/p_i} \end{bmatrix} + D_G^B \begin{bmatrix} \vec{a}_G \\ \dot{\vec{\omega}} \end{bmatrix} \tag{63}$$

with n is the number of appendages in this case (n=2)
befor we present the general dynamic model we need to modulate the seconde

appendage that has the same charactiristics ((ω_i)frequency of the flexible mode. (ξ_i): damping ratio of the flexible mode.,mass...) but the joint point situate in the other side. model of the appendage is now governed by the following differential equations:

$$\begin{bmatrix} \vec{F}_{P_1} \\ \vec{T}_{P_1} \end{bmatrix}_{RP_1} = D_{P_1}^{A_1} \begin{bmatrix} \vec{a}_{P_1} \\ \dot{\vec{\omega}} \end{bmatrix} + {L_{p_1}}^T \ddot{\xi}_{P_1} \tag{64}$$

in general case the dynamic model:

$$\begin{bmatrix} \vec{F}_{P_n} \\ \vec{T}_{P_n} \end{bmatrix}_{RP_n} = D_{P_n}^{A_n} \begin{bmatrix} \vec{a}_{P_n} \\ \dot{\vec{\omega}} \end{bmatrix} + {L_{p_n}}^T \ddot{\xi}_{P_n} \tag{65}$$

After building the model of the flexible appendage, we connect it to the satellite system, by simply putting it in place of the block of the appendage in the next models. Figure (16) illustrates a block diagram of a flexible appendage.

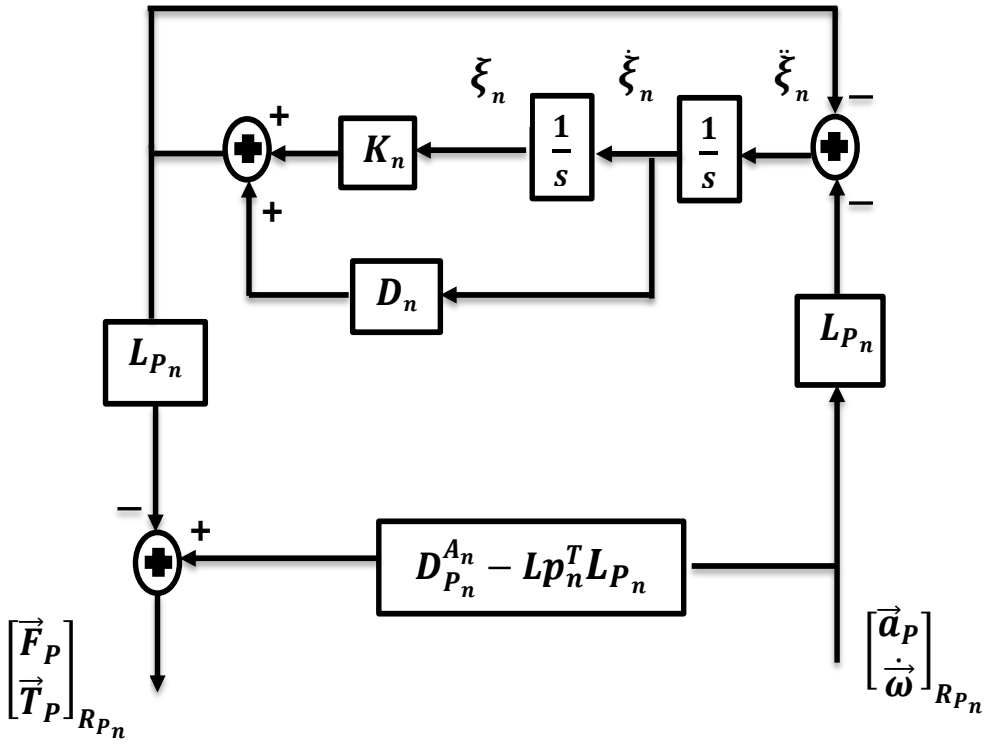


Figure 16: Appendage dynamic model $M_{P_n}^{A_n}(s)$: block-diagram representation

and thefollowing figure shows the Simulink model of a satellite hub with two flexible appendages.

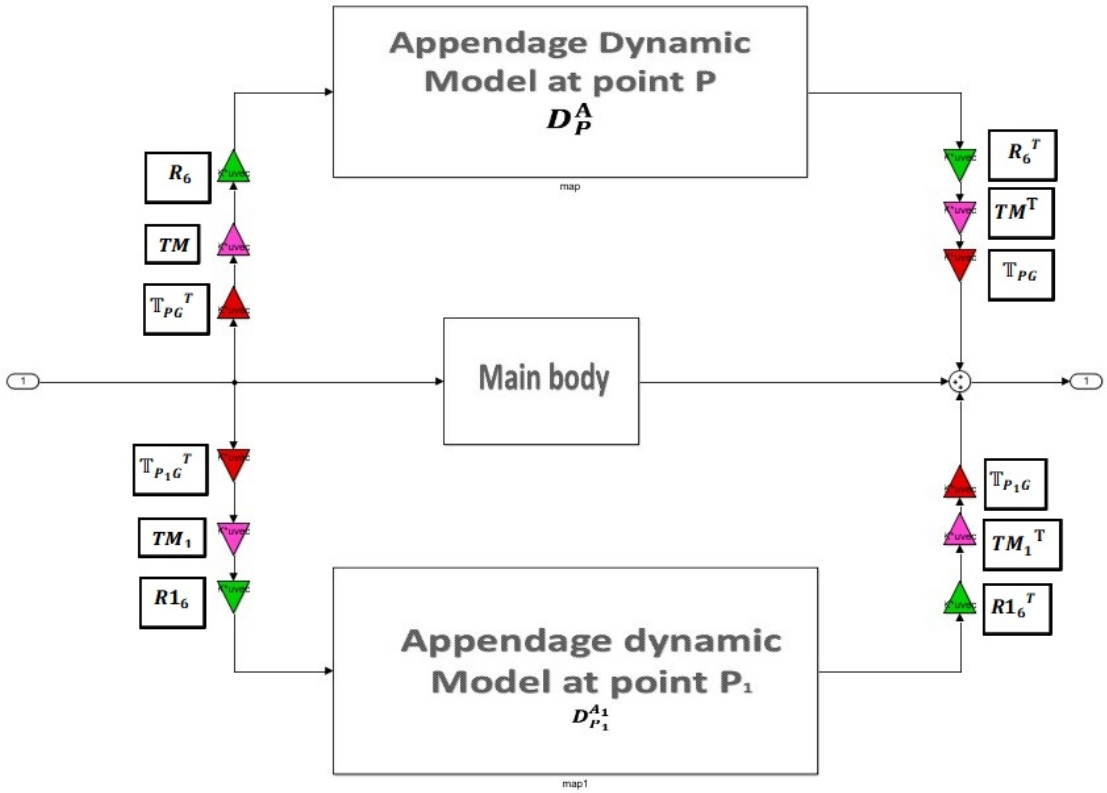


Figure 17: dynamic model of a satellite with two flexible solar array

we have introduced the transition matrix and the next figure present the matrix angles :

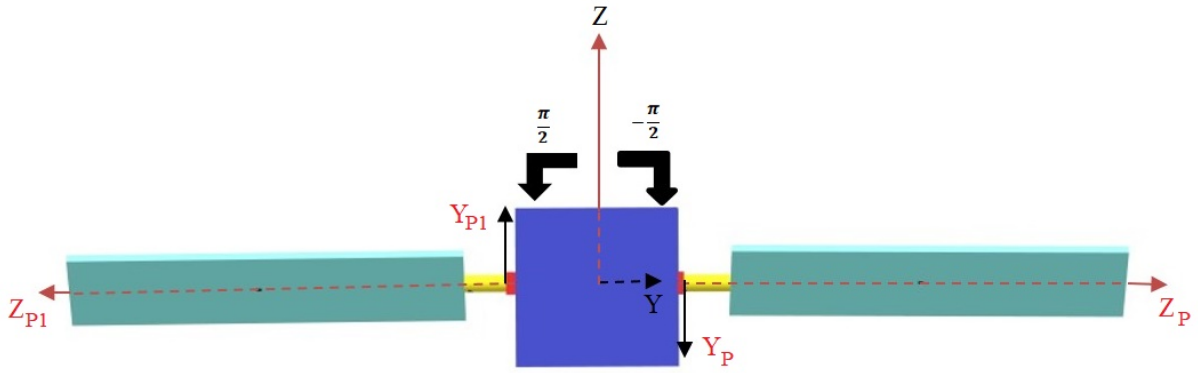


Figure 18: transition matrix angels

and the m-file data :

```
close all
clear all
clc
%DATA

teta=10*pi/180;%angle of rotation of solar panrl 1 (rad)
teta1=-10*pi/180;%angle of rotation of solar panrl 2 (rad)
alpha=-pi/2%angle of rotation use in transition matrix of solar panrl 1 (rad)
alpha1=pi/2,%angle of rotation use in transition matrix of solar panrl 2 (rad)
ma=100;%solar pannel 1,2 mass (kg)
m=2000;%main body mass (kg)
lp=[0 0 5 90 0 0;0 0 0 0 14 0;-3 0 0 0 0 119;0 0 4 62 0 0];%the model contribution of beams at point P,P1
wis=2*pi*[0.04 0.111 0.13 0.27];%angular frequencies of the K=4 flexible mode
[rgp]=[0 2 0;-2 0 0;0 0 0];%antisymmetric matrix associated with the vector rgp(distance from G to P)
[rcp]=[0 8 0;-8 0 0;0 0 0];%antisymmetric matrix associated with the vector rcp(distance from p to c)
psis=0.001;%damping ratio of the flexible mode

[rgp1]=[0 -2 0;2 0 0;0 0 0];%antisymmetric matrix associated with the vector rgp1(distance from G to P1)
[rcp1]=[0 -8 0;8 0 0;0 0 0];%antisymmetric matrix associated with the vector rcp1(distance from p1 to c1)

tt=[1 0 0;0 cos(alpha) sin(alpha);0 -sin(alpha) cos(alpha)]%rotaion matrix use in transition matrix solar panel 1
tm=[tt zeros(3,3);zeros(3,3) tt];%transition matrix solar panel 1
tt1=[1 0 0;0 cos(alpha1) sin(alpha1);0 -sin(alpha1) cos
```

```
    (alpha1)]%rotaion matrix use in transition matrix
    solar panel 2
tm1=[tt1 zeros(3,3);zeros(3,3) tt1]%transition matrix
    solar panel 2

i=eye(3,3);%is the 3*3 identity matrix
jac=[7000 0 0;0 200 0;0 0 10000];%moment of inertia
    tesor of the appendage with respect to C,C1 written
    in Rp,Rp1
r3=[cos(teta) sin(teta) 0;-sin(teta) cos(teta) 0;0 0
    1];%the 3*3 rotation matrix solar pannel 1
r31=[cos(teta1) sin(teta1) 0;-sin(teta1) cos(teta1) 0;0
     0 1];%the 3*3 rotation matrix solar pannel 2
jbg=[2000 100 50;100 8000 80;50 80 8000];%moment of
    inertia tensor of the main body with respect to G
    writen in RG
KS=diag(wis.*wis);%flexible mode ration
DS=diag(2*psis.*wis);%flexible stiffnes coefficient

%pannel 1

[rcp2]=[rcp]*[rcp];%[rcp] power tow
a=ma*i;
b=ma*[rcp];
c=-(ma*[rcp]);
d=jac-(ma*[rcp2]);
dap=[a b;c d];% static-dynamic model of A at point p
tgp=[eye(3,3) [rgp];zeros(3,3) eye(3,3)];%transport
    direct dynamic model of the hub body B from G to P
tpg=inv(tgp);%the invers of tgp
dcgain=dap-(lp'*lp);%residual mass matrix rigidly
    contelivred to the base B at point P
cgp=[r3 zeros(3,3);zeros(3,3) r3];%the 6*6 rotation
    matrix
dag=[ma*i zeros(3,3);zeros(3,3) jac];% static-dynamic
    model of A at point G

%pannel 2

```

```
[rcp21]=[rcp1]*[rcp1];%[rcp] power tow
a1=ma*i;
b1=ma*[rcp1];
c1=-(ma*[rcp1]);
d1=jac-(ma*[rcp21]);
dap1=[a1 b1;c1 d1];% static-dynamic model of A at point
    p
cgp1=[r31 zeros(3,3);zeros(3,3) r31];%the 6*6 rotation
   matrix
tgp1=[eye(3,3) [rgp1];zeros(3,3) eye(3,3)];%transport
   direct dynamic model of the hub body B from G to P1
tpg1=inv(tgp1);%the invers of tgp1
dcgain1=dap1-(lp'*lp);% static-dynamic model of A1 at
   point G

%main body

dbg=[m*i zeros(3,3);zeros(3,3) jbg];% static-dynamic
   model of B at point G
```

4.5.5 Use actuator in the pivot joint:

If the pivot joint is motorized with a motor applying a torque C_m around Z_p axis (i.e. a torque applied by the base **B** on the appendage **A**), the dynamic model of the appendage at point **P** becomes:

$$\begin{bmatrix} \vec{F}_p \\ \vec{T}_p \\ C_m \end{bmatrix} = M_p^A(s)(7\times 7) \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \\ \ddot{\theta} \end{bmatrix} \qquad (66)$$

Therefore, a new input $\ddot{\theta}$ and a new output C_m are introduced to the whole dynamic model.

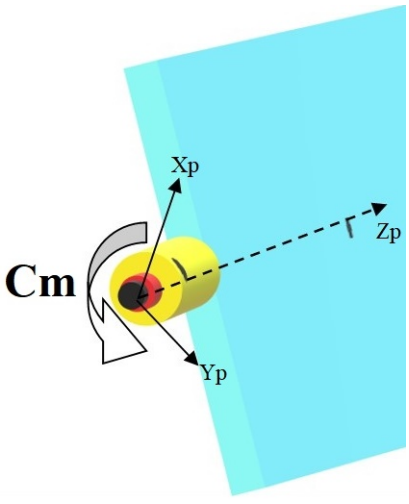


Figure 19: A schematic illustration of the inputs and outputs when pivot joints are added.

The objective is to compute the augmented direct model $M_G^P(s)(7 \times 7)$. Because of the revolute joint, the projection of the torque $\vec{T}_p$ exerted by the base on the appendage at point **G**,along (G_x,G_y,G_z) axis is either: null in case of a free revolute joint or equal to C_m in case of an actuated joint

$$C_m = \vec{T}_{G/A,P} * \vec{r}_{G_P} \tag{67}$$

where $\vec{r}_{G_P}$ is the distance from **p** to **G** . and **P** is the connection point between the mian body and the appendage
Expressing the direct dynamics model of the appendage at point **A** in frame R_p enables us to write that:

$$\begin{bmatrix} \vec{F}_{B/A} \\ \vec{T}_{B/A,P} \end{bmatrix} = M_p^A(s) \begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} + \ddot{\theta}_{RP} \end{bmatrix} \tag{68}$$

From (67) and (68), we can write the augmented direct model (7×7) of the appendage $[M_G^P(s)]_{R_G}$ at point **P** and expressed in frame R_G [7] :

$$\begin{bmatrix}\begin{bmatrix} \vec{F}_{B/A} \\ \vec{T}_{B/A,P} \end{bmatrix}\\ C_m\end{bmatrix} = \begin{bmatrix} I_6 \\ 0\ 0\ 0\ x_{r_{GP}}\ y_{r_{GP}}\ z_{r_{GP}} \end{bmatrix} [M_p^A]_{R_G} \begin{bmatrix} 0 \\ 0 \\ 0 \\ I_6 \quad x_{r_{GP}} \\ y_{r_{GP}} \\ z_{r_{GP}} \end{bmatrix} \begin{bmatrix}\begin{bmatrix} \vec{a}_p \\ \dot{\vec{\omega}} \end{bmatrix}\\ \ddot{\theta}\end{bmatrix} \tag{69}$$

The direct model :

$$[M_G^p(s)]_{R_G} = \begin{bmatrix} & I_6 & \\ 0\ 0\ 0 & x_{r_{GP}}\ y_{r_{GP}}\ z_{r_{GP}} & \end{bmatrix} [M_p^A]_{R_G} \begin{bmatrix} & 0 & \\ & 0 & \\ & 0 & \\ I_6 & x_{r_{GP}} & \\ & y_{r_{GP}} & \\ & z_{r_{GP}} & \end{bmatrix} \tag{70}$$

Now we can write :

$$\begin{bmatrix} \begin{bmatrix} \vec{F}_{B/A} \\ \vec{T}_{B/A,P} \end{bmatrix} \\ C_m \end{bmatrix} = [M_G^P(s)(7\times7)]_{RG} \begin{bmatrix} \begin{bmatrix} \vec{a}_p \\ \dot{\vec{\omega}} \end{bmatrix} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} P_p^A(s)_{11}(6\times6) & P_p^A(s)_{12}(6\times1) \\ P_p^A(s)_{21}(1\times6) & P_p^A(s)_{22}(1\times1) \end{bmatrix} \begin{bmatrix} \begin{bmatrix} \vec{a}_p \\ \dot{\vec{\omega}} \end{bmatrix} \\ \ddot{\theta} \end{bmatrix} \tag{71}$$

where:

$P_p^A(s)_{11}(6\times6)$=$[M_p^A]_{Ra}$: is the transfer function between $\begin{bmatrix} \vec{F}_{B/A} \\ \vec{T}_{B/A,P} \end{bmatrix}$ *and* $\begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix}$

$P_p^A(s)_{12}(6\times1)$: is the transfer function between $\begin{bmatrix} \vec{F}_{B/A} \\ \vec{T}_{B/A,P} \end{bmatrix}$ and $\ddot{\theta}$

$P_p^A(s)_{21}(1\times6)$: is the transfer function between C_m and $\begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix}$

$P_p^A(s)_{22}(1\times1)$: is the transfer function between C_m and $\ddot{\theta}$

The block diagram of Figure represents this operation. It also shows the connection of the first six inputs and outputs between $[M_p^A]_{R_P}$ and the hub's direct model $[D_G^B]$ in order to get the assembly model $[D_G^{stellite}]$, expressed in frame R_G . Taking into account a revolute joint between the hub and an appendage.

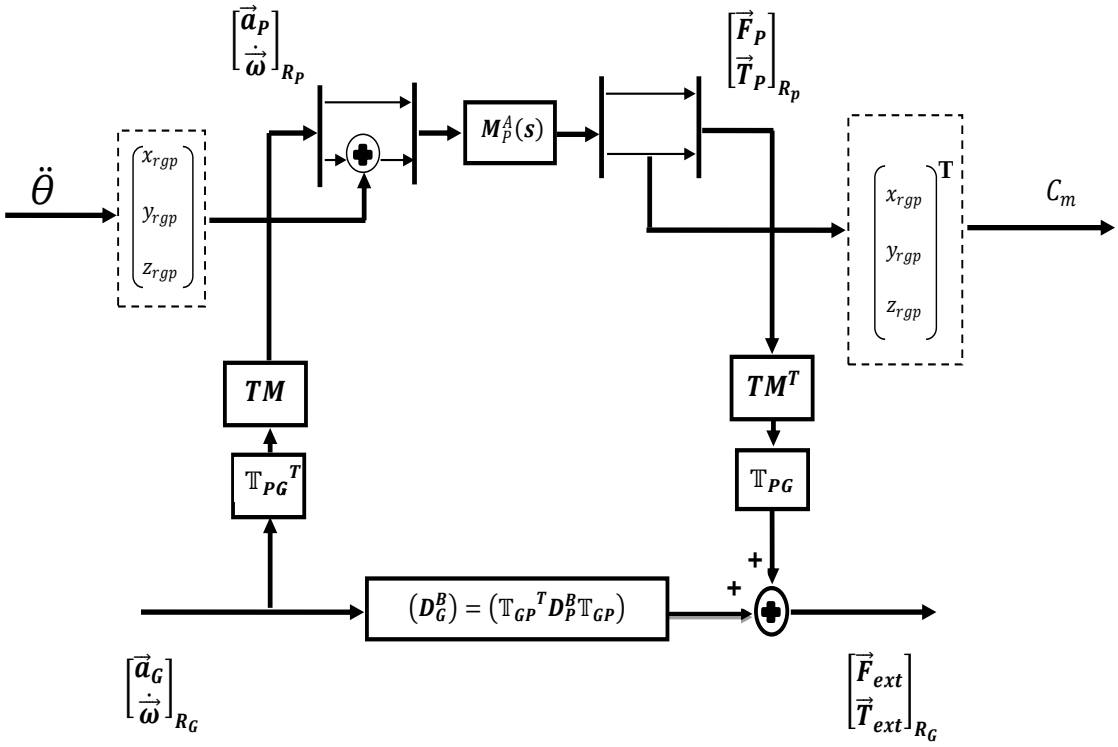


Figure 20: Direct dynamics model (7×7) block diagram of the assembly hub + appendage with revolute joint, expressed in frame R_G.

$r_{GP} = [x_{r_{GP}}, y_{r_{GP}}, z_{r_{GP}}]$ expresse $\ddot{\theta}$ in the frame R_p

using this block diagramme the simulink model described by the following figure

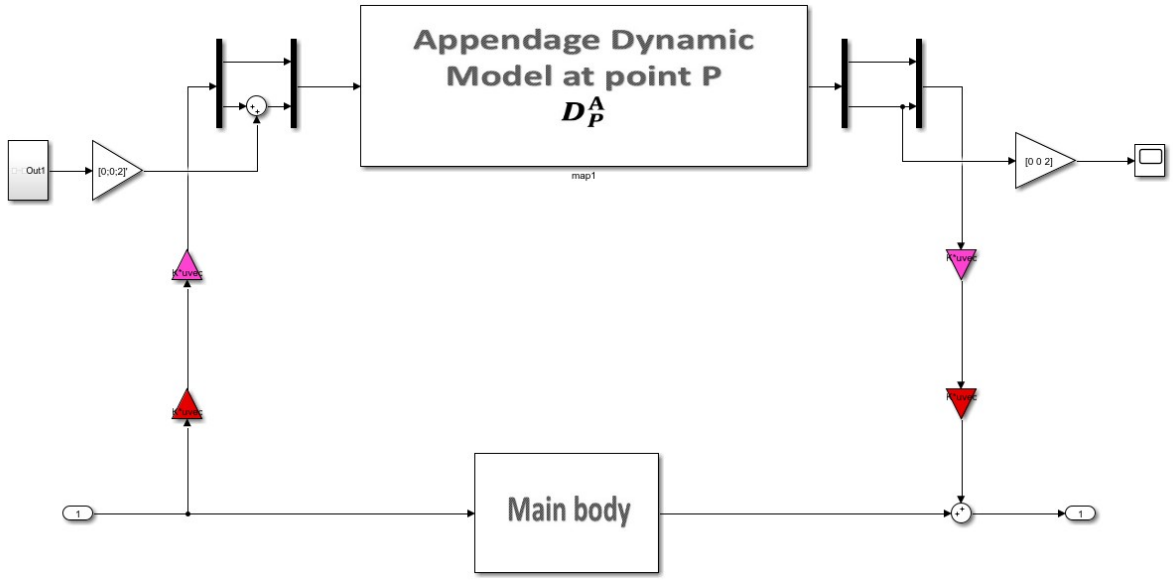


Figure 21: the augmented direct model(7×7)

4.5.6 Inverse dynamic model:

the inverse dynamic model is obtained by :

$$\begin{bmatrix} \vec{a}_P \\ \dot{\vec{\omega}} \end{bmatrix} = M_p^A(s)^{-1} \begin{bmatrix} \vec{F}_{B/A} \\ \vec{T}_{B/A,P} \end{bmatrix} \tag{72}$$

To fit into the block-diagram as many (rigid or flexible) appendages as possible through other feedbacks on $[(D_G^B)^{-1}]$,this model was developed in (see [3]) .

4.6 Control theory and application:

4.6.1 PID with ideal configuration:

An alternate version of the PID equation designed such that the gain (K_p) affects all three actions is called the Ideal or ISA equation:

$$u(t) = K_p(e(t) + \frac{1}{K_i}\int_0^t e(\tau)d\tau + K_d\frac{de(t)}{dt}) \tag{73}$$

Here, the gain constant (Kp) is distributed to all terms within the parentheses, equally affecting all three control actions. Increasing Kp in this style of PID controller makes the P, the I, and the D actions equally more aggressive. We may show this mathematically, by breaking the "ideal" equation up into three different parts, each one describing its contribution to the output (Δu):

$$\begin{cases} \Delta u = K_p\Delta_e & proportional\,action \\ \Delta u = \dfrac{K_p}{K_i}\Delta_e & integral\,action \\ \Delta u = K_pK_d\dfrac{de}{dt} & derivative\,action \end{cases} \tag{74}$$

the transfer function is given by :

$$F(s) = K_p[1 + K_i\frac{1}{s} + K_d\frac{Ns}{s+N}] \tag{75}$$

such us the integral action in s domain is realized by a filter $[\frac{Ns}{s+N}]$ and N is the filter coefficient the next figure will illustrated PID in ideal configuration

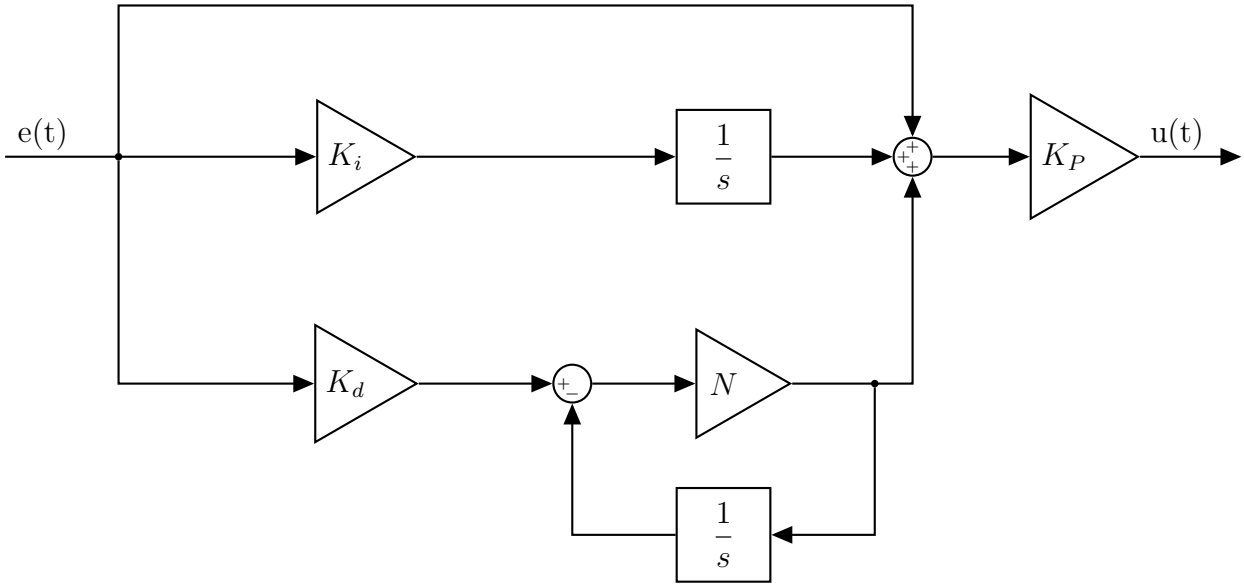


Figure 22: PID in ideal form

As you can see, all three portions of this PID equation are influenced by the gain (K_p) owing to algebraic distribution, but the integral and derivative tuning parameters (K_i and K_d) act independently within their own terms of the equation.

4.6.2 controler Tuning MIMO-PID using particale swarm optimization :

As we have defined The particle swarm optimization (PSO) algorithm, which updates particles by considering their past momentum and current direction, has demonstrated its power in several optimization applications[13]. However, the updating strategy followed by the standard PSO mainly aims to learn from the global optimum, which often leads to PSO suffering from premature convergence. Using the past momentum can result in the overshoot problem, which usually slows down convergence in complex optimization problems. Inspired by the massive success of the proportional-integral-derivative (PID) controller in automatic control,The proposed PSO utilizes the past, current, and change in global best together to update the search direction.a concrete design procedure to determine the design parameters (K_p, K_i, K_d, N) by solving the optimization problem,first we establish a connection between the PSO process and the PID controller-based control system[12].

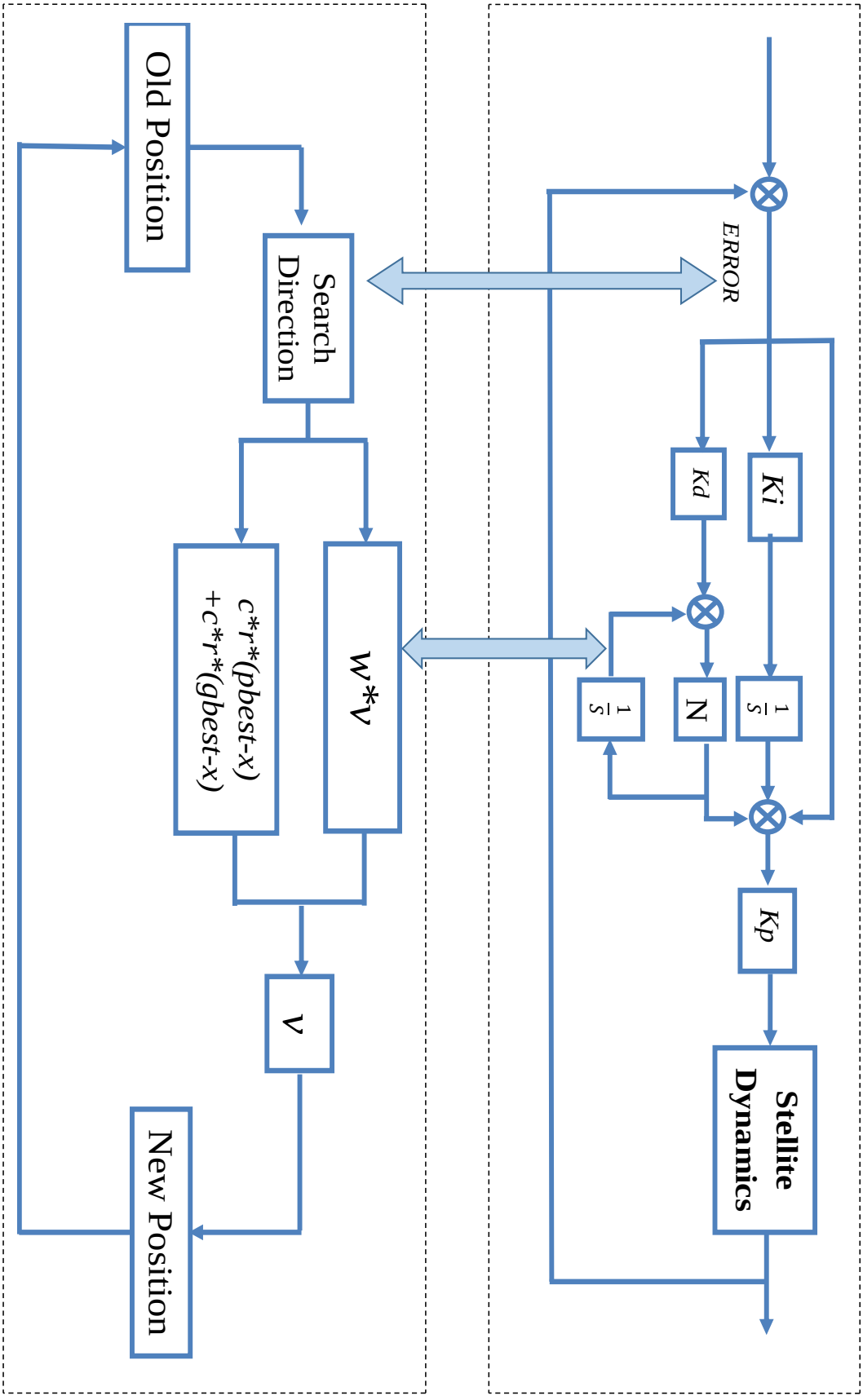


Figure 23: PID control system

the pso programme are addapted to optimise the PID gain after get the search limit (Stability area) usin matlab tuning toolbox , the most important here is to obtain a stable search area because if y are not measurable this will Break the process, the pso programme are given by:

```
%% *Particle Swarm Optimization*
Juml_Particles = 5 ;  % Particle Number
MaxIt = 10 ;  % Iteration
Juml_Variabel  = 24  ;  % The number of parameters to
   be optimized
c2 = 2  ;  % PSO parameter C1 (Social Constant)
c1 = 2  ;  % PSO parameter C2 (Cognitive Constant)
w  = 0.99;  % PSO momentum or inertia


%Calculate the fitness of each particle in each
   iteration
%=========================================================


%-----------------------------------------%
%   initialization parameters %
%-----------------------------------------%

for ir=1:Juml_Particles
for n=1:MaxIt
fitness=0*ones(ir,n); % Allocated Memory
end
for k=1:Juml_Variabel
R1=rand(k,ir); % R1 is Random Jum_Var and Jum_Particle
R2=rand(k,ir); % R2 is Random Jum_Var and Jum_Particle
end
end


%-----------------------------------------------------------%
% Swarm Initialization, Velocities, and Position
   Particles  %
%-----------------------------------------------------------%

```

```
% Search limit :
Rb(1) =0.1;          Ra(1) =0.9;
Rb(2) =0.1;          Ra(2) =0.9;
Rb(3) =0.1;          Ra(3) =0.9;
Rb(4) =0.1;          Ra(4) =0.9;
Rb(5) =0.1;          Ra(5) =0.9;
Rb(6) =0.1;          Ra(6) =0.9;

Rb(7) =0.1;          Ra(7) =0.9;
Rb(8) =0.1;          Ra(8) =0.9;
Rb(9) =0.1;          Ra(9) =0.9;
Rb(10) =0.1;         Ra(10) =0.9;
Rb(11) =0.1;         Ra(11) =0.9;
Rb(12) =0.1;         Ra(12) =0.9;

Rb(13) =0.1;         Ra(13) =0.9;
Rb(14) =0.1;         Ra(14) =0.9;
Rb(15) =0.1;         Ra(15) =0.9;
Rb(16) =0.1;         Ra(16) =0.9;
Rb(17) =0.1;         Ra(17) =1;
Rb(18) =0.1;         Ra(18) =0.9;

Rb(19) =0.10;        Ra(19) =0.9;
Rb(20) =0.1;         Ra(20) =0.9;
Rb(21) =0.1;         Ra(21) =0.9;
Rb(22) =0.1;         Ra(22) =0.9;
Rb(23) =0.1;         Ra(23) =1;
Rb(24) =0.1;         Ra(24) =0.9;

for ir=1:Juml_Particles
posisi_particle1=Rb(1)+(Ra(1)-Rb(1))*0.1*rand(1,ir); %
   Starting position(Xij)
posisi_particle2=Rb(2)+(Ra(2)-Rb(2))*0.1*rand(1,ir);
posisi_particle3=Rb(3)+(Ra(3)-Rb(3))*0.1*rand(1,ir);
posisi_particle4=Rb(4)+(Ra(4)-Rb(4))*0.1*rand(1,ir);
posisi_particle5=Rb(5)+(Ra(5)-Rb(5))*0.1*rand(1,ir);
posisi_particle6=Rb(6)+(Ra(6)-Rb(6))*0.1*rand(1,ir);

posisi_particle7=Rb(7)+(Ra(7)-Rb(7))*0.1*rand(1,ir);
```

```
71 posisi_particle8=Rb(8)+(Ra(8)-Rb(8))*0.1*rand(1,ir);
72 posisi_particle9=Rb(9)+(Ra(9)-Rb(9))*0.1*rand(1,ir);
73 posisi_particle10=Rb(10)+(Ra(10)-Rb(10))*0.1*rand(1,ir)
   ;
74 posisi_particle11=Rb(11)+(Ra(11)-Rb(11))*0.1*rand(1,ir)
   ;
75 posisi_particle12=Rb(12)+(Ra(12)-Rb(12))*0.1*rand(1,ir)
   ;
76
77 posisi_particle13=Rb(13)+(Ra(13)-Rb(13))*0.1*rand(1,ir)
   ; % Starting position(Xij)
78 posisi_particle14=Rb(14)+(Ra(14)-Rb(14))*0.1*rand(1,ir)
   ;
79 posisi_particle15=Rb(15)+(Ra(15)-Rb(15))*0.1*rand(1,ir)
   ;
80 posisi_particle16=Rb(16)+(Ra(16)-Rb(16))*0.1*rand(1,ir)
   ;
81 posisi_particle17=Rb(17)+(Ra(17)-Rb(17))*0.1*rand(1,ir)
   ;
82 posisi_particle18=Rb(18)+(Ra(18)-Rb(18))*0.1*rand(1,ir)
   ;
83
84 posisi_particle19=Rb(19)+(Ra(19)-Rb(19))*0.1*rand(1,ir)
   ; % Starting position(Xij)
85 posisi_particle20=Rb(20)+(Ra(20)-Rb(20))*0.1*rand(1,ir)
   ;
86 posisi_particle21=Rb(21)+(Ra(21)-Rb(21))*0.1*rand(1,ir)
   ;
87 posisi_particle22=Rb(22)+(Ra(22)-Rb(22))*0.1*rand(1,ir)
   ;
88 posisi_particle23=Rb(23)+(Ra(23)-Rb(23))*0.1*rand(1,ir)
   ;
89 posisi_particle24=Rb(24)+(Ra(24)-Rb(24))*0.1*rand(1,ir)
   ;
90
91 posisi_particle=[posisi_particle1;posisi_particle2;
   posisi_particle3;
92 posisi_particle4;posisi_particle5;posisi_particle6;
93 posisi_particle7;posisi_particle8; posisi_particle9;
```

```
posisi_particle10;posisi_particle11;posisi_particle12;
posisi_particle13;posisi_particle14; posisi_particle15;
posisi_particle16;posisi_particle17;posisi_particle18;
posisi_particle19;posisi_particle20; posisi_particle21;
posisi_particle22;posisi_particle23;posisi_particle24];
end

% v_ij=v_min+(v_max-v_min)*rand(.)----> harusnya
v_max1=0.1*(Ra(1)-Rb(1));
v_max2=0.1*(Ra(2)-Rb(2));
v_max3=0.1*(Ra(3)-Rb(3));
v_max4=0.1*(Ra(4)-Rb(4));
v_max5=0.1*(Ra(5)-Rb(5));
v_max6=0.1*(Ra(6)-Rb(6));

v_max7=0.1*(Ra(1)-Rb(1));
v_max8=0.1*(Ra(2)-Rb(2));
v_max9=0.1*(Ra(3)-Rb(3));
v_max10=0.1*(Ra(4)-Rb(4));
v_max11=0.1*(Ra(5)-Rb(5));
v_max12=0.1*(Ra(6)-Rb(6));

v_max13=0.1*(Ra(1)-Rb(1));
v_max14=0.1*(Ra(2)-Rb(2));
v_max15=0.1*(Ra(3)-Rb(3));
v_max16=0.1*(Ra(4)-Rb(4));
v_max17=0.1*(Ra(5)-Rb(5));
v_max18=0.1*(Ra(6)-Rb(6));

v_max19=0.1*(Ra(1)-Rb(1));
v_max20=0.1*(Ra(2)-Rb(2));
v_max21=0.1*(Ra(3)-Rb(3));
v_max22=0.1*(Ra(4)-Rb(4));
v_max23=0.1*(Ra(5)-Rb(5));
v_max24=0.1*(Ra(6)-Rb(6));

v_min=0;

for ir=1:Juml_Particles
```

```
133 kec_particle1=(v_max1-v_min).*rand(1,ir)+v_min;   % starting Velocity (Vij)
134 kec_particle2=(v_max2-v_min).*rand(1,ir)+v_min;
135 kec_particle3=(v_max3-v_min).*rand(1,ir)+v_min;
136 kec_particle4=(v_max4-v_min).*rand(1,ir)+v_min;
137 kec_particle5=(v_max5-v_min).*rand(1,ir)+v_min;
138 kec_particle6=(v_max6-v_min).*rand(1,ir)+v_min;
139
140 kec_particle7=(v_max7-v_min).*rand(1,ir)+v_min;
141 kec_particle8=(v_max8-v_min).*rand(1,ir)+v_min;
142 kec_particle9=(v_max9-v_min).*rand(1,ir)+v_min;
143 kec_particle10=(v_max10-v_min).*rand(1,ir)+v_min;
144 kec_particle11=(v_max11-v_min).*rand(1,ir)+v_min;
145 kec_particle12=(v_max12-v_min).*rand(1,ir)+v_min;
146
147 kec_particle13=(v_max13-v_min).*rand(1,ir)+v_min;
148 kec_particle14=(v_max14-v_min).*rand(1,ir)+v_min;
149 kec_particle15=(v_max15-v_min).*rand(1,ir)+v_min;
150 kec_particle16=(v_max16-v_min).*rand(1,ir)+v_min;
151 kec_particle17=(v_max17-v_min).*rand(1,ir)+v_min;
152 kec_particle18=(v_max18-v_min).*rand(1,ir)+v_min;
153
154 kec_particle19=(v_max19-v_min).*rand(1,ir)+v_min;
155 kec_particle20=(v_max20-v_min).*rand(1,ir)+v_min;
156 kec_particle21=(v_max21-v_min).*rand(1,ir)+v_min;
157 kec_particle22=(v_max22-v_min).*rand(1,ir)+v_min;
158 kec_particle23=(v_max23-v_min).*rand(1,ir)+v_min;
159 kec_particle24=(v_max24-v_min).*rand(1,ir)+v_min;
160
161 kec_particle=[ kec_particle1;kec_particle2;
    kec_particle3;
162 kec_particle4;kec_particle5;kec_particle6;
163 kec_particle7;kec_particle8;kec_particle9;
164 kec_particle10;kec_particle11;kec_particle12;
165 kec_particle13;kec_particle12;kec_particle15;
166 kec_particle16;kec_particle17;kec_particle18;
167 kec_particle19;kec_particle20;kec_particle21;
168 kec_particle22;kec_particle23;kec_particle24];
169 end
```

```
170
171 posisi_terbaik_lokal=posisi_particle; % p_best=x_ij
    Local Best Position (Pbest)
172
173 %-------------------------------------------------------%
174 % Evaluation of Population Initialization  %
175 %-------------------------------------------------------%
176
177 % Evaluate the fitness of each particle
178
179 for ir=1:Juml_Particles
180 Kp1=posisi_particle(1,ir);
181 Kp2=posisi_particle(2,ir);
182 Kp3=posisi_particle(3,ir);
183 Kp4=posisi_particle(4,ir);
184 Kp5=posisi_particle(5,ir);
185 Kp6=posisi_particle(6,ir);
186
187 Ki1=posisi_particle(7,ir);
188 Ki2=posisi_particle(8,ir);
189 Ki3=posisi_particle(9,ir);
190 Ki4=posisi_particle(10,ir);
191 Ki5=posisi_particle(11,ir);
192 Ki6=posisi_particle(12,ir);
193
194 Kd1=posisi_particle(13,ir);
195 Kd2=posisi_particle(14,ir);
196 Kd3=posisi_particle(15,ir);
197 Kd4=posisi_particle(16,ir);
198 Kd5=posisi_particle(17,ir);
199 Kd6=posisi_particle(18,ir);
200
201 N1=posisi_particle(19,ir);
202 N2=posisi_particle(20,ir);
203 N3=posisi_particle(21,ir);
204 N4=posisi_particle(22,ir);
205 N5=posisi_particle(23,ir);
206 N6=posisi_particle(24,ir);
207
```

```
208 % Calculate fitness for each particle
209 sim('oneapp')
210 t=time;
211 y=out;
212 for i=1:18
213 error(i)=abs(y(i)^2)*t(i);
214 end
215 ITAE=sum(error);
216 fitness_particle(ir)=ITAE;
217 end
218 % Looking for the best
219 fitness_terbaik_lokal=fitness_particle;
220 [fitness_terbaik_global,indeks]=min(
       fitness_terbaik_lokal); % Look for the Minimum Value
221
222 for ir=1:Juml_Particles
223 posisi_terbaik_global(:,ir)=posisi_terbaik_lokal(:,
       indeks);
224 end
225
226 %Velocity Particle Update
227 kec_particle=w *kec_particle+c1*(R1.*(
       posisi_terbaik_lokal-posisi_particle))+c2*(R2.*(
       posisi_terbaik_global-posisi_particle));
228 %Particle Position Update
229 posisi_particle = posisi_particle+kec_particle;
230
231 % Looping PSO
232 hfig = figure;
233 hold on
234 title('Convergence of PSO Algorithm Graphic');
235 set(hfig, 'position', [50,40,600,300]);
236 set(hfig, 'DoubleBuffer', 'on');
237 hbestplot = plot(1:MaxIt,zeros(1,MaxIt),'-');
238 xlabel('Iteration');
239 ylabel('Fitness Function');
240 hold off
241 drawnow;
242
```

```
243
244 It=1;
245 while It<=MaxIt
246 %       Iteration=It
247 for ir=1:Juml_Particles
248 Kp=posisi_particle(1,ir);
249 Kp1=posisi_particle(1,ir);
250 Kp2=posisi_particle(2,ir);
251 Kp3=posisi_particle(3,ir);
252 Kp4=posisi_particle(4,ir);
253 Kp5=posisi_particle(5,ir);
254 Kp6=posisi_particle(6,ir);
255
256 Ki1=posisi_particle(7,ir);
257 Ki2=posisi_particle(8,ir);
258 Ki3=posisi_particle(9,ir);
259 Ki4=posisi_particle(10,ir);
260 Ki5=posisi_particle(11,ir);
261 Ki6=posisi_particle(12,ir);
262
263 Kd1=posisi_particle(13,ir);
264 Kd2=posisi_particle(14,ir);
265 Kd3=posisi_particle(15,ir);
266 Kd4=posisi_particle(16,ir);
267 Kd5=posisi_particle(17,ir);
268 Kd6=posisi_particle(18,ir);
269
270 N1=posisi_particle(19,ir);
271 N2=posisi_particle(20,ir);
272 N3=posisi_particle(21,ir);
273 N4=posisi_particle(22,ir);
274 N5=posisi_particle(23,ir);
275 N6=posisi_particle(24,ir);
276
277 sim('oneapp')
278 t=time;
279 y=out;
280 for i=1:18
281 error(i)=abs(y(i)^2)*t(i);
```

```
end
ITAE=sum(error);
fitness_particle(ir)=ITAE;
end

% (e) Compare particle's fitness evaluation with particle's p_best
%     If current value is better than p_best, then set p_best value
%     equal to the current value and the p_best location equal to
%     the current location in j-dimensional space

% Lokal
for ir=1:Juml_Particles
if fitness_particle(ir)< fitness_terbaik_lokal(ir)
fitness_terbaik_lokal(ir)=fitness_particle(ir);
posisi_terbaik_lokal(:,ir)=posisi_particle(:,ir);
end
end

% Global
[fitness_terbaik_global_particle(It),indeks] = min(fitness_terbaik_lokal);
% (f) Compare fitness evaluation with the population's overall
%     previous best.It the current value is better than g_best
%     , then reset g_best to the current particle's array indeks and value
if fitness_terbaik_global_particle < fitness_terbaik_global
fitness_terbaik_global = fitness_terbaik_global_particle;

for ir=1:Juml_Particles
posisi_terbaik_global(:,ir) = posisi_terbaik_lokal(:,indeks);
end
```

```
311 end
312
313 % Update Velocity
314 kec_particle=w *kec_particle+c1*(R1.*(
        posisi_terbaik_lokal-posisi_particle))+c2*(R2.*(
        posisi_terbaik_global-posisi_particle));
315
316 % (h) Update Position Particle
317 posisi_particle = posisi_particle+kec_particle;
318
319 fprintf(1,'Iteration: %d,  Fitness: %f\n',It,
        fitness_terbaik_global_particle(It))
320 plotvector=get(hbestplot,'Ydata');
321 plotvector(It)=fitness_terbaik_global_particle(It);
322 set(hbestplot,'Ydata',plotvector);
323 drawnow
324 It=It+1;
325 %       fitness_global=fitness_terbaik_global_particle(It
        )
326 end
```

4.6.3 Simulation and results:

first the used simulation procedures were:
Define the initial position $(a_x, a_y, a_z$ and $\omega_x, \omega_y, \omega_z, \ddot{\theta})$=0. Define the referance value $(a_x, a_y, a_z, \ddot{\theta})$=0.5 (m/s^2) and $(\omega_x, \omega_y, \omega_z)$=0.1 (rad/s^2) and define the rotation angel of the appendage θ=10°≈0.1745 rad use filter to the input step signal see (3.3.3)

Simulation result for the main body with flexible solar array:

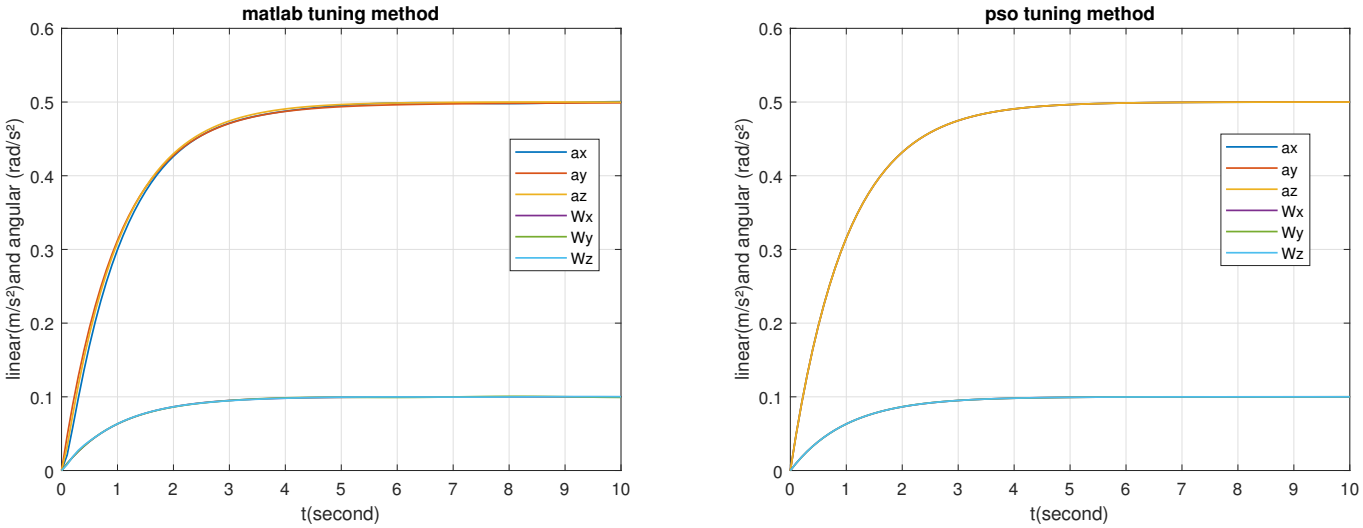


Figure 24: time response of the 6×6 PID controled system

PID gain	K_p	K_i	K_d	N
matlab toolboox	$\begin{bmatrix}-0.002\\-0.043\\-0.005\\-0.003\\-1.895\\0.001\end{bmatrix}$	$\begin{bmatrix}0.011\\0.016\\0.009\\0.021\\0\\0.003\end{bmatrix}$	$\begin{bmatrix}0.146\\0.260\\0.010\\0.007\\7.070\\0.002\end{bmatrix}$	$\begin{bmatrix}0.014\\0.167\\0.510\\5.494\\0\\0.053\end{bmatrix}$

	K_p	K_i	K_d	N
pso	$\begin{bmatrix}0.172\\0.155\\0.127\\0.137\\0.150\\0.170\end{bmatrix}$	$\begin{bmatrix}0.147\\0.128\\0.158\\0.101\\0.154\\0.171\end{bmatrix}$	$\begin{bmatrix}0.131\\0.177\\0.133\\0.108\\0.137\\0.173\end{bmatrix}$	$\begin{bmatrix}0.118\\0.103\\0.131\\0.166\\0.106\\0.106\end{bmatrix}$

	Risetime	SettlingTime	SettlingMin	SettlingMax	Overshoot	Undershoot	Peak	Peaktime
			pso	tuning	method			
a_x	2,204	3,929	0,454	0,499	0	0	0,499	10
a_y	2,205	3,928	0,454	0,499	0	0	0,499	10
a_z	2,203	3,921	0,454	0,499	0	0	0,499	10
ω_x	2,200	3,919	0,090	0,099	0,0002	0	0,099	9,800
ω_y	2,193	3,970	0,0900	0,100	0	0	0,100	10
ω_z	2,195	3,913	0,090	0,099	0	0	0,099	10
			tuning	matlab	toolbox			
a_x	2,230	4,308	0,454	0,500	0	0	0,500	10
a_y	2,267	4,184	0,450	0,498	0	0	0,498	10
a_z	2,195	3,925	0,453	0,500	0,025	0	0,500	8,500
ω_x	2,196	3,910	0,090	0,099	0	0	0,099	10
ω_y	2,159	3,515	0,089	0,100	1,406	0	0,100	8,100
ω_z	2,214	4,020	0,090	0,100	0,138	0	0,100	9,400

Simulation result for the main body with two flexible solar array:

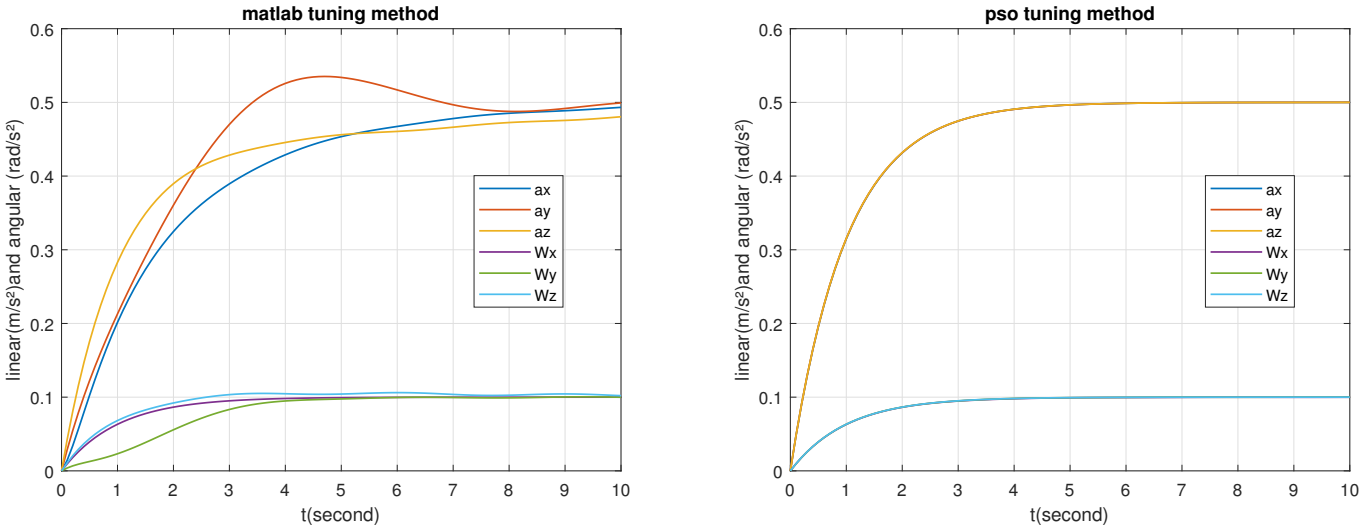


Figure 25: time response of the 6×6 PID controled system

PID gain	K_p	K_i	K_d	N
matlab toolboox	$\begin{bmatrix} 0.001 \\ -0.002 \\ -0.001 \\ 885.13 \\ 0 \\ 0.001 \end{bmatrix}$	$\begin{bmatrix} 0.710 \\ -0.786 \\ -2.079 \\ 0.488 \\ -4.216 \\ 0.136 \end{bmatrix}$	$\begin{bmatrix} -0.310 \\ -2.312 \\ -80.619 \\ -0.009 \\ -6.563 \\ -0.343 \end{bmatrix}$	$\begin{bmatrix} 2.611 \\ 0.565 \\ 0.159 \\ 97.292 \\ 4.167 \\ 2.593 \end{bmatrix}$
pso	$\begin{bmatrix} 0.039 \\ 0.129 \\ 0.104 \\ 0.120 \\ 0.147 \\ 0.170 \end{bmatrix}$	$\begin{bmatrix} 0.078 \\ 0.099 \\ 0.143 \\ -0.002 \\ 0.093 \\ 0.069 \end{bmatrix}$	$\begin{bmatrix} 0.131 \\ 0.100 \\ 0.088 \\ 0.160 \\ -0.043 \\ -0.047 \end{bmatrix}$	$\begin{bmatrix} 0.134 \\ 0.188 \\ 0.090 \\ 0.126 \\ 0.042 \\ -0.013 \end{bmatrix}$

	Risetime	SettlingTime	SettlingMin	SettlingMax	Overshoot	Undershoot	Peak	Peaktime
			pso	tuning	method			
a_x	2,209	3,927	0,454	0,499	0	0	0,499	10
a_y	2,207	3,938	0,454	0,499	0	0	0,499	10
a_z	2,203	3,922	0,454	0,500	0	0	0,500	10
ω_x	2,206	4,006	0,090	0,099	0,024	0	0,099	8
ω_y	2,197	3,889	0,090	0,100	0,060	0	0,100	9,400
ω_z	2,192	3,896	0,090	0,099	0,005	0	0,099	9,400
			tuning	matlab	toolbox			
a_x	4,274	7,655	0,445	0,493	0	0	0,493	10
a_y	2,589	8,642	0,452	0,535	7,188	0	0,535	4,700
a_z	3,076	7,659	0,432	0,480	0	0	0,480	10
ω_x	2,196	3,910	0,0909	0,099	0	0	0,099	10
ω_y	3,067	5,167	0,090	0,100	0,512	0	0,100	9,300
ω_z	1,888	9,300	0,092	0,106	4,025	0	0,106	6

Simulation result for the main body with actuated flexible solar array:

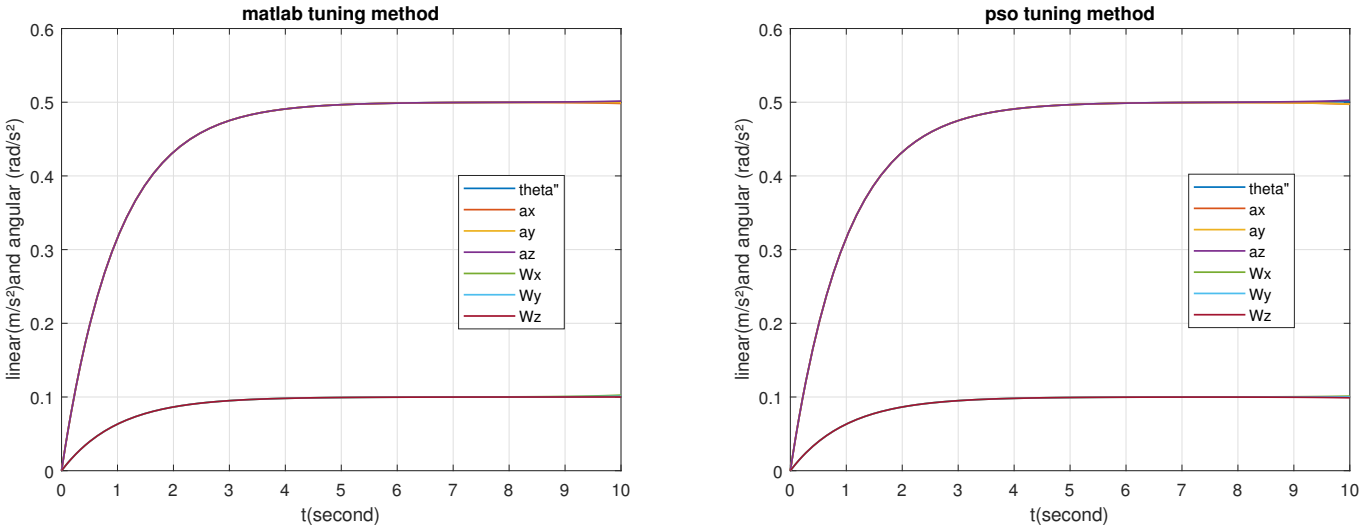


Figure 26: time response of the 7×7 PID controled system

PIDgain	K_p	K_i	K_d	N
matlab toolboox	$\begin{bmatrix} 123.254 \\ 65.297 \\ 99.574 \\ 57.556 \\ 27.566 \\ 89.628 \\ 83.190 \end{bmatrix}$	$\begin{bmatrix} 2.345 \\ -0.855 \\ 11.330 \\ 9.693 \\ 16.189 \\ 29.264 \\ 7.559 \end{bmatrix}$	$\begin{bmatrix} 153.254 \\ 99.258 \\ 246.324 \\ 46.984 \\ 18.883 \\ 29.377 \\ 0.741 \end{bmatrix}$	$\begin{bmatrix} 9.494 \\ -2.164 \\ 14.106 \\ 5.872 \\ 8.383 \\ 12.183 \\ 65.602 \end{bmatrix}$

	K_p	K_i	K_d	N
pso	$\begin{bmatrix} 116.70 \\ 17.596 \\ 11.434 \\ 53.015 \\ 1.309 \\ -66.330 \\ 83.190 \end{bmatrix}$	$\begin{bmatrix} 0.642 \\ 0.842 \\ 19.301 \\ 0.022 \\ 7.681 \\ -0.592 \\ 7.559 \end{bmatrix}$	$\begin{bmatrix} 2.212 \\ 80.418 \\ 54.358 \\ 0.059 \\ 100 \\ -0.047 \\ 0.741 \end{bmatrix}$	$\begin{bmatrix} 51.623 \\ 13.00 \\ 2.215 \\ 5.073 \\ 7.931 \\ 11.248 \\ 65.602 \end{bmatrix}$

	Risetime	SettlingTime	SettlingMin	SettlingMax	Overshoot	Undershoot	Peak	Peaktime
			pso	tuning	method			
a_x	2,213	3,990	0,454	0,500	0	0	0,500	10
a_y	2,149	3,672	0,454	0,499	0,488	0	0,499	7,69
a_z	2,153	3,689	0,454	0,499	0,445	0	0,499	7,895
ω_x	2,247	4,217	0,454	0,502	0	0	0,502	10
ω_y	2,313	4,820	0,092	0,101	0	0	0,101	10
ω_z	2,212	3,987	0,090	0,100	0	0	0,100	10
$\ddot{\theta}$	2,107	3,488	0,089	0,099	0,994	0	0,099	7,495
			tuning	matlab	toolbox			
a_x	2,256	4,268	0,457	0,503	0	0	0,503	10
a_y	2,169	3,759	0,449	0,499	0,277	0	0,499	7,902
a_z	2,186	3,860	0,449	0,499	0,079	0	0,499	8,502
ω_x	2,226	4,073	0,457	0,501	0	0	0,501	10
ω_y	2,460	8,740	0,092	0,102	0	0	0,102	10
ω_z	2,291	4,558	0,091	0,100	0	0	0,100	10
$\ddot{\theta}$	2,198	3,920	0,091	0,100	0	0	0,100	10

4.6.3.1 Discussion Of The Resulting:

As can be seen in the previos figures of the three controlled model the two used methodes shows great performance in reference tracing as a result of good choice of research domain for pso algorithm and tuning toolbox parameters,But in gain value the pso method give the optimal one,At the same time,the augmentation of system complexity cause the expansion in gain value as result of increment of the flexibility affect,for this reason the model with actuator in the pivot joint simulate the real model if we use the PID designed for the first and the second model this last will be unstable

4.6.4 Conclusion:

to fit into the block-diagram as many (rigid or flexible) appendages as possible through other forwards on (D_G^B) to minimize the number of occurrences of each physical parameters of each body, to directly access to them and finally to take easily into account their uncertainties. Finally, a last table lists the major parameters of the pointing control system where the given parameters are used as nominal scenario (look at Nicolas Guy et al. 2014, Alazard et al. 2013 - Murali, H. et al. 2015)

main body data	
m : hub mass(kg)	2000
$\mathbb{J}_G^B$: moment of inertia of the body with respect to G w.r.t in R_G(kg.m2).	$\begin{bmatrix} 2000 & 100 & 50 \\ 100 & 8000 & 80 \\ 50 & 80 & 8000 \end{bmatrix}$
$\vec{OG}$: distance from one hub nook to G w.r.t in R_i (m).	$[2\ \ 2\ \ 2]^T$
$\vec{r}_{GP}$: distance from center of hub G to point P w.r.t in R_G(m).	$[0\ \ 0\ \ 2]^T$
r : hub radius (m).	4
ω_0: angular velocity of satellite in its orbit (m /s).	7657 m/s or 27565.2 km/h
R : altitude of the satellite in its orbit (km).	420
Flexible appendage data	
m_A : solar array masses (kg).	100
$\mathbb{J}_A^C$:moment of inertia of appendage with respect to C w.r.t in R_P (kg.m2).	$\begin{bmatrix} 7000 & 0 & 0 \\ 0 & 2000 & 0 \\ 0 & 0 & 10000 \end{bmatrix}$
$\vec{r}_{CP1}, \vec{r}_{CP2}$: distance from point P1 & P2 to center of appendage C w.r.t in RP (m).	$[0\ \ 0\ \ 8]^T$
ω_i: angular frequencies of the k =4 flexible modes.	$2\pi[0.04\ \ 0.111\ \ 0.13\ \ 0.27]$
ξ_i : damping ratio of the flexible mode.	0.001
L_P : Modal Participation.	$\begin{bmatrix} 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 4 \\ 90 & 0 & 0 & 62 \\ 0 & 14 & 0 & 0 \\ 0 & 0 & 119 & 0 \end{bmatrix}^T$

5 General conclusion:

In this work , the linear dynamic model of a spacecraft composed of a rigid base and various flexible appendages connected to the base has been developed with two approaches. The possibility of improving the pointing accuracy of a flexible satellite, by utilizing direct state-space methods for the controller of simple design, which should improve the pointing accuracy of flexible satellites–even those having a complex structure. but there is limitation in modelisation because we take two flexible mode (two point masses) one in each appendage, to approach to the reality we should take more modes this cause complex model to deal with this approach is more applicable for microsatellite where there is small effect of flexibility and for the free hybrid approach has two main contributions useful for satellite advanced design phase.First,it introduces a generic modeling approach for satellites with flexible appendages.This approach allows to take easily into account uncertainties on physical parameters and ensures to obtain a minimal realization of the overall system. Secondly ,when tilted flexible appendages are considered, And is built by performing basic operations which is Transportation of a dynamic model from one point to another ,Connection of two dynamic models ,Utilization of effective masses to handle dynamic mass matrix for the flexible appendages ,Subdivision of the dynamic mass matrix to take into account a pivot joint ,All these operations can be simply represented by a block diagram and can be performed recursively to model any kind of open mechanical chain.

in large spacecraft this approach is a powerful tool in the same time effortless in addition of appendages (simple numerical application see)and can accept N flexible mode

References

[1] Bekhiti, Belkacem. (2018). Automatic Control of Aircraft and Missiles-Part I: Dynamic Modeling.

[2] Di Lorenzo, R., Santinelli, A., & Sciacovelli, D. (1975). On some attitude control syntheses for satellites with flexible appendages. Automatica, 11(2), 161-170.

[3] Tantawi, K. H. (2007). Linear dynamic modeling of satellites with various flexible appendages (Doctoral dissertation, Master Thesis, Ecole Nationale Supérieure de l'Aéronautique et de l'Espace-SupAero).

[4] Bekhiti, Belkacem. (2018). Multivariable Control System Design using the Theory of Matrix Polynomials. 10.13140/RG.2.2.12184.29441.E.

[5] Dr. Slami SAADI," Introduction à l'Optimisation Méta-heuristique : Cours, Problèmes Résolus & Recueil de Travaux Pratiques", édition OPU, 2017.

[6] Guy, N. (2013). Modèle et commande structurés: application aux grandes structures spatiales flexibles (Doctoral dissertation).

[7] Alazard, D., & Cumer, C. (2014). Satellite dynamics toolbox: Principles, user guide and tutorials. URL http://personnel. isae. fr/daniel-alazard/matlab-packages/satellite-dynamics-toolbox. html.

[8] Mohsenipour, R., Nemati, H., Nasirian, M., & Nia, A. K. (2013). Attitude control of a flexible satellite by using robust control design methods. Intelligent Control and Automation, 4(3), 313-326.

[9] Springer International Publishing Switzerland 2016,L. Mazzini, Flexible Spacecraft Dynamics, Control and Guidance,Springer Aerospace Technology, DOI 10.1007/978-3-319-25540-8-9

[10] Tao, J., Zhang, T., & Nie, Y. (2018). Attitude Maneuvering and Vibration Reducing Control of Flexible Spacecraft Subject to Actuator Saturation and Misalignment. Shock and Vibration, 2018.

[11] Theoretical computer science, 344(2-3), 243-278. Li, J., Xia, Y., Li, B., & Zeng, Z. (2020). A Pseudo-dynamic Search Ant Colony Optimization Algorithm with Improved Negative Feedback Mechanism. Cognitive Systems Research.

[12] Xiang, Z., Ji, D., Zhang, H., Wu, H., & Li, Y. (2019). A simple PID-based strategy for particle swarm optimization algorithm. Information Sciences, 502, 558-574.

[13] Kim, T. H., Maruta, I., & Sugie, T. (2008). Robust PID controller tuning based on the constrained particle swarm optimization. Automatica, 44(4), 1104-1110.

[14] Naka. S, Genji. T., Yura. T., Fukuyama. Y., " A Hybrid Particle Swarm Optimization for Distribution State Estimation", IEEE Transactions on power Systems, VOL. 18, NO. 1, pp 60-68. February 2003.

[15] Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey.

[16] Floudas, C.A., Pardolos, P.M.: Encyclopedia of Optimization, 2nd edn. Springer, Heidelberg (2009)

[17] Dorigo, M.: Optimization, Learning and Natural Algorithms. PhD thesis. Politecnico di Milano, Italy (1992)

[18] Talbi, E.G.: Metaheuristics: From Design to Implementation. Wiley, Chichester (2009)

[19] Åström, K. J. (2002). Control system design lecture notes for me 155a. Department of Mechanical and Environmental Engineering University of California Santa Barbara, 333.

[20] Mudry, F. (2002). Ajustage des paramètres d'un régulateur PID. Ecole d'ingénieurs du Canton de Vaud-Département d'électricité et informatique.

[21] Bassi, S. J., Mishra, M. K., & Omizegba, E. E. (2011). Automatic tuning of proportional-integral-derivative (PID) controller using particle swarm optimization (PSO) algorithm. International Journal of Artificial Intelligence & Applications, 2(4), 25.

[22] Hu, K., Vlahopoulos, N., & Mourelatos, Z. P. (2002). A finite element formulation for coupling rigid and flexible body dynamics of rotating beams. Journal of Sound and vibration, 253(3), 603-630.

[23] M.D. Shuster, S.D. Oh, Three-axis attitude determination from vector observations. J. Guidance Control 4(1), 70–77 (1981)

[24] 2. M.L. Psiaki, Extended quest attitude determination filtering. In Flight Mechanics Symposium, NASA Goddard Space Flight Center (1999)

[25] R. B. Noll, J. Zvava and J. J. Deyst, "Effect of Structural Flexibility on Spacecraft Control Systems," Journal of Spacecraft and Rockets, 1969.

[26] T. siamak, "On Attitude Recovery of Spacecraft Using Nonlinear Control," Ph.D. Thesis, Concordia University, Montreal, 2005.

[27] O. Z. Hayrani, "Dynamics and Control in Modal-Space of Flexible Spacecraft," Ph.D. Thesis, Virginia Polytechnic Institute and State University, Blacksburg, 1979.

[28] Yang, X. S. (2011, May). Metaheuristic optimization: algorithm analysis and open problems. In International Symposium on Experimental Algorithms (pp. 21-32). Springer, Berlin, Heidelberg.

[29] Floudas, C.A., Pardalos, P.M.: A Collection of Test Problems for Constrained Global Optimization Algorithms. LNCS, vol. 455. Springer, Heidelberg (1990)

[30] D. C. Hyland, J. L. Junkins and R. W. Longman, "Active Control Technology for Large Space Structures," Journal of Guidance, Control and Dynamics, Vol. 16, No. 5, 1993, pp. 801-821. doi:10.2514/3.21087

[31] O. Morgul, "Control and Stabilization of a Flexible Beam Attached to a Rigid Body," International Journal of Control, Vol. 51, No. 1, 1990, pp. 11-31. doi:10.1080/00207179008934048

[32] Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1995)

[33] Bai, Q. (2010). Analysis of particle swarm optimization algorithm. Computer and information science, 3(1), 180.

[34] Kirkpatrick, S., Gellat, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220, 670–680 (1983)

[35] A. M. Bloch, P. S. Krishnaprasad J. E. Marsden and G. Sánchez de Alvare, Stabilization of rigid body dynamics by internal and external torques, Automatica 28 (1992), 745-756.

[36] S. Gaulocher, C. Pittet and J. Chretien, Six- DOF Formation Flying Modeling and Control with an Application to Space Interferrometry, 6th International ESA Conference on Guidance, Navigation and Control Systems, (2005), Loutraki, Greece.

[37] Ahmed A. Shabana, Flexible Multibody Dynamics: Review of Past and Recent Developments, MULTIBODY SYSTEM DYNAMICS 1 (1997) 189-222

[38] K. Hu, N. Vlahopoulos and Z. P. Mourelatos, A Finite Element Formulation For Coupling Rigid and Flexible Body Dynamics of Rotating Beams, Journal of Sound and Vibration 253 (2002), 603-630

[39] W. Pan and E.J. Haug, Flexible multibody dynamic simulation using optimal lumped inertia matrices, Computer Methods in Applied Mechanics and Engineering 173 (1999), pp. 189–200.

[40] J.F. Imbert and A. Mamode, The effective mass concept in base motion dynamics and its application to solar array dynamics, NASTRAN User's Conf. proceeding (1977), Munich.

[41] C. Cumer and J.P. Chretien. Minimal lft form of a spacecraft built up from two bodies, AIAA Guidance, Navigation, and Control Conference, (2001). ONERA/DCSD, AIAA.

[42] Sunada, S., & Harayama, T. (2007). Design of resonant microcavities: application to optical gyroscopes. Optics express, 15(24), 16245-16254.

[43] Liebe, C. C. (1995). Star trackers for attitude determination. I E E E Aerospace and Electronic Systems Magazine, 10(6), 10-16. https://doi.org/10.1109/62.387971

[44] E. J. Post, "Sagnac effect," Rev. Mod. Phys. 39, 475-493 (1967)

[45] P. W. LIKINS and A. H. GALE: The analysis of interactions between attitude control system and flexible appendages. Prec. 19th Congr. LA.F. Vol. 2 (1968).

[46] P. W. Lx Ns and G. E. FLEISHER: Results of flexible spacecraft attitude control studies utilizing hybrid coordinates. J. Spacecraft, 264 (1971)

[47] M. GAMBA, A. SANmmNELLI and G. FERRERO : Research on the Dynamic Response of the TV Broadcasting Satellite with Flexible Paddles, Controlled by a Rigid Flywheel. A Study carried out by FIAT Centre Elettronico Avio for ESRO, November 1971 (D7664CE).

[48] G. PORCELLX: Attitude control of flexible space vehicles. AIAA Jl 10 (1972).

[49] M. ATHANS: The role and use of the stochastic linear quadratic-gaussian problem in control system design. 1EEE Trans. Aut. Control, December (1971).

[50] Chiha, I., Liouane, N., & Borne, P. (2012). Tuning PID controller using multiobjective ant colony optimization. Applied Computational Intelligence and Soft Computing, 2012.

[51] Chen, W. H., Zhang, T., & Cui, M. Q. (2011). Study of robot trajectory based on quintic polynomial transition. Meikuang Jixie(Coal Mine Machinery), 32(12), 49-50.