Interplanetary Trajectory Planning with Monte Carlo Tree Search

Daniel Hennes and **Dario Izzo**
European Space Agency
Advanced Concepts Team
Noordwijk, The Netherlands
daniel.hennes@esa.int, dario.izzo@esa.int

Abstract

Planning an interplanetary trajectory is a very complex task, traditionally accomplished by domain experts using computer-aided design tools. Recent advances in trajectory optimization allow automation of part of the trajectory design but have yet to provide an efficient way to select promising planetary encounter sequences. In this work, we present a heuristic-free approach to automated trajectory planning (including the encounter sequence planning) based on Monte Carlo Tree Search (MCTS). We discuss a number of modifications to traditional MCTS unique to the domain of interplanetary trajectory planning and provide results on the *Rosetta* and *Cassini-Huygens* interplanetary mission design problems. The resulting heuristic-free method is found to be orders of magnitude more efficient with respect to a standard tree search with heuristic-based pruning which is the current state-of-the art in this domain.

1 Introduction

Interplanetary trajectory optimization holds its most challenging aspect in its combinatorial part, that is the selection of the planetary encounters.

Tree searches with heuristic-based pruning, implementing problem knowledge, are the state-of-the-art in the aerospace industry for tackling these problems. A notable example is the software STOUR in use at NASA, Jet Propulsion Laboratory [Longuski and Williams, 1991] which targets the automated design of trajectories with multiple fly-bys and has been used in several important mission design works [Heaton *et al.*, 2002; Petropoulos *et al.*, 2000]. Some attempts have been made in the last decade to advance the state-of-the-art by proving the use of advanced combinatorial search paradigms such as Ant Colony Optimization [Ceriotti and Vasile, 2010], genetic algorithms [Deb *et al.*, 2007; Gad and Abdelkhalik, 2011; Izzo *et al.*, 2014], tree search strategies [Izzo *et al.*, 2014; Petropoulos *et al.*, 2014], or bi-level optimization setups [Englander, 2013]. While somewhat successful, these advanced methods all make use of problem knowledge to define heuristics and their application to different data sets or domains requires substantial tuning of many internal parameters and often result in an inefficient set-up.

A relatively recent method to tackle extremely complex combinatorial problems is that of the Monte Carlo Tree Search (MCTS) paradigm. Born in the context of two-person zero-sum games with perfect information; in MCTS a node is evaluated by averaging the final outcome of several random simulations. Four steps, namely: selection, expansion, simulation, and back-propagation, are performed iteratively up to when a stopping criteria is reached. Many papers have studied MCTS and applied its modifications and variants to different domains with great success; for an excellent overview see [Browne *et al.*, 2012].

In this paper we propose a heuristic-free approach based on MCTS to tackle the complex problem of interplanetary trajectory planning, and in particular the planetary encounter selection problem. We start by giving a brief background on the interplanetary trajectory design problem and MCTS. Next, we look at trajectory design as a planning task, defining the possible actions and their relation to the final trajectory. We then describe a number of modifications to traditional MCTS, unique to the domain of interplanetary trajectory planning. We discuss our experimental set-up, in particular parameter search and runtime analysis. The resulting heuristic-free method is found to be orders of magnitude more efficient with respect to standard tree search with heuristic-based pruning which is the current state-of-the art in this domain [Longuski and Williams, 1991; Heaton *et al.*, 2002; Petropoulos *et al.*, 2000].

2 Background

2.1 Interplanetary trajectories

We begin with a brief outline of the fundamentals of trajectory design. Even today's most powerful launch systems are not able to send a spacecraft on a direct trajectory to target bodies in the outer solar system. Therefore, most interplanetary missions require a well designed trajectory that guides the spacecraft through a number of gravity assist maneuvers, also called *fly-bys*. Each fly-by provides the spacecraft with a gravitational kick by "stealing" a small amount of the planet's orbital energy. A sequence of carefully planned and executed fly-bys allows the spacecraft to save propellant (or time) and enables trajectories otherwise impossible.

In this paper we model an interplanetary spacecraft trajectory using a multiple gravity assist impulsive propulsion model (known as the MGA model [Izzo *et al.*, 2007]), which is a very good preliminary approximation for a spacecraft equipped with chemical propulsion featuring a high thrust capability. In the MGA model, the spacecraft is assumed to be able to perform powered fly-bys; at each gravity assist maneuver an impulse, modeled as a discontinuity ΔV in the spacecraft velocity, can be used to correct the fly-by geometry or decelerate the spacecraft to *rendezvous* with a target body. Impulses at any other point of the trajectory, i.e. deep space maneuvers, are not considered in this work. Using this model, a trajectory $\mathcal{T}$ is defined by a pair $(\mathcal{P}, \tilde{T})$ where $\mathcal{P}$ is an ordered set of cardinality N containing the planetary encounter sequence (e.g. $\mathcal{P} = \{\text{Earth}, \text{Venus}, \text{Jupiter}\}$), and $\tilde{T}$ is an ordered set of cardinality N containing the epochs of the planetary encounters (e.g. $\tilde{T} = \{t_0, t_1, t_2\}$). It is convenient for the definition of time grids to use, instead of $\tilde{T}$, the ordered set T containing, rather then the epochs, the various time of flights (e.g. $T = \{t_0, t_1 - t_0, t_2 - t_1\} = \{t_0, T_1, T_2\}$).

Given a trajectory $\mathcal{T} = (\mathcal{P}, T)$, we may evaluate the cumulative change of velocity (total ΔV) required to fly it. This quantity is of fundamental importance in interplanetary trajectory design, as each required velocity change relates directly to the propellant mass via the Tsiolkovsky equation, $\Delta V = I_{sp} g_0 \ln(m_i/m_f)$, where m_f is the spacecraft mass at the end of the maneuver and m_i its mass at the beginning. Trivially, $m_f = m_i - m_p$ where m_p is the propellant mass spent.

We start by computing the positions $\boldsymbol{r_i}$ and velocities $\boldsymbol{v_i}$ of the planets at the encounter epochs $\forall i \in [0..N-1]$. For this computation we use the analytical planet ephemerides defined by NASA / Jet Propulsion Laboratory[1]. Given two position vectors of two sequential encounters $\boldsymbol{r_{i-1}}$ and $\boldsymbol{r_i}$, and the time of flight T_i, we can then determine the spacecraft's absolute velocities $\boldsymbol{v_{i-1}^+}$ and $\boldsymbol{v_i^-}$ at the encounters (i.e. $\forall i \in [1..N-1]$) by solving Lambert's problem [Izzo, 2014][2]. We may then compute, at each planet, the spacecraft relative velocities before and after the encounter $\tilde{\boldsymbol{v}}_i{}^- = \boldsymbol{v_i^-} - \boldsymbol{v_i}$, $\tilde{\boldsymbol{v}}_i{}^+ = \boldsymbol{v_i^+} - \boldsymbol{v_i}$. Two Lambert legs, e.g. Earth-Venus and Venus-Jupiter, can then be "patched" together by calculating the required ΔV to patch $\tilde{\boldsymbol{v}}^{-,+}$ at Venus. The required ΔV maneuver, computed accounting for the hyperbolic fly-by trajectory, depends on the the magnitude of the relative arrival and departure velocities at Venus but also on the angle β between the relative velocities [Izzo *et al.*, 2007].

In addition to the various ΔV_i computed at the encounters, the contributions at launch and arrival must be considered. The relative departure velocity at launch is either fully or partially supplied by the launch system. We thus discount the launcher's maximum velocity v^{LS} from the departure relative velocity at Earth $\tilde{\boldsymbol{v}}_0^-$:

$$\Delta V_{Launch} = \max\left(0, |\tilde{\boldsymbol{v}}_0^-| - v^{LS}\right) .$$

At the final encounter, a maneuver is performed to capture the spacecraft in the gravitational field of the target body. We compute this as $\Delta V_{rndvz} = |\tilde{\boldsymbol{v}}_{N-1}^+|$, a contribution, called *rendezvous* maneuver, necessary to match the velocity of the target body. A generic MGA trajectory $\mathcal{T} = (\mathcal{P}, T)$ is then associated to a required ΔV_{tot} defined by:

$$\Delta V_{tot} = \Delta V_{Launch} + \sum_{i=1..N-2} \Delta V_i + \Delta V_{rndvz} .$$

Finding the correct planetary sequence $\mathcal{P}$ and time schedule T that allow for the minimization of this ΔV_{tot} is the problem we tackle and solve in the remainder of this paper.

2.2 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a technique widely applicable in domains that require sequential decision making, including game-tree search and planning problems. The MCTS paradigm combines informed tree search with the generality of Monte Carlo simulations. Although, MCTS exist in many variants, all are based on the concept of incrementally building an internal tree to inform its search policy. MCTS algorithms are any-time algorithms that repeat the following four basic steps until the computational budget is depleted [Chaslot *et al.*, 2008]:

1. **Selection:** Starting from the root a selection policy is deployed to descend through the tree while balancing exploration and exploitation.
2. **Expansion:** Once the tree reaches a leaf node, the state is advanced by performing a random available action and the resulting state is added as a new node to the tree.
3. **Simulation:** A Monte Carlo simulation is run with random action selection. Heuristic knowledge can be used to give higher weight to promising actions.
4. **Back-propagation:** Once a final state is reached, the value is back-propagated upwards trough the search tree and each node selected in step 1 is updated accordingly.

Let us now discuss a popular choice of MCTS for planning problems, Upper Confidence bounds for Trees (UCT) [Kocsis and Szepesvári, 2006], in further detail. UCT is based on the Upper Confidence Bound (UCB) [Auer *et al.*, 2002] selection strategies for multi-armed bandit problems. The multi-armed bandit problem is a classical toy problem addressing the *exploration-exploitation dilemma*. In particular, Auer et. al [Auer *et al.*, 2002] propose a policy UCB1 that has a bounded regret for arbitrary reward distributions with support in $[0, 1]$ after any number of plays (i.e. in finite-time). The same paper also discusses various other selection policies, most notably ϵ-greedy and UCB1-Tuned. The authors make the following observations: an optimally tuned ϵ-greedy policy performs almost always best; UCB1-Tuned performs comparably to a well-tuned ϵ-greedy policy but without a proven regret bound. The UCB1-Tuned policy takes into account the measured variance of rewards and is thus less sensitive to the reward distribution than UCB1.

[1]The approximated ephemerides were used as defined in http://ssd.jpl.nasa.gov/?planet_pos [accessed November 2014]

[2]In this work we only consider 0-revolution Lambert solutions. The extension of our methods to multiple revolution is straight forward.

UCT follows the MCTS approch outlined above and deploys the following selection policy:

$$\arg\min_i \bar{X}_i + C_p\sqrt{\frac{\ln n}{n_i}}\ , \tag{1}$$

where $\bar{X}_i$ is the estimated reward for child i (or action i), n is number of times the current node has been selected, and n_i is the number of times the child i has been updated. The parameter C_p has been introduced to balance exploration-exploitation behavior. If $C_p = \frac{1}{\sqrt{2}}$, then Equation (1) is equivalent to the UCB1 policy as introduced in [Auer *et al.*, 2002]. The ϵ-greedy policy is computationally less demanding and thus of interest in complex planning problems with many tree updates. Equation (2) shows a slightly modified version of ϵ-greedy as introduced in [Sabharwal *et al.*, 2012]:

$$\arg\min_i \bar{X}_i + \frac{\epsilon n}{n_i}\ . \tag{2}$$

Both selection policies (1) and (2) require tuning of an exploration–exploitation parameter, C_p and ϵ respectively. The UCB1-Tuned policy uses the measured variance of rewards:

$$\arg\min_i \bar{X}_i + \sqrt{\frac{\ln n}{n_i}\min\{\frac{1}{4}, V_i\}}, \tag{3}$$

where V_i is the variance for child i:

$$V_i = (\frac{1}{2}\sum_{j=1}^{n_i} X_{i,j}^2) - \bar{X}_i^2 + \sqrt{\frac{2\ln n}{n_i}} = {\sigma_i}^2 + \sqrt{\frac{2\ln n}{n_i}}\ . \tag{4}$$

All three variants have been applied as selection policies in MCTS variants.

3 MCTS for Trajectory Planning

In order to apply UCT to interplanetary trajectory design, we must transcribe the problem of finding an interplanetary trajectory $\mathcal{T} = (\mathcal{P}, T)$ as a planning task. The initial state (i.e. root of the tree) is pre-launch at Earth and the first action is selecting the departure date $t_0 \in [\overline{t}_0, \underline{t}_0]$ where $\overline{t}_0$ and $\underline{t}_0$ define the boundaries of the launch window. The consecutive moves alternate between selecting the next planetary encounter $P_i \in \{\text{Venus}, \text{Earth}, \text{Mars}, ...\}$, and time of flight $T_i \in [\overline{T}_i, \underline{T}_i]$ where the limits $\overline{T}_i$ and $\underline{T}_i$ are chosen looking at the orbital periods τ_{i-1}, τ_i of P_{i-1} and P_i and setting the wide bounds $\underline{T}_i = 0.1*(\tau_i+\tau_{i-1}), \overline{T}_i = 2*(\tau_i+\tau_{i-1})$. The action sequence: $\{t_0 = 5110, P_1 = \text{Venus}, T_1 = 150, P_2 = \text{Jupiter}, T_2 = 2000\}$ thus defines a trajectory departing from Earth[3] at time 5110 MJD2000 (Modified Julian Day 2000) with two Lambert legs: from Earth to Venus in 150 days and from Venus to Jupiter in 2000 days. Contrary to the usual approach for trajectory design, this problem description allows us to simultaneously tackle the selection of the planetary encounter sequence and the timing of flyby. As most of the search space is spanned by very costly actions (resulting in infeasible trajectories requiring unrealistic propellant quantities) we also consider a threshold on the ΔV of 10 [km/s], so that if at any node the cumulative ΔV exceeds this threshold the state is considered terminal.

[3] $P_0 = \text{Earth}$ is implied.

3.1 Ephemeris grid

The UCB selection policy used in UCT is based on finite multi-armed bandits problems; as such the action space must be a set of discrete choices. However, the initial action t_0 and the TOF-actions T_i are continuous in nature. A straightforward solution is to use a regular tiling of the time domain, e.g. grid points every 5 days. Although, regular-tiling is often used as a discretization technique of the state, action and time domain in robotics, it is highly inefficient for Keplerian motions. Planets in the inner solar system have orbital periods of just 88 days (Mercury) to 687 days (Mars), while outer planets take 12 years (Jupiter) to 29 years (Saturn) to complete an orbit. We thus define a 1D grid for each body based on its orbital period τ_i and a grid resolution parameter l [deg]:

$$G = \left\{ t_0 + j\frac{l\tau_i}{360^\circ}, j \in N \right\}. \tag{5}$$

The action set $A(t_0)$ is a subset of G^{Earth} in agreement with the launch window; each TOF action set $A(T_i)$ is a subset of G^{P_i} such that $t_0 + \sum_{j=1}^{i-1} T_j + T_i$ falls on G^{P_i} in agreement with the bounds on the time of flight. This ensures that all ephemeris calculations, i.e. the calculation of the location $\boldsymbol{r}$ and velocity $\boldsymbol{v}$, of a body are aligned with its grid. The complexity of the planning problem increases exponentially with decreasing grid resolution parameter l as shown in Figure 1.

3.2 Selection policy

The UCB1 and UCB-Tuned policies require a reward distribution with support in the interval $[0, 1]$. The evaluation of a trajectory is the sum of all required ΔV maneuvers including final rendezvous and thus in the half-closed interval $\Delta V_{tot} \in [0, \infty)$. We thus map ΔV^{tot} to the reward for simulation j as $X_j \in [0,]$ as follows:

$$X_j = \max\left(0, \frac{\Delta V_{max} - \Delta V_{tot,j}}{\Delta V_{max}}\right). \tag{6}$$

Furthermore, we redefine the node value estimator $\bar{X}$ in Equations (1), (2), and (3) as follows:

$$\bar{X}_i = \max_{j=1}^{n_i} X_j\ , \tag{7}$$

where X_j is the reward of simulation j. For a given action sequence, each of the Lambert problems has a unique solution. As such, the patching velocities and the launch and rendezvous ΔV calculations are deterministic in nature. We can thus use the max-estimator instead of the average over X_j as each X_j gives a guaranteed lower bound on true value of node j. In previous work, the "max-style" estimator is either used in combination with the average estimator for MCTS in stochastic single-player games [Schadd *et al.*, 2008] or as the sole node estimator in deterministic domains [Sabharwal *et al.*, 2012].

3.3 Expansion

In MCTS, the node expansion occurs at a leaf node after descending through the internal tree. A previously untried action is chosen at random and the corresponding node is added

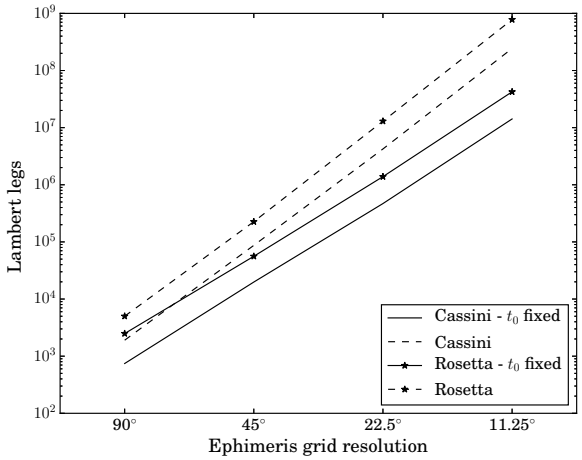


Figure 1: Problem complexity as a function of grid size.

Table 1: Global minima for $l = 11.25°$.

Problem	fixed t_0	full
Cassini-Huygens	6.760 km/s	6.571 km/s
Rosetta	6.498 km/s	5.887 km/s

to the tree, at which point the Monte Carlo simulation starts. As stated above, the evaluation of a simulation step provides a guaranteed bound. As such there is no need to re-evaluate the same trajectory again either in part or in whole. We thus expand the internal tree by all nodes encountered during the Monte Carlo simulation.

3.4 Contraction

In addition to the four steps of MCTS (i.e, selection, expansion, simulation, and backpropagaion), we introduce a fifth step: *contraction*. When MCTS is applied to game-tree search, the computational budget is usually relatively limited and thus it is with low probability that the internal tree reaches the depth of the full search tree (i.e., a leave of the internal tree is a final state). This is especially the case in the early phase of the game; later in the game, when the internal tree reaches final states, end-game databases have been proven to be more efficient [Browne *et al.*, 2012]. Trajectory planning problems span a very broad tree, with a branching factor as high as 500 and limited depth, i.e. a trajectory rarely includes more than 5-10 fly-bys and thus a search depth of 11-21. In addition, we are not concerned with executing the expected best move at the root node, as the case in traditional MCTS, but rather in the full trajectory. The modified *expansion* step and a rather broad but shallow tree, allow our algorithm to grow an internal tree that indeed reaches final nodes. Any subtree of the internal tree that fully covers the corresponding problem search space (i.e., each child is played at least once and all leaves are final states) can be ignored by further search iterations and is thus removed from the internal tree.

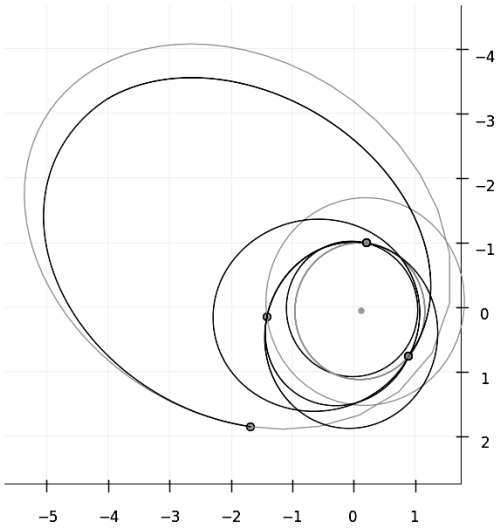


Figure 2: Example of a possible *Rosetta* mission trajectory.

3.5 Flat Monte Carlo

In addition to the UCT with the various selection policies we establish a baseline by including experiments with Flat Monte Carlo search. Flat Monte Carlo does not gradually build a tree but rather performs simulations starting from the root until the computational budget is depleted, at which point the best encountered solution is returned [Browne *et al.*, 2012].

4 Experimental Evaluation

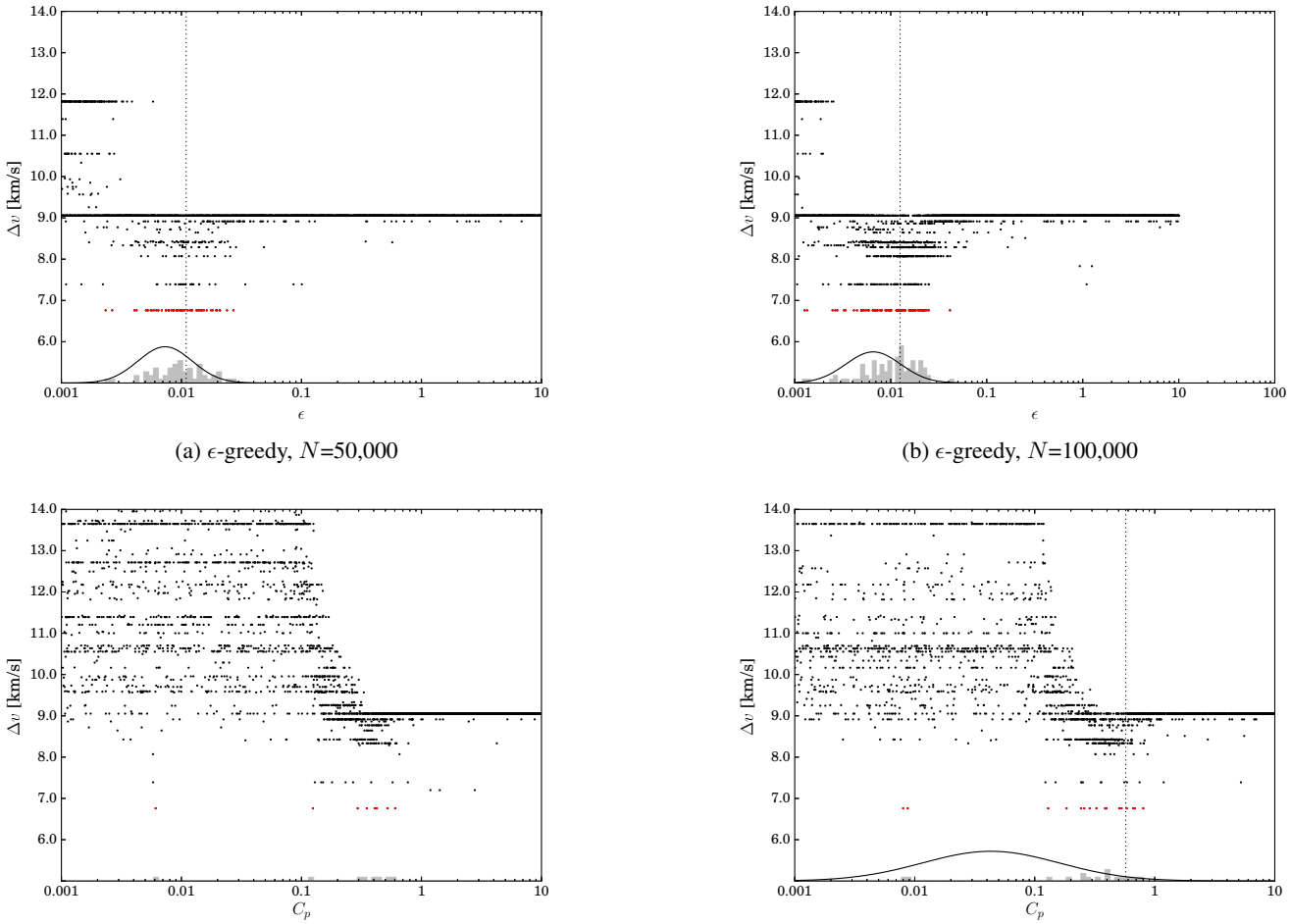
4.1 Problem set

We evaluate our approach on two well-known missions: *Cassini-Huygens* and *Rosetta*. *Cassini-Huygens* is a joint NASA/ESA mission launched in 1997 sent to Saturn. The spacecraft *Cassini* arrived at its destination in 2004; it has since successfully deployed the *Huygens* lander to Saturn's moon Titan and studied many of Saturn's satellites.

Rosetta is an ESA-operated mission launched in 2004. *Rosetta* arrived at its destination, comet 67P/*Churyumov-Gerasimenko*, in August 2014. It is the first mission in history to rendezvous with a comet, escort it as it orbits the Sun, and successfully deploy a lander probe, named *Philae*, for a triple landing on the comet's surface.

We transcribe the two missions as planning problems following the approach outlined in Section 3. For both missions we define two variants, the fixed t_0 variant and the full variant. The fixed-t_0 variant, fixes the first action of the planning problem to a launch date close to the actual mission launch; $t_0 = 1551.31$ MJD2000 for *Rosetta* and $t_0 = -787.53$ MJD2000 for *Cassini-Huygens*. The full variant includes the first action within a 200 day and 365 day launch window for *Cassini-Huygens* and *Rosetta* respectively. An example of a *Rosetta* trajectory as described by our trajectory planning model is shown in Figure 2.

The complexity in terms of Lambert legs is shown in Figure 1. The graph shows the number of Lambert legs re-



(a) ϵ-greedy, N=50,000

(b) ϵ-greedy, N=100,000

(c) UCB1, N=50,000

(d) UCB1, N=100,000

Figure 3: Parameter search for the fixed-t_0 *Cassini-Huygens* problem with $l = 11.25°$ grid resolution.

quired to exhaustively search the problem space by depth-first search. Table 1 lists the minimum ΔV solutions for both problems found by exhaustive search with the finest grid resolution $l = 11.25°$.

4.2 Parameter search

The UCB-1 (see Equation (1)) and ϵ-greedy (see Equation 2) selection policies require one parameter choice each. Performance is expected to greatly depend on careful tuning of these parameters and we can not rely on "classical" choices as no prior work has been done in the intersection of MCTS and interplanetary trajectory planning. Therefore, we deploy a parameter search, inspired by the methodology described in [Kuipers *et al.*, 2013]. We sample 4000 parameter instances uniformly on a logarithmic scale. For each parameter instance, one run of UCT with the selected policy is performed until the computational budget of N Lambert legs is depleted. In addition to the approach in [Kuipers *et al.*, 2013], we fit a log-normal distribution to the parameter samples that result in runs that reached the minimum ΔV solution. The mean of the fitted distribution is used as a parameter choice to conduct the performance evaluation.

4.3 Performance evaluation

To investigate the performance of the different selection policies, we determine the expected runtime of the search. The expected runtime $\mathbb{E}(\mathrm{RT})$ is computed with respect to the number of Lambert legs. In particular, the formulation in [Auger and Hansen, 2005] has been followed, where $\mathbb{E}(\mathrm{RT})$ is defined as:

$$\mathbb{E}(\mathrm{RT}) = \frac{1 - p_s}{p_s} N + \mathbb{E}(\mathrm{RT}_s) \,,$$

where $\mathbb{E}(\mathrm{RT}_s)$ is the expected number of Lambert legs for a successful trial, p_s is the probability of convergence to the target value, and N is the budget for a trail. The target value is defined as $\Delta V_t = \min \Delta V + \epsilon$, where $\min \Delta V$ is the minimum ΔV solution and $\epsilon = 50$ m/s a convergence threshold.

5 Results

Figure 3 shows the result of the parameter search in the *Cassini-Huygens* fixed-t_0 problem for 4000 sample points.

Table 2: Mean values for fitted log-normal distributions; parameters in bold are used for further simulations.

Cassini-Huygens	fixed t_0		full	
Lambert Legs	50K	100K	100K	1M
ϵ-greedy (ϵ)	**0.0110**	0.0125	—	**0.0139**
UCB1 (C_p)	—	**0.5730**	—	—

Rosetta	fixed t_0		full	
Lambert Legs	50K	100K	100K	1M
ϵ-greedy (ϵ)	**0.0146**	0.0203	**0.0406**	—
UCB1 (C_p)	**0.4427**	0.6078	**1.8454**	—

Table 3: Expected runtime for problems with fixed launch date t_0; budget per run N=50,000.

Cassini-Huygens	p_s	RT$_s$	$\mathbb{E}$(RT)
ϵ-greedy	0.07550	33,446	645,697
UCB1	0.00980	19,305	5,097,510
UCB1-Tuned	0.0	—	∞
Flat Monte Carlo	0.0	—	∞

Rosetta	p_s	RT$_s$	$\mathbb{E}$(RT)
ϵ-greedy	0.39100	31,330	109,207
UCB1	0.00850	33,652	5,866,005
UCB1-Tuned	0.00075	21,503	66,638,169
Flat Monte Carlo	0.0	—	∞

The runs that reached minimum ΔV values are highlighted in red. The mean values of the log-normal distribution are reported in Table 2. We see a number of "solution-bands", most dominantly around 9 km/s; too little or too much exploitation traps the search in local minima. The experiment presented in Figure 3 (c) resulted in less than 10 successful runs, thus an estimate for C_p was not attempted. Table 2 lists parameter choices for the *Cassini-Huygens* and *Rosetta* missions as identified by the parameter search and distribution fitting. For the full *Cassini-Huygens* problem, a parameter estimate for ϵ could only be established for runs with N=1,000,000.

Table 3 and Table 4 show the expected runtime for both problems with Lambert leg budget N=50,000 for fixed-t_0 and Lambert leg budget N=100,000 for the full problems. The ϵ-greedy selection policy performs overall best, UCB1 is one order of magnitude worse in the fixed-t_0 case and fails to reach the minimum solution in the full problem. UCB1-Tuned performs worse than UCB1. Only well-tuned ϵ-greedy selection found the minimum solution given the limited computational budget.

6 Discussion

We have presented an approach to transcribe the preliminary-phase of an interplanetary trajectory design as a planning problem to which we applied the MCTS paradigm. Contrary to what is commonly done in most approaches to trajectory design, the sequence of planetary encounters (i.e. the combinatorial part of the problem) was not fixed a priori. The final result for a generic mission is a number of trajectory options, each comprised of an encounter sequence $\mathcal{P}$ and a time line T.

Table 4: Expected runtime for full problems; budget per run N=100,000.

Problem	p_s	RT$_s$	$\mathbb{E}$(RT)
Cassini-Huygens	0.00025	64,607	399,964,607
Rosetta	0.23588	68,602	392,543

With respect to a depth-first search (also using a pruning threshold of 10 km/s), MCTS makes use of orders of magnitude less Lambert leg computations to find the global optimal solution. The real strength of the approach, though, lies in its ability to find very good solutions with a very limited computational budget. Best–first search, such as A^* or beam search, would only be as successful as the employed future–cost heuristic which is not readily available in planetary trajectory planning; whereas our algorithm is heuristic–free.

As such, our MCTS-based approach seems to be a very promising candidate to substitute current algorithms aimed at helping mission designers to identify good planetary encounter options for preliminary mission design. As shown in the case of the *Cassini-Huygens* mission, our approach was able to find the best sequence of encounters and a time line that is very close to the one flown by the real mission. As expected, in case of the *Rosetta* mission the suggested trajectory options were incomplete. The fly-by sequence actually used in the Rosetta mission was not found; although other good options were located by MCTS. This is due to the fact that employed MGA trajectory model, while computationally advantageous, is not capable to describe ΔV-EGA (ΔV - Earth Gravity Assist) maneuvers [Sims *et al.*, 1997]. These maneuvers are critical to the use of the fly-by sequence E–E–M–E–E–67P which was adopted for the *Rosetta* mission. This is also the reason why fixing the launch date produces a more difficult search problem than the open launch window. In the open launch window, a multitude of solutions with different flyby sequences can be found that score within the $\min \Delta V_t$ threshold.

Our results are consistent with past findings [Auer *et al.*, 2002; Browne *et al.*, 2012] with respect to the effectiveness of a well-tuned ϵ-greedy policy. Furthermore, our results show that UCB1-Tuned does not outperform UCB1. We employ the "max-style" node value estimator for all selection polices. UCB1-Tuned uses a measured variance to scale the upper bound which seems to be ineffective in combination with this max-estimator. We aim to address this issue in future work by using the variance of child node values as opposed to measured variance of the back-propagated values. In addition, the use of more sophisticated trajectory models and the inclusion of a second objective to score trajectories (i.e. total time of flight) are interesting future extensions of this work. We expect that the application of the MCTS paradigm would produce reliable and fast results also in these cases.

References

[Auer *et al.*, 2002] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[Auger and Hansen, 2005] A. Auger and N. Hansen. Performance evaluation of an advanced local search evolutionary algorithm. In *Congress on Evolutionary Computation (CEC 2005)*, volume 2, pages 1777–1784, Piscataway, NJ, USA, 2005. IEEE Press.

[Browne *et al.*, 2012] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *Computational Intelligence and AI in Games, IEEE Transactions on*, 4(1):1–43, 2012.

[Ceriotti and Vasile, 2010] Matteo Ceriotti and Massimiliano Vasile. MGA trajectory planning with an ACO-inspired algorithm. *Acta Astronautica*, 67(9):1202–1217, 2010.

[Chaslot *et al.*, 2008] Guillaume Chaslot, Sander Bakkes, Istvan Szita, and Pieter Spronck. Monte-carlo tree search: A new framework for game ai. In *AIIDE*, 2008.

[Deb *et al.*, 2007] Kalyanmoy Deb, Nikhil Padhye, Ganesh Neema, and V Adimurthy. Interplanetary trajectory optimization with swing-bys using evolutionary multi-objective optimization. *Lecture Notes in Computer Science*, 4683:26–35, 2007.

[Englander, 2013] Jacob Englander. *Automated trajectory planning for multiple-flyby interplanetary missions*. PhD thesis, University of Illinois at Urbana-Champaign, 2013.

[Gad and Abdelkhalik, 2011] Ahmed Gad and Ossama Abdelkhalik. Hidden genes genetic algorithm for multi-gravity-assist trajectories optimization. *Journal of Spacecraft and Rockets*, 48(4):629–641, 2011.

[Heaton *et al.*, 2002] Andrew F Heaton, Nathan J Strange, James M Longuski, and Eugene P Bonfiglio. Automated design of the europa orbiter tour. *Journal of Spacecraft and Rockets*, 39(1):17–22, 2002.

[Izzo *et al.*, 2007] Dario Izzo, Victor M Becerra, DR Myatt, Slawomir J Nasuto, and J Mark Bishop. Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories. *Journal of Global Optimization*, 38(2):283–296, 2007.

[Izzo *et al.*, 2014] Dario Izzo, Luís F. Simões, Chit Hong Yam, Francesco Biscani, David Di Lorenzo, Bernardetta Addis, and Andrea Cassioli. GTOC5: Results from the European Space Agency and University of Florence. *Acta Futura*, 8:45–55, 2014.

[Izzo, 2014] Dario Izzo. Revisiting Lamberts problem. *Celestial Mechanics and Dynamical Astronomy*, pages 1–15, 2014.

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, pages 282–293. Springer, 2006.

[Kuipers *et al.*, 2013] Jan Kuipers, Aske Plaat, JAM Vermaseren, and H Jaap van den Herik. Improving multivariate horner schemes with monte carlo tree search. *Computer Physics Communications*, 184(11):2391–2395, 2013.

[Longuski and Williams, 1991] James M Longuski and Steve N Williams. Automated design of gravity-assist trajectories to mars and the outer planets. *Celestial Mechanics and Dynamical Astronomy*, 52(3):207–220, 1991.

[Petropoulos *et al.*, 2000] Anastassios E Petropoulos, James M Longuski, and Eugene P Bonfiglio. Trajectories to jupiter via gravity assists from venus, earth, and mars. *Journal of Spacecraft and Rockets*, 37(6):776–783, 2000.

[Petropoulos *et al.*, 2014] Anastassios E. Petropoulos, Eugene P. Bonfiglio, Daniel J. Grebow, Try Lam, Jeffrey S. Parker, Juan Arrieta, Damon F. Landau, Rodney L. Anderson, Eric D. Gustafson, Gregory J. Whiffen, Paul A. Finlayson, and Jon A. Sims. GTOC5: Results from the Jet Propulsion Laboratory. *Acta Futura*, 8:21–27, 2014.

[Sabharwal *et al.*, 2012] Ashish Sabharwal, Horst Samulowitz, and Chandra Reddy. Guiding combinatorial optimization with uct. In *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems*, pages 356–361. Springer, 2012.

[Schadd *et al.*, 2008] Maarten PD Schadd, Mark HM Winands, H Jaap Van Den Herik, Guillaume MJ-B Chaslot, and Jos WHM Uiterwijk. Single-player monte-carlo tree search. In *Computers and Games*, pages 1–12. Springer, 2008.

[Sims *et al.*, 1997] Jon A Sims, James M Longuski, and Andrew J Staugler. V8 leveraging for interplanetary missions: Multiple-revolution orbit techniques. *Journal of Guidance, Control, and Dynamics*, 20(3):409–415, 1997.