Acta Futura 5 (2012) 39-52
DOI: 10.2420/AF05.2012.39

Acta Futura

Real-Time Planetary Landing Site Selection – A Non-Exhaustive Approach

Luís F. Simões [*]
Advanced Concepts Team, PPC-PF, European Space Agency, ESTEC, 2201 AZ Noordwijk, The Netherlands
CA3 – Computational Intelligence Research Group, UNINOVA, 2829-516 Caparica, Portugal

Clément Bourdarias [†]
Navigation Sensors & Image Processing, ASTRIUM Space Transportation, 78133 Les Mureaux CEDEX, France

Rita A. Ribeiro [‡]
CA3 – Computational Intelligence Research Group, UNINOVA, 2829-516 Caparica, Portugal

Abstract. In a spacecraft's Hazard Detection and Avoidance architecture, the Piloting function is tasked with selecting in real-time, during a descent onto a planetary surface, an adequate landing site which meets mission, safety and reachability requirements. These requirements are assessed by Hazard Mapping and Trajectory Planning functions, capable of producing for each candidate landing site a quantification of its quality according to multiple criteria. We developed a Piloting function, IPSIS, based on a dynamic multi-criteria decision making model, that uses a procedure based on fuzzy sets for normalizing and handling the uncertainty in this input data. For the identification of the best landing sites at each instant we use PSO-Tabu, a hybrid algorithm combining Particle Swarm Optimization and Tabu Search. This Site Selection process is aided by a Retargeting process that tracks during descent the ratings of identified high quality sites, enabling a more efficient exploration of the dynamic search space. We present here an experimental validation of this novel approach to landing site selection. Solutions of the highest quality are shown being consistently identified from evaluations of non-exhaustive subsets of visible alternatives, on an architecture seen achieving exceptional levels of performance in multiple hardware platforms.

1 Introduction

Future in-situ exploration missions, like the joint ESA/NASA Mars Sample Return, will need autonomous, precise and safe landing. Achieving that, will require the introduction of hitherto untested technologies.

Hazard Detection and Avoidance (HDA) architectures will be responsible for the selection, in real-time, of safe landing sites which meet mission, safety and reachability requirements. Safety requirements are assessed by a Hazard Mapping (or Hazard Detection) function, which uses imaging sensor(s), such as camera or LIDAR, to map the terrain in terms of hazards at touch-

[*]Corresponding author. E-mail: luis.felismino.simoes@esa.int, lfs@uninova.pt
[†]E-mail address: clement.bourdarias@astrium.eads.net
[‡]E-mail address: rar@uninova.pt

down (craters, rocks, high slopes, shadows, …). The reachability of potential landing sites is evaluated using a specific guidance function, called Trajectory Planning. Strong links with absolute navigation allow for taking into account mission constraints, e.g. to land as close as possible to a pre-defined site of high scientific interest. These functions provide the necessary inputs for producing spatial-temporal multi-criteria assessments of candidate landing sites' quality. Drawing from these metrics, a Piloting function monitors in real-time the suitability of its current target, and provides the autonomous decision-making for whether to retarget, and where to.

We developed an innovative Piloting function, IPSIS ("*Intelligent Planetary SIte Selection*") [1], based on a dynamic multi-criteria model [4], which uses a procedure based on fuzzy sets for normalizing this input data, while handling its inherent uncertainty [14, 21, 8]. The identification of the best landing sites at each instant is performed by PSO-Tabu, a novel hybrid algorithm combining Particle Swarm Optimization and Tabu Search [22, 20]. This Site Selection process is aided by a Retargeting process that tracks during descent the ratings of previously identified high quality sites, thus taking benefits from time redundancy for enabling a more efficient exploration of the dynamically changing solution space. Usage of this algorithm makes it possible to find an appropriate landing site without the need to exhaustively evaluate the quality of all visible alternatives. Consequently, a very high computational efficiency is achieved.

We present here the architecture implemented for HDA, built around this new IPSIS function. Special emphasis is given to those components that allow for efficient identification of the best candidate landing sites, from a minimal sample of the available alternatives. The optimality-efficiency trade-offs involved in such a setup are analyzed, based on extensive experimental evaluations on simulated landing scenarios.

Results are also given for the execution times on multiple hardware platforms, of all of the architecture's functions (Hazard Mapping, Trajectory Planning and Piloting). Two different Piloting functions are compared in these tests: IPSIS and IVN+. Unlike IPSIS, IVN+ implements the traditional approach of exhaustively evaluating all the landing sites visible from the lander at every given instant.

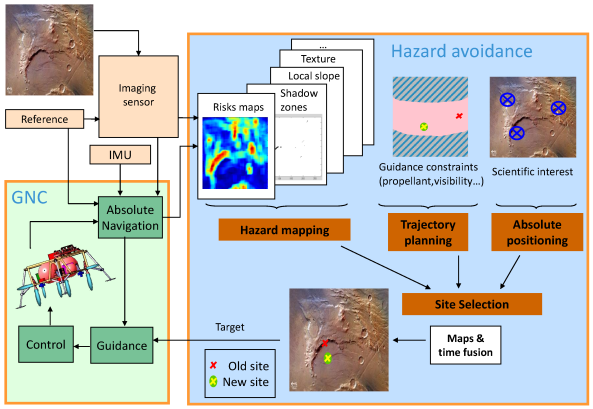

FIGURE 1. *Hazard Detection and Avoidance architecture and links with Guidance, Navigation and Control*

2 Planetary Landing Site Selection

The HDA subsystem is in charge of autonomously selecting a safe landing site in real-time during the landing phase on a distant body. On a planet with atmosphere, this landing phase begins after parachute jettisoning (on Mars, depending on the mission, this happens around the altitude of 2 km), as the vehicle uses its retro rockets to decrease its vertical velocity until touchdown.

The HDA system processes inputs from imaging sensors (typically a combination of camera, RADAR or LIDAR) and from the navigation function in order to provide a targeted site to the guidance. It is typically made up of the following components:

- Hazard Mapping (or Hazard Detection), which uses imaging sensor(s) to detect hazards;
- Trajectory Planning, which evaluates the possibility to reach the sites that are being mapped and estimates the fuel consumption;
- Landing site selection (or Piloting), which chooses the landing site according to hazard mapping, trajectory planning and other mission constraints like scientific interest or pre-defined landing area, using absolute navigation information as an input if available.

Connections between HDA and Guidance, Navigation and Control (GNC) are shown on Figure 1.

The goal of the Piloting function is to select in real-time an adequate landing site which meets mission, safety and reachability requirements:

DOI: 10.2420/AF05.2012.39

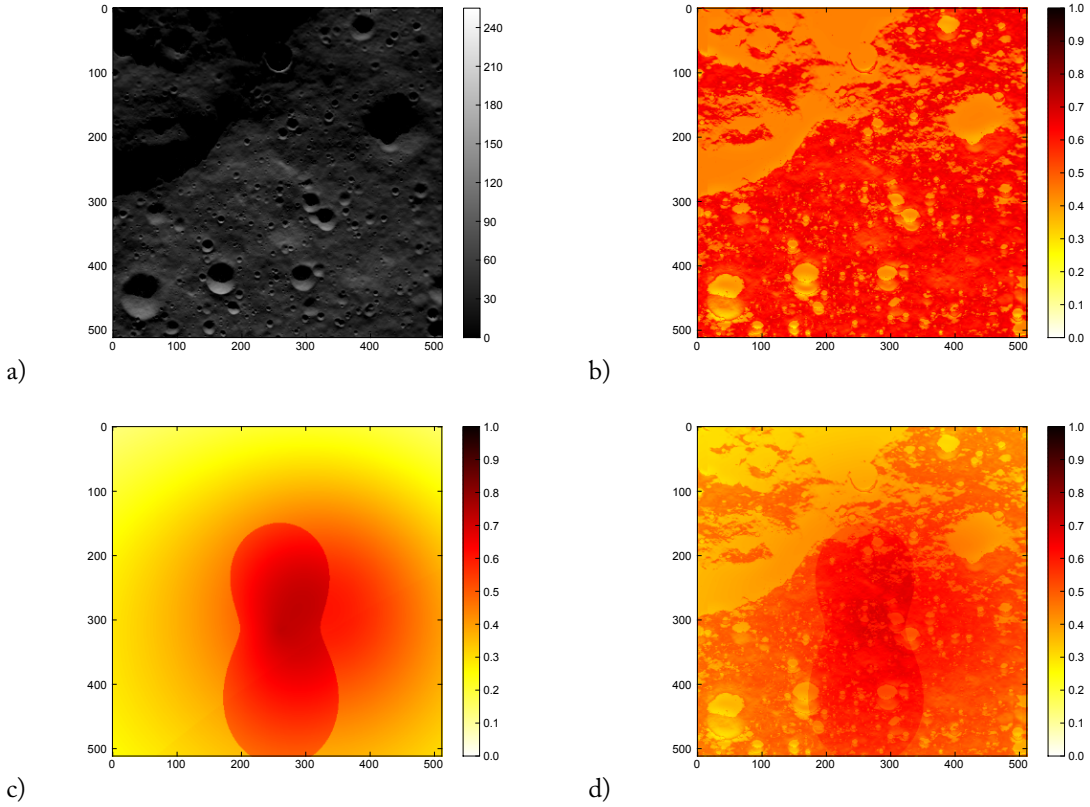

Figure 2. *Visualizations of the moon_02 landing scenario, at an altitude of approximately 650 m (iteration 33), including exhaustive evaluations of all visible sites' ratings according to different criteria. a) camera image, b) partial aggregation of the criteria representing terrain hazards (shadow, texture and slope), c) partial aggregation of the criteria representing guidance and mission constraints (fuel, reachability, distance and scientific interest), d) full aggregation of all the criteria rating landing site quality*

- the site must be compliant with mission constraints such as scientific interest, visibility from Earth etc.;
- the site must be safe with respect to local slope, illumination level and terrain roughness;
- the site must be reachable with the remaining fuel and the spacecraft propulsive capabilities;
- the site should remain visible to the imaging sensor throughout the descent so that its estimated characteristics can be continuously updated.

More precisely, the seven criteria considered in this paper and that will lead to the choice of a new target are:

shadows: derived from camera image grey levels;

slope: estimated through processing of either from camera, LIDAR or RADAR data;

texture or roughness: obtained for example from a multi-resolution variance analysis of the camera image;

fuel: needed to reach the considered target (trajectory planning algorithm);

reachability: of the considered target: feasibility of a retargeting, taking into account guidance limitations and camera visibility constraints along trajectory (trajectory planning algorithm);

distance: of the considered target to the current target;

scientific interest: distance to pre-defined landing sites with known absolute position (e.g. sites of scientific interest).

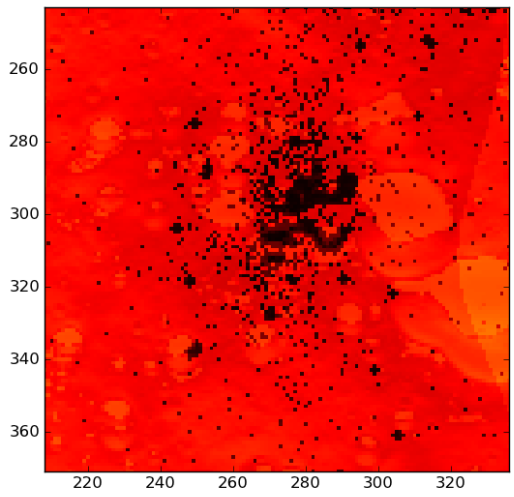

FIGURE 3. *Application of the non-exhaustive approach to landing site selection on the search space depicted in Figure 2d, here zoomed to the region centered on the global best site; sites evaluated during a search are shown in a darkened color*

Visualizations of hazard maps resulting from normalizing and aggregating [14] this input data can be seen in Figure 2.

3 Non-Exhaustive Approach to Site Selection

In our previous work [8] using a spatial-temporal (dynamic) multi-criteria decision making approach [4], pixels on the input hazard maps were seen as the alternatives undergoing selection, and were thus exhaustively evaluated. Though optimal in terms of identifying the landing sites with highest quality, such an approach is hardly feasible in a real-time computing environment with the foreseen CPU platforms (e.g. LEON3).

The solution introduced by UNINOVA and ASTRIUM Space Transportation [22] consists in the application of non-exhaustive search methodologies [3]. The aggregation of processed input values [14] produces a gradual variation in the ratings of neighboring landing sites (Figure 2), amenable to exploration by metaheuristic algorithms. Site selection thus becomes an optimization problem, in which one seeks to locate the site coordinates that maximize the rating function.

The multiple components of the proposed non-exhaustive approach to site selection, described in detail below, can be observed in operation in Figure 3: the reevaluation of tracked sites by Steepest Ascent Hill Climbing can be discerned in the small areas of contiguous evaluated sites, the movements of the particle performing a Tabu Search (including activations of its *teleportation* mechanism) are identifiable in the large areas of contiguous evaluated sites, and finally, Particle Swarm Optimization's global exploration of the search space is seen in the scattering of evaluated sites.

The approach followed here is in line with applications of the Swarm Intelligence paradigm to image analysis [19, 17]. In our model, the tabu list in the PSO-Tabu algorithm can then be seen as providing a form of stigmergic self-organization.

3.1 Particle Swarm Optimization

In the Particle Swarm Optimization [13, 5, 18] algorithm (PSO), a swarm of n particles collectively explores the search space. Each particle $i \in \{0, \ldots, n-1\}$ is defined by $\vec{x}_i$, its current position in the search space, $\vec{p}_i$, the best position the particle has visited so far, and $\vec{v}_i$, its velocity. At each step, each particle asynchronously updates its velocity vector, and afterwards updates its position in the search space with $\vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i$.

The core of the PSO algorithm lies in the equation used to update particles' velocities. Though several alternatives exist, the canonical algorithm uses the equation with "constriction coefficients" [6]:

$$\vec{v}_i \leftarrow \chi(\vec{v}_i + \vec{U}(0,\phi_1) \otimes (\vec{p}_i - \vec{x}_i) + \vec{U}(0,\phi_2) \otimes (\vec{p}_g - \vec{x}_i)),$$

where $\vec{U}(0,\phi_i)$ represents a vector of random numbers uniformly distributed in $[0,\phi_i)$, randomly generated at each step and for each particle, and $\otimes$ is the component-wise multiplication. The constriction factor χ is usually set to 0.7298, along with ϕ_1 and ϕ_2, also called the acceleration coefficients, both set to 2.05 [18]. $\vec{p}_g$ refers to the best position visited so far by the particle's neighbors. A commonly used swarm topology has particles arranged in a ring, connected to all particles up to a radius of k: particle i has as neighbors the particles in $\{(i+j) \bmod n : j = \pm 1, \pm 2, \ldots, \pm k\}$. We use here a swarm size of $n = 25$, and $k = 2$.

We initialize particles with no velocity, $\vec{v}_i = \vec{0}$, and in random initial positions, $\vec{x}_i = \vec{p}_i = \vec{U}(0, X_{max})$, where $X_{max} = 512$ stands for the dimensions of our search spaces (images of 512 by 512 pixels). Particles'

 DOI: 10.2420/AF05.2012.39

continuous positions are decoded into the problem's discrete domain by a simple, and computationally efficient, truncation prior to evaluation (in case of improvement, $\vec{p}_i$ is still overwritten with the continuous $\vec{x}_i$ position).

A maximum velocity V_{max} is imposed: any component of $\vec{v}_i$ going outside the $[-V_{max}, +V_{max}]$ range is brought back to the nearest valid value, before the particle's position update takes place. A value of $V_{max} = \frac{X_{max}-0}{2} = 256$ was set, following the empirical rule in [5]. Particles will still occasionally fly outside the search space. When that happens, a rating of 0 is assigned to the unfeasible position. Subsequent $\vec{v}_i$ updates will then quickly pull the particle back into the search space. For a tight control over the number of steps PSO is allowed to iterate over, these "evaluations" of positions outside the image still count towards the total budget for the execution.

3.2 Tabu Search

Tabu Search [11, 3] can be described as a local search method incorporating techniques for escaping local optima and avoiding cycling. A step of the algorithm can be described as follows: exhaustively evaluate all of the current solution's neighboring points, and select the best one from amongst those, that is not considered tabu; move there, and add the new current solution to the tabu list.

The tabu list contains all solutions that are considered tabu, and thus avoided as destinations. Solutions stay there for a period equal to the tabu tenure value, one of the system's parameters. In our implementation, this period is defined as a number of steps carried out by Tabu Search, and set to 250.

An aspiration criterion was implemented: when in one step all neighboring solutions are considered tabu, instead of terminating the search, the tabu status of the "less" tabu solution is revoked, i.e., the one that has been tabu for a longer period leaves the tabu list, and Tabu Search moves to it again.

As neighborhood structure, we consider the 8 pixels directly adjacent to a given one (all pixels at a Chebychev distance of 1), eventually discarding from the neighborhood pixels that would fall outside the image. A significant overlap exists between the pixels in the neighborhoods of consecutive solutions (2 or 4 out of the 8 pixels). In our implementation, a data cache keeps track of all landing sites evaluated in an iteration. By enabling spatial indexing (efficiently, in terms of time and space complexity), previously evaluated sites can be retrieved, and the wasted computation of reevaluations averted. Still, so as to exert a tight control over the execution's computational cost, multiple considerations of the same site during search all count towards the total budget of available site evaluations.

3.3 PSO-Tabu

Memetic algorithms are a class of metaheuristics in which a population-based optimization algorithm is hybridized with a local search procedure. Combining the global exploratory power of one, with the other's capacity for local refinement, this approach has proved to be extremely successful at solving many complex problems. For a study on the properties and performance of memetic algorithms that include PSO in their composition, see [16].

Having previously successfully applied both PSO and Tabu Search, in isolation, to the problem of landing site selection [22], we went on to combine them both into a memetic algorithm: PSO-Tabu. This algorithm follows the specification outlined in Section 3.1 for PSO, but differs in that now one of the particles behaves according to the Tabu Search specification outlined in Section 3.2. This tabu-particle has no velocity $\vec{v}_i$, but it does keep a memory $\vec{p}_i$ of the best solution it has visited up to that point.

The tabu-particle is tasked with performing at all times a local search on the best identified region of the search space. To that end, the swarm's topology is slightly altered: the ring topology described in Section 3.1 is preserved, but additional unidirectional connections are established from all the remaining particles towards the tabu-particle. That is to say, through its $\vec{p}_g$ the tabu-particle has at all times knowledge as to the best solution identified so far in the swarm. Also, solutions it identifies will influence the rest of the swarm's dynamics, by way of its $\vec{p}_i$, as it propagates back through the ring structure.

A *teleportation* mechanism enables the tabu-particle to make use of the information at its disposal: at the beginning of its update step, the tabu-particle checks whether a solution has been found in the swarm, that is better than the best it has previously visited. Should that be the case, the tabu-particle gets relocated to that solution, adds it to the tabu list, and then proceeds with its search from there. No other solutions have their tabu status affected as a consequence of this relocation, and should the tabu-particle later move towards a region it was previously exploring, it will avoid it if its tabus have

not yet expired.

3.4 Improving optimization in a dynamic and noisy environment

As the spacecraft descends, the search space for site selection gradually transforms itself, as new surface details become discernible. Uncertainties in hazard estimation decrease, but are ever present, and need to be handled [12]. The ability to convert between landing sites' image (2D) and terrain (3D) coordinates enables a mapping of alternatives across different iterations (though with some error), and this feature is then exploited to enhance the Piloting function's performance.

The solution implemented in IPSIS generates and makes use of a memory of distinct high quality alternatives in the search space: 20 sites from amongst the best found, having distances between themselves exceeding a parameterized amount, have their ratings tracked across multiple iterations of the descent (for reasons of computational efficiency, the data cache is reset between iterations, and only this smaller memory is carried over). Because multiple ratings can be averaged for the same tracked site, some of its ratings' noise can be canceled. Different sites will enter and leave this memory during descent, but by a process that favors the accumulation of knowledge, over instantaneous overwriting with the current, potentially over-estimated, best solutions.

At the beginning of each iteration, multiple Steepest Ascent Hill-Climbing (SAHC) applications explore the regions around the current image coordinates of each tracked alternative. SAHC is a local search method that at each step evaluates all neighbors to the current solution, and then deterministically moves to the best one if its quality exceeds that of the current solution. It uses here a von Neumann neighborhood, which comprises the 4 pixels at a Manhattan distance of 1 from a given pixel. Each SAHC application stops upon locating the nearest optima, or upon reaching the bounds of the region defined for that search, which is dynamically sized so as to match the magnitude of the errors in projection of coordinates. By climbing the local gradient after a search space change, this procedure will be either tracking optima that are changing due to more surface characteristics becoming visible, or will be repairing the current image coordinates of the terrain coordinates that had been stored for the tracked alternative.

Upon completing the reevaluation stage, we have, with high probability, identified at least one high quality site in the new search space. Assuming the search algorithm being applied for site selection is one of PSO or PSO-Tabu, this knowledge is then used in the particles' initialization stage: a single particle in the swarm will be initialized not at random, like the remaining particles, but at the best alternative observed during the reevaluation stage (in the case of PSO-Tabu, the tabu-particle will be the one thus initialized).

4 Experimental Analysis

This section evaluates the risk to the spacecraft of failure by the Piloting function in identifying appropriate landing sites.

A validation campaign was carried out with the goal of assessing the optimality-efficiency tradeoffs involved in the transition from an exhaustive to a non-exhaustive approach to landing site selection. In addition, we address the research questions of measuring, for this problem, the benefits gained by hybridizing the global search algorithm with a local search one, and by handling the search problems' dynamic nature through the incorporation of a memory mechanism.

4.1 Experimental Design and Setup

Simulated landings were performed on four different scenarios, depicted in Figure 4. Terrain images were generated by PANGU [15], a scene generator for interplanetary terrains developed by the University of Dundee in the frame of an ESA contract. A camera resolution of 512 by 512 pixels was used across all scenarios, and the non-exhaustive Site Selection process configured to evaluate only 1% of those pixels per iteration (2621 candidate landing sites). In our tests, the Piloting function runs at 1 Hz and there are approximately 40 to 60 iterations in a descent.

Simulations were carried out in open-loop, meaning there was no connection between the output of HDA and the input of Guidance: the trajectory is precomputed and remains fixed throughout the descent[1]. This ensures the search problems encountered by different Piloting configurations at the same point of the descent on a given scenario always match exactly.

The conducted experiments tested both landing with no tracking of sites, and with it. The first setup would not be used in a real setting, but is tested here to assess the mechanism's contribution to solutions' quality.

[1]For an evaluation of IPSIS in closed-loop simulations, see [1].

 DOI: 10.2420/AF05.2012.39

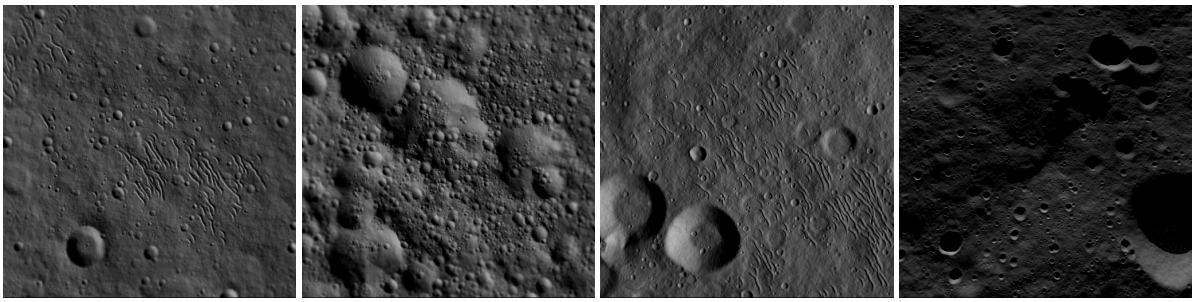

Figure 4. *Landing scenarios considered in the experiments (camera images at altitudes of approximately 2 km, time of the first call to the Piloting function). From left to right: craters_dunes, craters_only, mars_01, moon_02*

In both setups, constant explicit reevaluation of the lander's current target was switched off. This way, there is no bias simplifying the search process at the very first iteration, and only knowledge acquired during reevaluation of tracked sites will end up being used in the Site Selection process' initialization stage[2]. Site reevaluation, though application of SAHC on a bounded region around each tracked site, incurs a cost in terms of site evaluations: some of the total number of evaluations available for the whole Piloting execution is used up, decreasing the budget available for the Site Selection process. The conducted experiments sought a better understanding of the costs and benefits inherent in the tracking of alternatives.

At lower altitudes, the terrain area represented by a pixel becomes smaller than the lander's area. Evaluation of a landing site's quality then requires consideration of multiple pixels at once. This is handled in IPSIS by a Regions Aggregation process: all pixels in a region matching the lander's area are evaluated, and their ratings aggregated into a single value [21], which is then assigned to the alternative represented by the region's central pixel. With only 1% of the pixels being evaluated per iteration, the Regions Aggregation process effectively reduces the number of landing sites that can be evaluated in iterations at lower altitudes. So as to provide the Site Selection process with uniform conditions across all iterations in these experiments, the Regions Aggregation process was switched off. All search problems are then being solved through a number of site evaluations equal to 1% of the number of pixels. Because at low altitudes a single pixel's rating might become highly dissociated from the rating of the whole region centered on it, and erroneously conflicting with ratings being tracked for the same terrain coordinates by the Retargeting process, the Piloting function is then applied in these tests only down to altitudes of approximately 325 m (resulting in 31, 31, 36 and 46 iterations per landing scenario, respectively). By that point, both the chosen landing site and its alternatives are considered to have already been thoroughly mapped, and Piloting gradually drops the role of global exploration for superior alternatives, to focus exclusively on the monitoring for changes in the rating of the lander's current target, commanding a retargeting towards one of the best alternatives it has been tracking should it deem it appropriate.

In each iteration, landing sites will have ratings ranging in different values, depending on the specific conditions encountered in them. Identifying a landing site with a rating of 0.6 might in one iteration mean the optimal landing site was found, while in some other such a rating might be very far from optimal. So as to obtain comparable performance measures, across iterations and landing scenarios, "Oracles" were generated for all search problems on which Piloting was executed. An Oracle is an exhaustive evaluation of all sites visible in a given iteration, that is done separately from the processes involved in the Piloting function, but using the data preparation processes defined for it. By computing the Oracles, two measures can then be used for evaluating the quality of a site in global terms:

Rank A site's rank is defined as the number of sites on the map better than itself. Special handling is required when assigning ranks to multiple sites having the exact same rating. Example: if sorting a

[2]In landings where no alternatives are tracked, all particles are initialized uniformly at random in the multiple search spaces. When tracking alternatives, one of the particles (the Tabu Search particle, if PSO-Tabu is being used) is initialized at the coordinates of the best site identified during the reevaluation stage.

list of sites' ratings by decreasing order results in $[0.9, 0.8, 0.8, 0.7]$, then the corresponding sites are assigned the ranks $[0, 1, 1, 3]$. Given the image resolution being used (512 by 512 pixels), ranks can then take values in $\{0, ..., 512^2 - 1\}$;

Normalized Rating Having knowledge of the ratings of the global best and worst sites, ratings can be normalized, so that a rating of 1.0 always corresponds to the best site(s) and 0.0 to the worst site(s).

These two measures provide different insights into performance. A site may have a very bad rank, and yet have a normalized rating close to optimal, because all sites better than it have a very similar rating. Conversely, sites having bad normalized ratings, and yet good ranks will be characteristic of search problems where fewer good solutions exist.

Given the stochastic nature of Particle Swarm Optimization, 1000 landings were simulated per scenario. So as to more accurately pinpoint differences in performance to the actual differences between Piloting configurations, the same random number generator seeds were shared across executions tackling the same search problems[3]. Aggregation of performance values across Piloting executions on search problems encountered in the experiments was achieved through the following measures:

Optimality Rate (OR): fraction of executions in which the solution with rank 0 (the global optimum) was found;

Mean Best Quality (MBQ): average, over the executions being considered, of the best found solution's quality;

Average Evaluations to Optimal (AEO): average number of site evaluations required in order to reach the optimal solution (discards executions in which the optimal was not found);

Worst Case (WC): quality of the best solution found in the execution where worst performance was observed.

[3]A matrix of random number generator seeds was defined for the experiments, having dimensions of *1000 landings* $\times$ *46 iterations*. This matrix was shared between experiments with different Piloting configurations, and as well between experiments on different landing scenarios.

These are standard performance measures used in the evolutionary computing literature [9, ch. 14], that are here slightly renamed for clarity. A fifth measure was taken, "Average Evaluations to Convergence", identical to AEO but averaging over all the considered executions the number of evaluations required in order to reach the solution, optimal or not, upon which the algorithm was then not able to improve on anymore. Its results were to a large extent identical to those of the AEO measure, and are therefore ignored here, in favor of the more widely used AEO measure.

4.2 Results

Tables 1 and 2 present the experimental results evaluating performance of the IPSIS Piloting function, broken down so as to enable an analysis of the contribution given by different elements of the architecture. Table 1 compares the performance of PSO both in isolation, and when exploiting at every iteration knowledge acquired in preceding search problems. Table 2 presents matching results, but using instead PSO-Tabu in the Site Selection process. Figures 5, 6 and 7 present the empirical cumulative distribution functions related to the MBQ (in terms of rank and normalized rating) and AEO measures, resulting from the aggregate results of the 1000 landings on the four scenarios, when using the four different Piloting configurations.

Overall, the hybridization of PSO with Tabu Search improved the OR by approximately 1%. In terms of solution quality there was little to no improvement, as can be seen in the overlapping curves in Figures 5 and 6. Looking at Tables 1 and 2, when no alternatives are tracked, the addition of Tabu Search actually led to a slight worsening in MBQ_rank and WC (the reverse is observed when alternatives are being tracked; see Figure 5). Tabu Search's biggest contribution comes from the marked improvement in convergence speed, with a reduction in AEO on the order of 15% fewer site evaluations. This all the more clear in Figure 7.

As might be expected, initializing the search for the best landing site in the presence of a degree of knowledge about the search space's structure leads to very significant gains in performance. Overall, OR increases by approximately 10%, with a remarkable increase of over 20% seen in the craters_dunes landing scenario (where AEO more than halves). Convergence speed is equally greatly improved, with AEO seeing overall a reduction of approximately 25% in the number of site evaluations required to find the optimal site.

 DOI: 10.2420/AF05.2012.39

Table 1. ***Particle Swarm Optimization*** *performance in multiple landing scenarios (values taken over* 1000 *descents per scenario;* ranks *take values in* $\{0, ..., 512^2 - 1\}$*, 0 being optimal, and* normalized ratings *range in* $[0, 1]$*, 1 being optimal)*

landing scenario	number of Piloting executions	optimality rate	mean best quality		average evaluations to optimal	worst case	
			rank	normalized rating		rank	normalized rating
No tracking (consecutive search problems during descent are treated independently)							
craters_dunes	31000	0.440	3.08 ± 7.27	0.9897 ± 0.0146	1361 ± 599	442	0.9143
craters_only	31000	0.666	0.75 ± 1.65	0.9975 ± 0.0050	1104 ± 505	69	0.9571
mars_01	36000	0.877	0.20 ± 0.72	0.9998 ± 0.0006	1089 ± 479	17	0.9913
moon_02	46000	0.765	0.73 ± 3.49	0.9994 ± 0.0016	1250 ± 528	214	0.9779
aggregate results	144000	0.702	1.11 ± 4.13	0.9970 ± 0.0082	1185 ± 528	442	0.9143
20 alternatives tracked (the problem's dynamic nature is exploited)							
craters_dunes	31000	0.681	0.65 ± 1.35	0.9955 ± 0.0095	587 ± 754	20	0.9479
craters_only	31000	0.779	0.34 ± 0.91	0.9987 ± 0.0038	703 ± 641	59	0.9571
mars_01	36000	0.887	0.17 ± 0.62	0.9999 ± 0.0005	1024 ± 563	46	0.9913
moon_02	46000	0.806	0.40 ± 1.15	0.9996 ± 0.0012	1122 ± 629	25	0.9879
aggregate results	144000	0.793	0.38 ± 1.06	0.9986 ± 0.0051	907 ± 675	59	0.9479

Table 2. ***PSO-Tabu*** *performance in multiple landing scenarios (values taken over* 1000 *descents per scenario;* ranks *take values in* $\{0, ..., 512^2 - 1\}$*, 0 being optimal, and* normalized ratings *range in* $[0, 1]$*, 1 being optimal)*

landing scenario	number of Piloting executions	optimality rate	mean best quality		average evaluations to optimal	worst case	
			rank	normalized rating		rank	normalized rating
No tracking (consecutive search problems during descent are treated independently)							
craters_dunes	31000	0.444	3.25 ± 8.92	0.9894 ± 0.0149	1139 ± 681	442	0.9134
craters_only	31000	0.695	0.75 ± 1.84	0.9977 ± 0.0050	941 ± 614	78	0.9553
mars_01	36000	0.885	0.22 ± 0.87	0.9999 ± 0.0006	943 ± 561	46	0.9895
moon_02	46000	0.779	0.79 ± 4.06	0.9994 ± 0.0017	1133 ± 596	214	0.9779
aggregate results	144000	0.716	1.17 ± 4.96	0.9970 ± 0.0084	1035 ± 609	442	0.9134
20 alternatives tracked (the problem's dynamic nature is exploited)							
craters_dunes	31000	0.673	0.65 ± 1.34	0.9953 ± 0.0097	425 ± 665	29	0.9479
craters_only	31000	0.824	0.28 ± 0.82	0.9989 ± 0.0033	556 ± 616	32	0.9606
mars_01	36000	0.903	0.15 ± 0.59	0.9999 ± 0.0005	860 ± 614	15	0.9913
moon_02	46000	0.828	0.37 ± 1.10	0.9997 ± 0.0011	1001 ± 669	18	0.9852
aggregate results	144000	0.812	0.36 ± 1.02	0.9986 ± 0.0051	762 ± 680	32	0.9479

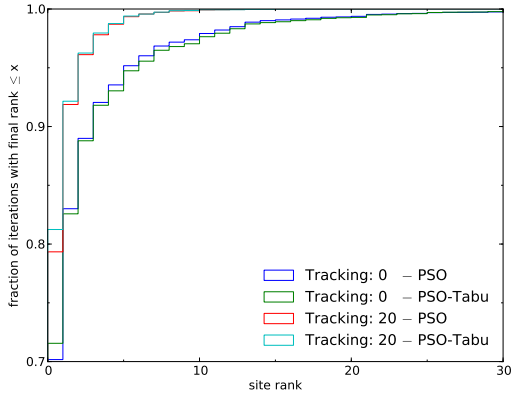

Figure 5. *Rank of the best site identified in each Piloting execution (empirical cumulative distribution functions)*

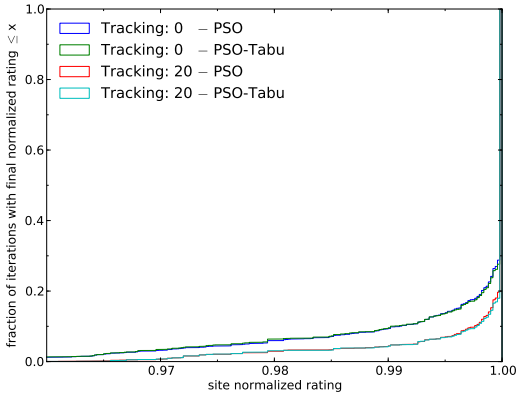

Figure 6. *Normalized rating of the best site identified in each Piloting execution (empirical cumulative distribution functions)*

As can be seen in Figure 7, already in the reevaluation stage, within the median of 146 site evaluations consumed by SAHC, over 20% of Piloting executions were able to locate the iteration's optimal site. When Tabu Search was then put to search locally in the vicinity of the best of the tracked sites, a distinct behavior was observed, with a significant fraction of Piloting executions being able to locate the optimal site within a few hundreds of site evaluations. In contrast, PSO takes longer to make effective use of the same initial information, given particles initially have high velocities and thus overshoot the known good site, in favor of a more global search.

Figures 5 and 6 show that when using PSO-Tabu, and the best identified solutions are tracked throughout the descent, there are negligible chances of a Piloting execution ending up with solutions that are not amongst the 10 best, or having normalized ratings worse than 0.96.

4.3 Discussion and new avenues

Though these results are by no means exhaustive, they show a Piloting function capable of discarding 99% of the evaluations carried out by exhaustive approaches, while still identifying solutions of the highest quality. We have shown this to be the case in four distinct landing scenarios, posing different challenges to the non-exhaustive Site Selection process. Even in craters_dunes, the scenario that posed the greatest challenge, all measures show the problem solution being perfectly approximated, when Piloting is executed making use of the local search and memory mechanisms.

Optimality is obviously desired, but there is actually in this problem a degree of tolerance for sub-optimal solutions. We envision up to two corrective maneuvers (retargetings) being performed in a descent (in 2 of the 40 to 60 total iterations). Site Selection's outputs are therefore not acted upon at every iteration, but rather continuously supplied to the Retargeting process, as sets of high quality landing sites for it to choose from should at some point in the descent the current target be found to no longer be suitable. Momentary failures in identification of the optimal landing site can therefore be tolerated, and improved upon in the following iterations, making this approach all the more suitable.

A possibility for future improvement will be the addition of mechanisms that explicitly search for distinct sets of top rated sites [2], as a way to better support the Retargeting process' tracking of alternatives. Currently, these are selected from amongst the sites evaluated in the search for the optimal landing site.

The great improvements in convergence speed brought by the mechanisms enhancing the Site Selection process acquire added relevance in a scenario such as ours where real-time responsiveness is critical. Should external factors force the Piloting function to provide a solution in a shorter amount of time than normal, Tabu Search and the reevaluation of tracked alternatives will have enabled this solution to be of a higher quality.

Random-restarts are frequently used in heuristic op-

 DOI: 10.2420/AF05.2012.39

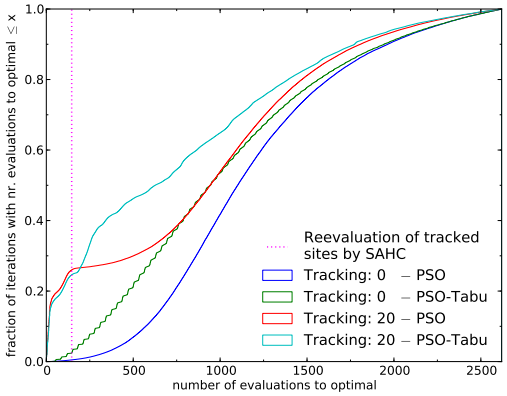

Figure 7. *Number of site evaluations required to reach the optimal solution, in those Piloting executions where it was found (empirical cumulative distribution functions)*

timization algorithms, so as to reintroduce lost diversity when the algorithm has converged on a region of the search space, and thus increase the likelihood of a superior solution being still found. We currently perform random-restarts, but synchronously with switches between descent iterations, and therefore with changes to the search space. The convergence time measurements indicate there is room for applying random-restarts also within the same iteration, possibly leading to yet additional gains in solution quality. With Tabu Search handling local search around the best found solutions, PSO can be safely given a more exploratory behavior. The results obtained with the current experiments could be used for implementing a restart scheduling strategy having optimal expected utility, as described in [10]. Alternatively to striving for increasing performance in terms of solution quality, these results point also to the suitability of increasing performance in terms of computational cost: the number of pixel evaluations could be safely reduced by a significant margin, with small degradation to solution quality. Doing so, the Tabu Search component would be expected to provide a greater contribution to solution quality than it does now. Yet another possibility these results indicate as being feasible, would be to increase the resolution at which candidate landing sites are mapped.

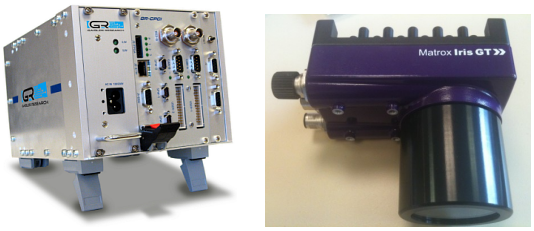

Figure 8. *Hardware test benches. Left: GR-RASTA LEON3 platform. Right: Matrox Iris GT300 camera*

5 Hardware Implementation & Profiling

To evaluate the benefits of IPSIS over more classical solutions, software profiling was done on specific hardware to measure the execution times.

5.1 Hardware test benches

Two hardware test benches were considered. The first one is a LEON3 board, supported by ESA and representative of future space hardware [7]. The second one is a commercial off-the-shelf "smart camera" from Matrox.

The GR-RASTA is used as a hardware platform for the implementation of the LEON3 system together with IP cores which implement multiple bus interfaces (Figure 8 left image). The LEON3 core is a 32-bit synthesizable processor core based on the SPARC V8 architecture. LEON3 is distributed as part of the GRLIB IP library, and the library contains LEON3 template designs for several popular FPGA prototyping boards. In the test bench, the tested algorithm is coupled with a real-time operating system (RTOS) which provides evaluation tools such as a timer for time measurements.

The Matrox Iris GT300 (Figure 8 right image) is a smart camera with a monochrome 640×480 CCD sensor, 1.6 GHz Intel Atom CPU, 256 MB RAM. It runs an embedded version of Windows XP. The camera is plugged to the host machine using an Ethernet link: the binary can be copied directly onto the 1GB flash disk and executed.

5.2 Software description

The complete HDA function (Hazard Mapping, Trajectory Planning and Piloting) was executed in open-loop in two different versions:

- IPSIS, relying on a non-exhaustive search methodology where the criteria values are evaluated only at

explored pixels;

- IVN+, using a more traditional approach where the criteria values are evaluated for the whole images, and used as a reference here.

IVN+ is a complete HDA function developed by ASTRIUM Space Transportation and implemented in the ESA "EAGLE" simulator. It is an evolution of the HDA function that was developed for the ESA IVN (*"Integrated Vision and Navigation"*) simulator [23]. Unlike IPSIS, its Piloting function evaluates exhaustively all the landing sites visible from the lander at a given instant.

IPSIS being only the Piloting part of HDA, the IVN+ Hazard Mapping and Trajectory Planning functions were used for building a complete HDA system, so that the two solutions could really be compared. When working together with IVN+, hazard mapping and trajectory planning are executed for all pixels. On the contrary, in the IPSIS architecture, the Piloting process becomes the director of which pixels to evaluate: the IVN+ Hazard Mapping and Trajectory Planning functions are then computing their results only at these specific positions.

The exact same code was compiled on the two hardware test benches. This code was extracted from the HAL (*"Hazard Avoidance and Landing"*) simulator, developed at ASTRIUM Space Transportation, and written in C++ with a strict compliance to the ISO standard to ensure portability.

5.3 Profiling results

The interest of the profiling lies here in the relative execution times between IPSIS and IVN+. Absolute execution times are strongly linked with code optimizations and with the chosen Hazard Mapping and Trajectory Planning algorithms, which are not the main aspects of interest for this study dedicated to the Piloting architecture. These absolute execution times are still given as they can be informative.

We present in Table 3 the average execution time for one iteration of the HDA. The average is taken over the whole trajectory from 2 km above ground level to touchdown (this landing phase lasts for approximatively 60 seconds). HDA frequency was set to 1 Hz, which means that the measured execution time shall be less than 1 second for a real-time execution. As before, we used images of 512 by 512 pixels generated by PANGU.

Table 3. *Average execution times per iteration, of the complete HDA system (Hazard Mapping, Trajectory Planning and Piloting), on different hardware platforms*

Piloting function used	hardware test bench		
	LEON3	Matrox Iris GT300	PC
IVN+	659 s	4.33 s	1.31 s
IPSIS	22 s	0.11 s	0.03 s
IPSIS as percentage of IVN+ time	3.3%	2.5%	2.3%
IPSIS speedup w.r.t. IVN+	30×	39×	44×

Execution times measured on a desktop PC (Intel Pentium E2200, 2.2 GHz) are also included in the table for comparison purposes.

When using the new architecture, one can clearly see the benefits of IPSIS: the total execution time for HDA is 30 to 44 times less compared to the standard IVN+ solution, depending on the platform. We remind that we are exploring only 1% of the pixels, so a gain of 100 compared to the old architecture could have been predicted. However, currently some computations still need to be done on the whole image on the Hazard Mapping algorithms, like the image reduction for roughness estimation. Also, IPSIS and IVN+ implement different strategies for evaluating site quality (fuzzy logic and evidence theory, respectively), having different computational costs per pixel. Finally, though small, the coordination of which sites to evaluate in the non-exhaustive approach does incur a computational cost.

These results show that the HDA system built around IPSIS is already capable of real-time execution on the Matrox camera, even when using the IVN+ Hazard Mapping and Trajectory Planning functions, that are not optimized at all for real-time processing.

This is not true on the LEON3. Still, it is important to note that the LEON3 hardware used in this study is running on a FPGA at 50 MHz and is thus not completely representative of the "real" LEON3 performance. Running on an ASIC SCOC3 at 80 MHz would bring a noticeable performance gain. Optimization of Hazard Mapping and Trajectory Planning will also permit an important gain. If not sufficient, the most demanding algorithms, like image processing, will have to be executed on a dedicated hardware accelerator (e.g.

 DOI: 10.2420/AF05.2012.39

FPGA).

6 Conclusion

The risk in using a non-exhaustive approach for the selection of landing sites during a descent onto a planetary body lies in the quality of identified solutions that is traded-off for the achievement of the necessary reductions in computational cost. Extensive validation experiments comparing executions of the described Site Selection process, with exhaustive evaluations of the same search spaces, showed it to be capable of consistently producing solutions of the highest quality, at a fraction of the computational cost. The very worst solutions identified in these experiments still provided adequate landing sites. The combination of the global exploratory power of Particle Swarm Optimization, with the improved local search given by Tabu Search, plus the tracking of multiple sites across the changing search space, proved crucial in the attainment of these performance levels.

The introduction of the new HDA architecture built around the IPSIS Piloting function shows impressive performance gains (30 to 44 times faster by comparison with IVN+) on different hardware platforms. Future steps include testing IPSIS with other Hazard Mapping and Trajectory Planning functions optimized for real-time execution on the LEON3 processor, or selecting algorithms to be implemented on a dedicated hardware accelerator, so as to obtain real-time feasibility on hardware likely to be used in future missions.

Acknowledgments

This work was partially financed by EADS ASTRIUM Space Transportation under contract ASTRIUM-4572019617, and by ESA under contract ESTEC 21744/08/NL/CBI.

References

[1] C. Bourdarias, P. Da-Cunha, R. Drai, L. F. Simões, and R. A. Ribeiro. Optimized and flexible multi-criteria decision making for hazard avoidance. In *33rd Annual AAS Rocky Mountain Guidance and Control Conference*, Breckenridge, Colorado, February 2010. American Astronautical Society.

[2] R. Brits, A. P. Engelbrecht, and F. van den Bergh. Locating multiple optima using particle swarm optimization. *Applied Mathematics and Computation*, 189(2):1859–1883, June 2007.

[3] E. K. Burke and G. Kendall, editors. *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. Springer, 2005.

[4] G. Campanella and R. A. Ribeiro. A framework for dynamic multiple-criteria decision making. *Decision Support Systems*, 52(1):52–60, Dec. 2011.

[5] M. Clerc. *Particle Swarm Optimization*. ISTE, London, Feb. 2006.

[6] M. Clerc and J. Kennedy. The particle swarm–explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, Feb. 2002.

[7] S. De Florio, E. Gill, S. D'Amico, and A. Grillenberger. Performance comparison of microprocessors for space-based navigation applications. In *7th IAA Symposium on Small Satellites for Earth Observation*, Berlin, Germany, May 2009. International Academy of Astronautics. IAA-B7-0915P.

[8] Y. Devouassoux, S. Reynaud, G. Jonniaux, R. A. Ribeiro, and T. C. Pais. Hazard avoidance developments for planetary exploration. In *GNC 2008: 7th International ESA Conference on Guidance, Navigation & Control Systems*, Tralee, Ireland, June 2008.

[9] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Natural Computing series. Springer, Berlin / Heidelberg, 2003.

[10] A. S. Fukunaga. Restart scheduling for genetic algorithms. In A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN V*, volume 1498 of *Lecture Notes in Computer Science*, pages 357–366, Berlin / Heidelberg, 1998. Springer.

[11] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, 1997.

[12] Y. Jin and J. Branke. Evolutionary optimization in uncertain environments–a survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, June 2005.

[13] J. Kennedy, R. C. Eberhart, and Y. Shi. *Swarm Intelligence*. Morgan Kaufmann Series in Evolutionary Computation. Morgan Kaufmann, 2001.

[14] T. C. Pais, R. A. Ribeiro, and L. F. Simões. Uncertainty in dynamically changing input data. In D. Ruan, editor, *Computational Intelligence in Complex Decision Systems*, volume 2 of *Atlantis Computational Intelligence Systems*, chapter 2, pages 47–66. Atlantis Press, Paris, France, June 2010.

[15] S. M. Parkes, I. Martin, M. Dunstan, and D. Matthews. Planet Surface Simulation with PANGU. In *Eighth International Conference on Space Operations (SpaceOps 2004)*, Montreal, Canada, May 2004. American Institute of Aeronautics and Astronautics.

[16] Y. G. Petalas, K. E. Parsopoulos, and M. N. Vrahatis. Memetic particle swarm optimization. *Annals of Operations Research*, 156(1):99–127, Dec. 2007.

[17] R. Poli. Analysis of the publications on the applications of particle swarm optimisation. *Journal of Artificial Evolution and Applications*, 2008:1–10, Jan. 2008. Article ID 685175.

[18] R. Poli, J. Kennedy, and T. Blackwell. Particle swarm optimization - an overview. *Swarm Intelligence*, 1(1):33–57, June 2007.

[19] V. Ramos and F. Almeida. Artificial ant colonies in digital image habitats - a mass behaviour effect study on pattern recognition. In M. Dorigo, M. Middendorf, and T. Stützle, editors, *Proceedings of ANTS'2000 - From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms*, pages 113–116, Brussels, Belgium, Sept. 2000.

[20] S. Reynaud, M. Drieux, C. Bourdarias, C. Philippe, L. F. Simões, and B. V. Pham. Science driven autonomous navigation for safe planetary pin-point landing. In *3rd European Conference for Aero-Space Sciences (EUCASS 2009)*, Versailles, France, July 2009. EUCASS2009-148.

[21] R. A. Ribeiro, T. C. Pais, and L. F. Simões. Benefits of full-reinforcement operators for spacecraft target landing. In S. Greco, R. A. Marques Pereira, M. Squillante, R. R. Yager, and J. Kacprzyk, editors, *Preferences and Decisions: Models and Applications*, volume 257 of *Studies in Fuzziness and Soft Computing*, pages 353–367. Springer, Berlin / Heidelberg, 2010.

[22] L. F. Simões, T. C. Pais, R. A. Ribeiro, G. Jonniaux, and S. Reynaud. Search methodologies for efficient planetary site selection. In *IEEE Congress on Evolutionary Computation (CEC 2009)*, pages 1981–1987. IEEE Press, May 2009.

[23] S. E. Strandmoe, T. Jean-Marius, and S. Trinh. Toward a vision based autonomous planetary lander. In *AIAA Guidance, Navigation, and Control Conference and Exhibit, Portland, OR, Aug. 9-11, 1999, Collection of Technical Papers. Vol. 2 (A99-36576 09-63)*, pages 1134–1144. American Institute of Aeronautics and Astronautics, 1999. AIAA-1999-4154.

 DOI: 10.2420/AF05.2012.39