INCREMENTAL PLANNING OF MULTI-GRAVITY ASSIST TRAJECTORIES

Author: Mr. Juan Manuel Romero Martin
University of Strathclyde, United Kingdom, juan.romero-martin@strath.ac.uk
Mr. Luca Masi
University of Strathclyde, United Kingdom, luca.masi@strath.ac.uk
Dr. Massimiliano Vasile
University of Strathclyde, United Kingdom, massimiliano.vasile@strah.ac.uk
Dr. Edmondo Minisci
University of Strathclyde, United Kingdom, edmondo.minisci@strath.ac.uk
Dr. Richard Epenoy
Centre National d'Etudes Spatiales (CNES), France, Richard.Epenoy@cnes.fr
Mr. Vincent Martinot
Thales Alenia Space France, France, vincent.martinot@thalesaleniaspace.com
Dr. Jordi Fontdecaba Baig
Thales Alenia Space France, France, jordi.fontdecababaig@thalesaleniaspace.com

The paper presents a novel algorithm for the automatic planning and scheduling of multi-gravity assist trajectories (MGA). The algorithm translates the design of an MGA transfer into a planning and scheduling process in which each planetary encounter is seen as a scheduled task. All possible transfers form a directional graph that is incrementally built and explored simultaneously forward from the departure planet to the arrival one and backward from the arrival planet to the departure one. Nodes in the graph (or tree) represent tasks (or planetary encounters). Backward and forward generated transfers are then matched during the construction of the tree to improve both convergence and exploration. It can be shown, in fact, that the multi-directional exploration of the tree, allows for better quality solutions for the same computational cost. Unlike branch and prune algorithms that use a set of deterministic branching and pruning heuristics, the algorithm proposed in this paper progressively builds a probabilistic model over all the possible tasks that form a complete trajectory. No branch is pruned but the probability of selecting one particular task increases as the algorithm progresses in the search for a solution. The effectiveness of the algorithm is demonstrated on the design optimization of the trajectory of Marco Polo, JUICE and MESSENGER missions.

I. INTRODUCTION

A gravity assist manoeuvre takes advantage of the relative movement and gravity field of planets or other massive celestial body to change the direction and speed of a spacecraft without the use of its propulsion system. This is achieved by the momentum exchange between the spacecraft and the celestial body during the close passage (swing-by or fly-by) of the spacecraft. As a result, the use of an optimal sequence of gravity assist manoeuvre enables the access to high ΔV targets in the Solar System, like Jupiter and the four planets beyond. The optimality of a sequence of gravity assist maneuvers rests on the optimal selection of the swing-by planets and of the encounter time with each planet. The resulting optimization problem of selecting the optimal sequence of swing-by planets and encounter times can be quite complex combinatorial problem.

Deterministic algorithms for the solution of the MGAP are those that solve a problem in a systematic manner, non-probabilistic, producing the same result in each trial. Deterministic algorithms for the solution of the MGAP are based on simplified models and an enumerative search like PAMSIT [1], or a two-level approach in which the problem is split in two sub-problems: find the optimal sequence of planetary encounter from an analysis of the Tisserand's graph or from simple energetic considerations [2], and find the optimal dates for the optimal sequences of planetary encounter by performing a branch and prune type of procedure for each of the sequences.

Bio-inspired algorithms, such as Particle Swarms, Genetic Algorithms or Ant Colony, have become an appealing technique to solve NP-hard problems in

combinatorial optimization due to their use of a reasonable computational time. Bio-inspired techniques for the solution of the MGAP can be found in [3], [5], [6]. In [3] the authors proposed a hybrid branch & prune and evolutionary process that could automatically generate sequence and optimal multi-gravity assist transfer with Deep Space Manoeuvres (DSM's) in a single loop. In [5] and [6] proposed to divide the problem in two loops: the outer loop and inner loop. The outer loop generates the planet sequence by the use of the Hidden Genes Genetic Algorithm (HGGA) that is passed to the inner loop to compute the optimal time sequence with a Monotonic Basic Hopping algorithm (MBH). In [4] a planning and scheduling approach is proposed to solve the MGAP; problem is translated into a planning and scheduling problem, and then the solution is incrementally built with a modified Ant Colony Optimization strategy.

The bio-inspired heuristic presented in this paper takes inspiration from the behaviour of a simple amoeboid organism, the *Physarum Polycephalum*, that is endowed by nature with simple heuristics that can solve complex discrete decision making problems. For example, it was shown that the *Physarum Polycephalum* is able to find the shortest path through a maze [9], recreate the Japan rail network, reproduce the designed highway network among several Mexican cities [7], solve multi-source problems with a simple geometry [8], [9], mazes [10] and transport network problems [11].

The algorithm presented in this paper is applied to three different instances of real MGA problems. First, it is applied to an Earth-Near Earth Asteroid transfer type (MARCO POLO mission), and then to an Earth-Jupiter transfer type (JUICE mission). Finally, it is applied to an Earth-Mercury transfer type (MESSENGER mission).

II. MULTI-DIRECTIONAL DISCRETE DECISION MAKING

The optimization algorithm proposed in this paper takes inspiration from the biology of the *Physarum Polucephalum*, and can be seen as a branch and prune algorithm in which the decision to branch or prune is made probabilistically rather than deterministically. To be more specific in the following branches are never really pruned but the probability of selecting them falls to almost zero. The mechanism is analogous to the most commonly known Ant Colony Optimization algorithm although with the distinctive novelty that the exploration of the tree of decisions proceeds in multiple directions. In analogy with A* type of path planning or with dynamic programming algorithms when the search proceed forward from the source to the sink the backward branches work as the heuristic function and vice versa when the search proceeds backward.

The *Physarum's* mathematical model is composed mainly by two main parts: 1) decision network exploration and 2) decision network growth in multiple directions. They are presented in this section along with a restart procedure that mitigates the risk of stagnation. The main parameters of the modified *Physarum* solver are summarized in **Table I**. The complete pseudocode of the multidirectional incremental modified *Physarum* solver is provided in **Algorithm 1**.

m	Linear dilation coefficient, see **Eq. (6)**.
ρ	Evaporation coefficient, see **Eq. (3)**
GF	Growth factor, see **Eq. (5)**
N_{agents}	Number of virtual agents.
p_{ram}	Probability of ramification.
λ	Weight on ramification, see **Eq. (8)**

Table I Setting parameters for the modified Physarum solver

Multidirectional Incremental Physarum Solver Pseudocode

initialize $m, \rho, GF, N_{agents}, p_{ram}, \lambda$
for each generation **do**
  for each virtual agent in all direction (DF and BF) **do**
    if current node ≠ end node **then**
      if $v \in U(0,1) \leq p_{ram}$ **then**
        using Eq. (8) create a new decision path, building missing links and nodes
      else
        Move on existing graph using Eq. (4)
      end if
    end if
  end for
  Look for possible matching, see **Sec. II**.
  Contract and Dilate veins using Eqs. (2), (3), (5)
  if r_{ij} exceeds upper radius, see Eq. (7) **then**
    Block radius increment
  end if
  Update fluxes and probabilities using Eqs. (1), (4)
  if *restart condition* **then**
    Update veins' radii using Eq. (9)
    Update fluxes and probabilities using Eqs. (1), (4)
  end if
end for

Algorithm 1 Pseudocode of the Multidirectional Physarum solver

II.I Decision network exploration

The Decision network exploration is based on the flux through the net of *Physarum* veins. The flux of the

Physaryn veins can be modelled as a classical Hagen-Poiseuille flow in cylindrical ducts with variable diameter that varies with time [8, 10, 11]:

$$Q_{ij} = \frac{\pi r_{ij}^4}{8\mu} \frac{\Delta p_{ij}}{L_{ij}} \quad (1)$$

where

- Q_{ij} is the flux between *i* and *j*
- μ is the dynamic viscosity
- r_{ij} is the radius of the vein
- L_{ij} is the length of the vein
- Δp_{ij} is the pressure gradient

For a better understanding of these parameters, they have been illustrated in a simple graph in **Fig. 1.**

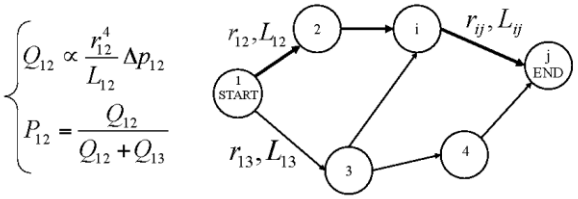

Fig. 1 Figure illustrate a simple graph where the thinker arrows represent higher fluxes. In this example $\boldsymbol{Q_{12} > Q_{13} \rightarrow P_{12} > P_{13}}$

A variation in the diameter of the veins allows for a change in the flux. The dilation of the veins due to an increase in the flowing nutrients can be modelled using a monotonic function of the flux:

$$\left.\frac{d}{dt} r_{ij}\right|_{dilation} = f\left(Q_{ij}\right) \quad (2)$$

where $f(0) = 0$, i.e., linear, sigmoidal, etc. It can be assumed that the dynamics of the veins is sufficiently slow for the flow to be considered in steady state [10] regime. The contraction of the veins, due to evaporative effects, can be assumed to be directly proportional to their radius:

$$\left.\frac{d}{dt} r_{ij}\right|_{contraction} = -\rho r_{ij} \quad (3)$$

where $\rho \in [0,1]$ is a pre-defined evaporation coefficient.

The probability associated with each vein connecting the node *i* and the node *j* is computed using a simple adjacency probability matrix based on fluxes:

$$P_{ij} = \begin{cases} \frac{Q_{ij}}{\sum_{j \in N_i} Q_{ij}}, & if\ j \in N_i \\ 0, & if\ j \notin N_i \end{cases} \quad (4)$$

where N_i is the set of neighbouring veins to a node *i*.

The original *Physarum* logic was modified by introducing a further term in the dilation process. The new term takes inspiration from the behaviour of the amoeba *Dictyostelium discoideum.* In its aggregative and slug stages, amoebae are chemotactically sensitive to a chemical known as cyclic Adenosine Monophosphate (cAMP). A starving pacemaker amoeba starts to emit cAMP, that is a call for aggregation and subsequent collective behaviour. In a computational algorithm, pacemaker can be considered the agent with best objective function. A linear dilation for the pacemaker, which is defined as the best path so far in the decision graph in terms of objective function, was here chosen:

$$\left.\frac{d}{dt} r_{ij_{best}}\right|_{elasticity} = GF r_{ij_{best}} \quad (5)$$

where *GF* is the growth factor of the best chain of veins and $r_{ij_{best}}$ the veins' radii. This pacemaker call can be interpreted as a variable elasticity of the veins with time: best veins increase their capacity of dilation with a percentage *GF*. This is an additive term in the veins' dilation process, whose first main term is expressed in **Eq. (2)**

The set of **Eqs (1)-(4)** can be implemented following the method proposed in [8] and resembles classical Ant Colony Optimization algorithms. Nutrients inside veins are interpreted as virtual agents that move in accord with the adjacency probability matrix in **Eq. (4)** on the existing graph, see line 8 of ***Algorithm 1***. In accordance to **Eq. (1)**, the flux in each vein is proportional to the fourth power of the radius and inversely proportional to the length. Once a vein is selected by a virtual agent in a generation, its radius is incremented using **Eq. (2)**. In the present work, a function linear with respect to the product between the radius $r_{ij}^{(k)}$ of the veins traversed by agent *k*, and the inverse of the total cost of the decision taken by agent *k*, i.e., the total length $L_{tot}^{(k)}$, will be used for the veins' dilation:

$$\left.\frac{d}{dt} r_{ij}^{(k)}\right|_{dilation} = m \frac{r_{ij}^{(k)}}{L_{tot}^{(k)}} \quad (6)$$

where the coefficient *m* is here called linear dilation coefficient. Evaporation is taken into account using **Eq.(3)** for each agent. Fluxes are then calculated using **Eq. (1)** and probabilities are updated in accordance with **Eq. (4)**. This mechanism of veins' diameter and flux

updating corresponds to *line 13* of **Algorithm 1**, where **Eq. (5)** contributes to the veins' growth of the pacemaker. An upper limit on the maximum vein radius was introduced in order to avoid veins' flux explosion and limit the converge rate. If the radius r_{ij} exceeds a maximum value r_{max}, the vein dilation is stopped until the radius returns again below r_{max} for the effect of evaporation. This upper limit, called $k_{explosion}$, is given as the ratio between r_{ij} and r_{ini}:

$$k_{explosion} = \frac{r_{ij}}{r_{ini}} \quad (7)$$

where r_{ini} is the initial radius of the veins. This mechanism corresponds to *lines 14* to *16* of **Algorithm 1**.

II.II Multidirectional Growth of the Decision Network

Solutions are composed adding branches and nodes incrementally. Adding a node implies a decision so does traversing the tree along a particular path. The incremental growth of the decision network in one direction is performed in parallel by a set of virtual agents. At every node of the tree, each agent either generates a new branch or moves along an existing one. At each node, the agent has a probability p_{ram} of ramification towards new nodes that are not yet linked with the current one. On *line 5* of **Algorithm 1**, a random number v is drawn from a uniform distribution $U(0,1)$ and the condition $v < p_{ram}$ is verified. Assuming that the agent is at node *i*, if ramification is the choice, the agent evaluates the set of possible new branches and assigns a probability p_{ij} of constructing a new link from the current node *i* to a new possible node $j \in \bar{N}_i$, where $\bar{N}_i$ is the set of unlinked nodes (for example nodes 4 and 5 in **Fig. 2.a**), according to:

$$p_{ij} \propto \frac{1}{L_{ij}^{\lambda}} \quad (8)$$

where λ is a pre-defined exponent. **Fig. 2.a** shows a possible ramification from the start node: dotted lines represent feasible branches not yet existing. If an agent is at the start node it has a probability p_{ram} of ramification towards the unlinked nodes 4 and 5. If the agent decides to create a new link, a new node is selected according to **Eq. (8)**, see *line 6* of **Algorithm 1**.

If a set of linked nodes is available, the agent can decide, with probability $1 - p_{ram}$, to traverse the existing branches in the neighbourhood N_i (see *line 8* of **Algorithm 1**). In the case shown in **Fig. 2.a** when an agent is at the start node, it can explore the already linked nodes 2 and 3. Once at node 2 or 3, the only possibility in order to complete the decision path is a new link construction between the current node and ending node.

In order to explore the decision space from multiple starting points, multiple *Physarum* are simultaneously grown and expanded in multiple directions. In this paper, a bi-directional approach is presented in which two trees, called *DF* (Direct Flow) and *BF* (Back Flow), form a network made of two superposed graphs. While growing, the two expanding *Physarum* have the possibility of merging decision sequences: agents can build and traverse arcs that connect nodes belonging to *DF* and *BF Physarum* respectively forming a single path from the heart of one *Physarum* to the heart of the other *Physarum*, see *line 12* of **Algorithm 1**.

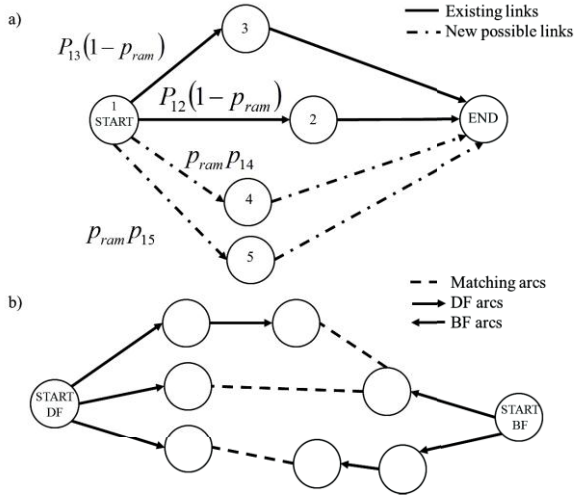

Fig. 2 Figure illustrating (a) ramification towards a new node and (b) merging between two decision routes, DF and BF routes.

Figure **Fig. 2.b** illustrates a simple case of merging sequences between the graphs associated to two amoebae (DF and BF). The merged decision path is given by the union of a route in the *DF* and a route in the *BF* through a merging arc.

The modified Multidirectional Physarum's merging method consists of taking the best n_{seq} BF and DF partial routes and then merge together by connecting them. The connection process randomly selects a pair of nodes along the two routes and tries to connect the two nodes with a merging arc. In the following, the top 10 routes generated in DF and BF are matched assuming an equal probability of cutting any of the arcs.

II.III Restart Procedure

Although the parallel multi-direction exploration of the decision trees increases the chances to find good solutions, there exists the risk of a premature stagnation due to an excessive increase of the decision probability along a particular path. This is equivalent to a premature explosion of the veins. In order to mitigate this problem, a restart procedure was added to the exploration process. If a certain condition, here called *restart condition*, is reached, the veins' radii are reset to:

$$r_{ij}=r_{ini} \tag{9}$$

The restart procedure is based on the number of nodes and arcs in common between two decision sequences: after comparing all decision sequences among each other, if the minimum number of nodes in common n_{min}^{com} exceeds a threshold n_{share}, the algorithm is restarted. The restart procedure is summarised at *lines 18* to *21* of **Algorithm 1**.

III. MULTI-GRAVITY ASSIST PROBLEM 3D LAMBERT MODEL WITH DEEP SPACE MANEUVER

The trajectory model used in this study is based on a lambert linked-conic trajectory. The MGAP is modeled using the actual ephemerides of the planets. The trajectories are composed of sequence of conic arcs linked together through discrete, instantaneous events. In particular, the sequence is continuous in position and piecewise continuous in velocity, i.e. each event introduces a discontinuity in the velocity of the spacecraft but not in its position.

Now, let's evaluate the case where a sequence of three planets {A, B, C} and three dates {T_A, T_B, T_C}, at which the spacecraft is at each planet, are given. The solution of the Lambert's problem provides the conic arc connecting each pair of two planets, as well as the corresponding velocity vectors at the beginning and at the end of each arc. If the planet B is a swing-by planet, the discrepancy of velocity at point B between the incoming velocity (velocity vector at the end of the A-B lambert arc) and the outgoing velocity (velocity vector at the beginning of the B-C lambert arc) is partially compensated by the gravity of the planet B by steering the incoming velocity. However not all incoming velocities can be naturally steered to match the outgoing velocities, due to the restriction on the altitudes at which the spacecraft is allowed to swing-by planet, as well as the immutability of the mass of the swing-by planet. Therefore, a propelled maneuver ΔV_i is required at the pericentre of the swing-by hyperbola to overcome the mismatch of velocity. The combination of powered maneuver and gravity steering is called powered gravity assist maneuver or powered swing-by.

III.I Formulation in Position

Given a vector of time $T^A = [T_1^A, T_2^A, \dots, T_i^A, \dots]^T$, the position and velocity of the first planet in a sequence, say *A*, are calculated from the ephemerides. For all the subsequent planets, up to the last one in the sequence, instead, the times are derived from the phase angles of the planet on its orbit (see **Fig. 3**). Assuming *B* is the next planet in the sequence, following *A*, and θ_1^B is the phase angle of *B* on its orbit at time T_1^A, the position, velocity and time T_j^B of *B* are computed for $\theta_j^B = \theta_1^B + \Delta\theta_j$, with $j = 1, \dots, n_B$ and $n_B = 2\pi/\Delta\theta_j$. The time corresponding to a given discrete phase angle can be computed from the time equation in the form $T_j^B = f\left(\theta_j^B + 2k_r\pi\right)$ with f the operator converting from true anomaly to time. The same model is applied also in reverse from the last planet to the first. In this case the position and velocity of the last planet are calculated from the ephemerides given a time vector that spans the desired arrival window.

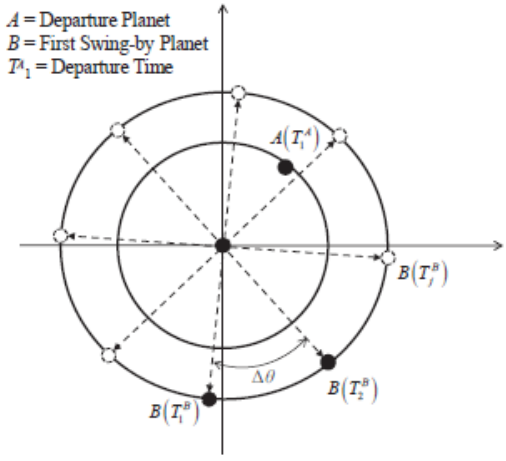

Fig. 3 Formulation in position for transfers between two planets.

III.II Generation of the Search Tree

If the vectors of encounter dates for planet *A* and *B* are respectively $T^A = [T_1^A, T_2^A, \dots, T_i^A, \dots]^T$ and $T^B = [T_1^B, T_2^B, \dots, T_j^B, \dots]^T$, then the set of possible transfers from *A* to *B* can be represented with a matrix where each element z_{ij}^{AB} of the matrix is the cost associated with a particular $T_i^A \rightarrow T_j^B$ transfer. If multiple alternative planets are available the matrix becomes

three dimensional, with the third dimension containing all possible planets. Note that each element of the matrix is a node in the tree of decisions that the *Physarum* incrementally builds, therefore only the nodes that the *Physarum* explores are actually generated and added to the tree. However, when a planet *A* the *Physarum* generates a new branch towards *B* all the transfers to *B* are evaluated and one selected to be added as a new node to the tree, the other transfers are stored for later addition of other nodes.

The cost z_{ij}^{AB} is the launch excess velocity Δv_0 if planet *A* is the departure planet, the powered swing-by cost Δv_i if *A* is a swing-by planet, or the sum of Δv_i and the arrival excess velocity Δv_f if *B* is the final planet. The cost of a complete transfer is then the sum of the departure Δv_0 plus all the Δv_i for all the planetary encounters and Δv_f. In the *Physarum* algorithm, the variables $L_{tot}^{(k)}$ in **Eq. (1)** and L_{ij} in **Eq. (6)** are then replaced by, respectively, $\Delta v_{tot}^{(k)}$ and z_{ij}^{AB}.

From a given planet at a particular node, a new planet is selected with a probability proportional to the inverse of the difference of the semimajor axis of the new planet with respect to the current one. Once the costs for the whole vector $\boldsymbol{T^B}$ is available, a transfer is selected, for example $T_2^A \to T_2^B$, with **Eq. (4)**, where only the costs z_{ij}^{AB} are used to compute the fluxes. If $v \in U(0,1) > p_{ram}$, the algorithm does not evaluate the cost for a new set of transfer arcs (i.e., does not build a new branch) but selects an existing arc among the available possibilities using **Eq. (4)**. The process is repeated until the final target planet is reached and a complete decision sequence is built. If, during the construction of a solution, no transfer arcs can be found that satisfy the constraints, then the construction is terminated and an infinite cost (or equivalently a zero probability) is associated to the resulting partial solution. **Eq. (6)** was slightly modified by substituting $L_{tot}^{(k)}$ with $L_{tot}^{(k)} + 1$ in order to avoid possible singularities that may appear with the MGA model.

III.III Local Solution Improvement Strategy

In order to improve the quality of the solutions, a local search procedure inspired to the *2-opt* local search strategy, commonly used in Ant Colony Optimization, was added to the algorithm. If $s = [A, T_2^A, B, T_7^B, C, T_{12}^C, D, T_{16}^D]^T$ is a solution vector, the local improvement checks whether a positive or negative increment of T_2^A, δT, improves the solution. If, for example, $\Delta v_{tot}(A, T_2^A, B, T_7^B, C, T_{12}^C, D, T_{16}^D) > \Delta v_{tot}(A, T_2^A + \delta T, B, T_7^B, C, T_{12}^C, D, T_{16}^D)$, then T_2^A is replaced by $T_2^A + \delta T$. The same δT is repeatedly added (or subtracted) to T_2^A till no improvement is registered. The process is then applied to T_7^B and the other dates till T_{16}^D, and repeated backwards from T_{16}^D to T_2^A. With this process the modified dates do not necessarily correspond to the discretised phase angles and a finer discretisation can be used for the local search.

III.IV Algorithm and problem settings

A number of additional quantities need to be defined to characterize a particular instance of the MGA problem along with the algorithm's parameters m, ρ, GF, N_{agents}, p_{ram}, r_{ini}, $k_{exploration}$ and λ introduced in Sec II. In particular, the departure planet P_0, the upper and lower boundaries on the swing-by altitude divided by the radius of the planet h_{low} and h_{up}, the set of available swing-by planets $P_s = \{P_1, P_2, \dots, P_{N_P}\}$, maximum number of swing-by's $n_{s_{max}}$, maximum number of resonances res_{max}, interval of dates defining the launch window T_{launch}, the interval of dates defining the arrival window $T_{arrival}$, the lower and upper boundaries on the time of flight ToF_{ij}^{low} and ToF_{ij}^{up} for each leg connecting two planets *i* and *j*, the final target planet P_{target}, the grid spacing in angle $\Delta\theta_{ij}$ and the upper and lower boundaries on launch and arrival velocities, respectively Δv_0^{max}, Δv_0^{min} and Δv_f^{max}, Δv_f^{min}. The value of parameters of the algorithm for the cases in this and following section are $m = 5 \times 10^{-3}$, $\rho = 10^{-4}$, $GF = 5 \times 10^{-3}$, $N_{agents} = 10$, $p_{ram} = 1$ and $\lambda = 0$, whilst r_{ini} is increased from 1 to 2, and $k_{exploration}$ consequently, in order to have a maximum radius of 5. Planets are identified with the following letters M (*Mercury*), V (Venus), E (Earth), Ma (Mars), J (Jupiter), S (Saturn), U (Uranus), N (Neptune), P (Pluto). For example the sequence EVVEJS means Earth-Venus-Venus-Earth-Jupiter-Saturn.

In comparison with the more commonly known Travelling Salesman Problem, if one considered, each planet-encounter time pair as a city, each city would be revisited k times, N_p cities would be available per each encounter and each pair of cities would require the evaluation of $(N_p k N_T)^2$ transfer arcs, where N_T is the number of discrete encounter dates, then the total number of transfers to be evaluated would be $kN_p\left(N_p k N_T\right)^2$. If the transfer arcs were put together in a sequence, the number of alternative sequences would be $kN_p{}^{(N_p k N_T)}$.

IV. CASE STUDIES

In this section, a number of real application case studies are used to evaluate the performances of the multi-directional discrete decision making *Physarum* algorithm. The first case study is applied to an Earth-Near Asteroid 1999 JU3 mission, similar to the Marco Polo, the second case optimizes the interplanetary trajectory to Jupiter for the JUICE mission and, finally, the third case optimizes part of the MESSENGER mission. All of the missions presented in this section are MGA with intermediate deep space manoeuvre.

All test cases presented in this section were executed in MATLAB 2013a on an Intel(R) Core(TM) i7-3770 3.40 Ghz and 8GB RAM under Windows 7.

IV.I Marco Polo Case Study

The trajectory studied in this section is the first phase of the Marco Polo mission, departing from Earth and arriving at 1999 JU3. No return transfer to Earth is considered.

Marco Polo was a proposed space mission study of a scientific return mission to the Apollo C-type asteroid 1999 JU3 [11]. Marco Polo is a MGA trajectory from Earth to 1999 JU3 following the sequence Earth – Earth – 1999 JU3 (EE-Asteroid) in its nominal trajectory and following the sequence Earth – Mars – Earth – 1999JU3 (EMaE-Asteroid) in its optional trajectory [12]. The nominal transfer (EE-Asteroid) has 2018/12/20 as the departure date from Earth and arrival date at the asteroid on 2022/02/14 with a ΔV cost of 3.7 km/s and a transfer time of about 3.2 years. The optional Earth-Mars transfer (EMaE-Asteroid) has 2017/12/21 as the launch date with a transfer time of 4.3 years, resulting in an arrival at the Asteroid on 2022/04/08. The optional Earth-Mars-transfer, with a ΔV cost of 4.3 km/s, has a higher ΔV than the nominal transfer of 0.5 km/s [12].

This case study is used to assess the sensitivity of the *Physarum* algorithm to some of the key parameters defining a particular family of MGA transfers. The parameters used for this sensitivity assessment are the Launch and Departure windows, as well as the grid spacing. Two different test cases are considered in this case study. Both of the test cases have the same setting parameters, only the launch and arrival windows are difference between them. Each of test cases are divided into two subtest cases which have different grid spacing settings to assess the impact of this key parameter on the performance of the algorithm.

IV.I.I Test Case 1

This test case considers the launch window that goes from 2017/07/05 to 2018/06/30 and the arrival window that goes from 2021/11/23 to 2022/11/18. The Marco Polo reference solution, for these departure and arrival windows, has a departure from Earth on 2018/11/20 and an arrival at the asteroid on 2022/02/14 with a total ΔV cost of 3.7 km/s and a transfer time of about 3.2 years.

Two different grid spacing were used to assess the convergence of the *Physarum* algorithm: **Table IV** is set to a have higher grid spacing resolutions (lower granularity) while **Table V** is set to a lower resolution (higher granularity)**.** Both cases use the same setting for the remaining parameters (see **Table II**). A set of three swing-by planets, P_s= {V E Ma}, is considered and the maximum total number of swing-by's was fixed to three with a maximum of two repeating planets in the same sequence. **Table III c**ontains the lower and upper boundaries on the Time of Flight for each possible leg connecting two planets. The setting for the grid spacing $\Delta\theta$, for test case 1.a and 1.b, can be found respectively in **Table IV** and **Table V**. **Table VI** and **Table VII** shows the best Top 5 trajectories found by the *Physarum* algorithm for both cases.

As it can be observed, the *Physarum* algorithm was able to find in both cases, low and high grid spacing resolution, the first phase of the nominal Marco Polo trajectory. The results for both test cases are basically identical proving that even with big difference in grid spacing resolution, the *Physarum* algorithm succeeded to find the best optimal solution. All the Top 5 results for both tests present the same sequence, EE-1999 JU3, as well as similar total ΔV cost of 3.6 km/s. This sequence EE-1999JU3 is similar to the nominal Marco Polo trajectory. It is remarkable how with a large launch and arrival windows (360 days respectively), the *Physarum* algorithm was able to a find an optimal solution with slightly better ΔV cost than the nominal Marco Polo trajectory, but with a transfer time longer of about 0.3 years (total transfer time around 4 years).

Parameter	Value
Set of Available Planets, P_s	{V E Ma}
ns_{max}	3
res_{max}	2
T_{launch}*[yyyy/mm/dd]*	2017/07/05 to 2018/06/30
T_{Step_launch}	10 day
$T_{arrival}$	2021/11/23 to 2022/11/18
$T_{Step\_arrival}$	10 day
$[\Delta v_0^{min}, \Delta v_0^{max} \Delta \mathrm{v_{max}}]$	[3.4, 5.5] km/s
$[\Delta v_f^{min}, \Delta v_f^{max} \Delta \mathrm{v_{max}}]$	[0.02, 2] km/s
h_{low} for { V E Ma}	[0.05, 0.1, 0.15]
h_{high} for {V E Ma}	[10, 70, 20]

Table II Test Case 1 Problem Definition Parameters

V	E	Ma	
[100, 500]	[30, 500]	[300, 2000]	**V**
[100, 200]	[200, 400]	[930, 1000]	**E**
[0, 0]	[60, 300]	[0, 0]	**Ma**

Table III Test Case 1: Lower and Upper Boundaries for Time of Flight [day]

V	E	Ma	
2	1	2	**V**
1.6	1	2	**E**
0	2	2	**Ma**

Table IV Test Case 1.a: Spacing Grid Definition $\Delta\theta$ [deg]

V	E	Ma	
10	10	10	**V**
10	10	10	**E**
10	10	10	**Ma**

Table V Test Case 1.b: Spacing Grid Definition $\Delta\theta$ [deg]

Rank	Cost (*km/s*)	Sequence		
1	3.6	Earth	Earth	1999 JU3
		6564.5	*7683.0*	*8056.5*
2	3.6	Earth	Earth	1999 JU3
		6524.5	*7638.9*	*8036.5*
3	3.6	Earth	Earth	1999 JU3
		6524.5	*7643.0*	*8166.5*
4	3.6	Earth	Earth	1999 JU3
		6514.5	*7626.8*	*8056.5*
5	3.6	Earth	Earth	1999 JU3
		7996.5	*7668.9*	*7996.5*

Table VI Test Case 1.a: Top 5 Sequence Results

Rank	Cost (*km/s*)	Sequence		
1	3.6	Earth	Earth	1999 JU3
		6514.5	*7643.8*	*8126.5*
2	3.6	Earth	Earth	1999 JU3
		6514.5	*7633*	*8086.5*
3	3.6	Earth	Earth	1999 JU3
		6514.5	*7624.3*	*8026.5*
4	3.6	Earth	Earth	1999 JU3
		6504.5	*7617.5*	*8016.5*
5	3.6	Earth	Earth	1999 JU3
		6564.5	*7676.5*	*8036.5*

Table VII Test Case 1.b: Top 5 Sequence Results

IV.I.II Test Case 2

This test case considers the launch window that goes from 2016/07/05 to 2017/06/30 and the arrival window that goes from 2021/11/23 to 2022/11/18. The Marco Polo reference solution for these departure and arrival windows has a departure from Earth on 2017/12/21 and an arrival at the asteroid on 2022/02/08 with a total ΔV cost of 4.3 km/s and a transfer time of about 4.3 years.

This test case uses the same settings as the previous test case for all the parameters: problem definition parameters (see **Table II**), lower and upper Time of Flight boundaries (see **Table III**), and grid spacing (see **Table IV** and **Table V**). The only difference is the use of a new launch and arrival window (see **Table VIII**). **Table IX** and **Table X** contain the best trajectories found by the *Physarum* algorithm for both test cases 2.a and 2.b.

As it can be observed, the *Physarum* algorithm was able to converge with the same solutions on both test cases. The solutions are similar to the optional Marco Polo's Earth-Mars transfer (EMaE-Asteroid), but with an additional resonance with Earth before Mars encounter (EEMaE-Asteroid). Although most of the found trajectories present slightly different sequence than the optional Earth-Mars transfer, they also present a substantial improvement on total ΔV cost with respect to the original. All the trajectories have a reduction of ΔV cost around 0.7 km/s. In the test case 2.a, the first trajectory with the sequence EMaE-Asteroid can be found at the position 6 of the ranking, while in the test case 2.b, it can be found at the position 19. **Fig. 4** and **Fig. 5** show the best trajectories for Test Case 2.a and 2.b respectively.

Parameter	Value
T_{launch}*[yyyy/mm/dd]*	2016/07/05 to 2017/06/30
T_{Step_launch}	10 day
$T_{arrival}$	2021/11/23 to 2022/11/18
$T_{Step\_arrival}$	10 day

Table VIII Test Case 2: Problem Definition Parameters

Rank	Cost *(km/s)*	Sequence				
1	3.57	Earth	Earth	Mars	Earth	1999JU3
		6049.5	*6579.3*	*7557.7*	*7667.7*	*8196.5*
2	3.58	Earth	Earth	Mars	Earth	1999JU3
		6229.5	*6594.7*	*7541.3*	*7640.4*	*8206.5*
3	3.6	Earth	Earth	Mars	Earth	1999JU3
		6229.5	*6594.7*	*7525.1*	*7636.7*	*8066.5*
4	3.62	Earth	Earth	Mars	Earth	1999JU3
		6049.5	*6578.9*	*7557.3*	*7667.3*	*8006.5*
5	3.63	Earth	Venus	Earth	1999JU3	
		6199.5	*6319.4*	*7646.4*	*8016.5*	
6	3.64	Earth	Mars	Earth	1999JU3	
		6199.5	*7525.9*	*7643.5*	*8026.5*	

Table IX Test Case 2.a: Sequence Results

Rank	Cost*(km/s)*	Sequence				
1	3.58	Earth	Earth	Mars	Earth	1999 JU3
		6189.5	*6554.7*	*7548.4*	*7658.8*	*8196.5*
2	3.58	Earth	Earth	Mars	Earth	1999 JU3
		6069.5	*6599.8*	*7566.5*	*7656.5*	*8156.5*
3	3.58	Earth	Earth	Mars	Earth	1999 JU3
		6189.5	*6554.7*	*7514*	*7665.7*	*8266.5*
4	3.59	Earth	Earth	Mars	Earth	1999 JU3
		6189.5	*6554.7*	*7524.9*	*7636.5*	*8236.5*
⋮						
19	3.64	Earth	Mars	Earth	1999JU3	
		6199.5	*7514.3*	*7636.4*	*8056.5*	

Table X Test Case 2.b: Sequence Results

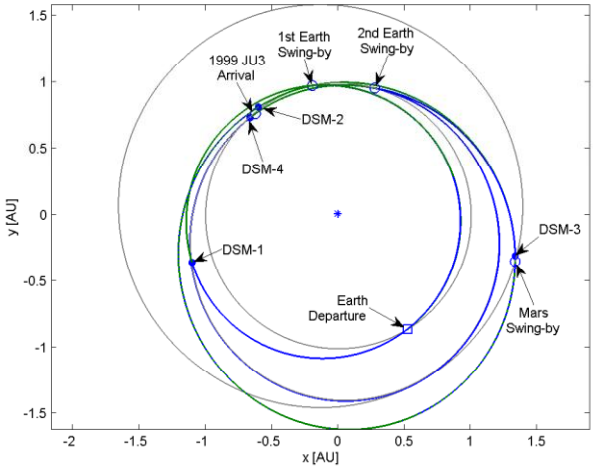

Fig. 4 Best Test Case 2.a Solution

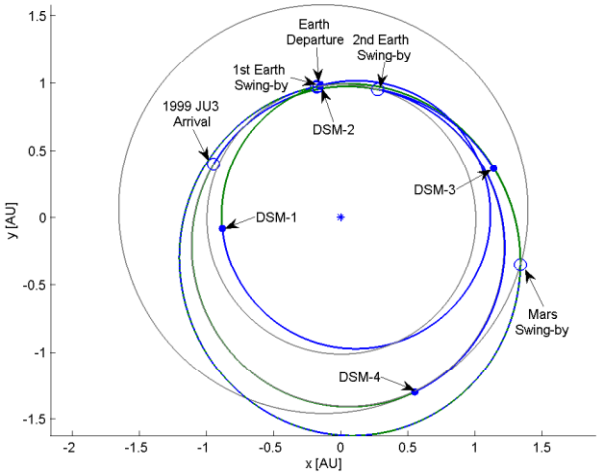

Fig. 5 Best Test Case 2.b Solution

IV.II JUICE Case Study

In this section, the trajectory studied is the reduced version of the JUICE mission, departing from Earth and arriving at Jupiter, avoiding the transfer phase to Jupiter's satellites.

JUICE (JUpiter ICy moons Explorer) is a scientific mission to explore the emergence of habitable worlds around the gas giant Jupiter and its satellites (Europa, Callisto and Ganymede) [14]. JUICE is a MGA trajectory from Earth to Ganymede following the sequence Earth – Venus – Earth – Earth – Ganymede (EVEE-GA) [14]. The mission is planned to be launched in mid-2022, with a 7.6 years of time of flight, arriving at Ganymedes around January 2030. The backup launch opportunity epoch is in August 2023 with a transfer time of 8 years, resulting in an arrival at Jupiter in August 2031 [13]. Both trajectories, the nominal and the backup, have a ΔV cost around 8.9 km/s [14].

This case study is used to assess the sensitivity of the *Physarum* algorithm to the launch and arrival windows key parameters under MGA problem with a large maximum of total number of swing-bys. Two different test cases are considered in this case study. Both of the test cases have the same setting parameters, only the launch and arrival windows have different setting between them. Test Case 1 has a wide launch and arrival windows of 360 days, while Test Case 2 has narrow windows of 60 days.

IV.II.I Test Case 1

This test case considers a launch window that goes from 2021/12/03 to 2022/11/28, and an arrival window from 2029/07/25 to 2030/07/20, 360 days window respectively.

The **Table XI** contains the parameters defining the problem for this particular case. A set of four planets, P_s= {V E Ma J}, is set as available swing-by planets. The maximum total number of swing-bys is six with a maximum of three repeating planets in the same sequence. **Table XII** contains the lower and upper boundaries on the Time of Flight for each possible leg connecting two planets. Additionally, **Table XIII** contains the setting for the grid spacing $\Delta\theta$ in degrees.

Table XIV shows the best 10 trajectories found by the *Physarum* algorithm after 6,000 function evaluations. The best sequence in table **Table XIV** is EEMaJ with a cost of 7.56 km/s and with total transfer time of 6.7 years. The first three best sequences present the same sequence and have very similar ΔV cost. The only difference is the epoch of the swing-by at Mars. None of the best 10 trajectories found have the same

sequence as the nominal JUICE mission, however all of them present a substantial improvement on the ΔV cost.

Parameter	Value
Set of Available Planets, P_s	{V E Ma J}
ns_{max}	6
res_{max}	3
$T_{launch}[yyyy/mm/dd]$	2021/12/03 to 2022/11/28
T_{Step_launch}	10 day
$T_{arrival}$	2029/07/25 to 2030/07/20
$T_{Step\_arrival}$	10 day
$[\Delta v_0^{min}, \Delta v_0^{max} \Delta \mathrm{v_{max}}]$	[3, 5] km/s
$[\Delta v_f^{min}, \Delta v_f^{max} \Delta \mathrm{v_{max}}]$	[0.02, 2] km/s
h_{low} for { V E Ma J}	[0.05, 0.05, 0.05, 0.1]
h_{high} for {V E Ma J}	[10, 70, 20, 80]

Table XI JUICE Test Case 1: Problem Definition Parameters

V	E	Ma	J	
[100, 500]	[30, 500]	[300, 2000]	[500, 3000]	**V**
[100, 200]	[200, 1000]	[930, 1000]	[800, 1500]	**E**
[0, 0]	[60, 300]	[0, 0]	[400, 1500]	**Ma**
[0, 0]	[0, 0]	[0, 0]	[0, 0]	**J**

Table XII JUICE Test Case 1: Lower and Upper Boundaries of Time of Flight [day]

V	E	Ma	J	
2	1	2	0.5	**V**
1.6	1	2	0.2	**E**
0	2	2	0.5	**Ma**
0	0	0	0	**J**

Table XIII JUICE Test Case 1: Spacing Grid Definition $\Delta\theta$ [deg]

Rank	Cost *(km/s)*	Sequence
1	7.56	Earth Earth Mars Jupiter 8336.5 8865.4 9738.8 10807.5
2	7.58	Earth Earth Mars Jupiter 8336.5 8865.4 9630.6 10837.5
3	7.68	Earth Earth Mars Jupiter 8336.5 8865.4 9732.8 10827.5
4	8.04	Earth Mars Jupiter 8056.5 9630.6 10837.5
5	8.05	Earth Mars Jupiter 8066.5 9646.8 10907.5
6	8.11	Earth Earth Mars Jupiter 8226.5 8753.4 9668.1 10877.5
7	8.13	Earth Mars Jupiter 8076.5 9652.3 10847.5
8	8.41	Earth Earth Mars Jupiter 8336.5 8865.4 9596.4 11017.5
9	8.49	Earth Venus Venus Earth Earth Mars Jupiter 8016.5 8160.7 8610.1 8763.7 9129 9650.6 11107.5
10	8.65	Earth Earth Mars Jupiter 8046.5 8572.9 9646.8 10907.5

Table XIV JUICE Test Case 1: Top 10 Trajectories

IV.II.II Test Case 2

This test case uses the same settings as the previous case, Test Case 1 (see **Table XI)**, as well as the same lower and upper boundaries of Time of Flight (see **Table XII** and grid spacing (see **Table XIII**). The only difference is the use of a narrow launch and arrival windows of 60 days. The new launch window goes from 2022/04/01 to 2022/06/01, and the arrival window from 2029/12/21 to 2030/02/21.

Table XVI shows the best 10 trajectories found by the *Physarum* algorithm for the Test Case 2. The best sequence in **Table XVI** is EEVEEJ with a total ΔV cost of 8.82 km/s and with a total transfer time of 7.7 years. The sequence of this solution is slightly similar to the nominal JUICE mission but with an extra swing-by at Earth before the encounter with Venus. Although this solution has an extra resonance with Earth, it presents an improvement on total ΔV cost, 0.1 km/s, and transfer time, 0.1 years. The second and the third trajectories present different sequences, as well as an increment of about 2 km/s on the total ΔV cost. **Fig. 6** illustrates the best trajectory from **Table XVI**.

Parameter	Value
$T_{launch}[yyyy/mm/dd]$	2022/04/01 to 2022/06/01
T_{Step_launch}	10 day
$T_{arrival}$	2029/12/21 to 2030/02/21
$T_{Step\_arrival}$	10 day

Table XV JUICE Test Case Problem Definition Parameters

Rank	Cost *(km/s)*	Sequence
1	8.82	Earth Earth Venus Earth Earth Jupiter 8140.5 8505.7 8688.9 8985.3 9715.8 10962.5
2	10.35	Earth Earth Venus Earth Jupiter 8140.5 9017.7 9356.4 9755.7 10972.5
3	10.56	Earth Earth Venus Venus Earth Earth Jupiter 8170.5 8535.7 8715.1 8939.8 9401.7 10132.1 10982.5
4	10.62	Earth Venus Venus Earth Jupiter 8140.5 8919.2 9383.5 9771.2 10962.5
5	10.90	Earth Earth Venus Earth Jupiter 8140.5 8505.7 8688.9 8985.3 10962.5
6	11.03	Earth Earth Venus Venus Earth Jupiter 8140.5 8505.7 8687.9 9351.3 9752.8 10982.5
7	11.10	Earth Earth Venus Earth Jupiter 8160.5 8525.7 9356.4 9755.7 10972.5
8	11.13	Earth Earth Mars Jupiter 8170.5 8696.3 9661.9 10962.5
9	11.14	Earth Earth Venus Earth Jupiter 8160.5 8525.7 9351.3 9752.8 10982.5
10	11.15	Earth Earth Venus Venus Earth Earth Jupiter 8170.5 8535.7 8715.1 8939.8 9401.7 10112.8 10982.5

Table XVI JUICE Test Case 2: Top 10 Trajectories

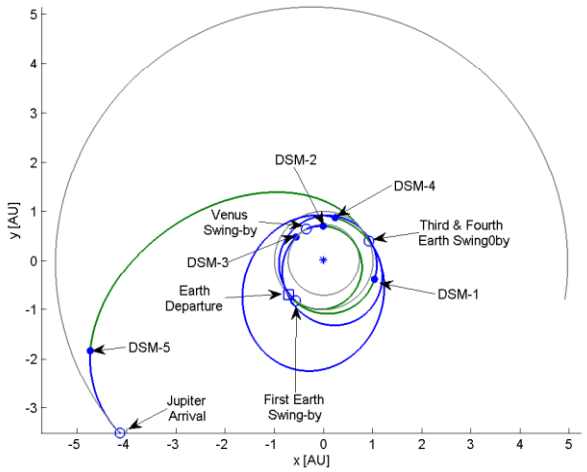

Fig. 6 The trajectory of the best solution found for the JUICE Test Case 2

IV.III MESSENGER Case Study

The trajectory studied in this section is a particular instance of the MESSENGER mission: the reduced version of the MESSENGER mission. The reduced MESSENGER mission departs from Earth and ends at the first encounter with Mercury, avoiding the following swing-bys with Mercury.

MESSENGER (MErcury Surface, Space Environment, Geochemistry, and Ranging) is a scientific mission to explorer Mercury to understand better its nature and evolution, as well as the high energy processes of the Sun [16]. The sequence swing-by planets for this MGA mission is given by Earth – Earth – Venus – Venus – Mercury – Mercury – Mercury (EEVVMMM) [15]. The mission was launched on 03/08/2004 with 6.6 years of transfer time; the spacecraft was inserted at the final Mercury orbit on 18/03/2011. The first encounter with Mercury was on 14/01/2008, around 3.8 years after the departure date.

This case study is used to assess the sensitivity of the *Physarum* algorithm to the time steps of the launch and arrival windows key parameters. Two different test cases are considered in this case study. Both of the test cases have the same setting parameters, only the launch and arrival windows have different settings, as well as its time steps. Test Case 1 has a launch and arrival windows of 30 days with a time steps of 1 day, while Test Case 2 has wider windows of 360 days with a time steps of 1 day.

IV.III.I Test Case 1

This test case considers a launch window that goes from 2004/07/19 to 2004/08/18 and an arrival window that goes from 2007/12/30 to 2008/01/29, both with a time step of 1 day. The **Table XVII** contains the parameters defining the problem for this particular case. A set of three planets, P_s= {M V E}, is set as available swing-by planets. The maximum total number of swing-bys is four with a maximum of two repeating planets in the same sequence. **Table XVIII** and **Table XIX** contain the lower and upper boundaries on the Time of Flight and the grid spacing $\Delta\theta$.

Table XX shows the best 5 trajectories found by the *Physarum* algorithm after 8,000 function evaluations (the default value of function evaluations). All the found solutions present the same sequence, EEVVM, which is the same as the nominal sequence of MESSENGER. The total ΔV cost varies from 8.62 to 8.72 km/s; and all the transfer time of about 3.4 years. The best found solution has a total ΔV cost of 8.62 km/s, slightly lower ΔV cost than the best solution found by F. Buscani and D. Izzo of 8.639 km/s; published at the ESA Global Trajectory Optimization Problems database [17], and total transfer time of 3.4 years. **Fig. 7** illustrates the best trajectory found in this test case.

Parameter	**Value**
Set of Available Planets, P_s	{M V E}
ns_{max}	4
res_{max}	2
T_{launch}*[yyyy/mm/dd]*	2004/07/19 to 2004/08/18
T_{Step_launch}	1 day
$T_{arrival}$	2007/12/30 to 2008/01/29
$T_{Step\_arrival}$	1 day
$[\Delta v_0^{min}, \Delta v_0^{max} \Delta \mathrm{v_{max}}]$	[1, 12] km/s
$[\Delta v_f^{min}, \Delta v_f^{max} \Delta \mathrm{v_{max}}]$	[5, 12] km/s
h_{low} for { M V E}	[0.05, 0.1, 0.15]
h_{high} for {M V E}	[10, 70, 20]

Table XVII MESSENGER Test Case 1 Problem Definition Parameters

M	V	E	
[30, 200]	[30, 300]	[30, 500]	M
[30, 300]	[30, 300]	[30, 300]	V
[30, 500]	[30, 300]	[30, 500]	E

Table XVIII MESSENGER Test Case 1: Lower and Upper Boundaries of Time of Flight [day]

M	V	E	
0.5	0.5	0.5	M
0.5	0.5	0.5	V
0.5	0.5	0.5	E

Table XIX MESSENGER Test Case 1: Spacing Grid Definition $\Delta\theta$ [deg]

Rank	Cost *(km/s)*	Sequence				
1	8.62	Earth	Earth	Venus	Venus	Mercury
		1667.5	*2032.7*	*2482.7*	*2713.7*	*2923.5*
2	8.70	Earth	Earth	Venus	Venus	Mercury
		1668.5	*2033.7*	*24980.1*	*2711.0*	*2922.5*
3	8.70	Earth	Earth	Venus	Venus	Mercury
		1668.5	*2033.7*	*2472.8*	*2705.5*	*2926.5*
4	8.71	Earth	Earth	Venus	Venus	Mercury
		1662.5	*227.7*	*2480.1*	*2711.0*	*2922.5*
5	8.72	Earth	Earth	Venus	Venus	Mercury
		1662.5	*2027.7*	*2486.9*	*2718.8*	2928.5

Table XX MESSENGER Test Case 1: Sequence Results

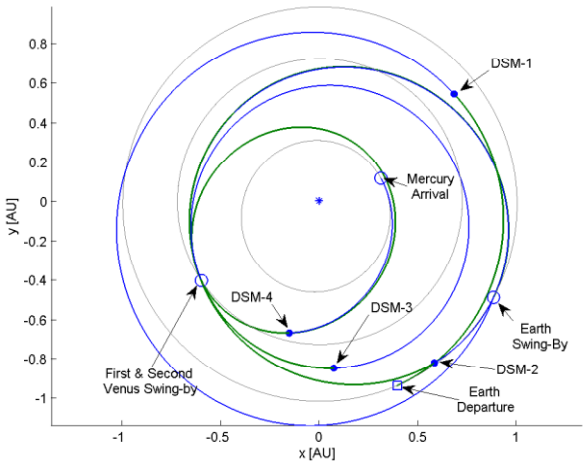

Fig. 7 The trajectory of the best solution found for the MESSENGER Test Case 1

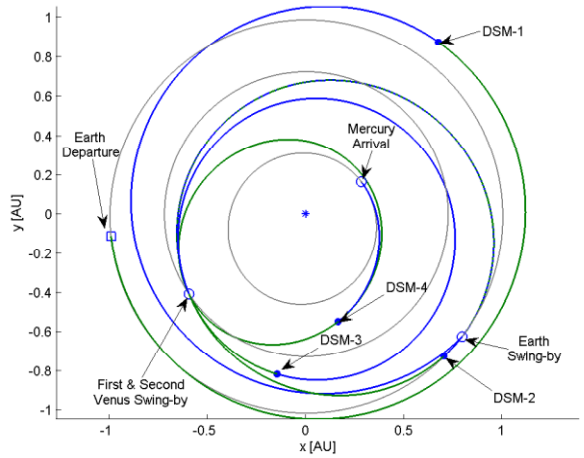

Fig. 8 The trajectory of the best solution found for the MESSENGER Test Case 2

IV.III.II Test Case 2

This specific test case considers a launch and arrival windows of 360 days with a time step of 1 day. The launch window goes from 2004/02/05 to 2005/01/30, and the arrival window goes from 2007/07/18 to 2008/07/12. This test case uses the same problem settings as previous test case 1 (see **Table XVII)**, together with the same lower and upper boundaries of time of flight (see **Table XVIII**) and grid spacing (see **Table XIX**). Additionally, in order to compensate the increment of the launch and arrival windows keeping a very low time step of 1 day, the maximum number of function evaluation is set to 16,000.

From **Table XXII**, it can be seen that even with a larger search space, the *Physarum* algorithm was able to find the same optimal solution of EEVVM. All these new solutions present even a lower ΔV cost than the best solution found on the test case 1. The best found solution has a total ΔV cost of 8.15 km/s which is 5.6% lower than the best know solution [17]. **Fig. 8** illustrates the best trajectory found in this test case.

Parameter	**Value**
T_{launch} *[yyyy/mm/dd]*	2004/02/05 to 2005/01/30
$T_{arrival}$	2007/07/18 to 2008/07/12

Table XXI MESSENGER Test Case 2: Problem Definition Parameters

Rank	Cost *(km/s)*	Sequence				
1	8.15	Earth	Earth	Venus	Venus	Mercury
		1535.5	*2053.7*	*2472.9*	*2704.1*	*2919.5*
2	8.20	Earth	Earth	Venus	Venus	Mercury
		1529.5	*2047.9*	*2466.3*	*2696.2*	*2922.5*
3	8.23	Earth	Earth	Venus	Venus	Mercury
		1527.5	*2045.9*	*2463.3*	*2693.3*	*2910.5*
4	8.25	Earth	Earth	Venus	Venus	Mercury
		1640.5	*2005.7*	*2458.4*	*2690.2*	*2912.5*
5	8.52	Earth	Earth	Venus	Venus	Mercury
		1495.5	*2017.2*	*2467.6*	*2697.2*	2923.5

Table XXII MESSENGER Test Case 2: Sequence Results

V. CONCLUSIONS

This paper introduced a novel bio-inspired algorithm for the automatic planning and scheduling of multi-gravity assist trajectories by translating the design of an MGA transfer into a planning and scheduling process and combining the multi-directional exploration of tree with the incrementally building of it.

The algorithm was applied to three MGA missions to demonstrate its good convergence performance with different type of MGAP. In the first case, the algorithm was applied to an Earth-NEA mission type showing the ability to find, with little parameter tuning, both nominal and optional trajectory of Marco Polo. In addition to the Marco Polo mission, the algorithm was applied to an Earth-Jupiter like transfer (JUICE mission) and to an Earth-Mercury like transfer (MESSENGER mission) providing the ability of the algorithm to find the nominal solutions of the reference missions, as well as to provide better solutions.

In summary, it was shown the ability of the *Physarum* algorithm to find optimal solutions even for large search space with no supervision during the simulation time and with little parameter tuning.

VI. ACKNOWLEDGMENTS

This research was partially supported by a grant from Thales-Alenia Space France and CNES.

VII. RENFERENCES

[1] A.E. Petropoulos, J.M. Longuski, E.P. Bonfiglio. "Trajectories to Jupiter via Gravity Assist from Venus, Earth and Mars." *Journal of Spacecraft and Rockets*, 2000: 37(6):776-783.

[2] S. Pessina, S. Campagnola, M. Vasile. "Preliminary Analysis of Interplanetary Trajectories with Aerogravity and Gravity Assist Manoeuvres." *54th International Astronautical Congress, Bremen, Germany*, 2003.

[3] M. Vasile, P. De Pascale. "On the Preliminary Design of Multiple Gravity-Assist Trajectories" *Journal of Spacecraft and Rockets*, 42(4): 794–805, 2006.

[4] M. Ceriotti, M. Vasile. "Automated MGA Trajectory Planning with an ACO-inspired Algorithm" *Acta Astronautica*, 67(910): 1202-1217, 2010.

[5] A. Gad, O. Abdelkhalik. "Hidden Genes Genetic Algorithm for Multi-Gravity-Assist Trajectory optimization" *Journal of Spacecraft and Rockets*, 48(4): 629–641, 2011.

[6] J.A. Englander, B.A. Conway, T. Williams. "Automated Interplanetary Trajectories Planning" *AIAA/AAS Astrodynamics Specialist Conference, Minneapolis, Minnesota*, 2012.

[7] A. Adamatzky, G.J. Martinez, S.V. Chapa-Vergara, R. Asomoza-Palacio, C.R. Stephens. "Approximating Mexican Highways with Slime Mould" *Natural Computing*, 10(3): 1195–1214, 2011.

[8] D.S. Hickey, L.A. Noriega. "Insights into Information Processing by the Single Cell Slime Mold Physarum Polycephalum" *UKACC Control Conference, Manchester, UK*, 2008.

[9] T. Nakagaki, H. Yamada, A. Toth. "Maze-Solving by an Amoeboid Organism" *Nature*, 407(6803): 470, 2000.

[10] A. Tero, K. Yumiki, R. Kobayashi, T. Saigusa, T. Nakagaki. "Flow-Network Adaptation in Physarum Amoebae" *Theory in Biosciences*, 127(2): 89–94, 2008.

[11] A. Tero, R. Kobayashi, T. Nakagaki. "Physarum Solver: a Biologically Inspired Method of Road-Network Navigation" *Physica: A Statistical Mechanics and its Applications*, 363(1): 115–119, 2006.

[12] M. Khan, P. de Pascale, J. Schoenmaekers "Marco Polo: Consolidated Report on Mission Analysis (CReMA)" *European Space Agency*, 2009

[13] O. Grasset, M.K. Dougherty, A. Coustenis, E.J. Bunce, C. Erd., D. Titov "Jupiter Icy moons Explorer (JUICE): an ESA mission to orbit Ganymede and to characterize the Jupiter system" *Planetary Pioneers Series* 78, 1-21, 2013

[14] A. Boutonnet, J. Schoenmaekers "JUICE: Consolidated Report on Mission Analysis (CReMA)" *European Space Agency*, 2012

[15] J. McAdams, D. Dunham, R. Farquhar, "Trajectory Design and Maneuver Strategy for the Messenger Mission to Mercury"*AIAA Space Flight Mechanics Conference*, AAS-05-173 2005

[16] B. Page, C. Bryan, E. Williams, A. Taylor, D. Stranbridge, P. Wolff, N. Williams "Messenger Navigation Operation During Mercury Orbital Mission Phase" *AAS 13/383*

[17] T. Vinko, D. Izzo "Global Optimization Heuristics and Test Problems for Preliminary Spacecraft Trajectory Design" *ACT Technical Report* ACT-TNT-INF-2008-GOHTPPSTD, 2008