AAS 19-406

# A RELAXATION APPROACH FOR HYBRID MULTI-OBJECTIVE OPTIMAL CONTROL: APPLICATION TO MULTIPLE DEBRIS REMOVAL MISSIONS

**Lorenzo A. Ricciardi*and Massimiliano Vasile†**

This paper presents a novel approach to the solution of multi-phase multi-objective hybrid optimal control problems. The proposed solution strategy extends previous work which integrated the Direct Finite Elements Transcription (DFET) method to transcribe dynamics and objectives, with a memetic strategy called Multi Agent Collaborative Search (MACS). The problem is reformulated as two non-linear programming problems: a bi-level and a single level one. In the bi-level formulation the outer level, handled by MACS, generates trial control vectors that are then passed to the inner level, which enforces the feasibility of the solution. Feasible control vectors are then returned to the outer level to evaluate the corresponding objective functions. The single level formulation is also run periodically to ensure local convergence to the Pareto front. In order to treat mixed integer problems, the heuristics of MACS have been modified in order to preserve the discrete nature of integer variables. For the single level refinement and the inner level of the bi-level approach, discrete variables are relaxed and treated as continuous. Once a solution to the relaxed problem has been found, a smooth constraint is added to systematically force the relaxed variables to assume integer values. The approach is first tested on a simple motorised travelling salesmen problem and then applied to the mission design of a multiple debris removal mission.

## INTRODUCTION

This paper proposes a method for the solution of multi-objective hybrid optimal control problems. The method is based on a direct transcription with Direct Finite Elements in Time (DFET) and a solution of the resulting multi-objective nonlinear programming (NLP) problem with a version of Multi Agent Collaborative Search (MACS) that implements a dual level optimisation and a single level optimisation through a Pascoletti-Serafini scalarization.

In DFET, the time domain is divided into a given number of segments, or elements, and the states and controls are approximated as arbitrary order polynomials on a given basis. DFET was successfully used to solve many difficult trajectory optimisation problems[1,2] and is a very robust transcription method with many interesting characteristics.[3–6]

MACS[7] is a memetic evolutionary multi-objective optimisation algorithm in which a population of agents is initially seeded in the search space with a Latin Hypercube sampling. All agents can perform a set of individual actions to explore their neighborhood and improve their position. Non-dominated solutions are saved to an external archive. Some agents can then perform social actions,

*Ph.D. Candidate, Aerospace Centre of Excellence, Mechanical & Aerospace Engineering, 75 Montrose Street, G1 1XJ, Glasgow, AIAA Student Member.

†Professor, Aerospace Centre of Excellence, Mechanical & Aerospace Engineering, 75 Montrose Street, G1 1XJ, Glasgow, AIAA Member.

exploiting the information coming from the archive or from other agents to collectively advance towards the Pareto front. Individual and social actions are repeated until a maximum number of function evaluations is reached. A special archiving strategy ensures a good distribution of points across the Pareto front.

MACS and DFET have been coupled by employing two different but synergistic formulations of the original Multi-objective optimal control problems, a bi-level formulation and a single level one. In the bi-level formulation the outer level, handled by MACS, generates trial control vectors that are then passed to the inner level, which enforces the feasibility of the solution by solving an NLP. Feasible control vectors are then returned to the outer level to evaluate the corresponding objective functions. The bi-level formulation is thus able to perform a global exploration of the search space and return uniformly spread solutions in criteria space. The single level formulation is also run periodically to ensure local convergence to the Pareto front. This synergistic approach has already been applied to complex test cases focusing on the optimal control of nominal ascent, re-entry and abort trajectory options for an airborne spaceplane launch system.[8]

The problems analysed so far with this approach were described by real variables only. Real world problems can also depend on some decisions which can only assume a countable number of possible values. The introduction of discrete decision variables results in a mixed integer problem, which is significantly more complex than a purely continuous or purely discrete problem. In case of an optimal control problem, a change of value of a discrete variable can completely alter the switching structure of the controls or even the number or the sequence of phases of the problem.

There is a vast amount of literature on the solution of mixed integer problems, but essentially there are two main approaches to solve them. The first class, called relaxation methods, treats discrete variables as if they were continuous and then employs some technique to restore the integer nature of the variables that were relaxed. The other class instead does not relax the discrete variables, thus usually resulting in a separate treatment of the continuous and discrete variables. The problem can then be tackled through a deterministic approach or through a stochastic approach.

Among the deterministic approaches, the most common method of the first class is the Branch and Bound algorithm,[9] while for the second kind there are the Outer Approximation method,[10,11] the Generalised Benders Decomposition,[12] or the use of deterministic discrete optimisation algorithms like the Mesh Adaptive Descent Method (MADS).[13,14] Stochastic approaches can ether be natively developed for discrete variables or be adapted to treat them by modifying the heuristics they employ, thus do not need any relaxation.

For mixed integer optimal control with general nonlinear dynamics and objetives, in the past have been proposed approaches coupling the Branch and Bound method with NLP solvers,[15,16] or coupling stochastic algorithms and NLP solvers.[17,18]

In the Branch and Bound based methods, the problem is initially relaxed and solved as purely continuous. This solution is used as the lower bound for the optimal solution of the problem, while a feasible solution for the non relaxed problem is used as an upper bound. In case all the relaxed variables assume integer values, the optimal solution has already been found. However, this will not happen in most of the cases, so one discrete variable will have to be branched, meaning that two different problems will be solved by imposing its value to be either $0$ or $1$, and leaving the other variables relaxed. Two new NLPs will be solved, which will update the lower bound and the upper bound for the solution. In case the lower bound of one branch is higher than the upper bound for the other, that branch cannot contain the optimal solution and is thus not further investigated.

This way, a finite, albeit potentially very large number of steps is produced that converges to the optimal solution. However, as also shown by Glocker,[15] it must be highlighted that for non convex problems it can happen that for the same values of the discrete variables there exist multiple locally optimal solutions with different values for the objective function. This can introduce the possibility that the branching procedure erroneously discards promising areas of the search space.

The method proposed by Englander[17] consists of a dual loop algorithm in which the outer loop solves a multi-objective problem handling a set of categorical variables through the multi-objective genetic algorithm NSGA-II,[19] and the inner loop solves a set of single objective constrained optimal control problems using Monotonic Basin Hopping.[20] Since the resulting optimal control problem was optimising only one objective (the mass of the vehicle), the local optimality with respect to the other objectives was not guaranteed. In addition, while a global search was performed for the discrete parameters, the continuous parameters were treated only locally by the NLP solver. As for the Branch and Bound based approaches, this can result in multiple local solutions for a given choice of the discrete parameters. This problem was partially addressed by using a Monotonic Basin Hopping method (MBH) in order to overcome this limitation.

The method proposed by Schlueter[18] instead was based on MIDACO,[21] a single objective Ant Colony Optimisation algorithm, coupled with an NLP solver. The MIDACO solver was treating both discrete and continuous variables without any relaxation, and the constraints were treated with a new penalisation scheme. However, the satisfaction of the constraints was not required to be strict in this global exploration phase. The MIDACO solver was allowed to run for a fixed maximum runtime, and the best solution found was then refined by an NLP algorithm.

Although the approach produced good results in the tests, there is no guarantee that the best weakly feasible solution found by MIDACO will not be significantly worsened by the NLP solver, which has to strictly enforce the constraints. Moreover, since the NLP does not modify the discrete variables, there is no guarantee that for a given choice of the discrete variables a fully feasible solution exists.

This work focuses on optimal control problems with a fixed number of phases. It will show how the existing method can be extended to deal with problems presenting discrete static variables. To deal with mixed integer variables with the approach proposed in this work, the heuristics implemented in MACS were modified in order to treat discrete variables. Thus, at the outer level, discrete variables remain discrete. Similarly to the approach proposed by Schlueter, all variables are handled simultaneously by the outer level global optimiser. However, the inner level will try to strictly satisfy the constraints.

In order to allow the maximum flexibility to the inner level NLP, the discrete variables are relaxed within the inner level. This gives the NLP the possibility of modifying even the discrete variables. This is particularly helpful, since it could be impossible for the NLP to satisfy some constraints for a fixed choice of the discrete variables, but this impossibility might require a large number of function evaluations before the NLP solver surrenders. Once the NLP has converged to a relaxed feasible solution, an additional set of constraints is imposed to enforce the relaxed variables to assume only discrete values, and the inner level NLP is solved again starting from the relaxed solution. This approach allows to strictly satisfy challenging mixed integer constraints and return fully feasible solutions at the outer level. When the single level gradient based refinement is invoked, it starts from a fully feasible solution and has thus a good chance of improving the guess it received.

Different numerical test cases will be shown. The first one is a multi-objective extension of the

motorised travelling salesmen problem (TSP). The second test case is a more complex version of the first, where the destinations visited by the travelling salesman do not have a fixed position in time. Finally, a relevant and challenging space application will be tackled: a multiple destination spacecraft trajectory optimisation typical of multi-debris removal missions or multiple target on-orbit servicing missions. Although the test problems have a small scale in terms of combinatorial complexity, the method is limited only by the size of the problems that the NLP can solve.

## PROBLEM FORMULATION

Multi-objective Hybrid Optimal Control problems can be formulated as:

$$\begin{aligned} &\min_{\mathbf{u}\in U,\boldsymbol{\sigma}\in\sigma,\mathbf{b}\in B,\boldsymbol{\mu}\in M} \mathbf{J} \\ &s.t. \\ &\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t) \\ &\mathbf{g}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t) \geq 0 \\ &\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\boldsymbol{\sigma}(t_0),\boldsymbol{\sigma}(t_f),\mathbf{b},\boldsymbol{\mu},t_0,t_f) \geq 0 \\ &t \in [t_0,t_f] \end{aligned} \tag{1}$$

where $\mathbf{J} = [J_1, J_2, ..., J_i..., J_m]^T$ is, in general, a vector of objectives $J_i$ that are functions of the state vector $\mathbf{x} : [t_0, t_f] \rightarrow \mathbb{R}^{n_x}$, continuous control variables $\mathbf{u} \in L^\infty(U \subseteq \mathbb{R}^{n_u})$, discrete control variables $\boldsymbol{\sigma} \in \sigma \subseteq \mathbb{Z}^{n_\sigma}$, continuous static parameters $\mathbf{b} \in B \subseteq \mathbb{R}^{n_b}$, discrete static parameters $\boldsymbol{\mu} \in M \subseteq \mathbb{Z}^{n_z}$ and time $t$. The functions $\mathbf{x}(t)$ belong to the Sobolev space $\mathcal{W}^{1,\infty}$ while the objective functions are $J_i : \mathbb{R}^{3n_x} \times \mathbb{R}^{n_u} \times \mathbb{Z}^{n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times \mathbb{R}^2 \longrightarrow \mathbb{R}$. The objective vector is subject to a set of dynamic constraints with $\mathbf{F} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{Z}^{n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times [t_0, t_f] \longrightarrow \mathbb{R}^{n_x}$, algebraic constraints $\mathbf{g} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{Z}^{n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times [t_0, t_f] \longrightarrow \mathbb{R}^{n_g}$, and boundary conditions $\boldsymbol{\psi} : \mathbb{R}^{2n_x} \times \mathbb{R}^{2n_u} \times \mathbb{Z}^{2n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times [t_0, t_f]^2 \longrightarrow \mathbb{R}^{n_\psi}$, where $\mathbf{F}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t)$, $\mathbf{g}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t)$ and $\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\mathbf{b},\boldsymbol{\mu},t_0,t_f)$ are vector fields.

When $N_p$ phases are present, dynamic constraints, path and boundary constraints, and objective functions are defined on each timeline. In order to connect different timelines, a set of $N_{ip}$ inter-phase constraints are introduced:

$$\psi_{s_p}\left(\mathbf{x}_{0,\mathbf{I}_{s_p}},\mathbf{x}_{f,\mathbf{I}_{s_p}},\mathbf{u}_{0,\mathbf{I}_{s_p}},\mathbf{u}_{f,\mathbf{I}_{s_p}},\boldsymbol{\sigma}_{0,\boldsymbol{I}_{s_p}},\sigma_{f,\mathbf{I}_{s_p}},\mathbf{b}_{\mathbf{I}_{s_p}},\boldsymbol{\mu}_{\mathbf{I}_{s_p}},t_{0,\mathbf{I}_{s_p}},t_{f,\mathbf{I}_{s_p}}\right) \geq 0 \quad s_p = 1, ..., N_{ip} \tag{2}$$

where the index vector $\mathbf{I}_{s_p}$ collects all the indexes of the phases that are connected by constraint $\psi_{s_p}$. Note that the number of phases is fixed, but their temporal order is actually defined by the inter-phase constraints (2).

The problem can be discretised following the procedure for DFET transcription illustrated by Ricciardi and Vasile.[8] This results in a Multi Objective Mixed Integer Nonlinear Programming (MOMINLP) problem coming from the transcription of problem (1), with the inclusion of inter-phase constraints (2). In vector form it can be written as:

$$\begin{aligned} &\min_{\mathbf{y}\in Y,\mathbf{p}\in\Pi,\mathbf{d}\in D} \tilde{\mathbf{J}} \\ &s.t. \\ &\mathbf{C}(\mathbf{y},\mathbf{p},\mathbf{d}) \geq 0 \end{aligned} \tag{3}$$

where $\mathbf{y}$ collects all the discretised state variables, $\mathbf{p}$ collects all the static and discretised dynamic control variables, $\mathbf{d}$ collects all the integer valued control variables and static parameters, and $\mathbf{C}$ collects all constraints, including boundary and interphase conditions.

## SOLUTION OF THE MIXED INTEGER OPTIMAL CONTROL PROBLEM

To solve Multi-Objective Optimal Control problems, Ricciardi and Vasile[8] proposed an algorithm combining two complementary approaches. The first one consists of a further reformulation of the multi-objective NLP into a bi-level optimisation problem. At the outer level, MACS uses its heuristics to generate trial solution vectors. This trial solution vector is received by the inner level, which uses it as a first guess to enforce the constraints through an NLP solver. Once the NLP converges, the solution is passed back to the outer level, which evaluates the objective functions and employs the dominance criterion and the Chebychev scalarisation to decide weather the solution is better than the existing ones. This provides global exploration capabilities and a good spreading of the solutions on the Pareto front.

In order to guarantee local optimality, a single level gradient based refinement step takes place every given number of iterations. This single level refinement reformulates the problem as a scalarised version of the original multi-objective problem, and allows a simultaneous handling of feasibility and optimality. Very importantly, the Chebychev scalarisation criterion used in the bi-level approach is consistent with the Pascoletti-Serafini scalarisation employed in the single level approach.

This approach is here extended in order to treat mixed-integer problems. For the outer level, treated with MACS, all variables were tagged as either real or discrete. The heuristics of MACS have been modified in order to guarantee that after their application the value of the discrete variables was integer and within the allowed bounds, while leaving unchanged the behaviour of the heuristics when operating on the continuous variables.The next section contains a detailed explanation of the modifications performed on the heuristics.

For the inner level and for the single level gradient based approach instead the discrete variables were relaxed in order to work with the NLP solver. This way the inner level is able to change the value of the relaxed variables in order to get a feasible solution. The following subsection explains in greater detail these modifications.

### Modification of the Heuristics of MACS

In order to preserve the behaviour and performances of the algorithm, the heuristics of MACS have been modified in the simplest, least invasive way possible. This means that no additional heuristic was introduced to deal with the discrete variables. In the following, the modifications applied to the *Inertia, Pattern Search* and *Differential Evolution* heuristics are explained in detail.

*Inertia.* The idea of this heuristic is to proceed searching along the same promising direction found at the previous iteration. The behaviour of this heuristic is thus strongly dependant on the heuristic which found the promising search direction at the previous iteration, i.e. either the Pattern Search or the Differential Evolution. However, since inertia produces a sample with a random step length along the chosen direction, it could result in the generation of non integer values for the discrete variables. For this reason, the inertia move is applied normally, but the components of the displacement vector corresponding to the discrete variables are rounded towards the closest integer. In case the rounding towards the closest integer would result in a zero displacement vector, a rounding towards the next larger integer is performed instead.

*Pattern Search.* This heuristic simply changes one component of the solution vector at a time, by a random amount:

$$x_{trial,j} = x_{ij} + \alpha \Delta_j \rho_i \tag{4}$$

where the component $j$ of the solution vector $x_i$ is perturbed by a random amount $\alpha$ scaled by the size of the interval for variable $j$ $\Delta_j$, and by a contraction factor $\rho_i$, which is decreased or increased at each iteration depending on weather a better solution is found or not.

In case this heuristic is currently operating on a discrete variable, the modification consists in rounding the perturbation $\alpha\Delta_j\rho_i$ to the closest integer. Similarly to the Inertia case, if this rounding would result in a zero length move, a rounding towards the next integer is performed instead.

*Differential Evolution.* This heuristic simultaneously changes all the components of the solution vector by combining it with other three solution vectors:

$$\mathbf{x}_{trial} = \mathbf{x}_i + \alpha\mathbf{e}\left((\mathbf{x}_i - \mathbf{x}_{i_1}) + F\left(\mathbf{x}_{i_2} - \mathbf{x}_{i_3}\right)\right) \tag{5}$$

where solution vector $x_i$ is perturbed by a random amount $\alpha$ scaled by a displacement vector generated by three randomly chosen solution vectors $x_1, x_2$ and $x_3$, which must all be different, and multiplied by a mask vector $\mathbf{e}$ which randomly assumes component values of $0$ or $1$.

In order to ensure that the discrete variables remain discrete, the same procedure for the Inertia case is adopted: the components of the displacement vector associated to the discrete variables are rounded towards the closest integer. In case this rounding would result in a zero displacement, a rounding by excess would be performed instead.

### Relaxation and Integrality constraints

Within the inner level and the single level refinement the discrete variables are relaxed and treated as continuous. This will likely generate a solution where the relaxed variables do not assume integer values. After a relaxed solution is found, the NLP solver is invoked again but the following constraint is imposed in order to force it to converge to solutions where the relaxed variables assume integer values:

$$\sin(\pi\tilde{\sigma}_j) = 0 \tag{6}$$

where $\tilde{\sigma}_j$ is a relaxed variable.

This simple and smooth constraint is satisfied only for integer values of the relaxed variable $\sigma_j$. However, since this constraint has many possible solutions, it can introduce several local optima. For this reason it is enforced only after a solution to the relaxed problem is found: the NLP solver is called again starting from the relaxed solution, but the constraint (6) is enforced. The delayed imposition of constraints (6) should allow the NLP solver to avoid the local minima introduced by this constraint and converge to the integer value closest to the relaxed solution.

### Treatment of combinatorial constraints

An interesting problem arises when an optimal control problem is posed to compute the optimal trajectory linking several targets but the order in which those targets must be visited can itself be optimised. This situation is typical of the class of problems known as Travelling Salesman Problem or Vehicle Routing Problem, but can occur also in the design of space missions.

Those problems can be formulated as multi-phase optimal control problems, where the boundary conditions of each phase can be expressed as:

$$\mathbf{x}(t_f; i) = \sum_j \mu_{i,j}\mathbf{P}_j \tag{7}$$

where $\mathbf{x}(t_f; i)$ indicates the boundary conditions at the end of phase $i$, $\mu_{i,j}$ are static optimisation variables which can only assume a value of 0 or 1, and $\mathbf{P}_j$ are the target boundary conditions. Variables $\mu_{i,j}$ can be considered as a choice on the target associated to each phase. In facts, if $\mu_{1,j} = 1$ while all other $\mu_{i,j} = 0$ the constraint will impose phase $i$ to terminate at the position of target $P_1$. However, exactly one $\mu_{i,j}$ per phase must be equal to one. In addition, it may be required to visit each target exactly once. This particular set of mutual exclusivity constraints needs a special treatment.

Von Stryk and Glocker[22] proposed to arrange the discrete variables $\mu_{i,j}$ in a matrix, relax the variables $\mu_{i,j}$ into $\tilde{\mu}_{i,j}$ and impose the following set of constraints

$$\begin{cases} \sum_i \tilde{\mu}_{i,j} = 1 \ \forall j \\ \sum_j \tilde{\mu}_{i,j} = 1 \ \forall i \end{cases} \tag{8}$$

where $\tilde{\mu}_{i,j}$ is the relaxed $i^{th}$ binary choice for phase $j$. For an $n$ targets problem, this results in a total of $2n$ constraints. The idea behind this set of constraints is to ensure that, if the variables assume a value of 0 or 1, only one element per row and column can be 1, thus resulting in a unique choice of target per each phase.

In the work of Von Stryk and Glocker, these constraints were employed to obtain a relaxed solution which constituted the upper bound for a branch and bound method: one relaxed variable was then split into a branch where it had a fixed value of 0 and a value of 1 in the other. The branch and bound method progressed until all relaxed variables were given a value of either 0 or 1, progressively discarding regions of the search space which were considered unpromising.

This method has the advantage of not requiring any further special treatment for the discrete variables, which are treated separately to the continuous ones. However, it has two main disadvantages: the first is that in order to obtain a solution to the full non-relaxed problem it has to solve many NLPs, in the best case equal to twice the number of binary variables and in the worst case equal to performing a full enumeration of the $2^n$ possible combinations.

The second, less obvious disadvantage, is that the method relies on the NLP to provide an upper bound for the solution. If the problem has multiple local solutions, the upper bound computed by the NLP might be incorrect and the method might discard the region of the search space where the true optimal solution lays. This will be further discussed in the results section.

Unfortunately, the set of constraints (8) has two major problems when applied in the current framework: the first one is that it is composed by $2n$ equality constraints. When adding constraints (6) to enforce each $\tilde{\mu}_{i,j}$ to assume a value of 0 or 1, the problem becomes overdetermined for the $\tilde{\mu}_{i,j}$, given that constraints (6) already impose an equality constraint per relaxed variable. Even if the constraints are redundant at the solution points, the NLP solver might have serious difficulties in handling this situation.

A second less obvious problem is shown in Figure 1: one should guarantee that the $\tilde{\mu}_{i,j}$ involved in each of constraints (8) are not all greater, or all less, of 0.5. The reason for this is that, in order to satisfy the constraints, the NLP solver performs a linearisation of the constraints. Thus, it will try to satisfy the constraints by pushing all the $\tilde{\mu}_{i,j}$ towards either 0 or 1. However, this conflicts with the constraints $\sum_i \tilde{\mu}_{i,j} = 1$. In other words, with a poor guess it might be impossible for the NLP to find a feasible solution to an otherwise simple problem.

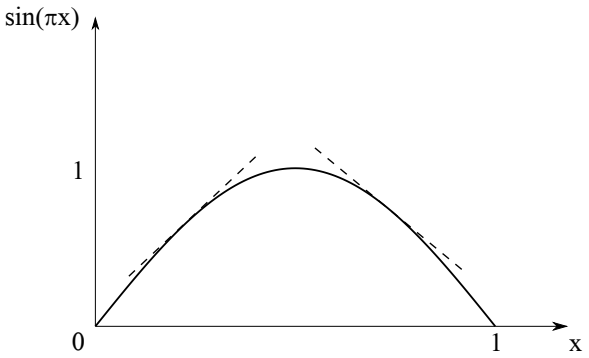



**Figure 1**: Graphic representation of constraints 6

In order to overcome all these obstacles, another approach is proposed. First, equality constraints (8) are rewritten as inequality constraints

$$\begin{cases} \sum_i \tilde{\mu}_{i,j} \leq 1 + \epsilon \ \forall j \\ \sum_i \tilde{\mu}_{i,j} \geq 1 - \epsilon \ \forall j \\ \sum_j \tilde{\mu}_{i,j} \leq 1 + \epsilon \ \forall i \\ \sum_j \tilde{\mu}_{i,j} \leq 1 - \epsilon \ \forall i \end{cases} \tag{9}$$

where $\epsilon$ is a fixed positive parameter lower than 1. This set of linear inequality constraints solves the issue of overdetermination of the system, because the only equality constraints remaining are from Eq. (6). Moreover it creates a feasible region of thickness $2\epsilon$ around the hyperplane defined by the constraints (8), which should be easier to satisfy than a strict equality constraint.

However, it is still possible that all elements $\tilde{\mu}_{i,j}$ of a row or column are on the same side of the constraint (6), i.e. they are all lower or greater than $0.5$. In order to solve this issue, an additional set of constraints is imposed:

$$\begin{cases} \sum_i \left( \tilde{\mu}_{i,j}^2 - \frac{1}{2} \right) \geq r^2 \ \forall j \\ \sum_j \left( \tilde{\mu}_{i,j}^2 - \frac{1}{2} \right) \geq r^2 \ \forall i \end{cases} \tag{10}$$

where $r$ is a parameter that can be chosen between $r_{min} = \frac{1}{2}$ and $r_{max} = \frac{\sqrt{n}}{2}$ and $n$ is the number of targets. For each row and column equality of Eq. (8) this additional set of constraints is creating a keep out sphere of radius $r$ centred around the point $\left(\frac{1}{2}, \cdots, \frac{1}{2}\right)$. As shown in Figure 2 for the $n = 2$ case, this combination of constraints (9) and (10) has a feasible region of finite size around the vertices of a unit hypercube that also le along the hyperplane parallel to the vector $(1, 1, \cdots, 1)$. Thus, when the integrality constraints (6) are imposed, the NLP solver has a good initial guess and can converge more easily towards a fully feasible solution.

It is important to remark that the feasible region of constraints (9) and (10) is disconnected and has a number of islands equal to the number of possible sequences. Thus, many different feasible

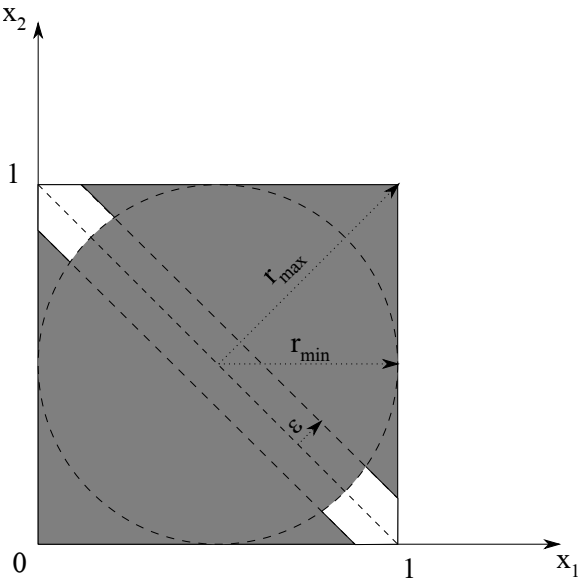



**Figure 2**: Feasible region of constraints 9 and 10

and locally optimal solutions are possible and a global exploration approach is necessary.

## TEST CASES

This section presents three test cases. Since the main goal of this paper is to perform first tests on the validity of the proposed approach, all the test cases are very simple in terms of combinatorial complexity. For the same reason, the first test case is is a multi-objective extension of a single objective problem already solved in the literature. This allows to directly compare the solutions obtained by the proposed approach with the known solutions.

In the first test case, referred to in the literature as a motirised Travelling Salesmen Problem, a vehicle has to visit several targets whose order is not specified. The second test case is an extension of the first, where the targets to visit do not have a fixed position over time. The third is a space application, where a spacecraft has to rendez-vous with multiple targets in order to deorbit or perform on-orbit servicing, and again the order in which the targets are visited is not given a priori.

The algorithm was implemented in Matlab® 2017b and run on a Linux workstation with 8 GB of RAM and an Intel i7-4790 cpu.

### A motorised Travelling Salesmen Problem

This section describes a multi-objective extension of a problem presented in:[22] a vehicle, described by a simple two dimensional dynamic model, starts from the origin of the plane and has to visit three target destinations before finally returning to its starting position. It is controlled by the magnitude of the acceleration and by the steering rate, both of which are limited. The order in which the three targets are to be visited is not specified a priori but has to be found as part of the solution, which in the original reference had to minimise the total mission time. The vehicle has to pass on each target point without any restriction on the velocity or direction of velocity at the rendez-vous, while at the final time it had to be at rest at the original position. This problem can be seen as an

extension of a classic Travelling Salesman Problem, where the presence of a dynamical model for the Salesmen significantly changes the nature of the problem and its complexity.

The objectives for this problem are the minimisation of the total time for the tour, and the minimisation of the energy required:

$$\min_{t_f,\mathbf{u},\boldsymbol{\mu}} (J_1, J_2)^T = \left( t_f, \int_{t_0}^{t_f} u_1^2 dt \right)^T \tag{11}$$

The dynamics of the vehicle are given by:

$$\begin{cases} \dot{x} &= v\cos(\alpha) \\ \dot{y} &= v\sin(\alpha) \\ \dot{v} &= u_1 \\ \dot{\alpha} &= u_2 \end{cases} \tag{12}$$

where $x$ and $y$ denote the position of the vehicle on the plane, $v$ the magnitude of its velocity and $\alpha$ the direction of the velocity vector. The vehicle is controlled by the magnitude of the acceleration $u_1$ and by the steering rate $u_2$, both of which are limited: $u_1^2 \leq 1$, $u_2^2 \leq 1$.

The vehicle starts at the origin at rest, and has to pass on 3 targets, whose location is

$$\begin{cases} P_1 = & (1,2) \\ P_2 = & (2,2) \\ P_3 = & (2,1) \end{cases} \tag{13}$$

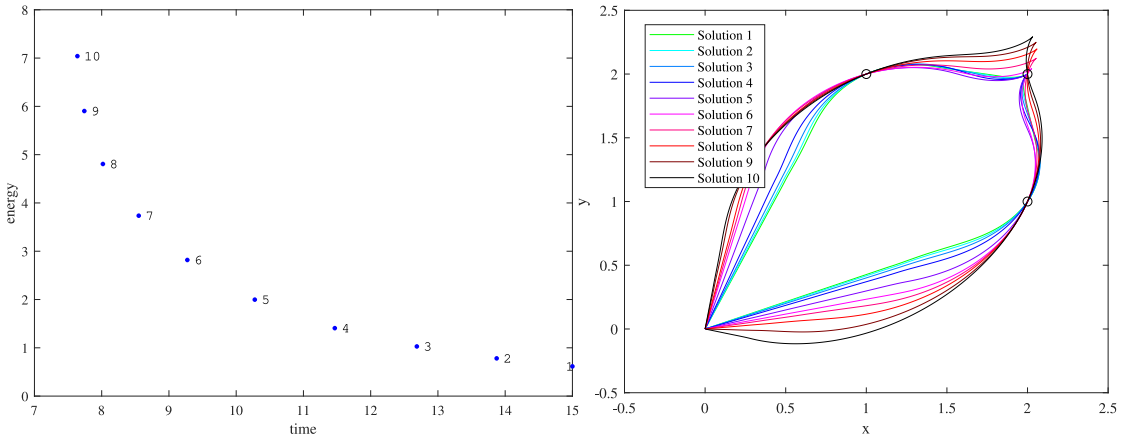
The problem was formulated as a 4 phase problem, with the last phase targeting the origin. Matching conditions were imposed between the states of the various phases, and final conditions for each phase were imposed as per Eq. (7). The problem was discretised using 3 DFET elements of order 7 for both states and controls and each phase, resulting in a problem with 638 variables. Bounds for the optimisation variables are reported in Table 1.

**Table 1**: Lower and upper bounds for the state, control and final time variables of the motorised Travelling Salesman Problem

| Variable | Lower bound | Upper bound |
|---|---|---|
| $x$ (m) | $-5$ | 5 |
| $y$ (m) | $-5$ | 5 |
| $v$ ($\mathrm{m\,s^{-1}}$) | $-10$ | 10 |
| $\alpha$ (rad) | $-\pi$ | $\pi$ |
| $u_1$ (#) | $-1$ | 1 |
| $u_2$ ($\mathrm{rad\,s^{-1}}$) | $-1$ | 1 |
| $t_f$ (s) | 0 | 15 |

The algorithm was run with 10 agents for a total of 40000 function evaluations with standard settings, and the single level gradient based refinement every 10 iterations. 10 solutions were kept into the archive. Figure 3 shows the Pareto front and the trajectories computed. Lower time solutions require more energy, as expected. The order in which the target locations are visited is the same

for all the trajectories, starting from $P_3$ and proceeding to $P_2$ and $P_1$. Rather unexpectedly, lower time solutions have longer trajectories with a marked cusp, while longer time solutions have shorter quasi rectilinear trajectories.



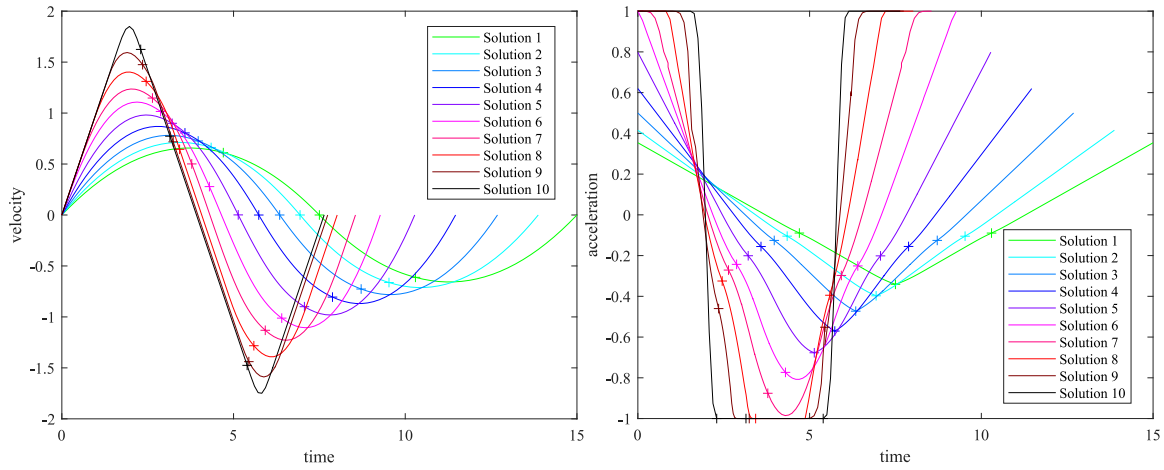

(a) Pareto front (b) Trajectories

**Figure 3**: Pareto front and Trajectories for the motorised Travelling Salesman Problem

To explain this unintuitive result it is useful to look at Figure 4, which shows the time histories of the magnitude of the velocity and the corresponding control, the acceleration. As expected, the minimum time trajectories are the result of a bang-bang control profile for the accelerations. In this case, the bang-bang switch happens twice: a full acceleration is followed by a full deceleration until velocity reaches zero. After that, the acceleration is still kept negative, meaning that the vehicle will proceed backwards, until a full acceleration will stop the vehicle again when it reaches the origin. The reason behind this unintuitive backwards arc is due to the limitations in the steering rate: the maximum steering rate is not sufficient to allow the vehicle to smoothly turn and visit all three targets while also accelerating. Thus, in order to reduce time, the optimiser found a solution which is overcoming this limitation.

Low energy solutions instead have a smooth double linear profile, corresponding to a milder acceleration/deceleration profile. Also for these solutions the vehicle proceeds backwards after the second target. In particular, the vehicle stops at that point even if this was not imposed by the boundary conditions. Since the velocities are lower for these solutions, a smaller curvature radius is achievable, resulting in rectilinear trajectories for most of the interval connecting two targets.

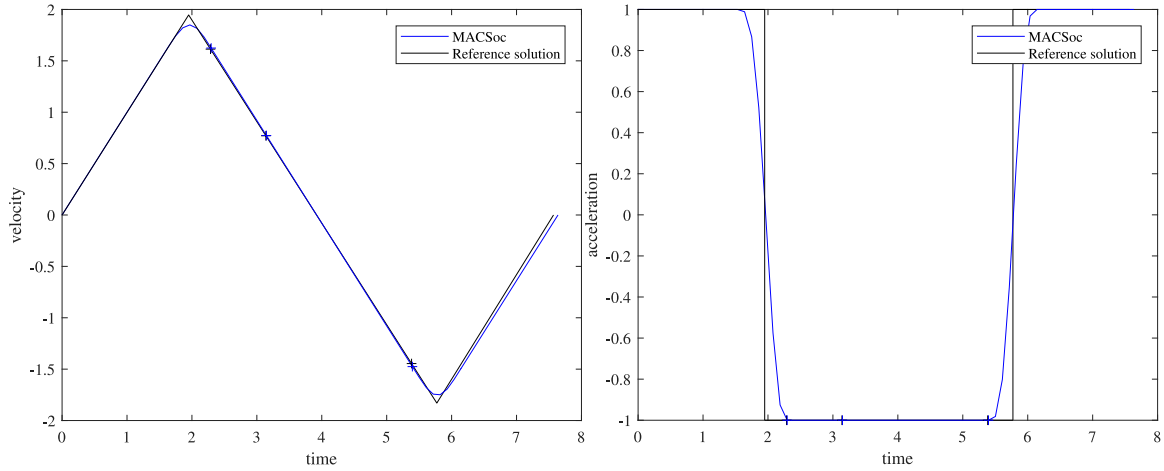
Solution 10 can be compared with the solution found in the reference,[22] which was solving the minimum time problem. Table 2 reports a comparison of the times when each solution visits each target and the total mission time. The difference in total mission time is below $0.3\%$ and can be attributed to the fact that the DFET transcription with Bernstein polynomials smooths the sharp discontinuities of bang-bang profiles, resulting in a slight penalisation of the objective function. This is also evident from Figure 5, which compares the reference solution with the solution obtained with the proposed method and shows a very good agreement. It is expected that with some iterations of mesh refinement, the bang-bang profiles of the solutions will be more sharp, thus decreasing the value of the mission time and evantually reaching the same value of the solution present in the literature.




(a) Time history of the velocity

(b) Time history of the acceleration

**Figure 4**: Time histories of velocity and acceleration for the Travelling Salesman Problem. $+$ marks when a target is visited




(a) Time history of the velocity

(b) Time history of the acceleration

**Figure 5**: Comparison of the solution obtained by current method and solution of Von Stryk and Glocker.[22] $+$ marks when a target is visited

**Table 2**: Comparison of intermediate and final times between the reference solution and solution 10 on the Pareto front

| Solution | $T(P_1)$ | $T(P_2)$ | $T(P_3)$ | $T_f$ |
|---|---|---|---|---|
| Reference | 5.3830 (s) | 3.1390 (s) | 2.286 (s) | 7.6166 (s) |
| Solution 10 | 5.3958 (s) | 3.1475 (s) | 2.296 (s) | 7.639 (s) |

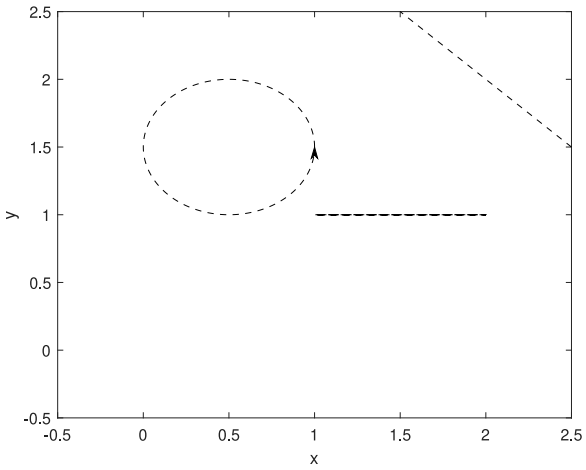



**Figure 6**: Trajectories of the targets for the time dependent motorised Travelling Salesmen Problem

An important fact is mentioned in the work of Von Stryk and Glocker: multiple local solutions exist even for a given order in which the targets are visited. It was not specified how the different solutions were generated, but the authors mentioned that different initial guesses had to be provided to the NLP solver in order to obtain different solutions. Since the approach proposed in this work is global and treats simultaneously both the discrete and the continuous variables, this kind of exploration is performed automatically, and, as shown, is able to return the best solution reported in the literature without requiring the user to generate different initial guesses for the NLP problem. Moreover, since the approach proposed in the reference was based on a deterministic branch and bound, it requires to compute all the solutions at the leaf level of the tree associated to the branch and bound. If the number of targets increases, this number can potentially become exceedingly large, resulting in intractable problems. With the approach proposed here, the maximum number of trials is specified a priori. Thus, even if larger problems become more and more difficult, the approach here proposed should still able to return a solution with a bounded number of function evaluations and computational time.

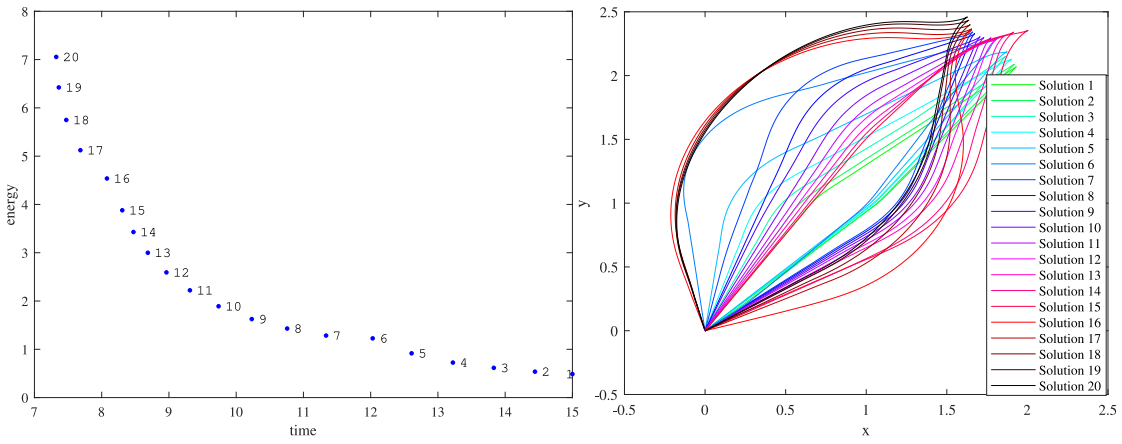
**A time dependent motorised Travelling Salesman Problem**

The problem described in the previous section was solved again, but this time the position of the targets is time dependent:

$$\begin{cases} P_1(t) = & (0.5 + 0.5\cos(t), 1.5 + 0.5\sin(t)) \\ P_2(t) = & (2 + \cos(2t), 2 - \cos(2t)) \\ P_3(t) = & (1.5 + 0.5\cos(3t), 1) \end{cases} \tag{14}$$

The time dependence of the position of the targets makes the problem particularly challenging, since the choice of the sequence in which the targets are to be visited cannot be decoupled from time. The time laws for the position of the targets were chosen in order to have periodic motions with different amplitudes and frequencies. The trajectories of the targets are shown in Figure 6.

The problem was solved with the same settings as the previous case, except that the number of agents was increased to 20, 20 solutions were kept in the archive, and a total of 100000 function evaluations was allowed.




(a) Pareto front (b) Trajectories

**Figure 7**: Pareto front and Trajectories for the time dependent motorised Travelling Salesman Problem

**Table 3**: Comparison of minimum time and minimum energy solution features between fixed and time dependent motorised Travelling Salesman Problem

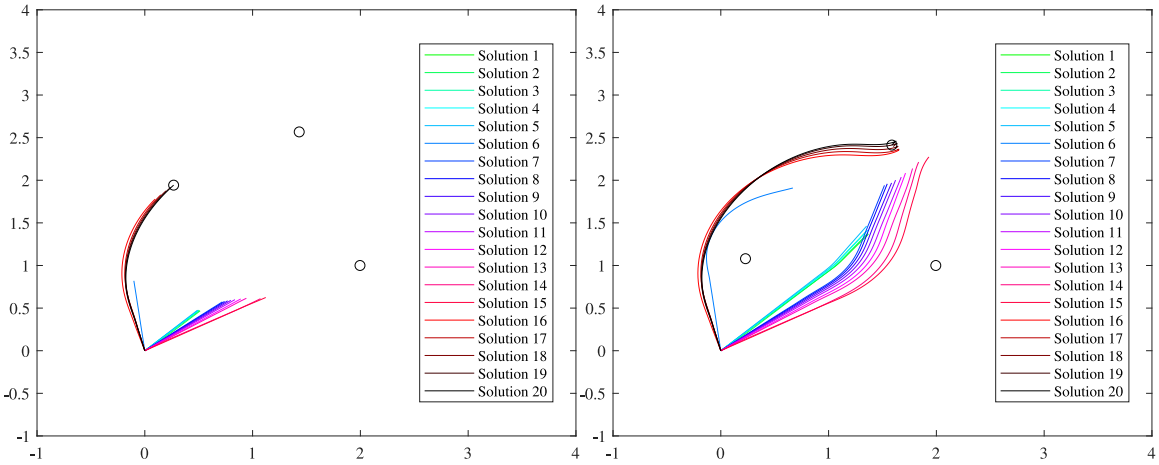
| Targets | Solution | $T(P_1)$ | $T(P_2)$ | $T(P_3)$ | $T_f$ | Energy |
|---|---|---|---|---|---|---|
| Fixed | Min $T_f$ | 5.3958 (s) | 3.1475 (s) | 2.296 (s) | 7.639 (s) | 7.040 |
| Moving | Min $T_f$ | 2.0546 (s) | 4.1399 (s) | 5.5458 (s) | 7.324 (s) | 7.055 |
| Fixed | Min $E$ | 10.623 (s) | 7.4999 (s) | 4.7023 (s) | 15 (s) | 0.616 |
| Moving | Min $E$ | 11.389 (s) | 7.1031 (s) | 3.3213 (s) | 15 (s) | 0.487 |

Figure 7 shows the Pareto front and the trajectories computed. As it is evident, there is still a trade-off between time and energy, but it appears to be composed of different branches . Solutions $1-5$ and $7-15$ visit the targets in the order $(P_3 \rightarrow P_2 \rightarrow P_1)$, while solutions 6 and $16-20$ visit the targets in the order $(P_1 \rightarrow P_2 \rightarrow P_3)$.

Interestingly, the minimum time solution takes less time to visit all targets than the minimum time solution of the previous case, indicating that the algorithm was able to exploit the relative motions of the targets to find a favourable timing for the various encounters, whose sequence is also different from the equivalent case of the previous problem. In order to do so, the solution also requires slightly more energy than the equivalent fixed targets case. Similarly, the minimum energy solution takes a sligthtly lower amount of energy than the fixed target minimum energy solution, while both trajectories take the same time to complete. Table 3 summarises these findings.

The trajectories followed by the various solutions are also more complex than in the previous case, although they still retain a similar overall shape. In order to convey a sense of the motion of the targets and of the vehicle, Figure 8 shows different snapshots of all trajectories. The snapshots were taken at the instants when the fastest solution (solution 20) reaches each target and finally when it returns to the origin.

Figure 9 shows the time histories of the velocity and of the acceleration. The profiles are similar to the previous case, although it is possible to see some differences. In particular, solutions $1-5$, $7-15$ and $16-20$ seem to belong to different families, with solution 6 making a class of its




(a) t = 2.0546

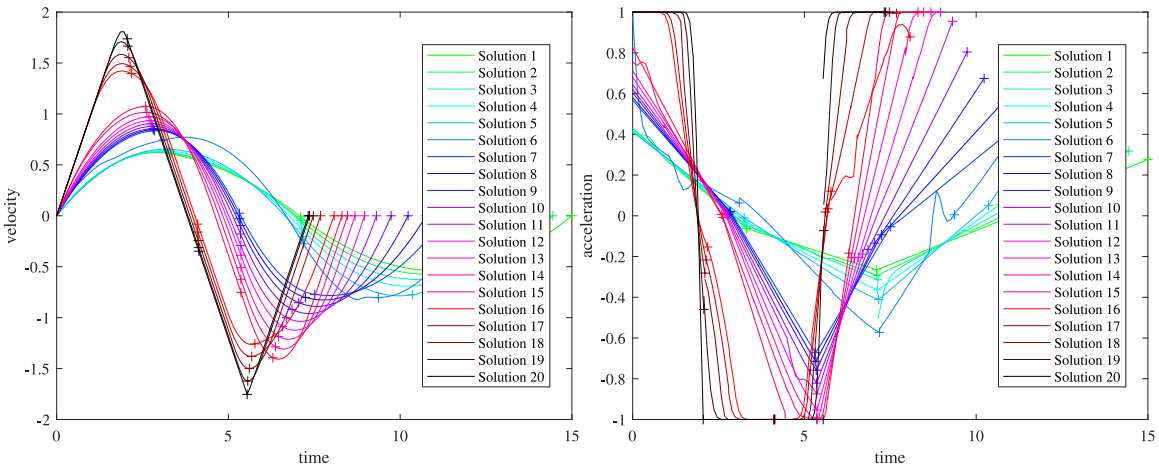
(b) t = 4.1399

(c) t = 5.5458

(d) t = 7.3237

**Figure 8**: Different snapshots of the trajectories for the time dependent motorised Travelling Salesman Problem

own. The same applies for the controls, where it is also possible to detect some defects where the NLP did not manage to converge to a locally optimal solution but stopped for small step size. Again, it would be possible to improve the quality and the resolution of these solutions by a mesh refinement procedure, but the overall trade-off and the main characteristics of the solutions were already captured at this stage.




(a) Time history of the velocity

(b) Time history of the acceleration

**Figure 9**: Time histories of velocity and acceleration for the time dependent Travelling Salesman Problem. $+$ marks when a target is visited

### Multi target debris removal

As a final test case, the algorithm is applied to the design of a space mission to remove three debris. The debris are assumed to be 3 defunct satellites located on the Geostationary orbit. A spacecraft is sent to apply a de-orbiting kit on each of those satellites. The spacecraft is assumed to start from an equatorial orbit. Thus it is convenient to express its dynamics using the full nonlinear equations of relative motion in Hill's frame,[23] restricted to a planar motion:

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{v_x} = 2\dot{f}\left(v_y - y\dfrac{\dot{r_c}}{r_c}\right) + x\dot{f}^2 + \dfrac{\mu}{r_c^2} - \dfrac{\mu(r_c+x)}{r_d^3} + \dfrac{uT\cos(\alpha)}{m} \\ \dot{v_y} = -2\dot{f}\left(v_x - x\dfrac{\dot{r_c}}{r_c}\right) + y\dot{f}^2 - \dfrac{\mu y}{r_d^3} + \dfrac{uT\sin(\alpha)}{m} \\ \dot{m} = \dfrac{-uT}{g_0 I_{sp}} \end{cases} \tag{15}$$

where $x$ and $y$ are the positions of the spacecraft relative to a reference point on the Geostationary orbit, $v_x$ and $v_y$ are the relative velocities, $\dot{f}$ is the rate of change of true anomaly of the Geostationary orbit, $r_c$ is the radius of the Geostationary orbit, $r_d$ is the distance of the current position to the centre of the Earth and and $m$ is the mass of the spacecraft, which is controlled by an engine

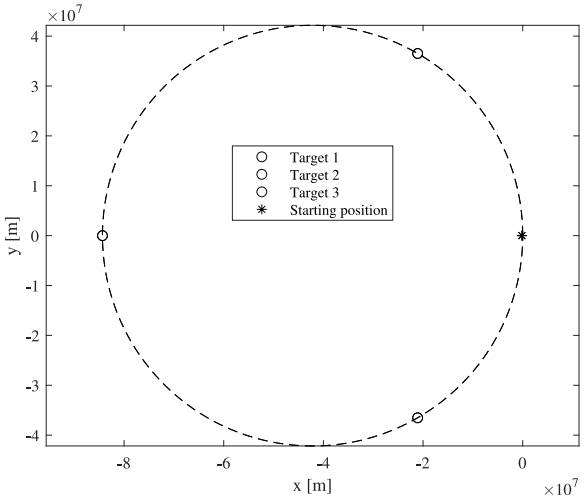



**Figure 10**: Positions of the targets and initial position of the spacecraft. Starting position lays inside the circle.

with maximum thrust $T$ and a specific impulse $I_{sp}$. The thrust vector is expressed in terms of throttling $u$ and direction $\alpha$. $g_0$ is the gravitational acceleration at sea level, while $\mu$ is the gravitational parameter of Earth.

The spacecraft has an initial mass of $1000\,\mathrm{kg}$, a minimum mass of $100\,\mathrm{kg}$ and is equipped with 3 de-orbiting kits, each with a mass of $30\,\mathrm{kg}$. The propulsion system is able to generate up to $10\,\mathrm{N}$ of thrust with a specific impulse of $300\,\mathrm{s}$. The spacecraft starts from an equatorial circular orbit with radius of $42\,000\,\mathrm{km}$, which is at a safe distance from the target orbit. This corresponds in the selected Hill's frame to the following initial conditions:

$$\begin{cases} x(0) = -164\,\mathrm{km} \\ y(0) = 0\,\mathrm{km} \\ v_x(0) = 0\,\mathrm{m\,s^{-1}} \\ v_y(0) = 5.9971\,\mathrm{m\,s^{-1}} \end{cases} \tag{16}$$

Since the reference frame is collocated on a point on the Geostationary orbit, the targets have a fixed position in time. This significantly simplifies the treatment of the boundary conditions, which no longer have a periodic time dependency over time. In this reference frame, they are assumed to be uniformly spread, forming an angle with the reference point on the Geostationary orbit of $30\,\mathrm{deg}$, $180\,\mathrm{deg}$ and $300\,\mathrm{deg}$. Since in this reference frame their positions are constant, their velocities are $0\,\mathrm{m\,s^{-1}}$. Figure 10 shows position of the targets and the initial position of the spacecraft in the relative frame.

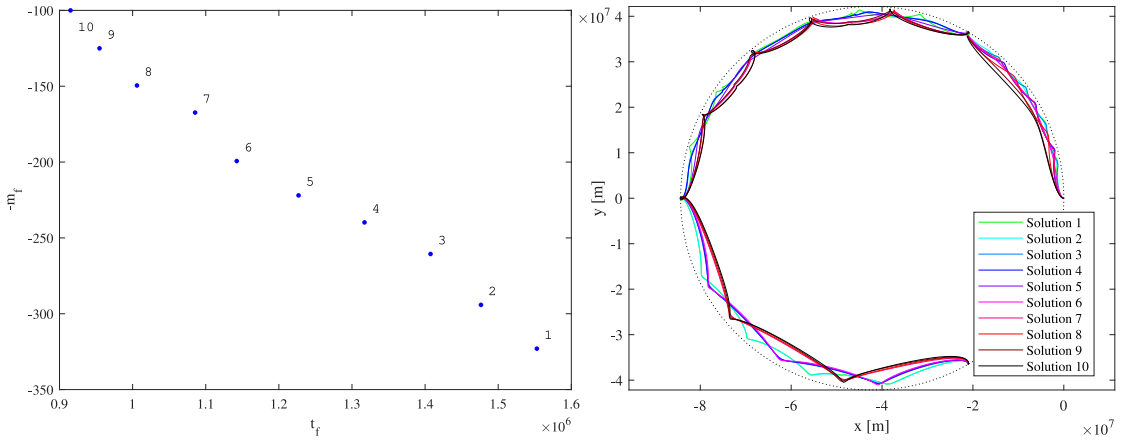
The spacecraft has to rendez-vous and dock with all targets in order to apply the de-orbit kit. The application of the de-orbiting kit is assumed to take $5\,\mathrm{h}$. The order in which targets are to be visited is not specified a priori. The objectives are the minimisation of the total mission time, and the maximisation of the final mass, corresponding to the minimisation of propellent consumption:

$$\min_{t_f, \mathbf{u}, \boldsymbol{\mu}} (J_1, J_2)^T = (t_f, -m_f) \tag{17}$$

The problem was formulated as a 3 phase problem, each phase was discretised with 3 DFET elements of order 7, resulting in a problem with 551 variables. The upper and lower bounds for state and control variables are reported in Table 4. The algorithm was run for 100000 function evaluations with standard settings, 10 agents storing 10 points on the Pareto front.

**Table 4**: Lower and upper bounds for the state and control variables of the multi debris removal mission

| Variable | Lower bound | Upper bound |
|---|---|---|
| $x$ (km) | $-126492$ | 42164 |
| $y$ (km) | $-84328$ | 84328 |
| $v_x$ (km s$^{-1}$) | $-20$ | 20 |
| $v_y$ (km s$^{-1}$) | $-20$ | 20 |
| $m$ (kg) | 100 | 1000 |
| $u$ (#) | 0 | 1 |
| $\alpha$ (rad) | $-\pi$ | $\pi$ |
| $t_f$ (d) | 0 | 40 |




(a) Pareto front

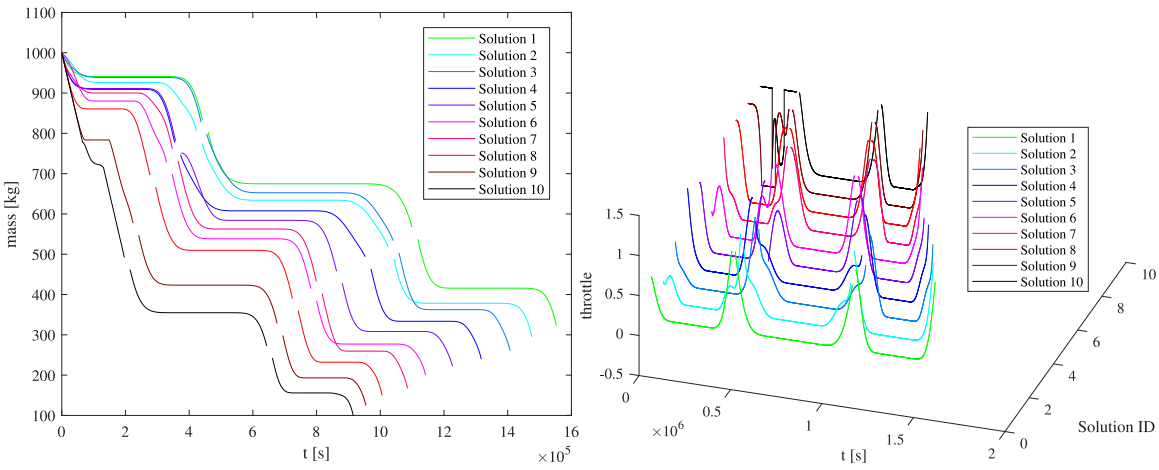
(b) Trajectories

**Figure 11**: Pareto front and Trajectories for the multiple debris removal mission

Figure 11 shows the Pareto front and the trajectories of the spacecraft in the relative frame. A tradeoff between mission time and final mass is present, as expected. Faster trajectories tend to be closer to the centre of the circle, meaning that in order to reduce mission time the optimiser exploited the relative rotation rate between the two orbits. All targets are visited in the same order. The epicycloidal shape of the trajectories in the relative frame means that the trajectories are more eccentric, while more circular shaped arcs mean that the corresponding trajectory is closer to circular.

Figure 12 shows the time histories of the mass of the vehicle and the throttling of the engine. The minimum time solution requires approximately 10 days and reached the lower bound of the final mass, meaning that it employed all available propellent. Interestingly, the minimum time solution does not have enough propellent to proceed at full throttle all the time, thus the optimiser had to save propellent and increase the mission time in order to reach all targets. In general, all solutions have

a bang-zero-bang structure, as expected, where shorter mission times solutions have comparatively longer burns. The throttle and mass profiles are not very sharp, meaning that the solutions would benefit from mesh refinement and that some improvements in the values of the objective functions are expected. However, the overall trend and trade-off between the solutions is well captured.




(a) Time history of the mass

(b) Time history of the throttle

**Figure 12**: Time histories of velocity and acceleration for the time dependent Travelling Salesman Problem

## CONCLUSIONS

This paper presented a new method for solving Multi-Objctive Hybrid optimal control problems. It is based on the coupling of the DFET transcription with the MACS memetic multi-objective optimisation algorithm. The resulting discretised problem is solved with a combination of two approaches, a bi-level approach, which allows for global exploration and a generation of a well spread set of solutions, and a single-level approach, which guarantees local optimality. An extension of the heuristics of MACS has been presented in order to allow it to work with mixed integer problems. Furthermore, a staged relaxation approach has been employed when the NLP solver is invoked. The relaxed variables are then forced to assume integer values with a simple set of smooth constraints. A special set of constraints was proposed to deal with problems where the integer variables encode mutually exclusive choices, like the choice of the targets in the Travelling Salesman problem. The approach was tested against a problem found in the literature, and against a multiple debris removal mission. Although the problems tackled here are relatively small scale, the main limitation of the proposed approach is given by the maximum size of problems that the NLP solver can handle.

## REFERENCES


[1] M. Vasile and A. E. Finzi, "Direct lunar descent optimisation by finite elements in time approach," *International Journal of Mechanics and Control*, Vol. 1, No. 1, 2000.

[2] M. Vasile and F. Bernelli-Zazzera, "Targeting a heliocentric orbit combining low-thrust propulsion and gravity assist manoeuvres," *Operational Research in Space & Air*, Vol. 79, 2003.

[3] C. L. Bottasso and A. Ragazzi, "Finite element and runge-kutta methods for boundary-value and optimal control problems," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 4, 2000, pp. 749–751.

[4] M. Borri and C. Bottasso, "A general framework for interpreting time finite element formulations," *Computational Mechanics*, Vol. 13, Dec 1993, pp. 133–142.

[5] M. Vasile, "Finite Elements in Time: A Direct Transcription Method for Optimal Control Problems," *AIAA/AAS Astrodynamics Specialist Conference*, 2010.

[6] L. A. Ricciardi and M. Vasile, "Direct Transcription of Optimal Control Problems with Finite Elements on Bernstein Basis," *Journal of Guidance, Control, and Dynamics*, oct 2018, pp. 1–15, 10.2514/1.g003753.

[7] L. A. Ricciardi and M. Vasile, "Improved Archiving and Search Strategies for Multi Agent Collaborative Search," *Computational Methods in Applied Sciences*, pp. 435–455, Springer International Publishing, jul 2018.

[8] L. A. Ricciardi, C. A. Maddock, and M. Vasile, "Direct Solution of Multi-Objective Optimal Control Problems Applied to Spaceplane Mission Design," *Journal of Guidance, Control and Dynamics*, 2018, p. Accepted for publication on 26 August 2018.

[9] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica: Journal of the Econometric Society*, 1960, pp. 497–520.

[10] M. A. Duran and I. E. Grossmann, "An outer-approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical programming*, Vol. 36, No. 3, 1986, pp. 307–339.

[11] R. Fletcher and S. Leyffer, "Solving mixed integer nonlinear programs by outer approximation," *Mathematical programming*, Vol. 66, No. 1-3, 1994, pp. 327–349.

[12] A. M. Geoffrion, "Generalized benders decomposition," *Journal of optimization theory and applications*, Vol. 10, No. 4, 1972, pp. 237–260.

[13] C. Audet and J. E. Dennis Jr, "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on optimization*, Vol. 17, No. 1, 2006, pp. 188–217.

[14] M. A. Abramson, C. Audet, J. W. Chrissis, and J. G. Walston, "Mesh adaptive direct search algorithms for mixed variable optimization," *Optimization Letters*, Vol. 3, No. 1, 2009, p. 35.

[15] M. Glocker and O. v. Stryk, "Hybrid optimal control of motorized traveling salesmen and beyond," *Proc. 15th IFAC World Congress on Automatic Control*, 2002, pp. 21–26.

[16] S. Sager, *Numerical methods for mixed-integer optimal control problems*. Der Andere Verlag Tönning, 2005.

[17] J. A. Englander, M. A. Vavrina, and A. R. Ghosh, "Multi-Objective Hybrid Optimal Control for Multiple-Flyby Low-Thrust Mission Design," *AAS/AIAA Space Flight Mechanics Meeting*, Williamsburg, Virginia, 11-15 Jan 2015.

[18] M. Schlueter, *Nonlinear mixed integer based optimization technique for space applications*. PhD thesis, University of Birmingham, 2012.

[19] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *Evolutionary Computation, IEEE Transactions on*, Vol. 6, No. 2, 2002, pp. 182–197.

[20] R. H. Leary, "Global Optimization on Funneling Landscapes," *Journal of Global Optimization*, Vol. 18, Dec 2000, pp. 367–383, 10.1023/A:1026500301312.

[21] M. Schlüter, J. A. Egea, and J. R. Banga, "Extended ant colony optimization for non-convex mixed integer nonlinear programming," *Computers & Operations Research*, Vol. 36, No. 7, 2009, pp. 2217 – 2229, https://doi.org/10.1016/j.cor.2008.08.015.

[22] O. Von Stryk and M. Glocker, "Numerical mixed-integer optimal control and motorized traveling salesmen problems. Commande numerique optimale a entiers mixtes et problemes de voyageur de commerce motorise," *Journal Européen des Systèmes Automatisés*, Vol. 35, No. 4, 2001, pp. 519–534.

[23] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems, Fourth Edition*. American Institute of Aeronautics and Astronautics, Inc., jan 2018, 10.2514/4.105210.