DEVELOPMENT, VALIDATION AND TEST OF OPTICAL BASED ALGORITHMS FOR AUTONOMOUS PLANETARY LANDING

Marco Ciarambino, Paolo Lunghi, Luca Losi, Michèle Lavagna

Politecnico di Milano
Department of Aerospace Science and Technology (DAER)
34, via La Masa, 20156, Milano, Italy

ABSTRACT

In this paper, a suite of tools and algorithms devoted to optical navigation for autonomous landing on planets and small bodies, currently under development at PoliMi-DAER, is presented. An hazard detection system based on a single camera and artificial neural networks selects the most suitable landing site for the lander. Autonomous guidance on board computes the trajectory to reach for the designated landing site, taking into account the lander dynamics and path constraints. The two systems are linked by vision-based navigation, which reconstructs the relative spacecraft states with respect to the ground, in particular by means of a camera through feature tracking and other typical sensors, such as Inertial Measurement Unit and altimeter. The ultimate goal is to create a whole Adaptive Guidance, Navigation and Control chain for autonomous landings. An experimental facility is currently under construction at PoliMi premises to verify, validate and test the aforementioned optical landing tools.

Index Terms— autonomous landing, optical navigation, adaptive guidance, hazard detection, experimental facility

1. INTRODUCTION

In recent years, a renewed interest in space exploration has induced investing a growing amount of human and financial resources to provide next generation spacecraft with enhanced autonomous navigation and landing capabilities. Complex missions in which close approach to and landing on uncooperative objects play a major role are being developed by numerous space agencies: in particular, ESA is working together with ROSCOSMOS on a cooperative programme for Mars, Phobos and Moon exploration: as our satellite is concerned, Luna-Resurs Lander (Luna-27) mission planned for 2020 and the Luna 29, the Lunar Sample Return mission to follow are involved, strongly focused on the South Pole landing to collect icy volatiles located in a very precise region of the huge Aitken crater. Part of the European contribution for the Luna-27 mission is the PILOT (Precise and Intelligent Landing using Onboard Technologies) subsystem, for enhancing autonomous landing capabilities in terms of high precision landing and hazard detection and avoidance functionalities. An analogous collaboration has been established for the ExoMars programme, which include a lander delivery first, followed by a rover release on Mars, to be launched in 2016 and 2018 respectively and for the exploration of the small Mars moon with the Phootprint Mission, a Phobos Sample Return mission planned for the Twenties. Since 2006, technologies for autonomous landing are studied by NASA in the frame of the Autonomous Landing Hazard Avoidance Technology (ALHAT) program [1]. ALHAT technologies, tested at the end of 2014 in free flight on the Morpheus lander demonstrator, are going to be integrated in future exploration missions. Recently, CNSA has performed its first lunar landing with a lander/rover system with the Chang'e 3 mission (with missions 4 and 5 already scheduled), while ISRO is planning to put a lander carrying a rover on the lunar surface by the early 2018 in the Chandrayaan 2 mission [2]. Among the various technologies under study, vision-based systems represent one of the most promising tools to provide the required level of accuracy, unattainable by classical technologies. In this paper, the research carried out at the Department of Aerospace Science and Technology (DAER) of Politecnico di Milano about algorithms and tools dedicated to autonomous navigation and hazard detection for landing is presented. The functional architecture for an optical autonomous landing system is shown Fig. 1: the navigation subsystem reconstructs the dynamic state of the lander through the camera and other sensors such as IMU and altimeter. Through an input frame of the landing region provided by the navigation camera, the hazard detector calculates the hazard map, exploited by landing site selection routine to identify the best landing site. The adaptive guidance subsystem uses all these informations to compute a feasible fuel-optimal trajectory, taking into account the constraints imposed by the available control authority.

To test and validate these algorithms, an experimental facility is under construction at PoliMi premises, featuring a robotic arm to simulate lander dynamics, a planetary surface mock-up and an illumination system.

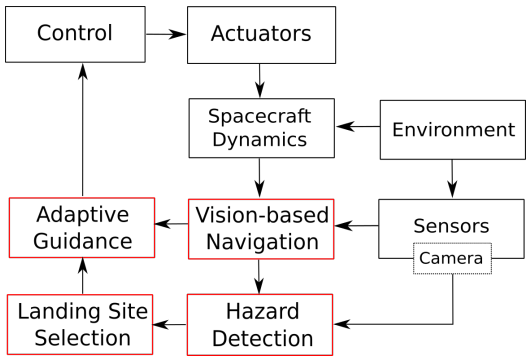


Fig. 1. Functional scheme of a vision-based autonomous landing system. Red boxes denote subsystems presented in this paper.

2. HAZARD DETECTION SYSTEM

The hazard detector in development at PoliMi - DAER features Artificial Neural Networks (ANN). Information extracted from a single-channel image provided by a monocular navigation camera are processed by ANNs to generate a hazard map of the landing area, exploited by the site selection routine to compute the new target. This design choice allows computational efficiency for real-time operations and cost-effectiveness.

2.1. System architecture

The hazard detection system architecture is shown in Fig. 2. The retargeting process is divided in 4 different subphases:

Input and preprocessing. It has been considered as input a 8-bit grey-scale frame with a resolution of 1024×1024 pixels. Since during the hazard detection maneuver the spacecraft is in a near vertical attitude, small deviations from nadir pointing are corrected by applying a perspective transformation.

Image processing and input assembly. The input image is analysed and different indices are extracted. This process computes low level information from the image in order to reduce the data space in which the network should detect the morphological features of the terrain. The same image processing techniques are computed at different scales to allow the ANN to grasp relative distances and depths [3]. Three scales have been used for this work, depending on the different mobile window or image pyramid level utilised: *small*, *medium*, *large*, that downsample the image respectively to 256×256, 128×128, 64×64 pixels. The final hazard map dimension coincides with the small scale sample. Specifically, the indices extracted from the image can be divided in *window-based* and *global* indices. First category is represented by mean μ and standard deviation σ of the current mobile window pixels. They are defined as:

$$\mu = \frac{\sum_{i=1}^{N} I_i}{N}, \qquad \sigma = \sqrt{\frac{\sum_{i=1}^{N}(I_i - \mu)^2}{N-1}} \tag{1}$$

where I_i corresponds to the intensity of the i-th pixel, N is the number of pixels inside the considered image window.
Global indices image gradient are instead computed through custom kernels convolution across the whole image and then downsampled to match the dimensions of the window-based indices. The image gradient *Grad* is calculated through an expanded 5×5 Prewitt filter [4] for vertical and horizontal directions. The square root of the sum of the square of every directional gradient yields the total image gradient magnitude. The Laplacian of Gaussian *LoG* combines a Gaussian filter with the Laplacian operator, and it is often used as an edge detector in images [5]. It is defined analytically as:

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{2}$$

where x, y are the image coordinates and σ the standard deviation. In this work it has been implemented as a 5×5 discrete linear kernel to convolve the image with.
Over these indices, also the Sun elevation angle above the horizon is assembled in the total input matrix, allowing the network to understand features at different illumination conditions.
Summarising, 13 indices are fed into the network to compute a single pixel of the hazard map: μ, σ, *grad*, *LoG* for three different image scales, Sun elevation angle.

Hazard map computation. The assembled input matrix is processed by a cascade neural network. In this kind of structure, each neuron is assembled in sequential hidden layers and has two inputs: the original input matrix plus the previous hidden layers outputs. During the training, the network is progressively increased adding one hidden layer at once, leading to a near-optimal configuration [6].
The output of the network represents a pixel of the hazard map, whose value spans from 0 (completely safe), to 1 (completely unsafe). To relate nearby points, a light blur filter is applied to the output of the neural network.

Target landing site selection. As the hazard map is available, a new target landing site is computed. All the sites that do not respect minimum requirements on safety or dimension (taking into account the lander footprint, margined with the expected dispersion at touchdown) are immediately discarded. The remaining candidate landing sites are then ranked according to three criteria: minimum hazard index, maximum landing area, minimum distance from the nominal landing site (to maximise the probability to find a landing site actually reachable with the divert capabilities of the lander). The influence of these parameters can be adjusted with three corresponding weights to satisfy user needs or to maximise the performances.

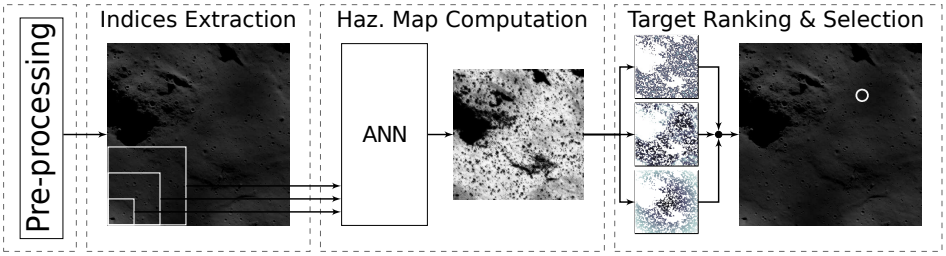


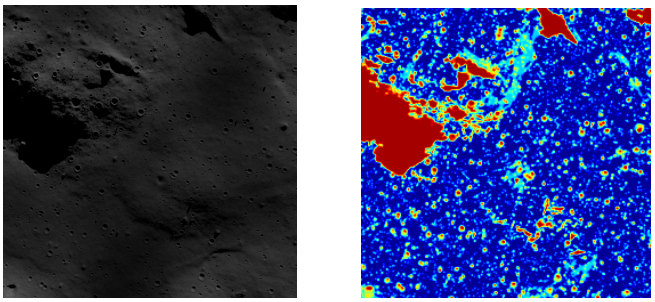
Fig. 2. Hazard detector working flow. After acquisition, image features are extracted and fed in the artificial neural network to estimate the hazard map. Landing site is then computed taking into account multiple parameters.

2.2. Lunar dataset

High resolution lunar DEMs (5-2 m/point) from LROC[1] have been used as the base to craft an artificial images dataset. Since real images usually lack of metadata like camera model, altitude and attitude of the lander and does not include a detailed 3D terrain model, we opted to generate the artificial images dataset. At first, resolution has been increased up to 0.3 m/point adding small craters, boulders and fractal noise [7, 8, 9]. The camera frame is then rendered through the *POV-ray* open source software[2] with realistic illumination conditions assuming a pinhole camera with 60° angle of view.
Ground truth is computed directly from DEM data: for each elevation point, a circular window of size equal to the lander footprint is considered: the mean plane of the DEM points in the window is calculated by least squares approximation. Computing mean plane inclination and the difference between the maximum and minimum deviation of the window points from the mean plane, the *slope* and the *roughness* of the terrain are respectively computed. Converting the two in camera coordinates, it is possible to render the slope and the roughness ground truth maps. Each point is considered safe if the local slope or the local roughness do not exceed the values imposed by the user. Also a shadow map rendering is performed applying a threshold of camera image.
The maximum hazard index of 1 is assigned to pixels in shadows, meaning *completely unsafe*. An index of 0.33 is assigned to a pixel whose slope or roughness is out of the limits imposed, 0.66 if both are exceeded, 0 if the pixel safe to land on. From the 1024×1024 pixels original resolution, the ground truth is downsampled through Gaussian pyramids to 256×256 pixels.

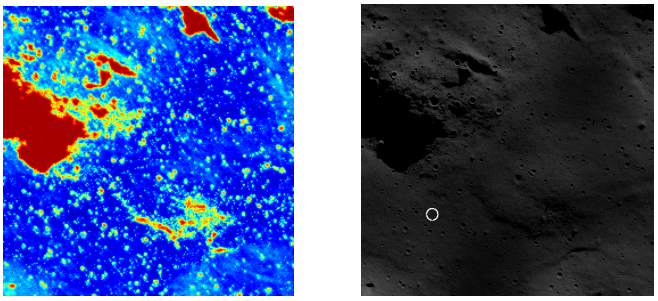
2.3. Performances

Performances are assessed comparing the landing sites found by the hazard detector onto the produced hazard map with the sites found onto the ground truth. The test set consists in 8 images of four lunar regions rendered at two different Sun inclination, 15° and 80°. A lander of 3 meters of diameter in footprint and a navigation error of 15 meters at 3σ has been considered. For the ground truth hazard calculations, terrain roughness over 0.5 meters and slopes over 15° are considered unsafe.



(a) Rendered lunar surface (b) Ground truth

Fig. 3. Example of lunar surface rendering from augmented LROC DEMs and corresponding ground truth hazard map. Blue are safe areas (0), red completely unsafe in shadow areas (0).



(a) Computed hazard map (b) Landing site found

Fig. 4. Computed hazard map and landing site found.

Computed hazard map and landing site found relative to the rendering in Fig. 3 are shown in Fig.4. The terrain fea-

[1]Courtesy of NASA and Arizona State University. URL: http://wms.lroc.asu.edu/lroc/rdr_product_select, last visit on: March 7, 2016.

[2]Persistence of Vision Raytracer (Version 3.7). Retrieved from http://www.povray.org/download/

tures are correctly interpreted by the network.
To assess the performances, landing sites found on the computed hazard map are compared to the ones found on the ground truth. In particular, a landing site can be classified as *True Positive* (TP), meaning a correctly identified landing site, a *False Positive* (FP) an actually unsafe site considered erroneously safe, a *False Negative* (FN), that is an actually safe site considered erroneously unsafe, or a *True Negative* (TN), a correctly identified unsafe landing site.
Defining the *Safety Ratio* r_S as the fraction of true positives with respect to the total number of landing sites found:

$$rs = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{3}$$

and defining the *Correctness Ratio* r_C as the fraction of correctly identified sites (TP) with respect to the true safe landing site in the image:

$$r_C = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

the probability to select an unsafe site is minimized maximizing r_S, whereas as r_C increases, the available landing area increases. The whole performance can be assessed with a unique index J expressed as:

$$J = r_S^5 R_C^{1/5} \tag{5}$$

where the two exponents privilege landing sites safety during J maximization. Hazard map safety thresholds between 0.04 and 0.3 for the landing sites computations in the hazard map have been tested. Maximum J is recorded for a safety threshold of 0.17. As outlined in Section 2.1, three weights are adjustable to have a greater flexibility in the landing sites selection. The greatest coefficient of 0.6 has been selected for the landing site area, to increase robustness to navigation errors. A value of 0.3 is given to the distance with respect to the nominal landing site and the lowest of 0.1 is given to the mean hazard index of the site, because the system takes already into account a very safe threshold of 0.17.
With these parameters, the system:

- always selects a True Positive as Target Landing site;
- the worst case in ranking of the first False Positive is position 39, with an average on the test dataset of 695;
- an average Safety Ratio of 0.9649, meaning that over 96% of the landing sites found are actually safe.

For what concerns *computational performances*, the hazard detection system software is coded in C++ for its "flight" version. Profiling has been executed with three different tools: *GperfTools*, sampling profiler capable to reach up to 250 Hz; the high resolution clock of the standard C++ *chrono* library; the *GNU time* command. The hazard detector ran forced on a single thread of the processor for 1000 times to mediate the effects of the operative system overheads on the CPU. All tests have been performed on a AMD A10-7700K APU, running 64 bit Ubuntu 14.04 GNU/Linux operative system. Gperftools hit 108 148 times at 250 Hz for a total time of 432.59 s, while CPU time resulted 432.67 s. The conceptual difference in the working method of the sampling Google profiler and the two CPU time profilers allow to assume a decent reliability in the computational times. As expected, the heaviest routine in the algorithm resulted the indices extraction, spanning almost 50% or the total hazard detection software runtime. However, due to recent improvements in space qualified hardware such as Field Programmable Gate Arrays [10, 11], it will be possible to split in parallel threads the heaviest parts of the algorithm –indices extraction in primis– with dramatic improvements expected.

3. ADAPTIVE GUIDANCE

Once a safe landing site is selected, the system must compute a new feasible trajectory toward the new target. A fuel-optimum criterion is followed, in order to maximize the attainable landing area in subsequent target updates, that may be required as soon as smaller terrain features become observable, with the decreasing of the altitude. Here the structure of the algorithm is summarized, and some numerical results are presented. For a detailed description, see [12].

A planetary landing is characterized by fast dynamics. The expected time of flight of the approach phase in which HDA tasks take place is in the order of $1\,\mathrm{min}$, and the mass is supposed to significantly change during the maneuver. In this case, distances, for both downrange and altitude, are small compared to the planet's radius; thus, the assumption of a constant gravity field with flat ground is appropriate. Furthermore, aerodynamic forces are neglected: the effects of the possible presence of atmosphere (especially for low densities, as in the case of Mars) could be omitted due to the relative low velocity (on the order of $100\,\mathrm{m\,s^{-1}}$) and the associated forces can be treated as disturbances [13]. The translational dynamics of the spacecraft are expressed in a ground reference system as:

$$\dot{\mathbf{r}} = \mathbf{v} \qquad \dot{\mathbf{v}} = \frac{\mathbf{T}}{m} + \mathbf{g} \qquad \dot{m} = -\frac{T}{I_{\text{sp}} g_0} \tag{6}$$

where $\mathbf{r} = [x, y, z]^T$, x is the altitude, y is the downrange direction and z is the cross-range; $\mathbf{g}$ is the constant acceleration of gravity vector of the planet, I_{sp} the specific impulse of the main engine, and g_0 the standard gravity acceleration on Earth. The thrust net magnitude is indicated with $T = \|\mathbf{T}\|$.

The thrust vector acts as the control variable. The mass equation is linked to the control acceleration by the thrust-to-mass ratio $\mathbf{P}$:

$$\mathbf{P} = \mathbf{T}/m = \dot{\mathbf{v}} - \mathbf{g} \tag{7}$$

Then, the mass equation in system (6) can be rewritten as a

first order linear ordinary differential equation:

$$\dot{m} = -\frac{P}{I_{\text{sp}} g_0} m \tag{8}$$

where $P = \|\mathbf{P}\|$.

The states $\mathbf{r}_0$, $\mathbf{v}_0$ and m_0 at the initial time t_0 are supposed to be known. At the end of the maneuver, at time t_f, the final states $\mathbf{r}_\text{f}$ and $\mathbf{v}_\text{f}$ are constrained to assume fixed values. Then, the optimal guidance problem is to find a control profile $\mathbf{T}(t)$ to bring the system from the initial to the target final states, compatibly with all the constraints imposed by the actual system architecture. For sake of simplicity is considered $t_0 = 0$.

The main thruster is assumed to be tightly connected to the spacecraft body. Then, the thrust vector depends only on the attitude of the spacecraft, expressed by the vector of Euler's angle $\mathbf{e}$ and on the thrust magnitude T. At the beginning of the maneuver, the attitude is assumed to be known. Then, the initial acceleration is function only of the initial thrust magnitude. Moreover, at the end of the maneuver, the lander's is required to be aligned with the local vertical on the Target Landing Site. In case of flat surface, this condition reduces to impose null horizontal acceleration. A total of 17 boundary constraints are then available for position, velocity and acceleration components: 6 on initial states, 3 on initial acceleration (function of initial thrust magnitude), 6 on target final states and 2 on the final acceleration due to the final attitude requirements

$$\begin{aligned} \mathbf{r}(0) &= \mathbf{r}_0 & \mathbf{r}(t_\text{f}) &= \mathbf{r}_\text{f} \\ \mathbf{v}(0) &= \mathbf{v}_0 & \mathbf{v}(t_\text{f}) &= \mathbf{v}_\text{f} \\ \dot{\mathbf{v}}(0) &= \mathbf{f}(T_0) & \dot{\mathbf{v}}(t_\text{f}) &= [\textit{free}, 0, 0]^T \end{aligned} \tag{9}$$

The 3 components of the acceleration can be expressed in a polynomial form. The minimum order needed to satisfy the boundary constraints is 2 for the vertical axis, 3 for the horizontal components:

$$\dot{\mathbf{v}}(t) = \begin{bmatrix} \dot{v}_\text{x} \\ \dot{v}_\text{y} \\ \dot{v}_\text{z} \end{bmatrix} = \begin{bmatrix} \dot{v}_{0\text{x}} + c_{1\text{x}} t + c_{2\text{x}} t^2 \\ \dot{v}_{0\text{y}} + c_{1\text{y}} t + c_{2\text{y}} t^2 + c_{3\text{y}} t^3 \\ \dot{v}_{0\text{z}} + c_{1\text{z}} t + c_{2\text{z}} t^2 + c_{3\text{z}} t^3 \end{bmatrix} \tag{10}$$

By integrating the acceleration twice and applying the boundary conditions, the trajectory is parametrized in terms of time-of-flight t_f and initial thrust magnitude T_0, that are considered as optimization parameters. Once the acceleration profile is defined, the thrust-to-mass ratio can be obtained from Eq. (7) and the thrust profile is:

$$\mathbf{T} = m\mathbf{P} \tag{11}$$

where the mass profile is obtained by solving Eq. (8). An analytical solution is not available, nevertheless, an approximate value of the integral is computed numerically. From the thrust vector a complete guidance profile, in terms of Euler angles and thrust magnitude, is easily obtained.

In addiction to boundaries constraints, the system is subject also to path and box constraints. The initial thrust magnitude is bounded to the thrust actually available on-board, while the time-of-flight must lie between its lower and upper limit:

$$0 < T_{\text{min}} \leq T_0 \leq T_{\text{max}} \tag{12}$$

$$0 < t_{\text{min}} \leq t_\text{f} \leq t_{\text{max}} \tag{13}$$

The theoretical t_{max} is determined by the amount of fuel on board m_{fuel}, whereas the adopted t_{min} corresponds to the time required by the lander to reach the ground with maximum thrust pointing downward. Evidently t_{min} does not corresponds to a feasible soft landing maneuver, and it is adopted as a theoretical lower limit to exclude singularities arising towards $t_\text{f} = 0$. The angular velocity of the spacecraft is limited by the actual control torques M_{Cmax} given by the Attitude Control System (ACS). The extrapolation of the exact torques from angles is not immediate, due to the coupled terms in the attitude dynamics. Torques are approximated by the decoupled term due to the angular acceleration, which is a sufficiently accurate approximation in case of small angles and low angular speed. Exploiting this approximation leads to the following inequality:

$$I_{\text{max}} \|\dot{\boldsymbol{\omega}}(t)\| \leq M_{\text{Cmax}} \tag{14}$$

in which $\dot{\boldsymbol{\omega}}$ is the derivative of the rotational velocity vector, and I_{max} is the maximum moment of inertia at initial time t_0. The rotational velocity $\boldsymbol{\omega}$ is obtained from the relation:

$$\boldsymbol{\omega} = \frac{\mathbf{a} \times \dot{\mathbf{a}}}{\|\mathbf{a}\|^2} \tag{15}$$

where $\mathbf{a} = \dot{\mathbf{v}} - \mathbf{g}$ is the control acceleration vector whose derivative $\dot{\mathbf{a}}$ is known exactly, being a polynomial in time. The spacecraft is required to remain in a cone pointed at the target and defined by the maximum slope angle δ_{max}. This constraint has has the dual purpose to assure that the the lander does not penetrate the ground, even in presence of bulky terrain features near the landing site, and to limit the angle of view on the target. In fact, the performances of vision-based navigation systems depend on the inclination between the trajectory and the ground [14, 15]. The constraint is expressed by the inequality:

$$r_\text{y}^2(t) + r_\text{z}^2(t) \leq r_\text{x}^2(t) \, \tan^2(\delta_{\text{max}}) \tag{16}$$

Path constraints need to be satisfied at every time instant during the landing. Pseudospectral techniques allow us to evaluate them discretely at Chebyshev-Gauss-Lobatto (CGL) points. Derivative terms of the rotational vector are obtained by the use of the Chebyshev differentiation matrix [16].

Finally, the final mass must be included between the initial value and the spacecraft dry mass. Since the mass trend

Table 1. Lunar landing MC analysis, large diversion maneuver case: parameters dispersion.

Quant.	Nominal	std	Units
$\mathbf{r}_0$	$[2000, -1062, 0]$	$[130, 600, 600]$	m
$\mathbf{v}_0$	$[-35, 30, 0]$	$[2, 2, 2]$	$\mathrm{m\,s^{-1}}$
$\mathbf{e}_0$	$[-55, 0, 0]$	$[5, 15, 0]$	deg
$\mathbf{r}_\mathrm{f}$	$[30, 0, 0]$	-	m
$\mathbf{v}_\mathrm{f}$	$[-1.5, 0, 0]$	-	$\mathrm{m\,s^{-1}}$
$\mathbf{e}_\mathrm{f}$	$[-90, 0, 0]$	-	deg
m_0	865	10	kg
m_dry	790	-	kg
g_moon	-1.624681	$5\,\%$	$\mathrm{m\,s^{-2}}$
I_sp	325	$5\,\%$	s
T_min	1000	$5\,\%$	N
T_max	2320	$5\,\%$	N
I_max	1000	$5\,\%$	$\mathrm{kg\,m^2}$

is strictly monotone by problem construction, the only constraint with respect the minimum mass is verified:

$$m(t_\mathrm{f}) \leq m_\mathrm{dry} \tag{17}$$

3.1. Optimization Problem

The optimization problem takes the form:
Find T_0 *and* t_f*, in the domain defined by the inequalities* (12)*, that minimize the fuel consumption computed by the Eq.* (8)*, subject to constraints* (14)*,* (16)*, and* (17)*.*
The optimization could be solved with any non-linear programming (NLP) solver: the choice of this solver has a huge impact over the final convergence properties and computational time. At DAER, a dedicated optimization algorithm based on Taylor Differential Algebra (DA) was developed.

Differential Algebra techniques were devised to attempt solving analytical problems through an algebraic approach [17]. Instead to be modelled as simple real numbers, quantities are represented as their Taylor expansion around a nominal point. In this way, DA variables carry more information rather than their mere punctual values. Computing the objective function as a DA variable it is possible to estimate its sensitivity to the variation of the optimization variables. Then through the operation of map inversion, stationary points of the function can be found in a restricted number of iterations, using only simple algebraic computation between Taylor coefficients. A detailed discussion about the developed optimization algorithm is included in [12].

3.2. Numerical Results

To estimate the performances of the proposed algorithm, a series of different simulations regarding a realistic case of lunar landing were carried out. Extensive MC simulations were exploited to assess the robustness and the flexibility of the proposed approach. The parameters considered in the initial dispersion include initial position, velocity, attitude, amount of fuel on board, specific impulse, spacecraft moment of inertia, available thrust and gravity acceleration. The case here reported as example represents a large scale diversion maneuver from a nominal altitude of $2000\,\mathrm{m}$. The initial parameters are summarized in Table 1, while the trajectories obtained are shown in Figure 5: in all the cases, the ordered diversion was found feasible by the guidance algorithm. These results were compared with the solutions computed with a general-purpose non-linear optimization software (SNOPT): in the worst case observed, the difference between the two solutions was less than $0.2\,\%$.

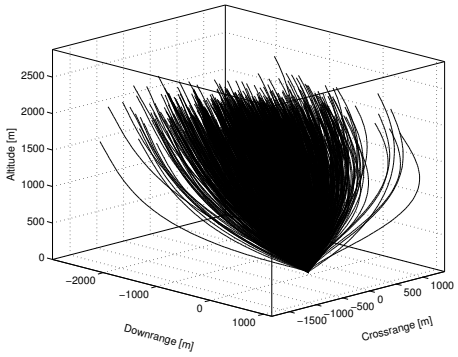


Fig. 5. Large scale diversion maneuver simulation (from Ref. [12]).

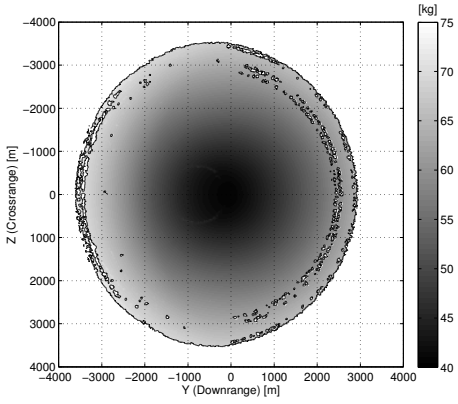


Fig. 6. Attainable area and fuel consumption comparison (from Ref. [12]).

A Monte Carlo (MC) simulation is exploited also to assess the algorithm performances in terms of attainable landing area and fuel consumption. A series of 1×10^5 random diversions between $\pm 4000\,\mathrm{m}$ along both the horizontal axes is ordered from the same nominal conditions of Table 1. The attainable landing area can be obtained by correlating optimization results together with the coordinates of the TLSs, as shown by Figure 6, in which only the feasible points (satisfying all the

constraints) are shown. The system is able to compute a feasible landing path in an approximately circular landing area of radius larger than 2300 m centred at the nominal landing site (at the origin of the figure), a performance better than what is required for similar scenarios [18, 19].

The simulation of Figure 6 was exploited also to obtain an estimation of the computation time. All the simulations were tested on a Intel® Core™ i7-2630QM CPU at 2 GHz of frequency. The mean computation time is 25.23 ms with a standard deviation (STD) of 7.16 ms. The algorithm is very fast (and further improvements are possible in code optimization), compatible with on-board computation.

4. OPTICAL NAVIGATION

The Optical Navigation algorithm relies on a single grayscale camera and is based on Simultaneous Localization and Mapping (SLAM) procedure. Information will be shared with all the other subsystems, with particular attention given to the cooperation of the generated map thread with the HDA. Monocular navigation is lately a topic of great interest because of the advantages of been reliable, precise and low cost, being the technology of digital cameras widely tested and consolidated for space applications.

4.1. System architecture

The current outline of the system is under development, inspired by latest advances in robotics while considering the typical constraints of space applications.
The proposed architecture for the Navigation System is shown in Fig. 7.
Currently the Feature detection, the Matching and the Tracking branches are under development, while the map thread is just defined conceptually and will be addressed in future.
The main idea is to follow the path introduced by [20], with tracking and mapping running in separate CPU threads, but in a more advanced way as in [21] and [22]. This would be actually possible thanks to the later development of new multicore processors specifically designed for space applications. Algorithm is bootstrapped by images coming from the camera and works as follow:

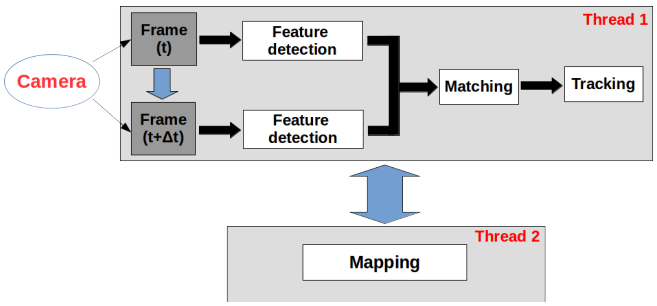


Fig. 7. Navigation System architecture. Grayscale frames from camera are sent to the algorithm which works on two threads. One dedicated to features extraction and tracking and the other dedicated to mapping.

Feature detection. Oriented FAST and Rotated BRIEF (ORB) features are extracted from the current frame, an upper bound limit on the number of those is set. These features are invariant to scale and rotation and are resistant to noise. For each feature a descriptor is also associated.

Matching. Once a couple of frames with their respective ORB features extracted (along with their descriptors) are available, matching exploiting Hamming distance is performed and is then exploited to discard outliers.

Tracking. After features correspondences between two consecutive frames are obtained, roto-translation which connects the two camera pose is computed. This is achieved exploiting 2D-2D correspondences of point features to solve the Five Point Relative Pose Problem [23]. Once rotation matrix and translation vector are calculated, the scale problem must be addressed since the translation vector is actually not scaled. At this step, without external measurements included (nor from IMU or Laser Altimeter), it is only possible to find relative scale exploiting triangulation of 3D points from 2D features.

Mapping. The mapping branch is designed to work independently on a separate thread, parallel to the already mentioned steps, similarly to [22] and [21]. Triangulated 3D points are given as input along with frames from which are obtained. The algorithm shall improve their 3D location (e.g. with Bundle Adjustment BA), and then build the map which will be used from the tracking thread in subsequent steps in order to improve pose estimation precision solving Efficient Perspective-n-Point (ePnP) problem.

4.2. Feature extraction

The first processing step for each frame sent by the camera to the system is the feature extraction.
Different approaches to this problem exist and may be devided into feature based methods, with matching or tracking exploiting optical flow, and direct methods, which work directly on pixel intensities. According to [22] feature based methods, despite a bit higher computational cost, assure the highest performances in terms of precision.
Most important objectives to be met by the feature detection system are scale invariance and low computational cost of the whole process. This because of the constraint of real time application and because the s/c, being on a landing trajectory, is expected to move increasingly close to the surface of a target body, thus giving as input images with a scale that will change drastically along the whole operation.

Given these considerations, ORB features extraction [24] has been chosen. ORB is a very fast binary descriptor based

on Binary Robust Independent Elementary Features (BRIEF) [25]. Features extracted are extremely fast to compute and match, while they have good invariance to viewpoint, scale and are resilient to different light conditions. The feature detector is built on the Feature from Accelerated Segment Test (FAST) corner detector and on the BRIEF descriptor, both techniques which have good performances and low computational cost.
At the moment ORB has been implemented and set to build for each frame in input an 8-level grey scale pyramid with a scale factor of 1.2 and to extract 500 features along with their descriptors. In Fig. 8 it is possible to see the application of ORB on one of the frames from a synthetic video of a landing trajectory on the Moon with camera pointing downwards. ORB runs faster than other feature detection methods as the

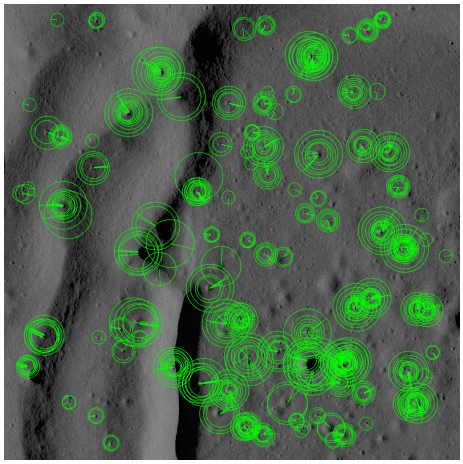


Fig. 8. Example of ORB features from data set image. Circle around keypoints with size according to scale and orientation are shown.

Scale Invariant Feature Transform (SIFT) or the Speed-Up Robust Feature (SURF) without loss of performance, resulting suited for real-time applications.

4.3. Matching

Matching is performed exploiting Hamming distance as similarity measure between two descriptors. For each descriptor in the first frame, the matcher finds the closest descriptor in the second frame by trying one by one. Hamming distance can be computed very efficiently between corresponding binary descriptor strings and is a peculiarity of ORB which makes the process very fast.

Since at least 5 matching are needed for the tracking to solve for roto-translation between a couple of frames, matching is performed with subsequent incoming frames and rejection is done until at last 5 matches are found. Once the matching is completed, result is fed into a routine which checks the distances of extracted matches and discards ones whose distance is above a certain threshold. This is an heuristic way to immediately discard clear outliers and retain only good matches. In Fig. 9 an example is given to show the obtained pairs even with frames separated by a wide baseline.

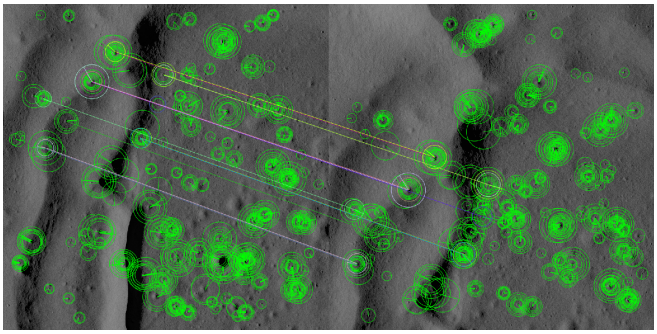


Fig. 9. Example of ORB features matched between two data set images for the landing trajectory with a wide baseline (60 frames interval). Circles around key-points with size according to scale and orientation are shown.

4.4. Tracking

Matched ORB features are given as input to the tracking block which has the task to reconstruct the motion of the spacecraft frame by frame, in order to implement Visual Odometry (VO).
VO intends to reconstruct the pose frame to frame from 2D to 2D image coordinate correspondences expressed in the two camera reference systems, from which rotation matrix and translation vector and consequently pose of the spacecraft are obtained.
In order to set the problem, along with feature correspondences, it is mandatory to know the calibrated camera matrix, representing the link from the 3D world seen by the camera to the 2D image plane.

First step is the calculation of the *Essential matrix (E)*, which relates corresponding 2D points on two different images according to epipolar constraints [26]. This can be done solving the Five-Point Relative Pose Problem [23].
The algorithm consists of computing the coefficients of a tenth degree polynomial in closed form and subsequently finding its roots. Cheirality check allows to select among the possible solutions (requiring for the reconstructed point correspondences to lie in front of the cameras), and results obtained are improved by RANSAC iterations to reject the presence of outliers. Rotation matrix and translation vector are thereafter computed solving a simple Singular Value Decomposition (SVD) problem from E.
Absolute scale of the translation can be computed only through data fusion, in which measurements coming from other sensors (IMU, Laser Altimeter), are properly filtered along with the ones coming from VO. At the moment these

measurements are not integrated.
Considering this limitation, relative scale between each subsequent frame can be computed and thus a scale factor to be applied to each iteration.
This scale factor is obtained triangulating 3D points from subsequent image pairs. From corresponding 3D points, the relative distances between any combination of two 3D points in the same frame can be computed. The proper scale can be then determined from the distance ratio between a 3D point pair obtained from the first image couple and from a 3D pair in the second image couple. This way, concatenating all the rotation matrices and translation vectors a coherent trajectory for the spacecraft is obtained up to a known relative scale factor.
About *computational performances* tests have been performed on a Intel Core i7 CPU, running 64 bit Ubuntu 15.10 GNU/Linux operative system.
All the algorithm has been coded in C++ exploiting the open source OpenCV-3.1.0 [27] library. Running the algorithm on a single CPU thread, an average time of 26.07 ms and 8.44 ms have been registered respectively for the feature detection and the feature matching subroutines. Being the tracking part under heavy development, no computational time is available.

5. EXPERIMENTAL FACILITY

To further increase the TRL of the aforementioned algorithms, an hardware-in-the-loop experimental facility is under setup at DAER premises. Since the scarce availability of complete real landing imagery datasets, vision-based algorithms development relies widely on synthetic images. To validate such approach, experiments are necessary. Moreover, the whole navigation system performance can be assessed only connecting the composing parts together, to verify mutual influences. As explained in details in next sections, the facility is composed by a robotic arm carrying the sensors suite to simulate lander dynamics, a 3D planetary mock-up, an illumination system, control and test computers. A sketch of the setup is present in Fig. 10. The goal is to reproduce the landing maneuver over a reproduced planetary environment with a realistic illumination condition. The system is designed to verify either hardware and software breadboards up to TRL 4, with possible further enhancements to qualify flight models to TRL 5.

5.1. Planetary mock-up

The selection of a proper scale factor is needed to properly determine the facility manufacturing requirements. It is assumed that the hazard detection starts at a maximum altitude of 2000 m. For reasons of cost containment, the choice of the robotic arm was restricted to the use of an already available hardware, with an operative envelope of 1 m. Then, a maximum scale factor of 2000:1 has been considered. The

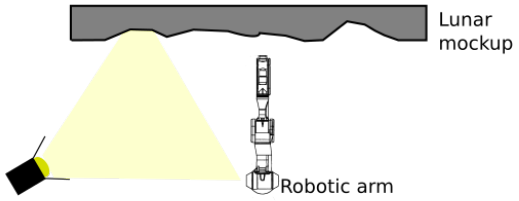


Fig. 10. Experimental facility room setup.

expected accuracy at touchdown is in the order of 10 m due to navigation errors, resulting equal to 5 mm in the scaled environment. To have a resolution of the terrain at least one order of magnitude greater than the landing accuracy, a resolution of at least 0.5 mm is necessary. The scale factor can be adapted to simulate closer range maneuvers with higher details, due to the fractal structure of the Moon surface.
The *Renshape® BM5460* polyurethane foam has been chosen as material for the lunar diorama due to its surface finish that yields the correct optical properties and because of its great workability. Considering the selected scale, the maximum envelope of the robotic arm and the maximum considered field of view of a landing camera (60°), the size of the terrain mock-up has been set to 2400×2000 mm, resulting from the assembly of 8 polyurethane foam sheets, each measuring 1200×500 mm. The total thickness of a sheet is 100 mm: 70 are available for milling, 30 are used to fix the mock-up to the support structure. The lunar DEM selected for the diorama has been chosen for its terrain features. Surface data come from NASA Lunar Reconnaissance Orbiter (LRO) mission[3]. Since the original resolution was too low for our purposes, it has been increased up to 0.25 m/px adding craters, boulders and fractal noise [8]. A 3D model of the physical milled surface to verify the required level of accuracy for ground-truth measures, is going to be obtained through dense matching photogrammetry or LASER scanning –both techniques are already available at PoliMi– with a resolution of at least 0.5 mm. A trade off is currently carried out to select the best technique. First tests of photogrammetry performed on a mock-up sample easily reached for the resolution requirement. The calibrated DEM is used as base onto which ground trajectories are reconstructed during the test phase. An example of lunar mock-up test is shown in Fig. 11, where two small samples are side by side to verify milling process accuracy ad the edges.

5.2. Robotic Arm

A Mitsubishi PA10-7C robotic arm is available at DAER. It features 7 DoF and it is capable to handle a 10 kg payload in a

[3]Courtesy of NASA and Arizona State University. URL: http://wms.lroc.asu.edu/lroc/rdr_product_select, last visit on: March 7, 2016.

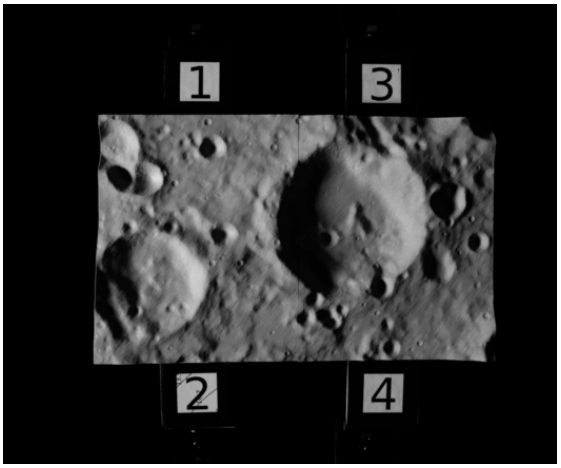


Fig. 11. Two small mockup polyurethane samples from a contiguous lunar DEM area side by side used for a preliminary test of Renshape® optical properties.

operative spatial range of 1.03 meters from its shoulder joint. On top of the end effector the robotic arm carries the sensor suite, that in case of the optical hazard detector in Section 2 is composed by a camera and a range sensor simulating the altimeter.

5.3. Illumination system

It must be able to guarantee the realistic light environment of the planetary surface during the simulation. In particular, for planets without atmosphere like the Moon, diffuse light must be avoided. It is composed by a CAME-TV LED array with narrow beam angle and temperature adjustable between 3200 K and 5600 K which can also be adjusted in position and light intensity; a *Dimming system* of non reflective black structure to prevent external Sunlight and internal reflections to jeopardize the simulation accuracy.

5.4. Inertial Measurement Unit

Since maneuver accelerations are scaled in the laboratory environment, whereas noise and disturbances are not, introducing an Inertial Measurement Unit in the sensor suite could not be representative of an actual landing sequence, with unrealistic signal to noise ratios in the software simulation. Trade-off studies are currently ongoing.

5.5. Test campaign

Preliminary tests are ongoing to trim open points in the design: illumination system consistency with the requested performances, taking into account also the optimal distance of the light source with the diorama; verification of needs of painting the mock-up milled surface. To evaluate the performances, shadows correspondence and the light intensity of shadowed and lit areas are under evaluation. As already mentioned, the milled surface will be calibrated to be able to compensate the simulated trajectory in the laboratory during the simulations. At setup completed, functional tests will be executed to verify the compliance with the requirements. Then, navigation and guidance algorithms are tested through a "bottom-up" procedure: first, single subsystems are checked for their standalone performance; then, subsystems are progressively joined together to check if relative couplings degrade the performances under the minimum requirements; eventually, closed loop simulations test and validate the overall facility in flight-like conditions. In this phase, performances are the same of a flight-like system: dispersion at touchdown, rate of successful landings, computational time requested.

6. CONCLUSION AND FUTURE DEVELOPMENTS

A suite of tools and algorithms for vision based navigation for autonomous landings in development at PoliMi-DAER have been presented. An hazard detector based on a single camera and artificial neural networks and an efficient semi-analytical adaptive guidance algorithm are in an advanced state, while the navigation is going to connect the two in near future. An experimental facility based on a robotic arm to simulate lander dynamics and a planetary mock up is being built for their validation and test.
The research team scope is to complete an entire Adaptive Guidance, Navigation and Control chain for autonomous landings with a single camera integrating the three subsystems aforementioned: the landing site selected on the hazard map is transformed in a 3D point in the physical world by the mapping process. The lander dynamics states of that point are then utilized by the guidance algorithm to compute the trajectory towards the landing site.

7. REFERENCES

[1] C.D. Epp and T.B. Smith, "Autonomous precision landing and hazard detection and avoidance technology (alhat)," in *Aerospace Conference, 2007 IEEE*. Institute of Electrical and Electronics Engineers, March 2007, pp. 1–7.

[2] "Chandrayaan Programme website," http://www.chandrayaan-i.com/index.php/chandrayaan-2.html, Last visit: 26th Feb. 2016.

[3] Ashutosh Saxena, Sung H. Chung, and Andrew Y. Ng, "Learning depth from single monocular images," in *Advances in Neural Information Processing Systems*, 2005, pp. 1161–1168.

[4] H. B. Kekre and S. M. Gharge, "Image Segmentation using Extended Edge Operator for Mammographic Im-

ages," *International Journal on Computer Science and Engineering*, vol. 2, no. 4, pp. 1086–1091, 2010.

[5] M. S. Nixon and A. S. Aguado, *Feature Extraction and Image Processing*, Newnes, 2002.

[6] Scott E. Fahlman and Christian Lebiere, "The cascade-correlation learning architecture," in *Advances in Neural Information Processing Systems 2*, D.S. Touretzky, Ed. 1990, pp. 524–532, Morgan-Kaufmann.

[7] U.J. Shankar, Wen-Jong S., T.B. Criss, and D. Adams, "Lunar terrain surface modeling for the alhat program," in *IEEE Aerospace Conference*, March 2008, pp. 1–10.

[8] Paolo Lunghi, Marco Ciarambino, and Michle Lavagna, "A multilayer perceptron hazard detector for vision-based autonomous planetary landing," Vail, CO, 2015, AAS/AIAA Astrodynamics Specialist Conference 2015.

[9] F. Hörz, R. Grieve, G. Heiken, P. Spudis, and A. Binder, *Lunar Surface Processes*, Cambridge University Press, 1991.

[10] G. Capuano, M. Severi, E. Della Sala, R. Ascolese, C. Facchinetti, and F. Longo, "Compact and high-performance equipment for vision-based navigation," in *63rd International Astronautical Congress (IAC)*, 2012.

[11] M. Dunstan and K. Hornbostel, "Image processing chip for relative navigation for lunar landing," in *9th International ESA Conference on Guidance, Navigation, and Control Systems (GNC 2014)*, 2014.

[12] Paolo Lunghi, Roberto Armellin, Pierluigi Di Lizia, and Michèle Lavagna, "Semi-analytical adaptive guidance computation based on differential algebra for autonomous planetary landing," Napa, CA, Aug. 2016, 26th AAS/AIAA Space Flight Mechanics Meeting.

[13] Behçet Açikmeşe and Scott R. Ploen, "Convex programming approach to powered descent guidance for Mars landing," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 5, pp. 1353–1366, 2007.

[14] Gregory Flandin, Bernard Polle, Noela Despré, Jacques Lheritier, Nicholas Perrimon, and Pierre Blanc-Paques, "Maturing vision based navigation solutions to space exploration," Toronto, Ontario Canada, Aug. 2010, AIAAGNC, AIAA Paper 2010-7601.

[15] Joseph Riedel, Andrew Vaughan, Robert A. Werner, Wang Tseng-Chan, Simon Nolet, David Myers, Nikolaos Mastrodemos, Allan Lee, Cristopher Grasso, Todd Ely, and David Bayard, "Optical navigation plan and strategy for the lunar lander Altair; OpNav for lunar and other crewed and robotic exploration applications," Toronto, Ontario Canada, Aug. 2010, AIAAGNC.

[16] Claudio Canuto, M. Yousuff Hussaini, Alfio Quarteroni, and Thomas A. Zang, *Spectral Methods in Fluid Dynamics*, Springer, New York, 1988.

[17] Martin Berz, *Differential Algebraic Techniques. Entry in Handbook of Accelerator Physics and Engineering*, World Scientific, New York, 1999.

[18] Jeff Delaune, Diego De Rosa, and Stephen Hobbs, "Guidance and control system design for lunar descent and landing," Toronto, Ontario Canada, 2010, AIAAGNC.

[19] Michael C. Johnson, "A parameterized approach to the design of lunar lander attitude controllers," Keystone, CO, 2006, AIAAGNC.

[20] Georg Klein and David Murray, "Parallel tracking and mapping for small AR workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*. IEEE, 2007, pp. 225–234.

[21] Christian Forster, Matia Pizzoli, and Davide Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 15–22.

[22] Raul Mur-Artal, JMM Montiel, and Juan D Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *Robotics, IEEE Transactions on*, vol. 31, no. 5, pp. 1147–1163, 2015.

[23] David Nistér, "An efficient solution to the five-point relative pose problem," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 6, pp. 756–770, 2004.

[24] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski, "ORB: an efficient alternative to SIFT or SURF," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2564–2571.

[25] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua, "Brief: Binary robust independent elementary features," *Computer Vision–ECCV 2010*, pp. 778–792, 2010.

[26] Richard Hartley and Andrew Zisserman, *Multiple view geometry in computer vision*, Cambridge university press, 2003.

[27] G. Bradski, ," *Dr. Dobb's Journal of Software Tools*.