JOURNAL OF SPACECRAFT AND ROCKETS
Vol. 59, No. 3, May–June 2022

Landing Site Selection Using a Geometrically Conforming Footprint on Hazardous Small Bodies

Joshua D. Nelson* and Hanspeter Schaub†
University of Colorado Boulder, Boulder, Colorado 80309

https://doi.org/10.2514/1.A35145

In recent years there has been a great deal of research and development pertaining to the landing of spacecraft on small bodies, such as asteroids. The capabilities to identify and avoid large rocks and other hazards on the surface of small bodies have seen significant improvement. However, many current techniques search for a location on the surface that contains no hazards within a scaled square, circular, or elliptical footprint. A challenge with this approach on asteroids with highly hazardous terrain is that such acceptable landing locations may be few and far between, or may not even exist at all. This paper proposes the use of a geometrically conforming footprint to significantly widen possible landing regions. An optimization technique that uses such a noncircular/elliptical footprint is formulated for a landing location selection algorithm. Coarse and fine variations for determining a landing location are developed and compared for their landing site selection performance as well as their computational effort. The algorithms to find a landing pose close to some desired landing location are constructed assuming a polygonal model of the lander and the obstacle regions. Numerical simulations illustrate the advantages of the geometrically conforming footprint over a circular one. Further, while the fine search algorithm shows better results in placing the lander close to an obstacle, the coarse search algorithm shows comparatively strong results with significantly less computational effort.

I. Introduction

SINCE the turn of the 21st Century, scientific interest in landing spacecraft on small celestial bodies has been on the rise. To further understand the origin and makeup of the solar system, spacecraft are being sent out to these bodies to perform in situ analysis or, more recently, sample extraction and return. Several missions, such as Rosetta, Hayabusa2, and OSIRIS-REx, have been mounted where the entry, descent, and landing (EDL) phase is critical to mission success. Missions such as these have shown that comets and asteroids contain a large amount of hazardous terrain, such as large rocks and steep slopes (Fig. 1), often in the most scientifically interesting regions [1]. Due to the long ground communication delay to Earth, EDL operations have a limited real-time human input and thus require a certain level of autonomy. The process of locating a safe place to land, known as hazard detection and avoidance (HDA), becomes challenging due to the abundance of hazardous terrain. In 2006 NASA launched the Autonomous Landing and Hazard Avoidance Technology (ALHAT) project as a means of fostering innovation in HDA technologies [2]. The primary goal of ALHAT was to develop autonomous landing site selection techniques that can satisfy any given safety and accuracy requirements. ALHAT has recently been succeeded by NASA's Safe and Precise Landing–Integrated Capabilities Evolution (SPLICE) project, which seeks to further innovate upon many of the HDA techniques introduced since the launch of ALHAT [3]. NASA's continued investment in landing technologies is reflective of the current momentum surrounding HDA research.

Several recent studies and developments in HDA focus primarily on hazard detection, which for a long time has been considered the deciding factor for successful landings. There are methods that attempt to match identified features on a 2D image to database of known hazards, such as Yu and Cui's affine invariant matching algorithm [4]. Many methods involve the construction of a digital elevation model (DEM) with either LIDAR or vision-based techniques [5–8]. Recently proposed techniques use artificial intelligence, such as neural networks, in combination with vision-based or LIDAR sensors to identify hazards [9–11]. With the wealth of existing and developing research into the identification of hazards, there is now an increasing trend toward a more refined level of landing site selection. For example, Wei et al. provide a method for avoiding regions that are closed environments, such as craters with a flat interior that may otherwise be selected as a safe landing location [12]. Further, Cui et al. propose a safety index method in which hazardous terrain is classified with varying levels of safety, and fuel consumption and touchdown performance are factored into an optimization problem for landing site selection [13]. However, these methods all contain the same constraint on their outcome: they search for landing locations that fit a scaled square, circular, or elliptical footprint. These types of footprints can be collectively referred to as nonconforming footprints.

The use of a nonconforming footprint presents a major issue on asteroids with highly hazardous terrain in that such acceptable landing locations may be few and far between, or may not even exist at all. Historically, these nonconforming footprints were necessary to account for uncertainties in landing pose in the order for 10 m [14]. However, recent innovations in both the hardware and software involved with hazard detection have greatly reduced these uncertainties [8,9,15,16]. Alongside the increasing capabilities of space-flight ready processors [17], these modern hazard detection techniques allow the possibility of using geometrically conforming footprints, as seen in Fig. 2, for landing among hazardous terrain. Note that the benefits of using a conforming footprint may only be perceivable when working with close to submeter navigation uncertainties. While this might not seem immediately feasible, Feng et al. and Bercovici show that small-body modeling techniques are fast approaching the required fidelity, which is supported by the nominal position uncertainty of OSIRIS-REx and Hayabusa2 of about 3 and 1 m, respectively [18–20].

This paper investigates a novel landing site selection technique that uses a geometrically conforming footprint. The formulation of this technique assumes that the surface terrain and hazards have been identified and processed into a two-dimensional (2D) safety map, parallel to the average local surface, in which hazards are represented with polygons [21]. Two-dimensional hazard projection is a technique used to desegregate various conditions that are considered unsafe into no-go regions for landing site selection [13]. Therefore

Received 15 April 2021; revision received 21 October 2021; accepted for publication 26 October 2021; published online 22 November 2021.

*Graduate Research Assistant, Ann and H. J. Smead Department of Aerospace Engineering Sciences; nelson.joshua@colorado.edu.

†Glenn L. Murphy Chair of Engineering, Ann and H. J. Smead Department of Aerospace Engineering Sciences. Fellow AIAA.

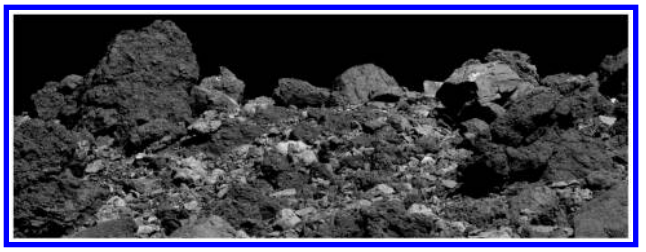


Fig. 1 The rocky terrain that covers much of Bennu (NASA/Goddard/University of Arizona).

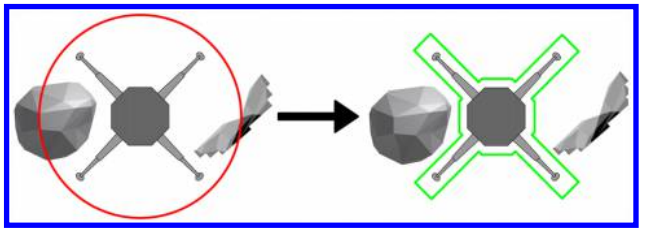


Fig. 2 A circular footprint (left) compared to a geometrically conforming footprint (right).

it is important to note that the polygons representing surface hazards in this paper are not necessarily literal representations of singular boulders, though they might be, but rather are, regions considered unsafe to land in. Figure 3 shows a simple example of what such a safety map might look like. The following section develops a cost function to minimize the distance between the landing site and some desired location; the geometrically conforming footprint is used to define nonintersection constraints with the hazards. Two variants of this method are discussed and are labeled as a coarse search and fine search, respectively.

II. Mathematically Defining the Geometrically Conforming Footprint

A. Problem Formulation

Consider first the lander and the surface expressed in the group $SE(3)$. The reference frame $\mathcal{F}$:$\{\boldsymbol{F}, \hat{\boldsymbol{f}}_1, \hat{\boldsymbol{f}}_2, \hat{\boldsymbol{f}}_3\}$ is created for the lander such that the frame origin $\boldsymbol{F}$ is located at the geometric centroid of the feet and is coplanar with the bottom faces of the feet. The third basis vector of frame $\mathcal{F}$ is defined to be orthogonal to the feet plane and pointed toward the lander (such that the entire lander geometry is in the $+\hat{\boldsymbol{f}}_3$ direction), whereas the other two basis vectors are defined such that $\mathcal{F}$ is right-handed, as seen in Fig. 4. Assuming that a 2D safety map has been constructed in advance for a region on the surface, the reference frame $\mathcal{S}$:$\{\boldsymbol{S}, \hat{\boldsymbol{s}}_1, \hat{\boldsymbol{s}}_2, \hat{\boldsymbol{s}}_3\}$ is defined such that the frame origin $\boldsymbol{S}$ is located at a corner of the finite plane that the safety map exists in. The first two basis vectors are defined such that entire safety map exists in the first octant. The third basis vector of frame $\mathcal{S}$ is chosen to be normal to safety map plane, and roughly antiparallel to the local gravity direction, with the other two basis vectors defined such that $\mathcal{S}$ is right-handed.

To operate within the 2D safety map, the basis vectors $\hat{\boldsymbol{f}}_3$ and $\hat{\boldsymbol{s}}_3$ are enforced to be parallel and point $\boldsymbol{F}$ is projected onto the safety map plane. These constraints truncate the operation space of this problem to $SE(2)$. Let there be two sets of input polygons: a set representing the spacecraft lander, and a set representing the surface in some confined rectangular region containing the desired landing location, denoted as point $\boldsymbol{L}$. Assume that both sets are rigid bodies such that all the surface polygons remain fixed in the $\mathcal{S}$ frame, and all the lander polygons remain fixed in the $\mathcal{F}$ frame. Considering the position of the lander relative to the surface origin $\boldsymbol{r}_{F/S}$ and the position of the desired landing location relative to the surface origin $\boldsymbol{r}_{L/S}$, this problem can be cast as a minimization of the quadratic cost function:

$$V(\boldsymbol{r}_{L/F}) = \frac{1}{2}\boldsymbol{r}_{F/S}^T Q \boldsymbol{r}_{F/S} + \boldsymbol{c}^T \boldsymbol{r}_{F/S} \tag{1}$$

where $Q \succeq 0$ scales the influence on the cost of each component of $\boldsymbol{r}_{F/S}$, and $\boldsymbol{c} = -Q\boldsymbol{r}_{L/S}$ offsets the minimum cost to be at $\boldsymbol{r}_{F/S} = \boldsymbol{r}_{L/S}$. Optimization of such a cost function subject to a set of linear inequality constraints is known as quadratic programming (QP) [22]. This QP is further expanded by constraints preventing the intersection of the surface and lander polygons.

B. Separating Axis Theorem and Approximate Convex Decomposition

The safety of a selected landing location materializes as the assurance that surface hazards do not penetrate the lander hull. This assurance is quantified by the nonintersection of the lander and surface obstacles represented through polygons. The foundation of

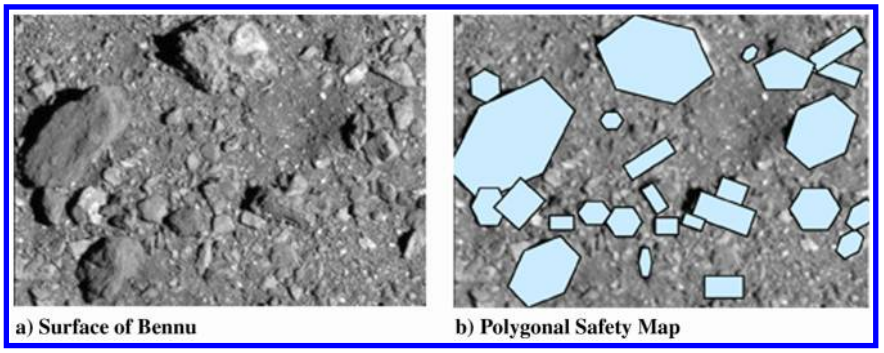


Fig. 3 Example of safety map.

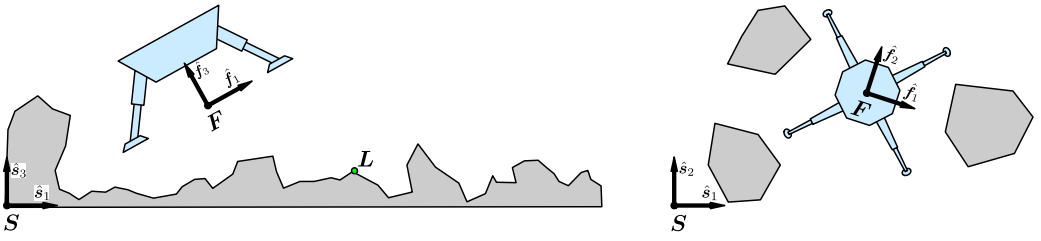


Fig. 4 Frame definitions for the lander and surface with $\hat{\boldsymbol{f}}_3$ and $\hat{\boldsymbol{s}}_3$ forced to be parallel on the right.

the nonintersection constraints comes from the separating axis theorem (SAT) [23], which says that any two convex polygons do not overlap if, and only if, there exists at least one spatial axis where the projections of those polygons do not overlap. Let $\boldsymbol{r}_{C/S}$ be a vector from the $\mathcal{S}$ frame origin to the centroid $\boldsymbol{C}$ of a polygon, and let $\boldsymbol{r}_{V/C}$ be a vector from $\boldsymbol{C}$ to any exterior point on the polygon, as seen in Fig. 5a. Then for any two nonintersecting polygons h and k, there exists an axis where the projection of $\boldsymbol{r}_{C/S,h} - \boldsymbol{r}_{C/S,k}$ is greater in magnitude than $\boldsymbol{r}_{V/C,k} + \boldsymbol{r}_{V/C,h}$, where $\boldsymbol{r}_{V/C,k}$ is the same direction as the projection, and $\boldsymbol{r}_{V/C,h}$ is the opposite direction. This can be seen in Fig. 5b, where the separating axis is shown to be along $\hat{\boldsymbol{s}}_2$.

While the SAT provides a mathematically reliable method to determine if two polygons intersect, note that it is defined specifically for *convex* polygons. This presents a natural issue, as surface hazards and spacecraft landers are seldom convex in their geometry. This issue is avoided by constructing a convex hull around any nonconvex polygon. However, this idea has substantial drawbacks; a convex hull over a highly nonconvex polygon would result in a large amount of empty space removed from consideration when fitting two polygons together. Thus, a more refined application of convex hulls can be used in form of a technique known as approximate convex decomposition [24] (ACD). The general idea behind ACD is to subdivide highly nonconvex polygons into multiple polygons that are less nonconvex than their parent before applying a convex hull to each new polygon. The number of subdivided polygons is determined by the maximum allowable level of nonconvexity, which is parameterized by the user. For the landing technique being proposed in this paper, it is assumed that the decomposition step has already been completed for the lander, and the safety map construction algorithm uses only convex polygons to represent hazards [21].

C. Constraint Formulation

The existence of a separating axis between a surface polygon k and a lander polygon h is readily represented with the inequality

$$(\boldsymbol{r}_{C/S,h})_\beta - (\boldsymbol{r}_{C/S,k})_\beta \geq (\boldsymbol{r}_{V/C,k})_\beta + (\boldsymbol{r}_{V/C,h})_\beta \tag{2}$$

where the subscript β represents some direction in $\mathcal{S}$ and $(\cdot)_\beta$ is the scalar projection of a vector on β. Testing for a separating axis in only one direction is insufficient, so this inequality must be applied over an encompassing set of directions. When testing over a set of directions, the SAT says that the above inequality only needs to be satisfied in one direction. In fact, this inequality will fail in some other directions, rendering this problem infeasible. Therefore, Eq. (2) must be adjusted as follows [25]:

$$(\boldsymbol{r}_{C/S,h})_\beta - (\boldsymbol{r}_{C/S,k})_\beta \geq (\boldsymbol{r}_{V/C,k})_\beta + (\boldsymbol{r}_{V/C,h})_\beta - D_i(1 - \epsilon_{hk,\beta}) \tag{3}$$

$$\sum_\beta \epsilon_{hk,\beta} \geq 1 \tag{4}$$

Here D_i is a scalar big-M coefficient and $\epsilon_{hk,\beta} \in \{0, 1\}$ are activation decision variables. With a large-enough value of D_i and with $\epsilon = 0$, the inequality in Eq. (3) becomes always true within the scope of the problem. Thus, for a value of $\epsilon_{hk,\beta} = 1$, the separating axis criterion becomes active along the β direction. As stated previously, the separating axis must exist in at least one direction for the two polyhedra to not intersect, which is enforced with the inequality in Eq. (4). The value of D_i may be chosen to be infinitely large; however, its minimum effective value is the length of the longest dimension of the operational area defined by the safety map.

These constraints are applied between every convex polygon in the lander set h and surface set k (they are not applied between two polygons contained in the same set). The centroid positions of lander set are known and constant in the $\mathcal{F}$ frame; however, these constraints are evaluated in the $\mathcal{S}$ frame. Thus Eq. (3) is modified by

$${}^{\mathcal{S}}\boldsymbol{r}_{C/S,h} = {}^{\mathcal{S}}\boldsymbol{r}_{F/S} + [SF]^{\mathcal{F}}\boldsymbol{r}_{C/F,h} \tag{5}$$

where $\boldsymbol{r}_{C/F,h}$ is constant in the $\mathcal{F}$ frame and $[SF]$ is a direction cosine matrix (DCM) that rotates a vector description from $\mathcal{F}$ to $\mathcal{S}$ [26]. This alteration now adds the position of $\boldsymbol{F}$ relative to $\boldsymbol{S}$ and the attitude of $\mathcal{F}$ relative to $\mathcal{S}$ as decision variables to this problem.

Because this problem is constrained to $SE(2)$, the attitude is represented by a single angle $\theta_{F/S}$ in-plane with the safety map [26]. To maintain these constraints as linear, the DCM parameterized by this angle must be linearized about some reference frame $\mathcal{R}$. This introduces an attitude angle $\theta_{F/R}$ that is linearized to the DCM:

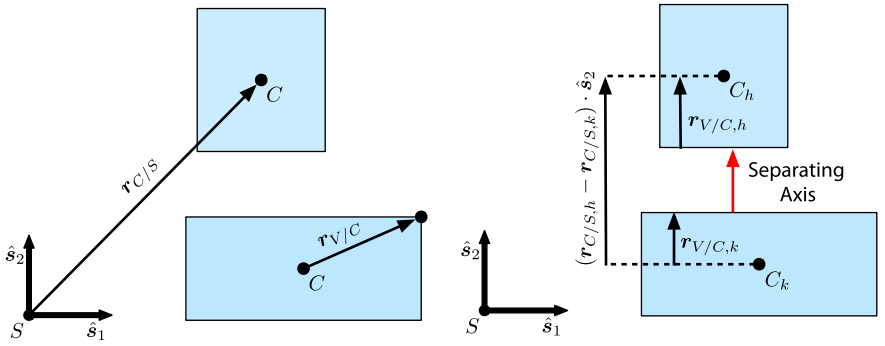
$$[C(\theta_{F/R})] = \begin{bmatrix} 1 & -\theta_{F/R} \\ \theta_{F/R} & 1 \end{bmatrix} \tag{6}$$

To avoid a significant amount of error in the DCM introduced by linearization, rotations defined by $\theta_{F/R}$ should be bounded within some small linear regime. Therefore, this problem does not consider the surface frame $\mathcal{S}$ as the frame $[C(\theta_{F/R})]$ is linearized about. Let frame $\mathcal{R}$ be an intermediate frame that relates to $\mathcal{S}$ by a rotation of some reference angle $\theta_{R/S}$, denoted by the DCM $[SR]$. To maintain linearity in these constraints, the angle $\theta_{R/S}$ is held constant when solving for a solution. To search for a landing pose throughout the entire problem space, this problem is to be solved several times over iterations of $\theta_{R/S}$, with the solution with the lowest cost chosen as the final solution.

To enforce the discussed constraints on attitude, the following bounds are added to the problem constraints:

$$\theta_{F/R,\min} \leq \theta_{F/R} \leq \theta_{F/R,\max} \tag{7}$$

Note that these bounds introduce a double inequality to the problem constraints. In fact, upper and lower bounds on ${}^{\mathcal{S}}\boldsymbol{r}_{F/S}$ must also be introduced to contain the problem within the known region of surface hazards. For consistency in the constraints, and because many effective QP solvers operate on double inequality constraints, an upper bound is added to Eqs. (3) and (4). First, isolating the decision variables in Eq. (3) to one side of the inequality and expressing the vectors in their known frame leads to



a) Vector Definitions **b) Depiction of Separating Axis**

Fig. 5 Example of the separating axis theorem.

$$({}^{\mathcal{S}}\boldsymbol{r}_{C/S,k} + {}^{\mathcal{S}}\boldsymbol{r}_{V/C,k})_\beta - D_i \le ({}^{\mathcal{S}}\boldsymbol{r}_{F/S} + [SR][C(\theta_{F/R})]^T \times ({}^{\mathcal{F}}\boldsymbol{r}_{C/F,h} - {}^{\mathcal{F}}\boldsymbol{r}_{V/C,h}))_\beta - D_i\epsilon_{hk,\beta} \quad (8)$$

$$1 \le \sum_\beta \epsilon_{hk,\beta} \quad (9)$$

Next, another big-M value, called D_{sr}, is introduced to Eq. (8) as an upper bound. This new value can be the same as D_i; however, if the surface area being considered for landing does not fully contain all known hazards, then D_{sr} may be the length of longest dimension of the surface area being considered. The upper bound for Eq. (9) is the number of search directions for a separating axis, labeled $\beta_{\max}$. Therefore, all constraints defined thus far are

$${}^{\mathcal{S}}\boldsymbol{r}_{F/S,\min} \le {}^{\mathcal{S}}\boldsymbol{r}_{F/S} \le {}^{\mathcal{S}}\boldsymbol{r}_{F/S,\max} \quad (10a)$$

$$\theta_{F/R,\min} \le \theta_{F/R} \le \theta_{F/R,\max} \quad (10b)$$

$$({}^{\mathcal{S}}\boldsymbol{r}_{C/S,k} + {}^{\mathcal{S}}\boldsymbol{r}_{V/C,k})_\beta - D_i \le ({}^{\mathcal{S}}\boldsymbol{r}_{F/S} + [SR][C(\theta_{F/R})]^T \times ({}^{\mathcal{F}}\boldsymbol{r}_{C/F,h} - {}^{\mathcal{F}}\boldsymbol{r}_{V/C,h}))_\beta - D_i\epsilon_{hk,\beta} \le D_{sr} \quad (10c)$$

$$1 \le \sum_\beta \epsilon_{hk,\beta} \le \beta_{\max} \quad (10d)$$

Note that Eq. (10c) exists for every direction being used to test for a separating axis, with each having a unique decision variable $\epsilon_{hk,\beta}$. For each pairing of surface and lander polygons, let the lower bound of Eqs. (10c) and (10d) form the vector $\boldsymbol{l}_{hk}$ and the upper bound form the vector $\boldsymbol{u}_{hk}$. Let there be decision variable vectors $\boldsymbol{x} = [{}^{\mathcal{S}}\boldsymbol{r}_{F/S}^T \quad \theta_{F/R}]^T$ and $\epsilon_{hk} = [\epsilon_{hk,1} \quad \dots \quad \epsilon_{hk,n}]^T$ for q directions in β. Then Eqs. (10c) and (10d) become

$$\boldsymbol{l}_{hk} \le [\Sigma_{hk} \quad D_{hk}] \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\epsilon}_{hk} \end{bmatrix} \le \boldsymbol{u}_{hk} \quad (11)$$

where $\boldsymbol{l}_{hk}, \boldsymbol{u}_{hk} \in \mathbb{R}^{q+1}$, $\Sigma_{hk} \in \mathbb{R}^{(q+1)\times 3}$, and $D_{hk} \in \mathbb{R}^{(q+1)\times q}$. Let the upper and lower bounds of Eqs. (10a) and (10b) be $\boldsymbol{x}_l$ and $\boldsymbol{x}_u$, respectively. For an example of how these constraints evolve, let there be two lander polygons $h \in \{1, 2\}$ and two surface polygons $k \in \{1, 2\}$. Then the inequality constraints would be

$$\begin{bmatrix} \boldsymbol{x}_l \\ \boldsymbol{l}_{11} \\ \boldsymbol{l}_{12} \\ \boldsymbol{l}_{21} \\ \boldsymbol{l}_{22} \end{bmatrix} \le \begin{bmatrix} I_{3\times3} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \Sigma_{11} & D_{11} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \Sigma_{12} & \boldsymbol{0} & D_{12} & \boldsymbol{0} & \boldsymbol{0} \\ \Sigma_{21} & \boldsymbol{0} & \boldsymbol{0} & D_{21} & \boldsymbol{0} \\ \Sigma_{22} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & D_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\epsilon}_{11} \\ \boldsymbol{\epsilon}_{12} \\ \boldsymbol{\epsilon}_{21} \\ \boldsymbol{\epsilon}_{22} \end{bmatrix} \le \begin{bmatrix} \boldsymbol{x}_u \\ \boldsymbol{u}_{11} \\ \boldsymbol{u}_{12} \\ \boldsymbol{u}_{21} \\ \boldsymbol{u}_{22} \end{bmatrix} \quad (12)$$

where $I_{3\times3}$ is the 3×3 identity matrix. Recall that every variable in $\boldsymbol{\epsilon}_{hk}$ is a binary decision variable, and thus, with the constraints defined, this problem becomes cast as a Mixed Integer Quadratic Program (MIQP) in the form

$$\text{Minimize} \quad V(\boldsymbol{z}) = \frac{1}{2}\boldsymbol{z}^T Q \boldsymbol{z} + \boldsymbol{c}^T \boldsymbol{z} \quad (13a)$$

$$\text{Subject To} \quad \boldsymbol{l} \le A\boldsymbol{z} \le \boldsymbol{u} \quad (13b)$$

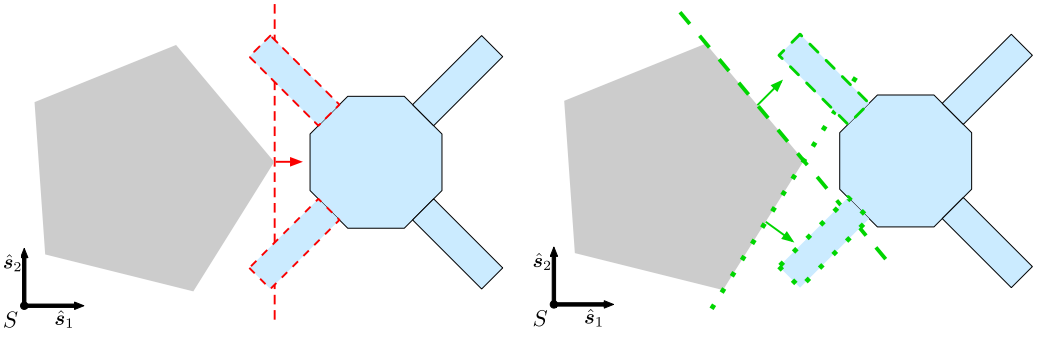
$$z_i \in \{0, 1\} \quad (13c)$$

where $\boldsymbol{z} \in \mathbb{R}^n$, $Q \in \mathbb{R}^{n\times n}$, $\boldsymbol{c} \in \mathbb{R}^n$, $\boldsymbol{l}, \boldsymbol{u} \in \mathbb{R}^m$, $A \in \mathbb{R}^{m\times n}$, and $i = 4, \dots, n$. Let K and H be the number of polygons in the surface set and lander set, respectively, and K_q be the sum of the amount of search directions for each surface polygon. Then the dimensions of this problem are $n = 3 + HK_q$ and $m = 3 + HK_q + HK$.

D. Search Directions

The set of directions to search for a separating axis must be encompassing; i.e. for two polygons sufficiently far from each other anywhere in $\mathbb{R}^2$, there must exist at least one valid direction in the set. An obvious choice for the set comprises the positive and negative of the two basis directions of the $\mathcal{S}$ frame. However, while such a set satisfies the encompassing requirement, it may not provide an ideal result when the two polygons are very close to one other. For example, if a separating axis is only checked along the basis directions of $\mathcal{S}$, then the case shown in Fig. 6a would fail, even though the lander is not intersecting the rock. Think of the separating axis as creating a plane defined by the direction of the axis, and the most extreme point in the axis direction on the polygon. Therefore, using only the $\mathcal{S}$ frame basis vectors as separating axes creates a frame-oriented bounding parallelogram. Luckily, a separating axis can be defined in any direction in $\mathbb{R}^2$, and thus the set of search directions can be chosen freely. Consider a set of search directions defined by the normal vectors that define the faces of one of the polygons, which is also an encompassing set. Because only one separating axis needs to exist to prove separation, then this new selection of possible axes allows the previously failed case to pass, as seen in Fig. 6b.

There are pros and cons to using either of the described search direction sets. The set containing the basis directions of $\mathcal{S}$ is consistent throughout the problem space, and it is thus simpler to preprocess. The negative aspect of this set is potential restrictions it places on two polygons that are closely placed together. The set containing the normal directions of a polygon's faces involves a greater amount of preprocessing, as each pair of polygons being compared requires a specific set of search directions. Because the surface hazard polygons remain constant in the $\mathcal{S}$ frame, they are used to define these search direction sets. Thus, each polygon in the surface set has a unique corresponding search direction set that must be defined in preprocessing. However, these types of search direction sets provide a more conforming fit between lander and surface polygons, which is ideal in



a) The dashed line represents the plane created by the separating axis in the $\hat{s}_1$ direction

b) The shape with the dashed/dotted border only passes the separating axis test with the dashed/dotted line

Fig. 6 Comparing the effectiveness of the two search methods.

the fine placement of the final landing position. Therefore, the search direction set containing the basis directions of $\mathcal{S}$ is referred to as the *coarse* searching set, and the set that contains the face-normal directions of the surface polygons is referred to as the *fine* searching set.

III. Algorithm to Solve for Landing Locations

The program described in Eq. (13) is known to be Nondeterministic Polynomial (NP)-hard due to the presence of integer decision variables [27]. This raises tractability concerns in regard to solving this problem on-board a spacecraft. Fortunately, recent studies [28,29] show tractable results for solving MIQPs on embedded systems. These techniques use the branch and bound (B&B) approach to solve MIQPs by subdividing the problem into a tree of relaxed QP problems that can be searched to find the optimal solution. The performance of this approach depends heavily on the method used to search the tree. This section describes an example of an algorithm for solving this problem with a B&B approach. This algorithm in particular is not developed with the intention of being flight ready, but rather as a means to compare how the two variants of Eq. (13) perform when solved via B&B.

Using the B&B approach allows the binary decision variables in this problem to be abstracted away, and replaced by continuous variables whose value is dictated by the tree of QP problems. Therefore, Eq. (13) is restructured to

$$\text{Minimize} \quad V(z) = \frac{1}{2} z^T Q z + \boldsymbol{c}^T z \tag{14a}$$

$$\text{Subject To} \quad \begin{bmatrix} \boldsymbol{l} \\ \bar{\boldsymbol{l}} \end{bmatrix} \leq \begin{bmatrix} A \\ \bar{A} \end{bmatrix} z \leq \begin{bmatrix} \boldsymbol{u} \\ \bar{\boldsymbol{u}} \end{bmatrix} \tag{14b}$$

where $\bar{\boldsymbol{l}}$, $\bar{A}$, and $\bar{\boldsymbol{u}}$ are the continuous representation of Eq. (13c). For every binary decision variable in the original problem, there is a row in $\bar{A}$ with a value of 1 at the column corresponding to that variable. Initially, the upper and lower bounds are $\bar{\boldsymbol{l}} = -\tau \mathbf{1}$ and $\bar{\boldsymbol{u}} = \tau \mathbf{1}$, where τ is a small feasibility tolerance. Each successive branch in the tree alters a single set of $\boldsymbol{\epsilon}_{hk}$, choosing one $\epsilon_{hk,\beta}$ to have its upper and lower bounds changed to $\bar{l} = 1 - \tau$ and $\bar{u} = 1 + \tau$. In context to the problem, this process begins by deactivating all of the separating axis constraints defined by Eq. (10c), and then reactivating one for each combination of surface and lander polygons as the tree is explored. A final solution is only accepted if it is found at the maximum depth of the tree. Because this process ensures the necessary activation of separating axis constraints, then the constraints defined by Eq. (10d) become unnecessary, and thus can be removed from the problem. This reduces the dimensionality of this problem such that $z, \boldsymbol{c}, \boldsymbol{l}, \boldsymbol{u} \in \mathbb{R}^n, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}} \in \mathbb{R}^m, Q, A \in \mathbb{R}^{n \times n}$, and $\bar{A} \in \mathbb{R}^{m \times n}$ with $n = 3 + HK_q$ and $m = HK_q$.

As stated previously, this algorithm assumes that the surface hazards and lander have already been identified and decomposed into sets of convex polygons $\{P\}$. Each polygon P in a set contains a set of vertices $\{V\}$ and face normal unit vectors $\{N\}$ that describe the polygon. Let there be two sets of polygons denoted with the subscripts k and h, which contain the polygons for the surface and lander, respectively. Within a polygon, every vertex $\boldsymbol{r}_V$ and face normal $\hat{\boldsymbol{n}}$ is described in the reference frame respective to the polygon's set. Furthermore, this algorithm is presented using the fine searching set and may be adjusted to use the coarse searching set by replacing all $\{N\}$ with the set of positive and negative basis vectors of the $\mathcal{S}$ frame.

Before the elements in Eq. (13b) can be assembled, the values of $\boldsymbol{r}_{V/C}$ and $\boldsymbol{r}_C$ (this notation is truncated from $\boldsymbol{r}_{C/S}$ and $\boldsymbol{r}_{C/F}$ for surface and lander polygons, respectively) must be found for each polygon, as shown in Algorithm 1. The value of $\boldsymbol{r}_C$ is simply the vertex-weighted centroid of each polygon. As previously discussed, $\boldsymbol{r}_{V/C}$ must be the vector from the centroid to the edge of the polygon in the search direction. This value is more easily found on a polygon from the surface set, because the search directions are defined in the $\mathcal{S}$ frame. As seen in line 8, the largest projection of the vertices relative to the centroid is found for each search direction vector. These scalar values are stored in a set $\{\rho\}$ that aligns with the search direction set. Due to the variability in attitude between the lander and the surface, $\boldsymbol{r}_{V/C}$ for a lander polygon cannot be predetermined using the search directions. Instead, the set $\{\rho\}$ is filled with the largest projections in the basis directions of the $\mathcal{F}$ frame.

Algorithm 1: Preprocessing the geometry

Input: The surface and lander sets of polygons $\{P_k\}$ and $\{P_h\}$, where each polygon contains a set of vertices $\{V\}$ and face normal unit vectors $\{N\}$ in their respective frames.

1 **for each** $P \in \{P_k\} \cup \{P_h\}$ **do**
2 $\quad \boldsymbol{r}_C = \mathbf{0}$; $j = 0$; $\{\rho\} = \varnothing$
3 $\quad$ **for each** $\boldsymbol{r}_V \in \{V\}$ **do**
4 $\quad\quad \boldsymbol{r}_C = \boldsymbol{r}_C + \boldsymbol{r}_V$; $j = j + 1$
5 $\quad \boldsymbol{r}_C = \frac{\boldsymbol{r}_C}{j}$
6 $\quad$ **if** $P \in \{P_k\}$ **then**
7 $\quad\quad$ **for each** $\hat{\boldsymbol{n}} \in \{N\}$ **do**
8 $\quad\quad\quad \{\rho\} \leftarrow \max\{\hat{\boldsymbol{n}} \cdot (\boldsymbol{r}_V - \boldsymbol{r}_C), \quad \forall\, \boldsymbol{r}_V \in \{V\}\}$
9 $\quad$ **else if** $P \in \{P_h\}$ **then**
10 $\quad\quad$ **for each** $i \in \{1, 2, 3\}$ **then**
11 $\quad\quad\quad \{\rho\} \leftarrow \max\{\hat{\boldsymbol{f}}_i \cdot (\boldsymbol{r}_V - \boldsymbol{r}_C), \quad \forall\, \boldsymbol{r}_V \in \{V\}\}$
12 $\quad P \leftarrow (\boldsymbol{r}_C, \{\rho\})$

With the geometry preprocessed, the vectors and matrices in Eq. (14) are constructed in the manner depicted in Algorithm 2. Lines 1–4 perform initialization, followed by the hessian and gradient value assignment. Because this problem is only concerned with minimizing the distance of $\boldsymbol{r}_{F/S}$ to some objective position $\boldsymbol{r}_{L/S}$ in the $\hat{\boldsymbol{s}}_1$ and $\hat{\boldsymbol{s}}_2$ directions, the hessian should only be nonzero (and positive) at the elements $Q_{1,1}$ and $Q_{2,2}$. Line 6 implements the upper and lower bounds for $\boldsymbol{r}_{F/S}$ and $\theta_{F/R}$. The solver is warm started with the vector z_0, which

Algorithm 2: Solver part 1

Input: The surface and lander sets of polygons $\{P_k\}$ and $\{P_h\}$, $\boldsymbol{r}_{F/S}$ hessian weights x and y, objective position $\boldsymbol{r}_{L/S}$, position lower and upper bounds $\boldsymbol{r}_{F/S,\min}$ and $\boldsymbol{r}_{F/S,\max}$, attitude lower and upper bounds $\theta_{F/R,\min}$ and $\theta_{F/R,\max}$, big-M values D_i and D_{sr}, reference angle $\theta_{R/S}$, tolerance τ, and max solver iterations Λ.

1 $H = \text{size}(\{P_h\})$; $K_q = 0$
2 **for each** $P \in \{P_k\}$ **do** $K_q = K_q + \text{size}(\{\rho\})$
3 $n = 3 + HK_q$; $m = HK_q$; $i, j = 4$; $t = 1$; $\{\Gamma\} = \varnothing$
4 $Q = \mathbf{0}_{n \times n}$; $\boldsymbol{c} = \mathbf{0}_n$; $A = \mathbf{0}_{n \times n}$; $\bar{A} = \mathbf{0}_{m \times n}$; $\boldsymbol{l} = \mathbf{0}_n$; $\boldsymbol{u} = \mathbf{0}_n$; $\bar{\boldsymbol{l}} = -\tau \mathbf{1}_m$; $\bar{\boldsymbol{u}} = \tau \mathbf{1}_m$
5 $Q_{1,1}, Q_{2,2} = x, y$; $c_1 = -Q_{1,1} * r_{L/S,1}$; $c_2 = -Q_{2,2} * r_{L/S,2}$
6 $A_{1:3,1:3} = I_{3\times 3}$; $\boldsymbol{l}_{1:2} = \boldsymbol{r}_{F/S,\min}$; $\boldsymbol{u}_{1:2} = \boldsymbol{r}_{F/S,\max}$; $\boldsymbol{l}_3 = \theta_{F/R,\min}$; $\boldsymbol{u}_3 = \theta_{F/R,\max}$
7 $z_0 = \mathbf{0}$; $z_{0,1} = r_{L/S,1}$; $z_{0,2} = r_{L/S,2}$
8 **for each** $P_h \in \{P_h\}$ **do**
9 $\quad$ **for each** $P_k \in \{P_k\}$ **do**
10 $\quad\quad \Gamma = \|\boldsymbol{r}_{L/S} - \boldsymbol{r}_{C/S,k}\|$; $\{\gamma\} = \varnothing$; $\beta = 1$
11 $\quad\quad\quad$ **for each** $(\hat{\boldsymbol{n}}, \rho) \in (\{N_k\}, \{\rho_k\})$ **do**
12 $\quad\quad\quad\quad \gamma = \hat{\boldsymbol{n}} \cdot (\boldsymbol{r}^*_{F/S} + [SR(\theta_{R/S})]\boldsymbol{r}_{C/F,h} - (\boldsymbol{r}_{C/S,k} + \rho\hat{\boldsymbol{n}}))$; $\{\gamma\} \leftarrow (\beta, \gamma)$
13 $\quad\quad\quad\quad \boldsymbol{r}_{V/C,h} = [\rho_{h,1} \quad \rho_{h,2}]^T$
14 $\quad\quad\quad\quad A_{i+\beta,1} = \hat{n}_1$; $A_{i+\beta,2} = \hat{n}_2$
15 $\quad\quad\quad\quad A_{i+\beta,3} = \hat{\boldsymbol{n}} \cdot (\boldsymbol{r}_{C/F,h,\theta} - \boldsymbol{r}_{V/C,h,\theta})$
16 $\quad\quad\quad\quad A_{i+\beta,j} = -D_i$; $\bar{A}_{t,j} = 1$; $u_{i+\beta} = D_{sr}$
17 $\quad\quad\quad\quad l_{i+\beta} = \hat{\boldsymbol{n}} \cdot (\boldsymbol{r}_{C/S,k} - \boldsymbol{r}_{C/F,h,\alpha} + \boldsymbol{r}_{V/C,h,\alpha}) + \rho - D_i$
18 $\quad\quad\quad\quad j = j + 1$; $t = t + 1$; $\beta = \beta + 1$
19 $\quad\quad$ sort($\{\gamma\}$); $\{\Gamma\} \leftarrow (j - 3, \Gamma, \{\gamma\})$; $i = i + \beta$
20 sort($\{\Gamma\}$)
21 Execute Algorithm 3 with Q, A, $\bar{A}$, $\boldsymbol{c}$, $\boldsymbol{l}$, $\boldsymbol{u}$, $\bar{\boldsymbol{l}}$, $\bar{\boldsymbol{u}}$, z_0, $\{\Gamma\}$, τ, and Λ

contains the objective position. Next, each combination of surface and lander polygons and their corresponding search directions are looped through to build the constraints defined by Eq. (10c). In the matrix A, every ith row is the start of a new polygons combination and β iterates through that combination's search directions. To improve the readability of lines 15–17, the notation of $\boldsymbol{r}_\alpha$ and $\boldsymbol{r}_\theta$ is defined for vectors rotated from the $\mathcal{F}$ frame to the $\mathcal{S}$ frame, such that

$$
\begin{aligned}
[SR][C(\theta_{F/R})]^{T\mathcal{F}}\boldsymbol{r} &= {}^{\mathcal{S}}\begin{bmatrix} r_1\cos\theta_{R/S} - r_2\sin\theta_{R/S} \\ r_1\sin\theta_{R/S} + r_2\cos\theta_{R/S} \end{bmatrix} \\
&\quad + {}^{\mathcal{S}}\begin{bmatrix} r_1\sin\theta_{R/S} + r_2\cos\theta_{R/S} \\ -r_1\cos\theta_{R/S} + r_2\sin\theta_{R/S} \end{bmatrix}\theta_{F/R} \\
&= {}^{\mathcal{S}}\boldsymbol{r}_\alpha + {}^{\mathcal{S}}\boldsymbol{r}_\theta\theta_{F/R} \qquad (15)
\end{aligned}
$$

This part of the algorithm also includes the construction of the B&B searching tree. For the overall structure of the tree, every level of depth represents a unique combination of surface and lander polygons, with the choice of $\epsilon_{hk,\beta}$ to activate existing laterally. Starting with the lateral structure of the tree, a set $\{\gamma\}$ is created for each combination of polygons. This set contains groupings of values β and γ for each separating axis search direction, where β is the position of the search direction in the block in $\bar{\boldsymbol{l}}$ and $\bar{\boldsymbol{u}}$ that represents this polygon combination. The value of γ, seen in line 12, is the signed distance of the lander polygon centroid, if $\boldsymbol{r}_{F/S} = \boldsymbol{r}_{L/S}$, from the edge of the lander polygon in the search direction, assuming that $\theta_{F/R} = 0$. The set $\{\gamma\}$ is then sorted in descending order of the value of γ in line 19. This step acts as a rough heuristic for the order in which the search direction constraint should be explored. Next, for the depthwise structure of the tree, a value Γ is determined for each combination of polygons in line 10. This value is the distance from the surface polygon centroid to the objective landing position, and it is added to the set $\{\Gamma\}$ along with the starting position of the block in $\bar{\boldsymbol{l}}$ and $\bar{\boldsymbol{u}}$ that represents this polygon combination and the set $\{\gamma\}$. Then in line 20, the set $\{\Gamma\}$ is sorted in descending order of the value of Γ. The method for ordering the tree laterally is more likely to be accurate the further the surface polygon in question is from the final landing position. Therefore, assuming that a feasible landing position exists anywhere near the objective position, starting the tree at the surface polygon furthest away from the objective position leads to the shortest number of iterations until a feasible solution is found. This method of searching from the furthest polygon is quicker because a valid separating axis of distant polygons is more likely to be closer in angle to the vector connecting the polygon to the objective position, thus providing quicker verification of nonintersection.

The branches of relaxed QP problems can be individually solved using any QP solver that operates on problems in the structure of Eq. (14) with $Q \succeq 0$ and can be warm started. The primary motivating factor in choosing a solver for this problem is how well it performs with limited computing power. The solver chosen for the development of this paper is the OSQP solver [30], which meets all the requirements and is shown to be robust and efficient enough to operate on low-power embedded systems [29].

Algorithm 3 depicts how the QP solver operates on the searching tree map to obtain a feasible solution to the problem. The lowest feasible cost value V and its associated solution z^* are initialized with ∞ and z_0, respectively. After the QP solver is initialized, the set $\{T\}$ is created to contain all active branches of relaxed QPs (the current deepest branch and all previous branches that links it back to the start of the tree). Every $T \in \{T\}$ contains the solution of the previous branch z (initialized with z_0), the current modification of the vectors $\bar{\boldsymbol{l}}$ and $\bar{\boldsymbol{u}}$, and the depth μ and lateral ν position of the current branch. The relaxed QP of the most recently added branch is solved in lines 6–8, using the previous solution for warm starting. If the solution of the relaxed QP is both feasible and less than the lowest feasible cost value, the first lateral position of the next depth level in $\{\Gamma\}$ is accessed to create a new branch to be added to $\{T\}$ in lines 13–15. Otherwise, the current branch is removed from $\{T\}$ and is replaced with the next lateral position of its depth level in lines 22–24. If there are no more lateral positions in the current depth level, the current branch is not replaced and the previous branch in $\{T\}$ is cycled through.

Algorithm 3: Solver part 2

Input: The problem matrices $Q, A, \bar{A}$ and vectors $\boldsymbol{c}, \boldsymbol{l}, \boldsymbol{u}, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}$, initial solution z_0, searching tree map $\{\Gamma\}$, tolerance τ, max solver iterations Λ, and max iteration reduction factor ζ.

1 Initialize the QP solver with Q, $\boldsymbol{c}$, $A^* = [A^T \quad \bar{A}^T]^T$, $\boldsymbol{l}^* = [\boldsymbol{l}^T \quad \bar{\boldsymbol{l}}^T]^T$, $\boldsymbol{u}^* = [\boldsymbol{u}^T \quad \bar{\boldsymbol{u}}^T]^T$
2 z, $z^* = z_0$; $V = \infty$; $\mu = 1$; $\nu = 1$; $\lambda = 0$; $\{T\} = \varnothing$; $\{T\} \leftarrow (z, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, \mu, \nu)$
3 **while** $\{T\} \neq \varnothing$ **do**
4 **if** $\lambda > \Lambda$ **then return** V, z^*
5 $\lambda = \lambda + 1$
6 $(z, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, \mu, \nu) \leftarrow$ last element of $\{T\}$
7 Update QP solver with $\boldsymbol{l}^* = [\boldsymbol{l}^T \quad \bar{\boldsymbol{l}}^T]^T$, $\boldsymbol{u}^* = [\boldsymbol{u}^T \quad \bar{\boldsymbol{u}}^T]^T$, and warm start z
8 Run QP solver, set $z =$ solution
9 **if** QP problem is feasible and $\frac{1}{2}z^TQz + \boldsymbol{c}^Tz < V$ **then**
10 **if** $\mu =$ size($\bar{\boldsymbol{l}}$) **then**
11 $V = \frac{1}{2}z^TQz + \boldsymbol{c}^Tz$; $z^* = z$; $\Lambda = \zeta\Lambda$
12 **goto** line 17
13 $(j, \Gamma, \{\gamma\}) \leftarrow \mu$th element of $\{\Gamma\}$; $(\beta, \gamma) \leftarrow$ 1st element of $\{\gamma\}$
14 $\bar{l}_{j+\beta} = 1 - \tau$; $\bar{u}_{j+\beta} = 1 + \tau$; $\mu = \mu + 1$; $\nu = 1$
15 $\{T\} \leftarrow (z, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, V, \mu, \nu)$
16 **else**
17 $(z, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, V, \mu, \nu) \leftarrow$ last element of $\{T\}$; Remove last element of $\{T\}$
18 $(j, \Gamma, \{\gamma\}) \leftarrow \mu$th element of $\{\Gamma\}$
19 **if** $\nu =$ size($\{\gamma\}$) **then**
20 **if** $\{T\} \neq \varnothing$ **then goto** line 17 **else return** V, z^*
21 **else**
22 $(\beta, \gamma) \leftarrow \nu$th element of $\{\gamma\}$; $\bar{l}_{j+\beta} = -\tau$; $\bar{u}_{j+\beta} = \tau$; $\nu = \nu + 1$
23 $(\beta, \gamma) \leftarrow \nu$th element of $\{\gamma\}$; $\bar{l}_{j+\beta} = 1 - \tau$; $\bar{u}_{j+\beta} = 1 + \tau$
24 $\{T\} \leftarrow (z, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, \mu, \nu)$

When the final depth level of the tree obtains an accepted solution, the values of V and z^* are updated in line 11. The max iteration value is also reduced here by some factor ζ that is given by the user. The purpose of the max iteration reduction is to give the user control over how much time the solver takes to find a better solution once a solution is found. Once this step is over, the tree searched for a better solution. This process continues until either the entire tree is searched, or some maximum number of iterations Λ is reached. At that point, the current values of V and z^* are returned as the solution of the MIQP. Note that the returned solution is not guaranteed to be optimal. The feasible region of this MIQP is nonconvex and will settle into a valley based on the construction of the searching tree. While brute force searching through the entire tree will find the optimal solution (if one exists), restrictions on computation time, controlled with Λ, may cause the algorithm to exit with a suboptimal solution.

IV. Numerical Results and Analysis

The following results come from two variations of the algorithm described in the previous section: one using the fine searching set, and the other using the coarse searching set. The algorithm has been implemented in C++ using the eigen linear algebra library (http://eigen.tuxfamily.org). A Python 3.7 script is used to call and time the algorithm with the appropriate inputs and to process the resulting data. Note that the algorithm implementation has not yet been optimized for solve times; the following solve times are presented for comparison between the fine and coarse searching methods. A MacBook Pro 2.8 GHz Intel Core i7 with 16 GB of RAM was used to collect these data.

Convex polygons were used to hand craft 2D models for both the lander and the surface, and the dimensions of these models were left unitless for simplicity. The four-legged lander is composed of five polygons that fit within a 1.75×1.75 unit box, and the feasibility

tolerance is $\tau = 1.11 \times 10^{-13}$. The maximum number of iterations Λ is set to be 20 times the depth of the searching tree, and the iteration reduction factor is $\zeta = 1/2$. Finally, the Python script, given an objective landing location, is set to run the solver 18 times over 20° intervals of $\theta_{R/S}$ for both searching sets.

A. Advantage over Circular Footprint

The initial tests for this solver algorithm were done on a surface composed of a grid of simple hazards. The searchable surface area is constrained to 10×10 units in $\hat{s}_1$ and $\hat{s}_2$, leading to the choice of big-M values to be $D_{sr} = D_i = 10$. These tests yielded results that clearly show the advantage of using a geometrically conforming footprint over a circular footprint. Figure 7 displays a solution for the placement of the geometrically conforming footprint (highlighted in blue) close to the objective landing location (marked with a red X) with a similarly scaled circular footprint centered on the solution. This is a situation where the circular footprint is violated by the arrangement of the hazards, creating a region on the surface where only a geometrically conforming footprint can find a solution. While this result showcases the usefulness of a geometrically conforming footprint, the following more complex set of tests was constructed for a deeper analysis of the solving algorithm and its two variants.

B. Coarse Versus Fine Searching Sets

The rest of the testing is done on a surface that is constrained to 20×20 units in the $\hat{s}_1$ and $\hat{s}_2$ directions and contains 14 polygons, with big-M values $D_{sr} = D_i = 20$. Several trials are run with hand-picked objective landing locations, such to stress test the solver algorithm. Three notable trials are presented graphically in Fig. 8 with some more analytic results in Table 1, where the offset value is the final distance from the center of the lander (point $\boldsymbol{F}$) to the objective landing location (point $\boldsymbol{L}$) as a percentage of the search region bound D_{sr}. Note that no assumptions are made as to if point $\boldsymbol{L}$ is actually reachable. Also shown in Fig. 8 are circles fit around the geometrically conforming footprint to demonstrate the feasibility of the circular footprint compared to the proposed one. Trial 1 demonstrates the expected outcome of this algorithm when the objective landing location is placed among a dense field of hazards that are similar in scale, or smaller, to the lander; the expectation being that both searching sets find a solution, and the fine searching set outperforms the coarse by a nonnegligible margin. However, this expectation changes when larger hazards are involved, which is best shown in trial 2, where the previously discussed drawbacks of the coarse searching set impact its performance significantly. While the result shown in Fig. 8d is among the most noticeable low-performance results of the geometry being tested here, it clearly demonstrates a lack of robustness for the coarse searching set when compared to the fine searching set. Finally, trial 3 demonstrates a case where the coarse searching set is able to outperform the fine, although this outcome is much less common than that of trial 1. The primary reason that the coarse searching set performed better is the aforementioned nonconvexity of the MIQP's feasible region; the solver got caught in a nonoptimal local minimum and returned a solution that is further from the objective than the coarse searching set. Of the results shown from this trial, the solution displayed in Fig. 8f best showcases the advantages of a geometrically conforming footprint over a circular footprint.

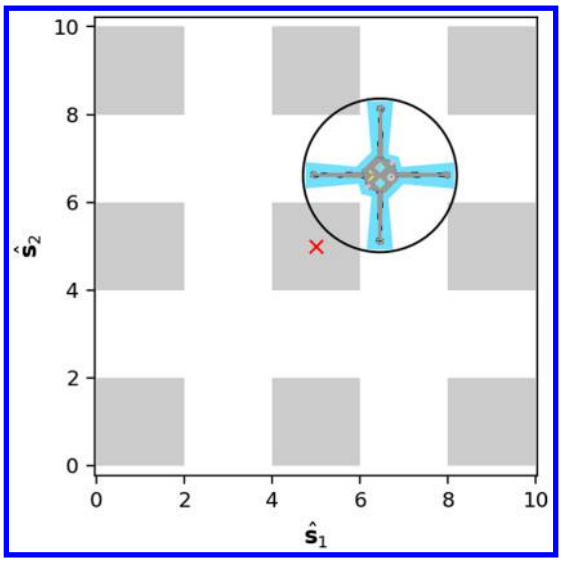


Fig. 7 Example of a geometrically conforming footprint fitting where a circular one cannot.

C. Monte Carlo Testing

Several other hand-picked trials were run to evaluate specific performance cases. Notable results from these trials indicate that the solver is generally robust to the placement of the objective landing location, including cases when the objective landing location is placed within a surface polygon. However, there are some cases where the solver was unable to return a solution when using the coarse searching set. These failures are a result of the iteration limit placed on the solver with the given Λ. Given unlimited time the solver will always return a solution, if one exists, however, a practical limit on solve time is crucial to HDA operations. To further quantify the performance of this solver with the given setting, a set of 1000 objective landing locations within the bounds of the surface area were randomly generated from a uniform distribution. This set was run through the Python script for the solver to produce solve time and objective offset data. Figure 9 shows the distribution of both searching set's objective offset over the generated set of data. Figure 10a shows the percent increase in time required to produce a solution of the fine searching set relative to the coarse, and Fig. 10b shows the objective offset of the fine searching set relative to the coarse for each individual trial, where a negative value indicates that the fine solution has a lower value (and thus higher performance) than the coarse.

The data collected from these 1000 trials further reinforce the expected behavior between the two variants of the solver. The data in Fig. 9a show that the average objective offset for the coarse searching set (when a solution is returned) is 6.3% of the search region bound with a maximum value of 29.3%, whereas Fig. 9b shows that the fine searching set's average objective offset is 4.7% of the search region bound with a maximum value of 24.3%. The indication from these results is that the use of the fine searching set, on average, provides a solution closer to the objective landing location than when the coarse searching set is used. Observation of Fig. 10b further bolsters this claim where, when the results of individual trials are compared, the fine searching set has, on average, an objective offset that is 1.8% of the search region bound less than that of the coarse searching set, up to a value of 18.9% of the search region less than the coarse searching set. While the fine searching set appears to be performing better than the coarse in regard to the objective offset, the previously discussed downside of using the fine searching set must also be examined. As expected, the solver using the fine searching set averages a longer duration of time to return a solution. The data in Fig. 10a show that the fine searching set has on average a 25.04% increase in solving time compared to the coarse, with a minimum and maximum increase of −43.84 and 94.37%, respectively. For reference to the actual scale of time being compared here, on the machine used to run this test, the average solving time for the fine set and coarse set is 27.8 and 36.7 s, respectively. Note that the implementation used for this test was not optimized for computational speed, which is why the solving time of the two variants is being compared relative to each other. Future work will include refinement and optimization of the searching algorithm, as well as warm starting for sequential solving, to meet computation times required for flight. Therefore, the bulk of the results presented here indicate that, with the problem configuration tested here, the fine searching set offers a landing solution that is about 2% closer to the objective landing location that costs about a 25% increase in solving time over the coarse searching set.

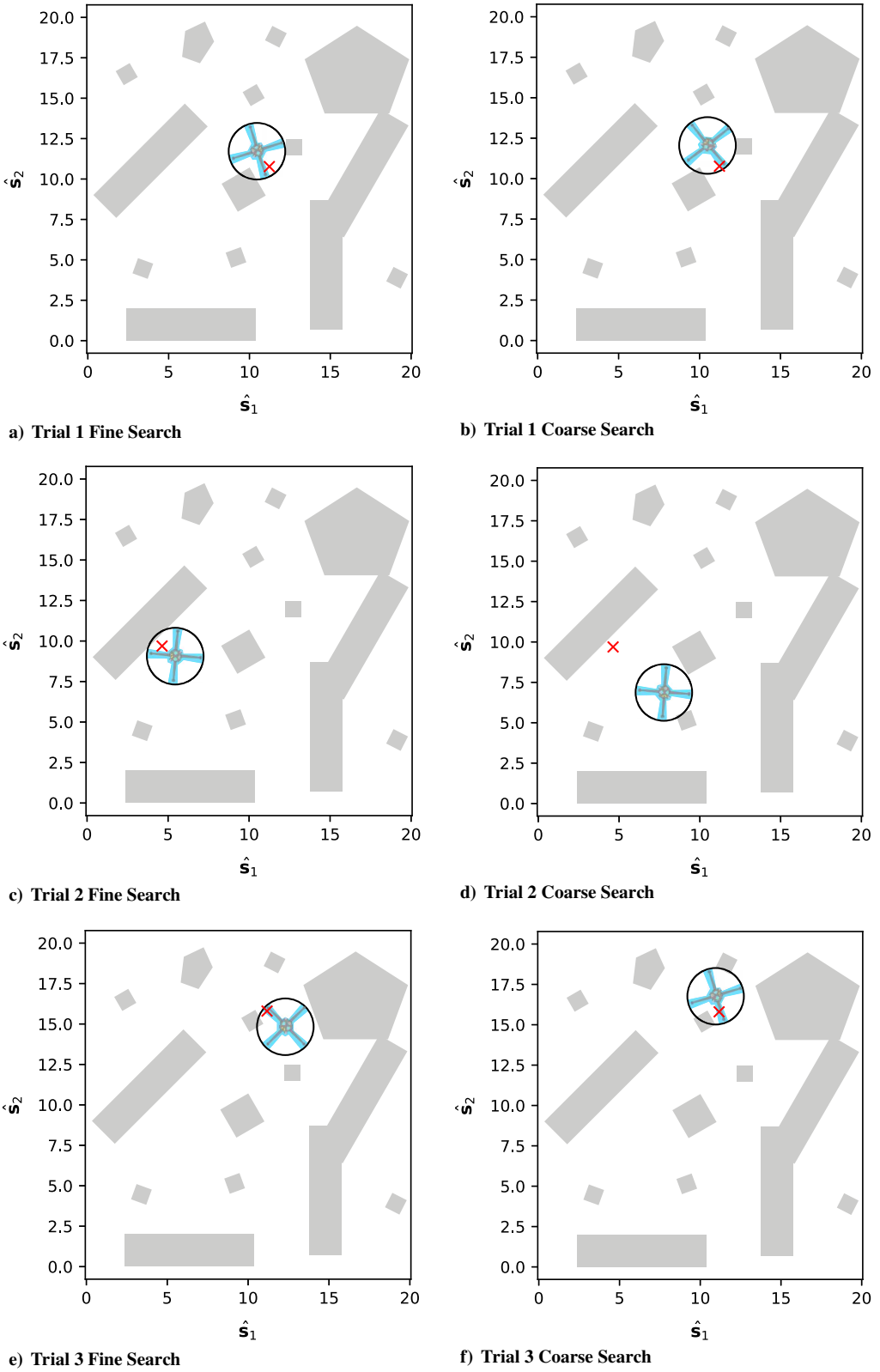


Fig. 8 Visual solutions of the three trials, where the objective landing location is marked by a red X, the geometrically conforming footprint is highlighted in blue, and a black circular footprint is centered over the solution.

Table 1 Numerical results of the three hand-picked trials

Trial number	Fine offset, %	Coarse offset, %
1	6.083	7.387
2	5.331	21.161
3	7.449	4.927

D. Likelihood of Returning a Solution

From the 1000 trials performed here, there were no failures to return a solution when using the fine searching set and there were 32 cases where the coarse searching set failed to return a solution. The cases where the coarse searching set failed had objective landing locations clustered around the $(\hat{s}_1, \hat{s}_2)$ pairs (4, 12.5) and (16, 17). Visual inspection of these regions indicates that, due to the use of the

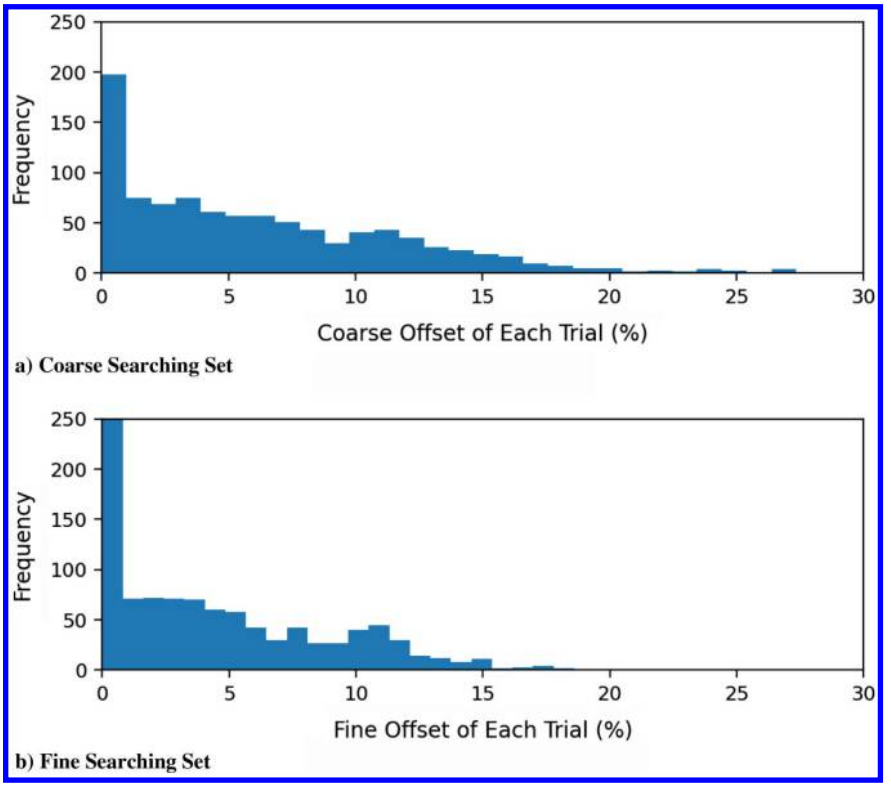


Fig. 9 The objective offset values in each trial for both searching sets.

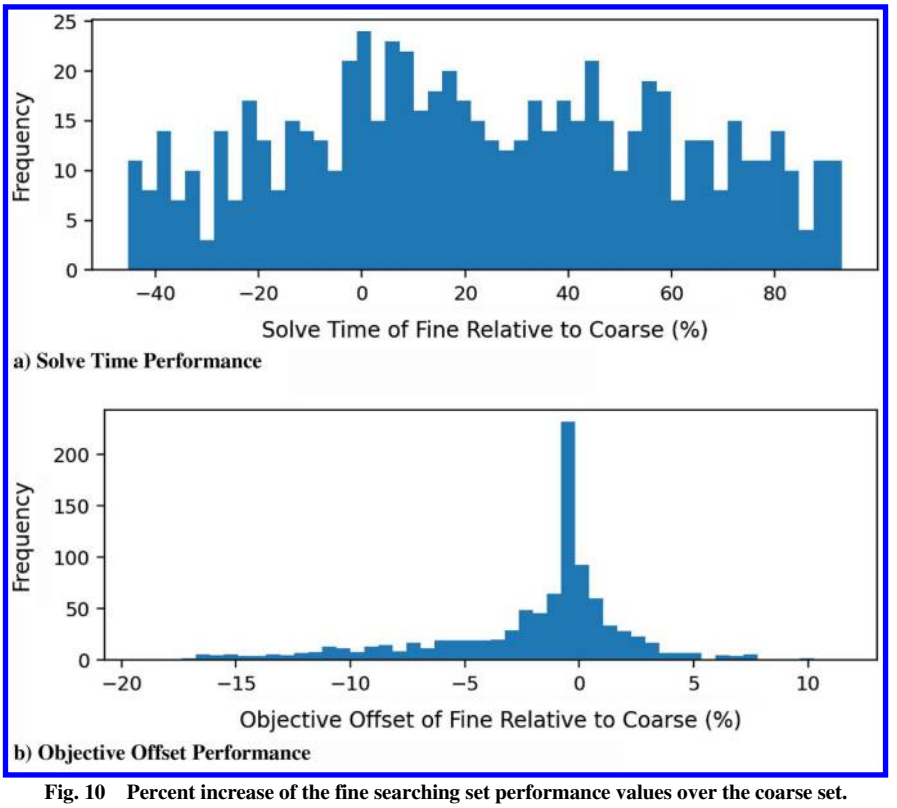


Fig. 10 Percent increase of the fine searching set performance values over the coarse set.

basis directions for separating axes, the feasible region for the coarse searching set is significantly far from these regions. The big disadvantage that the coarse searching set sees from the very large surface polygons becomes highly prevalent here, which opens up a discussion on the required scale and shape of polygons from a given safety map. As mentioned previously, the drawbacks of the coarse searching set become less significant when the local surface polygons are of the same scale or smaller than the lander. A natural solution then would be to further subdivide the larger polygons until every surface polygon is of the same scale as the lander. While this would certainly work, it has the potential to create a significant increase in the time required for the solver to return a solution. Recall the dimensionality

of the defined MIQP, with $Q \in \mathbb{R}^{(3+HK_q)\times(3+HK_q)}$, where H is the number of lander polygons, K is the number of surface polygons, and K_q is the sum of the search directions of each surface polygon. Because the coarse searching set only uses the $\mathcal{S}$ basis directions, the dimension of Q would be $(3 + 4HK) \times (3 + 4HK)$. Therefore, increasing the number of surface polygons by subdivision results in a quadratic increase in the size of the MIQP and thus would increase the duration of time required for the solver to find a solution. As the failures from using the coarse searching set were a direct result of the solver reaching the given iteration limit, raising that iteration limit (and thus giving the solver more time to find a solution) is also a valid adjustment to prevent failure. Thus, to avoid these failure conditions for a given problem set, some combination of subdividing polygons and raising the iteration limit is recommended on a case-by-case basis.

V. Conclusions

In this paper, a new technique is presented as the foundation for a new landing site selection algorithm. The core idea of this technique is to decompose a given 2D safety map of a small-body surface and a geometrically conforming footprint of a spacecraft lander into sets of convex polygons, and create an optimization problem with procedurally generated constraints that prevent the polygons from intersecting via the SAT. This problem is cast into an MIQP that minimizes the distance between a feasible landing location and a desired landing location. Two possible variations of this MIQP are discussed: one that uses the basis directions of the surface frame as possible separating axes, and one that uses the face normal vectors of the surface polygons, named the coarse and fine searching sets, respectively. An algorithm that uses a B&B framework to solve these MIQPs is formulated and has been tested on an illustrative scenario. The results of this scenario show the potential of using a geometrically conforming footprint over a circular one when subject to relatively low navigation uncertainties. Further, the coarse and fine variants of this MIQP are compared to identify the advantages and disadvantages of each variant. The numerical results from this comparison show that the fine searching set, on average, produces solutions that are 1.8% closer to a desired landing location than the coarse searching set, at the cost of a 25.04% increase in time required for the solver to return a solution. Finally, the fine searching set is shown to be more robust than the coarse searching set in the solver's ability to return a valid solution, given the same problem configuration. While this technique assumes a level of navigational uncertainty that is not presently feasible, the recent OSIRIS-REx and Hayabusa2 missions have shown that current technological advancements are on the cusp of attaining such feasibility. Future work includes the incorporation of navigational uncertainty into the MIQP and further development of a solving algorithm that is flight-ready. This technique, as it is presented in this paper, then serves as a foundation to guide the near-future cutting edge in hazard avoidance research.

References

[1] Serrano, N., "A Bayesian Framework for Landing Site Selection During Autonomous Spacecraft Descent," *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Inst. of Electrical and Electronics Engineers, New York, 2006, pp. 5112–5117, http://ieeexplore.ieee.org/document/4059234/. https://doi.org/10.1109/IROS.2006.282603

[2] Epp, C. D., Robertson, E. A., and Brady, T., "Autonomous Landing and Hazard Avoidance Technology (ALHAT)," *2008 IEEE Aerospace Conference*, Inst. of Electrical and Electronics Engineers, New York, 2008, pp. 1–7, http://ieeexplore.ieee.org/document/4526297/. https://doi.org/10.1109/AERO.2008.4526297

[3] Carson, J. M., Munk, M. M., Sostaric, R. R., Estes, J. N., Amzajerdian, F., Blair, J. B., Rutishauser, D. K., Restrepo, C. I., Dwyer-Cianciolo, A. M., Chen, G., and Tse, T., "The SPLICE Project: Continuing NASA Development of GN&C Technologies for Safe and Precise Landing," *AIAA Scitech 2019 Forum*, AIAA Paper 2019-0660, 2019. https://doi.org/10.2514/6.2019-0660

[4] Yu, M., and Cui, H., "Robust Hazard Matching Approach for Visual Navigation Application in Planetary Landing," *Aerospace Science and Technology*, Vol. 47, Dec. 2015, pp. 378–387, https://linkinghub.elsevier.com/retrieve/pii/S1270963815002953. https://doi.org/10.1016/j.ast.2015.09.028

[5] Woicke, S., and Mooij, E., "A Stereo-Vision Hazard-Detection Algorithm to Increase Planetary Lander Autonomy," *Acta Astronautica*, Vol. 122, May 2016, pp. 42–62, https://linkinghub.elsevier.com/retrieve/pii/S009457651600028X. https://doi.org/10.1016/j.actaastro.2016.01.018

[6] Barker, M., Mazarico, E., Neumann, G., Zuber, M., Haruyama, J., and Smith, D., "A New Lunar Digital Elevation Model from the Lunar Orbiter Laser Altimeter and Selene Terrain Camera," *Icarus*, Vol. 273, July 2016, pp. 346–355, https://linkinghub.elsevier.com/retrieve/pii/S0019
103515003450.
https://doi.org/10.1016/j.icarus.2015.07.039

[7] Mourikis, A., Trawny, N., Roumeliotis, S., Johnson, A., Ansar, A., and Matthies, L., "Vision-Aided Inertial Navigation for Spacecraft Entry, Descent, and Landing," *IEEE Transactions on Robotics*, Vol. 25, No. 2, 2009, pp. 264–280, http://ieeexplore.ieee.org/document/4801600/. https://doi.org/10.1109/TRO.2009.2012342

[8] Yan, Y., Qi, D., Li, C., Yu, M., and Chen, T., "A Holistic Vision-Based Hazard Detection Framework for Asteroid Landings," *IFAC-PapersOnLine*, Vol. 49, No. 17, 2016, pp. 218–223, https://linkinghub.elsevier.com/retrieve/pii/S2405896316315105. https://doi.org/10.1016/j.ifacol.2016.09.038

[9] Lunghi, P., Ciarambino, M., and Lavagna, M., "A Multilayer Perceptron Hazard Detector for Vision-Based Autonomous Planetary Landing," *Advances in Space Research*, Vol. 58, No. 1, 2016, pp. 131–144, https://linkinghub.elsevier.com/retrieve/pii/S0273117716301284. https://doi.org/10.1016/j.asr.2016.04.012

[10] Moghe, R., and Zanetti, R., "A Deep Learning Approach to Hazard Detection for Autonomous Lunar Landing," *Journal of the Astronautical Sciences*, Vol. 67, No. 4, 2020, pp. 1811–1830. https://doi.org/10.1007/s40295-020-00239-8

[11] Furfaro, R., Fink, W., and Kargel, J. S., "Autonomous Real-Time Landing Site Selection for Venus and Titan Using Evolutionary Fuzzy Cognitive Maps," *Applied Soft Computing*, Vol. 12, No. 12, 2012, pp. 3825–3839, https://linkinghub.elsevier.com/retrieve/pii/S1568494612000403. https://doi.org/10.1016/j.asoc.2012.01.014

[12] Wei, R., Jiang, J., Ruan, X., and Li, J., "Landing Area Selection Based on Closed Environment Avoidance from a Single Image During Optical Coarse Hazard Detection," *Earth, Moon, and Planets*, Vol. 121, No. 3, 2018, pp. 73–104. https://doi.org/10.1007/s11038-018-9516-2

[13] Cui, P., Ge, D., and Gao, A., "Optimal Landing Site Selection Based on Safety Index During Planetary Descent," *Acta Astronautica*, Vol. 132, March 2017, pp. 326–336, https://linkinghub.elsevier.com/retrieve/pii/S0094576516307214. https://doi.org/10.1016/j.actaastro.2016.10.040

[14] Bhaskaran, S., Nandi, S., Broschart, S., Wallace, M., Cangahuala, L. A., and Olson, C., "Small Body Landings Using Autonomous Onboard Optical Navigation," *Journal of the Astronautical Sciences*, Vol. 58, No. 3, 2011, pp. 409–427. https://doi.org/10.1007/BF03321177

[15] Perenzoni, M., Perenzoni, D., and Stoppa, D., "A 64 $\times$ 64-Pixels Digital Silicon Photomultiplier Direct TOF Sensor With 100-MPhotons/s/pixel Background Rejection and Imaging/Altimeter Mode With 0.14% Precision Up To 6 km for Spacecraft Navigation and Landing," *IEEE Journal of Solid-State Circuits*, Vol. 52, No. 1, 2017, pp. 151–160, http://ieeexplore.ieee.org/document/7756659/. https://doi.org/10.1109/JSSC.2016.2623635

[16] Seabrook, J., Daly, M., Barnouin, O., Johnson, C., Nair, A., Bierhaus, E., Boynton, W., Espiritu, R., Gaskell, R., Palmer, E., Nguyen, L., Nolan, M., and Lauretta, D., "Global Shape Modeling Using the OSIRIS-REx Scanning Laser Altimeter," *Planetary and Space Science*, Vol. 177, Nov. 2019, Paper 104688, https://linkinghub.elsevier.com/retrieve/pii/S0032063318304471. https://doi.org/10.1016/j.pss.2019.07.003

[17] Nakabeppu, S., Ide, Y., Takahashi, M., Tsukahara, Y., Suzuki, H., Shishido, H., and Yamasaki, N., "Space Responsive Multithreaded Processor (SRMTP) for Spacecraft Control," *2020 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS)*, Inst. of Electrical and Electronics Engineers, New York, 2020, pp. 1–3, https://ieeexplore.ieee.org/document/9097637/. https://doi.org/10.1109/COOLCHIPS49199.2020.9097637

[18] Feng, J., Hou, X., and Armellin, R., "Survey on Studies About Model Uncertainties in Small Body Explorations," *Progress in Aerospace Sciences*, Vol. 110, Oct. 2019, Paper 100549, https://linkinghub.elsevier.com/retrieve/pii/S0376042118301908. https://doi.org/10.1016/j.paerosci.2019.05.009

[19] Bercovici, B., "Mapping and Navigation of Small Bodies in the Presence of Uncertainty," Dissertation/Thesis, Univ. of Colorado, Boulder, CO, 2019, https://colorado.idm.oclc.org/login?url=https://www-proquest-com.colorado.idm.oclc.org/dissertations-theses/mapping-navigation-small-bodies-presence/docview/2296357079/se-2?accountid=14503.

[20] Kikuchi, S., Watanabe, S.-i., Saiki, T., Yabuta, H., Sugita, S., Morota, T., Hirata, N., Hirata, N., Michikami, T., Honda, C., Yokota, Y., Honda, R., Sakatani, N., Okada, T., Shimaki, Y., Matsumoto, K., Noguchi, R., Takei, Y., Terui, F., Ogawa, N., Yoshikawa, K., Ono, G., Mimasu, Y., Sawada, H., Ikeda, H., Hirose, C., Takahashi, T., Fujii, A., Yamaguchi, T., Ishihara, Y., Nakamura, T., Kitazato, K., Wada, K., Tachibana, S., Tatsumi, E., Matsuoka, M., Senshu, H., Kameda, S., Kouyama, T., Yamada, M., Shirai, K., Cho, Y., Ogawa, K., Yamamoto, Y., Miura, A., Iwata, T., Namiki, N., Hayakawa, M., Abe, M., Tanaka, S., Yoshikawa, M., Nakazawa, S., and Tsuda, Y., "Hayabusa2 Landing Site Selection: Surface Topography of Ryugu and Touchdown Safety," *Space Science Reviews*, Vol. 216, No. 7, 2020, p. 116. https://doi.org/10.1007/s11214-020-00737-z

[21] Zhao, W., Tong, X., Xie, H., Jin, Y., Liu, S., Wu, D., Liu, X., Guo, L., and Zhou, Q., "Simulation Experiment on Landing Site Selectionusing a Simple Geometric Approach," *ISPRS—International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XLII-3/W1, Jan. 2017, pp. 213–218, https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-3-W1/213/2017/. https://doi.org/10.5194/isprs-archives-XLII-3-W1-213-2017

[22] Best, M. J., *Quadratic Programming with Computer Programs*, 1st ed., Chapman and Hall/CRC, Boca Raton, 2017, Chap. 1, https://www.taylorfrancis.com/books/9781498735773. https://doi.org/10.1201/9781315120881

[23] Boyd, S. P., and Vandenberghe, L., *Convex Optimization*, Cambridge Univ. Press, Cambridge, England, U.K., 2004.

[24] Ghosh, M., Amato, N. M., Lu, Y., and Lien, J.-M., "Fast Approximate Convex Decomposition Using Relative Concavity," *Computer-Aided Design*, Vol. 45, No. 2, 2013, pp. 494–504, https://linkinghub.elsevier.com/retrieve/pii/S0010448512002370. https://doi.org/10.1016/j.cad.2012.10.032

[25] Fasano, G., "Solving Non-Standard Packing Problems by Global Optimization and Heuristics," *Springer Briefs in Optimization*, Springer International Publ., Cham, Switzerland, 2014, pp. 75–89. https://doi.org/10.1007/978-3-319-05005-8

[26] Schaub, H., and Junkins, J. L., *Analytical Mechanics of Space Systems*, 4th ed., AIAA Education Series, AIAA, Reston, VA, 2018, Chap. 3.

[27] Nemhauser, G. L., and Wolsey, L. A., *Integer and Combinatorial Optimization*, Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley, New York, 1988.

[28] Bemporad, A., and Naik, V. V., "A Numerically Robust Mixed-Integer Quadratic Programming Solver for Embedded Hybrid Model Predictive Control," *IFAC-PapersOnLine*, Vol. 51, No. 20, 2018, pp. 412–417, https://linkinghub.elsevier.com/retrieve/pii/S2405896318327265. https://doi.org/10.1016/j.ifacol.2018.11.068

[29] Stellato, B., Naik, V. V., Bemporad, A., Goulart, P., and Boyd, S., "Embedded Mixed-Integer Quadratic Optimization Using the OSQP Solver," *2018 European Control Conference (ECC)*, Inst. of Electrical and Electronics Engineers, New York, 2018, pp. 1536–1541, https://ieeexplore.ieee.org/document/8550136/. https://doi.org/10.23919/ECC.2018.8550136

[30] Stellato, B., Banjac, G., Goulart, P., Bemporad, A., and Boyd, S., "OSQP: An Operator Splitting Solver for Quadratic Programs," *Mathematical Programming Computation*, Vol. 12, No. 4, Dec. 2020, pp. 637–672. https://doi.org/10.1007/s12532-020-00179-2

O. Abdelkhalik
Associate Editor