Optimal and Risk-Aware Path Planning considering Localization Uncertainty for Space Exploration Rovers

J. Ricardo Sánchez-Ibáñez[1], Pedro J. Sanchez-Cuevas[2] and Miguel Olivares-Mendez[3]

***Abstract*— The reliability of autonomous traverses of rovers is critical. It may be jeopardized by the accumulation of errors and the uncertainty propagation of their localization systems. Moreover, space environments are usually harsh, challenging and unpredictable. Teleoperation is complex due to the significant and unavoidable delay. For these reasons, a path planner that provides some level of autonomy with guarantees could increase the success rate of planetary exploration missions. This paper proposes a path planning solution that tackles increasing localization uncertainty and makes a trade-off between the collision risk and the path length. The planner uses the the Fast Marching Method (FMM) to produce a costmap aware of this uncertainty and calculate the optimal path for a level of confidence. This paper additionally presents several simulation and experimental using a wheeled robotic vehicle within a lunar analogue facility.**

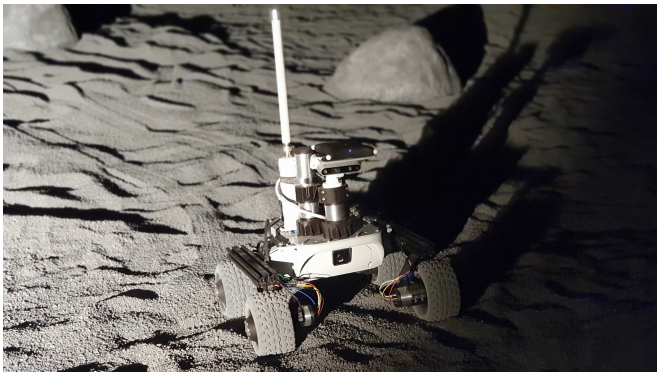


Fig. 1. Image of our LeoRover during the experiments in the LunaLab.

I. Introduction

Space exploration is a fundamental task to enable future missions in extraterrestrial environments, like building human settlements or exploiting space resources. In fact, both are part of the objectives and priorities of the leading space agencies (NASA, ESA, JAXA, ISRO, Australian Space Agency, LSA, etc.) and several companies (SpaceX, Honeybee robotics, Blue Origin, ispace, Space Applications Services...) along with the Artemis program [1]. As a part of the space exploration process, these agencies and companies use robotic vehicles, *rovers*, to accomplish *in-situ* planetary exploration [2]. Those must comply with the required capabilities of dealing with a great variety of situations and operating with a high grade of reliability and efficiency.

Planetary or lunar exploration usually start with a remote sensing operation using satellites to gather detailed environment information. Some samples are the Lunar Reconnaissance Orbiter (LRO) [3] or the Chandrayaan-1 [4] missions to collect images that help to map the surface of the Moon. This information is crucial to understand potential issues a rover could face while performing its mission, like slopes, obstacles, rocks and craters, among others. Actually, [5] applies super resolution algorithms to enhance the LRO images for future robotic space missions. The final goal of those studies is to provide as much information as possible to define future missions and create their preliminary plans. When the robot lands, its starting position is calculated using satellite observations or external information obtained by the lander. Then, the *in-situ* mission starts.

However, in most cases, it is challenging to assure the autonomous capabilities of those robots. Rovers navigate under harsh and varied conditions, which are difficult to predict. Moreover, their localization systems are mainly based in odometry methods [6], [7] such as the Visual Odometry (VO) method [8], Visual Inertial Odometry (VIO) [9], [10] or wheel odometry. These techniques accumulate a significant drift and uncertainty over time. This is due to the impossibility of using a Global Positioning System (GPS) or a Simultaneous Localization And Mapping (SLAM) technique that calculates the loop closure in real-time [11]–[13]. This produces an inaccurate propagation of the localization, which negatively impacts the reliability and the efficiency of the robotic system. Consequently, accomplishing a completely autonomous operation is significantly challenging. In most cases, it is usually very short. In the worst case, rovers can only be teleoperated which could be extremely complex due to the communication delay. Definitely, the accumulation of uncertainty arises as one of the biggest challenges of space rovers. This encourages the minimization of the operational risks originated from this uncertainty by developing reliable and robust techniques to assure the robot autonomy.

This problem has been previously studied from the controller point of view by several authors. Different solutions have been applied to ground robots [14], [15] and aerial vehicles [16], [17]. Other authors have addressed the problem at the planning level using algorithms such as A* [18] or RRT* [19]. In this paper, we propose a new path planning methodology based on the Fast Marching Method (FMM)

This work has been supported by the FiReSpARX project funded by the Luxembourg National Research Fund and the H2020 SESAME project funded by European Commission.

[1]Space Robotics Laboratory Dpt. of Systems Engineering and Automation, University of Malaga, Malaga, Spain. ricardosan@uma.es

[2]Advanced Centre for Aerospace Technologies (CATEC) Seville, Spain. psanchez@catec.aero

[3] Space Robotics (SpaceR) Research Group, Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg. miguel.olivaresmendez@uni.lu

[20]. This has been preliminary proved as a suitable technique for planetary rovers [21], and reconfigurable rovers [22]. It uses a grid but, unlike other algorithms like A*, used in [18], the path is not restricted to the grid segments and therefore it is shorter and smoother [23]. Besides, its computational complexity is known, while others such as RRT* [19] first find a feasible path and, thereafter, create more and more samples to converge to the optimal one [24]. The method results from the solution to the *eikonal* equation [25]. This models the propagation of a wave through the costmap. From this propagation FMM calculates the final path which is optimal, and continuous. The main drawback of this method is that it does not include zero or negative values in the cost to avoid local minima [26]. The cost only depends on scalar (isotropic) values and does not consider non-holonomic movement. The curvature of generated paths can be in some cases restricted to comply with robot turning restrictions [27].

The main novelty of our approach is the propagation of the uncertainty and its inclusion in the costmap. Thereafter, FMM is again used on this costmap to generate the path. A similar approach is done in [28] where the authors included a static uncertainty in the form of potential cost fields around obstacles. On the contrary, our method considers that the uncertainty increases as the robot moves. Our planner dilates the obstacles according to this uncertainty. The approach of [29] also dilates the obstacles according to the uncertainty the rover has whenever it stops. However, they still do not propagate this uncertainty over time in a single plan. Our algorithm has been validated through several simulations and an experimental campaign in our lunar analogue facility [30] (see Figure 1).

The rest of the paper is structured as follows: Section II presents the problem and describes the proposed planning method. Section III presents the validation setup, the system architecture, the robot and the testbed. Section IV aims to validate the method by assessing the planner performance in simulation and a real experimental campaign in our lunar analogue facility. Last, Section V closes the document with the conclusions and future works derived from this research.

II. Uncertainty Aware Planner

A. Problem statement

Navigating with localization uncertainty makes impossible to assure which is the specific position of the robot. Instead, it is only possible to know a bounded area where the robot could be with a specific level of confidence. This means that, even if the trajectory tracker performs accurately and the calculated path is assumed to be collision-free, the robot may still collide with an obstacle due to the localization uncertainty. The situation is illustrated in Figure 2. This evidences that it is necessary to include the uncertainty in the planning phase.

The planning method proposed in this paper aims to generate collision-free paths in a known environment considering localization uncertainty. The planner recognizes the uncertainty propagation and takes it into account during planning phase. This guarantees that the computed path is collision-free, at least, with a desired level of confidence.

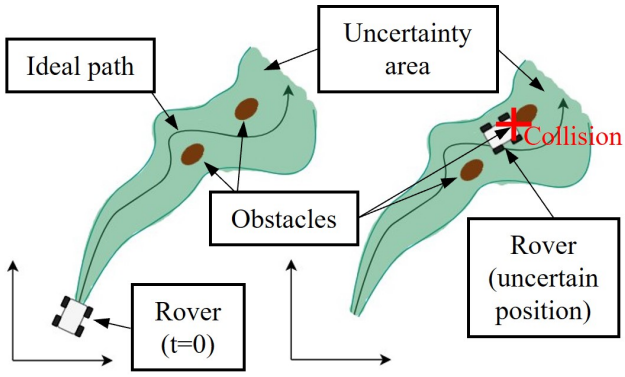


Fig. 2. Illustration of problem statement. Green background represents the uncertainty in the localization. The figure illustrates that not taking into account the uncertainty could led in a undesired collision because the actual position could be wrong.

B. Planner description

The planner addressed the scenario as a 2D costmap C. It differentiates between traversable and non-traversable areas. This means any generated path can trespass the former areas but not the latter ones. Any non-traversable area is composed of the area occupied by an obstacle, e.g. a rock the robot could collide with, and three dilations (see Figure 3). First, a geometric dilation surrounds the obstacle. This dilation (represented in Figure 3 as α) is defined by the diagonal size of the vehicle which is the minimum distance that prevents the robot from colliding with the obstacle. The second dilation (represented as ϵ) is defined by the maximum error expected by the trajectory controller. Lastly, the method proposes the novelty of including a dilation associated with the uncertainty. This dilation depends on the minimum distance the robot would drive to reach any point. This distance results from applying the FMM iteratively. To sum up, the first two dilations only depend on the distance to the obstacle, while the third one also depends on the starting position of the robot.

As stated before, FMM computes the minimum distance from the position of the robot, χ_{robot}, to any other point within the traversable area. In this case, it uses a regular square grid and assigns a value of distance d_{ij} to each node in the traversable area with index i, j. FMM uses the *eikonal* equation and an unit cost, as shown in (1), to perform this assignation. In this way, FMM calculates the optimal path between χ_{robot} and any other point by using the gradient descent method on the values of d_{ij}. The optimal path is defined as the one with minimum distance.

$$||\nabla d_{ij}|| = 1 \tag{1}$$

The method follows algorithm 1. It calculates the uncertainty dilation as a preliminary step and later generates the path going from χ_{robot} to a certain destination χ_{goal}. This

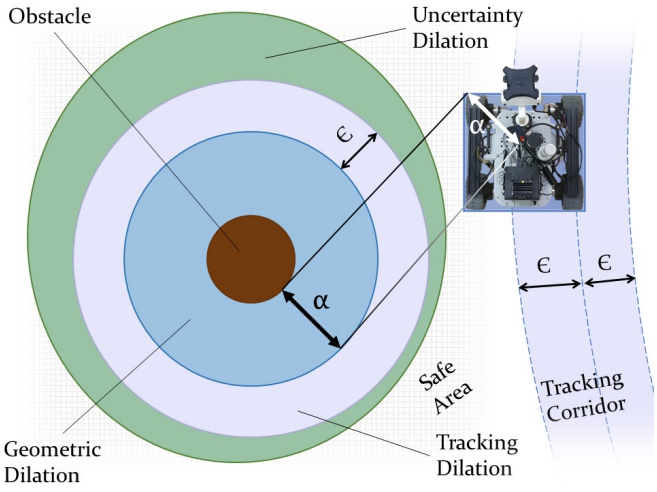


Fig. 3. Obstacle dilations considered by the planner. Geometric dilation is associated to the size of the robot. Tracking dilation represents the maximum error expected by the trajectory tracking. Uncertainty dilation is calculated through the propagation of the uncertainty for a confidence interval.

algorithm takes as inputs the costmap C and both positions χ_{robot} and χ_{goal}. At this point, the algorithm assumes the non-traversable areas of the costmap contain the geometric dilation and the tracking dilation (see Figure 3). Hence, the uncertainty dilation is yet to be calculated. From this costmap the algorithm calculates $\tilde{d}$, which is the proximity from any grid node to the nearest non-traversable area. This means $\tilde{d}_{ij} = 0$ for any node already in non-traversable area. This proximity is calculated by applying a distance transformation on the costmap [31]. Thereafter, a *While* loop produces this uncertainty dilation in an iterative way. In each iteration, the algorithm uses first FMM with the *computeDistance* function to calculate the values of distance d_{ij} from χ_{robot}. Thereafter, a copy of the costmap, $\tilde{C}$, is saved. With the computed values of the distances d_{ij}, *updateCostMap* updates C by adding new nodes to the uncertainty dilation if needed. When $\tilde{C} = C$, which means no nodes were added to this dilation in the last iteration, the loop stops. After this, the algorithm calculates the path Γ connecting χ_{robot} to χ_{goal} using *getPath* function.

Algorithm 1 Uncertainty Aware Path Planning

Input: C, $\chi_{robot}, \chi_{goal}$
Output: Γ
Initialisation :
$\tilde{d} \leftarrow$ distanceTransform(C)
$\tilde{C} = \varnothing$
while $\tilde{C} \neq C$ **do**
  $d \leftarrow computeDistance(C, \chi_{robot})$
  $\tilde{C} \leftarrow C$
  $C \leftarrow updateCostMap(d, \tilde{d})$
end while
$\Gamma \leftarrow getPath(L, \chi_{goal})$
return Γ

As mentioned, the *updateCostMap* function adds nodes to the localization uncertainty dilation if needed. To do this, it searches for traversable nodes that comply with condition (2). Here, f is a function that returns a value indicating the proportion of uncertainty σ_{ij} according to the confidence interval $\gamma \in [0, 100\%]$. For instance, in a normal distribution, the value of f, and hence the confidence interval, could be calculated using the standard normal table [32]. Any node complying with (2) is considered part of the uncertainty dilation and becomes non-traversable in the costmap. σ_{ij} comes as a function of the distance traversed d_{ij}. $\gamma \in [0, 100\%]$ represents the confidence interval. This variable is configurable by the user and it serves to tune the admissible risk of collision according to an uncertainty model.

$$\tilde{d}_{ij} \leq \epsilon + f(\gamma)\sigma_{ij}(d_{ij}) \tag{2}$$

III. Implementation

The effectiveness of the algorithm has been demonstrated in both simulation and experimental environments. The system architecture is depicted in Figure 4.

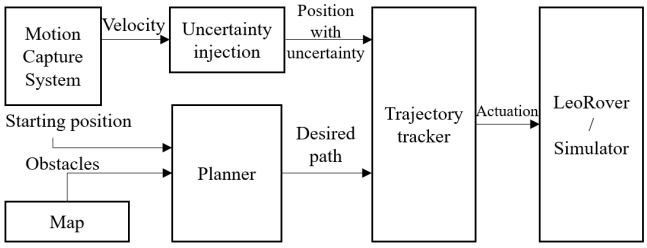


Fig. 4. System architecture. The uncertainty is injected in the system by adding noise to the velocity, measured by the MOtion CAPpture system (MOCAP). The trajectory tracker uses the resulting position with uncertainty to control the robot. The planner computes a collision-free path for a level of confidence using the starting position and a map provided by the MOCAP.

Figure 4 shows that the uncertainty is added to the velocity obtained by the motion capture system (MOCAP) or to the ground truth plugin of the simulator. The output of the uncertainty injection module is a new position that includes uncertainty. It has been calculated after integrating the measurements of a simulated velocity sensor according to the velocity constant first-order model (details in Section III-D). The planner uses the initial position and the map obtained with the ground truth, simulating that the mission starts in a known position that has been obtained with a satellite or a lander observation. This information is used by the path planner to calculate the collision-free path the robot should follow to reach the goal point. Once the desired path is calculated, the trajectory tracker is in charge of making the robot follow it. However, the unique feedback position available is the one with uncertainty. This simulates a situation in which the robot cannot be observed in real-time and, therefore, the uncertainty cannot be decreased.

The trajectory tracker is based on the *Conservative Pursuit* (C-Pursuit) algorithm [33]. It is in charge of controlling the movement of the robot with a constant velocity, v_m, in the

desired path. This is a variation of the well-known *Pure Pursuit* algorithm [34]. The main difference is that C-Pursuit prevents the robot from getting further than a configurable margin distance, ϵ, from the path. This ϵ creates the tracking corridor in Figure 3.

A. The rover

As a robotic vehicle, we have used the LeoRover[1]. This is a ROS-integrated open-source solution of a small planetary rover. It has 4-wheels and uses differential-drive to move. The size of the robot is 447 $\times$ 433 $\times$ 249 mm. Its empty weight is 6.5 kg and has a payload capacity of 5 kg. It has a Raspberry Pi 3B+ as the main computer and an STM32F4 as a real-time micro-controller. This hardware is running Ubuntu 16.04 and ROS (Robot Operating System) kinetic.

Fig. 5. LeoRover used during the experiments.

B. The LunaLab

The experimental validation has been carried out in the lunar analogue facility (LunaLab) of the University of Luxembourg [30]. This facility consists of a closed structure of 11 $\times$ 7 m filled with 20 tons of basalt to emulate the visual appearance of the Moon surface. It is equipped with 12 Optitrack cameras as MOtion CAPture system and 3 IP cameras to register the experiments.

Fig. 6. The LunaLab (Lunar Analogue facility of the University of Luxembourg) with some mock-up rocks and the LeoRover.

[1]https://www.leorover.tech/ (Accessed on 2021, August, 26)

C. The simulator

The experiments have been combined with ROS-based simulations to facilitate extensive algorithm testing and validate the proposed method deeply. The software of the LeoRover can be simulated using the tool provided by the manufacturer[2]. Moreover, this simulator uses Gazebo[3] to emulate the physics of the system.

D. Uncertainty injection

The planner and the tracker assume that the robot is going to move at a constant velocity, v_m. Then, a velocity constant first-order model has been used to simulate and inject the uncertainty in the system. This uncertainty is injected simulating a velocity sensor that can have a bias and a noise. To do so, the velocity is calculated by deriving the position provided by the ground truth. Then, the bias and the stochastic noise are added according to:

$$\widehat{v} = v + b + \nu \tag{3}$$

Where $\widehat{v}$ is the simulated velocity with uncertainty, v is the velocity obtained through the ground truth, b is the bias, and ν is the noise injected.

To calculate the uncertainty of one specific node of the grid, $\sigma_{ij}(d_{ij})$, it is necessary to define the velocity error, δv, as $\delta v = v - \widehat{v}$. Then, according to the first-order velocity model, the position with uncertainty and the position error δx verify that:

$$x_{k+1} = x_k + \Delta T \widehat{v_k} \tag{4}$$

$$\delta x_{k+1} = \delta x_k + \Delta T \delta v \tag{5}$$

Where ΔT is the step time of the discrete system, and $\widehat{v_k}$ is the velocity with uncertainty. x_k represents the position with the uncertainty that is feeding the trajectory tracker (see Figure 4), which is accumulating drift and uncertainty in each step time.

To simplify the implementation, it has been assumed that the sensor has no bias, $b = 0$, and it has a white Gaussian noise with variance $\delta v = \sigma_\nu^2$. However, the method described in this paper could be applied with different uncertainty models. With these assumptions, it can be concluded that the propagation of the error through the first-order model will be:

$$Var[\delta x_k] = k(\Delta T)^2 \sigma_\nu^2 \tag{6}$$

$$\sigma_{x_k} = \sqrt{t}\sqrt{\Delta T}\sigma_\nu \tag{7}$$

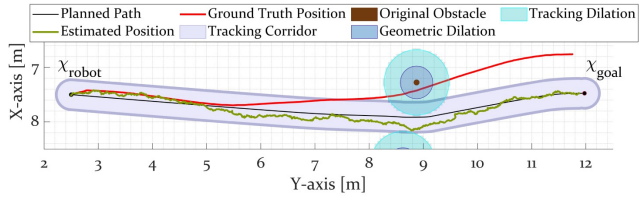
As the robot moves with a constant speed, v_m, then, $t_{ij} = d_{ij}/v_m$. The uncertainty of the node ij of the costmap can be expressed as follows:

$$\sigma_{ij} = \sqrt{d_{ij}/v_m}\sqrt{\Delta T}\sigma_\nu \tag{8}$$

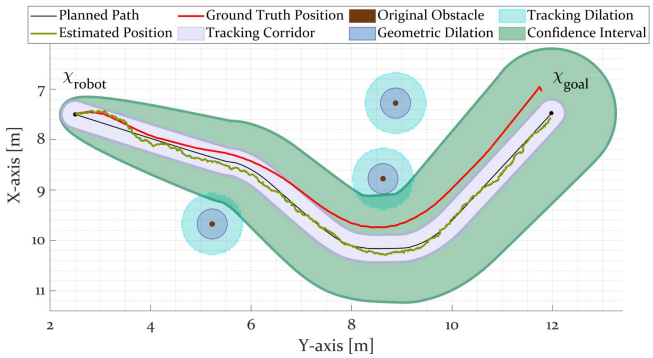
Where $d_{t_{ij}}$ represents the distance travelled by the robot to reach the node ij of the grid, according to the FMM.

[2]https://github.com/LeoRover (Accessed on 2021, August, 26)

[3]http://gazebosim.org/ (Accessed on 2021, August, 26)

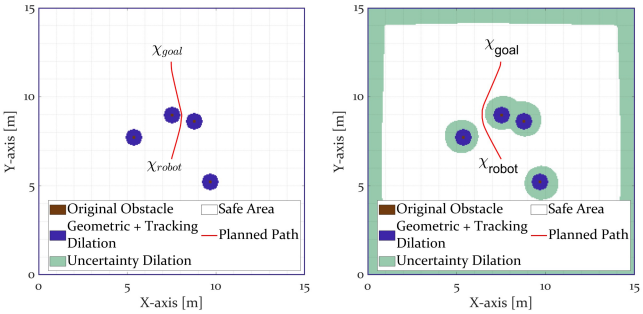


(a) Desired path and execution without considering uncertainty.

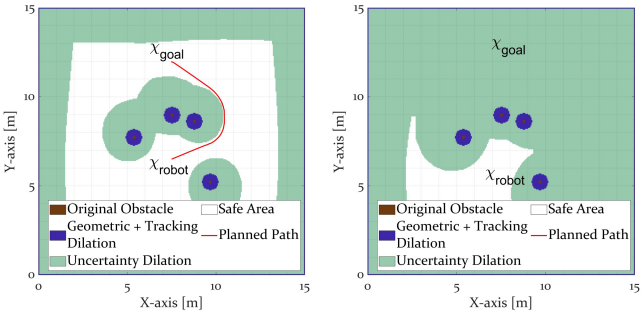


(b) Desired path and execution considering uncertainty.

Fig. 7. Planning results comparison. In a) the robot collides with the obstacles when the red line crosses the geometric dilation because the uncertainty has not been considered. b) shows how although the localization diverges, the uncertainty prevents the robot from colliding with the obstacle.



(a) Confidence Interval of 0%. (b) Confidence Interval of 68.27%.



(c) Confidence Interval of 95.45%. (d) Confidence Interval of 99.73%.

Fig. 8. Comparative of costmaps according to different values of σ. The resulting paths are coloured in red. As the level of confidence is increased, the path is safer but it is also longer, however, it is optimal for the confidence interval required. $\sigma_v = 0.3m/s$, $v_m = 0.1m/s$, $\Delta T = 0.1s$.

IV. Results

A. Method evaluation

Figure 7 illustrates the effect of considering or not the uncertainty in the planning phase. Figure 7(a) shows a simulated path execution without considering the uncertainty, and Figure 7(b) using the planning method presented in this paper. Although both results show that the estimated position is always inside the corridor of the trajectory tracking corridor, the ground truth sometimes violates this area due to the uncertainty. This situation, in fact, is leading to a clear collision situation in Figure 7(a) in which the desired path has been calculated without considering the uncertainty. In contrast, despite the drift is also present in Figure 7(b), thanks to the path has been calculated using the planning method presented in this paper for a confidence interval of 68.27% the robot does not collide.

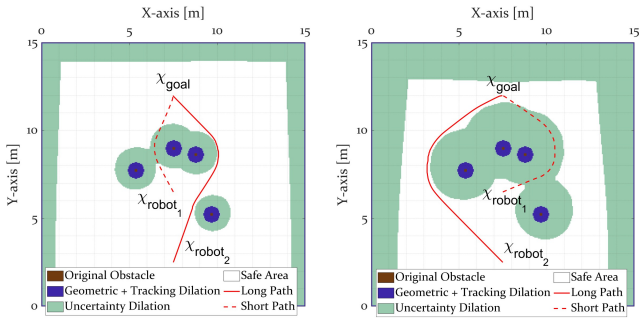
B. Influence of the uncertainty in the costmap

To clarify the effect of considering the uncertainty, Figure 8 depicts costmaps calculated using different values of confidence intervals. The scenario consists of four circular obstacles of $0.15m$ diameter arranged around the starting position and placed at a distance of $2.5m$ from it. The costmap was composed of 300 $\times$ 300 nodes with a resolution of $0.05m$, i.e. 15 $\times$ 15m. The four studied cases have been considering a confidence interval of 0%, 68.27% (σ), 95.45% (2σ) and 99.73% (3σ).

Figure 8(a) represents the path and the costmap without considering the uncertainty. The path generated by the planner is optimal, but according to the stochastic process, the confidence interval is null. This means that there are no guarantees to assure that the final path executed by the rover is not going to violate the geometric dilation which will led to collide with an obstacle. Figures 8(b) and 8(c) show that, as the confidence interval is increased, the generated path is safer and more conservative but also longer. However, FMM computes the optimal path for the considered uncertainty. Last, Figure 8(d) shows an extreme case in which the goal point cannot be reached with a required level of confidence. In this case, a confidence interval of 99.73% has been required in the planning phase.

C. Influence of the uncertainty in the starting position

This section focuses on evaluating the effect of changing the starting point in the desired path when the uncertainty is taken into account. Increasing the distance between the starting point and the obstacles produces an increase in the localization uncertainty when they approach the obstacles. In some cases, especially in those in which the robot is supposed to pass between two obstacles, the high value of the uncertainty does not assure that the robot is not going to collide with the obstacles for specific levels of confidence. Consecuently, the algorithm marks these areas as non-traversable in the costmap. Figures 9(a) and 9(b) illustrate this with two cases in which the desired path significantly changes when modifying the starting position of the robot. The new costmaps are also more conservatives. This is because the uncertainty in the localization close to the obstacle is higher as the starting position of the robot is farthest.



(a) Confidence Interval of 68.27%. (b) Confidence Interval of 95.45%.

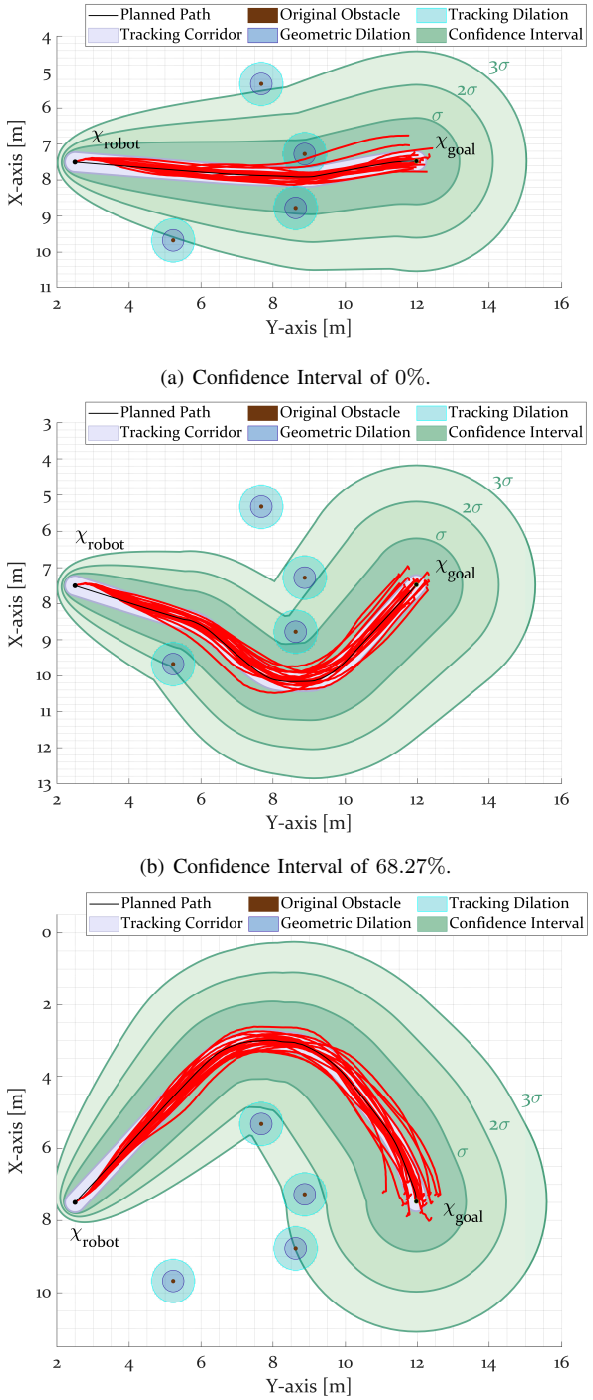
Fig. 9. Comparative of costmaps and maps according to different values of σ. The resulting paths are coloured in red. This figure shows that the path and the costmap change when the starting position varies. $\sigma_v = 0.3m/s$, $v_m = 0.1m/s$, $\Delta T = 0.1s$.

D. Path execution results

Figure 10 presents the results obtained after simulating the execution of some of the paths presented in Figures 8 and 9. Each path has been executed 40 times, 120 simulations in total. Figure 10 presents the results of 60 simulations, 20 per plot, to facilitate the visualization.

Those results show the benefits of including the uncertainty in the planner. For instance, Figure 10(a) shows that the rover violates the geometrical dilation of one of the obstacles due to the uncertainty. In this case, the uncertainty was not considered in the costmap. As it was explained in Section II-B and illustrated in 3, violating this dilation means that the robot has collided with the obstacle. The results show in Figures 10(b) and 10(c) that taking into account the uncertainty avoids violating this dilation. However, as a counterpart, it can be observed that the length of the paths significantly increase as more confidence is required. The path lengths obtained without considering the uncertainty were 5.55 m and 9.52 m from χ_{robot_1} and χ_{robot_2} respectively (see Figure 9). In contrast, the path lengths calculated with a confidence interval of $68.27\%(\sigma)$ were 5.92 m (6.67% longer) and 11.27 m (18.38% longer) and the ones considering $95.45\%(\sigma)$ were 8.69 m (56.58% longer) and 13.71 m (44.01% longer).

To quantify the risk, it is necessary to clarify that if the planner is not considering the uncertainty, the area out of the corridor should not be traversed by the robot. However, the uncertainty sometimes is producing a violation of this corridor. In fact, after analyzing the simulation results, it was calculated that the maximum violation of the corridor was in a 64.1% of the points executed by the path, the minimum was 0%, the average 14.35% and the standard deviation 19.07%. In contrast, the uncertainty areas associated to the level of confidence of $68.27\%(\sigma)$, $95.45\%(2\sigma)$ and $99.73\%(3\sigma)$ were not violated in any simulation. This evidences the advantages of considering the uncertainty in the planning phase, and consequently, the advantages of the method proposed.

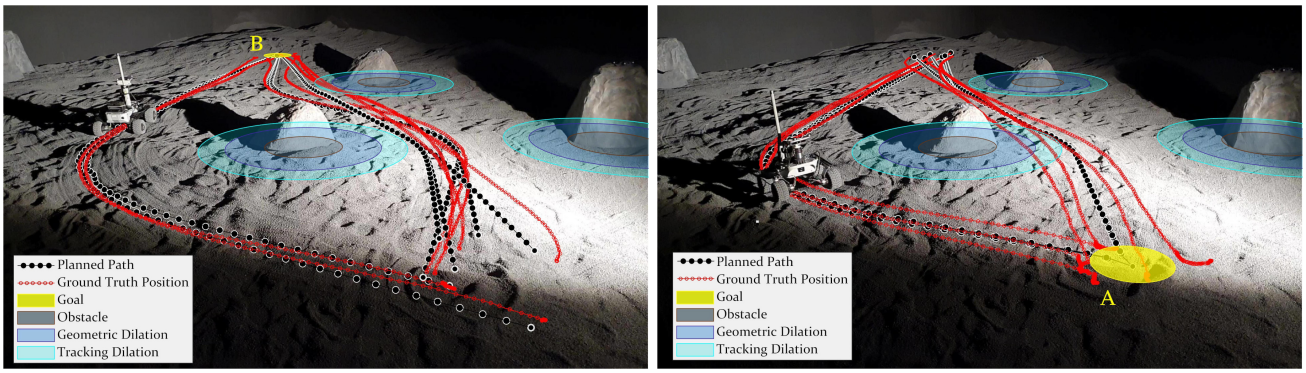


(a) Confidence Interval of 0%.

(b) Confidence Interval of 68.27%.

(c) Confidence Interval of 95.45%.

Fig. 10. Comparative of paths and simulated robot trajectories. $\sigma_v = 0.3m/s$, $v_m = 0.1m/s$, $\Delta T = 0.1s$.



(a) Traverses going to B position using three confidence intervals: 0% (Figure 12(a)), 68.27% (Figure 12(b)) and 95.45% (path at the left side).

(b) Traverses going to A position using two confidence intervals 0% (Figure 12(c)), 68.27% (Figure 12(d)).

Fig. 11. Experiment results in LunaLab. σ_v = 0.2 m/s, ϵ = 0.2 m.

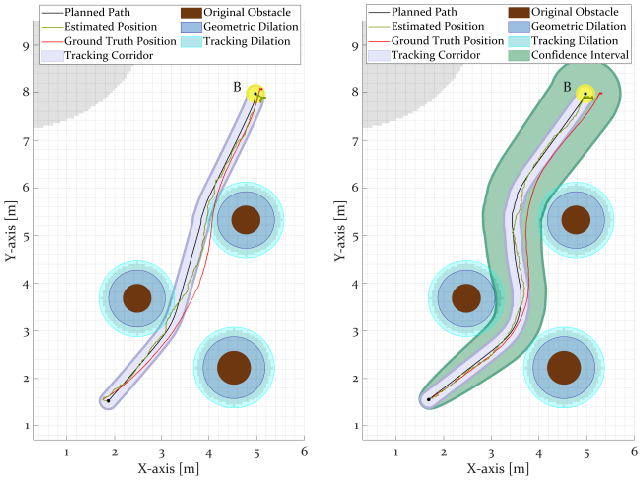
E. Experimental results

Finally, the planning method was validated in our LunaLab (see Section III-B). Obstacles and robot starting positions were measured using the MOCAP to generate an accurate costmap. It is important to remark that after this point, the MOCAP was only used to simulate the velocity sensor according to equation (3) and as a ground truth to compare the results and monitor the real position of the robot. In any case, the MOCAP has been used to provide a direct localization of the robot. The experiments consisted of driving between points depicted as A and B in Figure 11.

Figure 12 collects some of the experiments to illustrate and experimentally validate the effectiveness of the planner. In both cases, it can be observed that not considering the uncertainty provides risky paths which are supposed to be collision-free. However, in both Figures 12(a) and 12(c), the real position of the robot (red line) goes out of the corridor. This situation is considered unacceptable because the planned path has not considering that the robot will leave the corridor. In the hypothetical case of any other obstacle (or its geometric dilation) is located there, the robot would collide. In fact, a collision can be detected in Figure 12(c) when the red line cross the geometric dilation which is, in all the situations, a forbidden area. Figure 13 emphasizes this situation showing two images obtained during the experiments. In those experiments, the geometric dilation was defines longer than the maximum size of the robot to avoid putting in risk the rover.

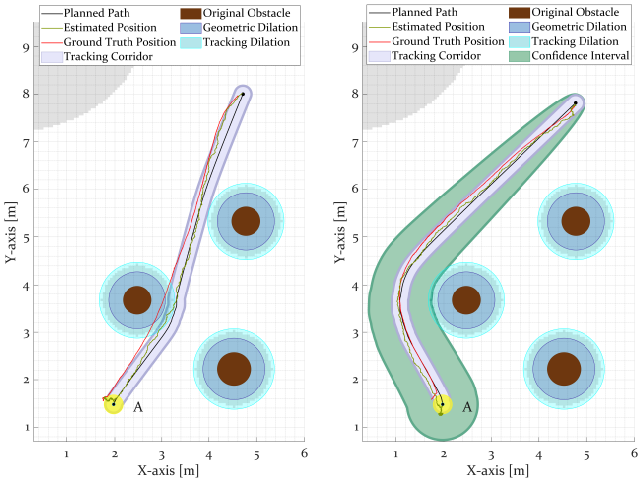
In contrast, Figures 12(b) and 12(d) show two results obtained considering the uncertainty. Although the fact of having some uncertainty does not allow us to guarantee that the path is collision-free, it is possible to assure that those are collision-free for level of confidence of 68.27%. The final paths obtained are safer and the results show that robot is always in the area that has been used by the planner, which, now, includes the uncertainty.

A video compilation of different experiments and simulations can be found in the link[4].



(a) Confidence interval 0% going to B.

(b) Confidence interval 68.27% going to B.

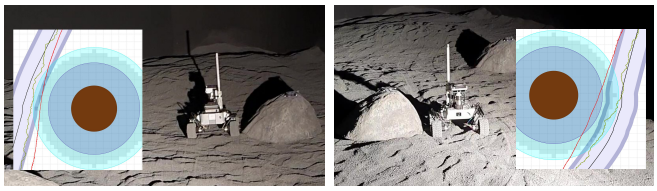


(c) Confidence interval 0% going to A.

(d) Confidence interval 68.27% going to A.

Fig. 12. Comparative of paths and executed robot trajectories σ_v = 0.2, ϵ = 0.2 m.

[4]https://dropit.uni.lu/invitations?share=921f8b4858e4b3f8e0d0



(a) Going to position B. (b) Going to position A.

Fig. 13. Two samples of distance violations during the experiments due to not considering the uncertainty.

V. Conclusions and Future Works

The results presented in this paper demonstrate the advantages of considering the localization uncertainty in the planner. In particular, this paper has introduced a strategy to use the Fast Marching Method, already used in prior path planning applications with rovers, along with a costmap based on this uncertainty. The simulation and the experiments have served to validate the proposed approach. This approach serves as the cornerstone for future implementations that may add more complexity to the uncertainty function. The main course of action is to land the presented approach to a more realistic setup, including uncertainty models from onboard sensors and refining the current path planning approach.

Future improvements to the path planning approach will consider cost due to turning and take into account how uncertainty increases with the variation of robot heading. Moreover, considering a model that acknowledges different kinds of uncertainty associated with different types of terrain seems promising.

References

[1] M. Smith, D. Craig, N. Herrmann, E. Mahoney, J. Krezel, N. McIntyre, and K. Goodliff, "The artemis program: An overview of nasa's activities to return humans to the moon," in *2020 IEEE Aerospace Conference*, pp. 1–10, 2020.

[2] Y. Gao and S. Chien, "Review on space robotics: Toward top-level science through space exploration," *Science Robotics*, 2017.

[3] G. Chin, S. Brylow, M. Foote, J. Garvin, J. Kasper, J. Keller, M. Litvak, I. Mitrofanov, D. Paige, K. Raney, *et al.*, "Lunar reconnaissance orbiter overview: Theáinstrument suite and mission," *Space Science Reviews*, vol. 129, no. 4, pp. 391–419, 2007.

[4] J. Goswami and M. Annadurai, "Chandrayaan-1: India's first planetary science mission to the moon," *Current science*, pp. 486–491, 2009.

[5] J. I. Delgado-Centeno, P. J. Sanchez-Cuevas, C. Martinez, and M. Olivares-Mendez, "Enhancing lunar reconnaissance orbiter images via multi-frame super resolution for future robotic space missions," *IEEE RA-L*, vol. 6, no. 4, pp. 7721–7727, 2021.

[6] J. Strader, K. Otsu, and A.-a. Agha-mohammadi, "Perception-aware autonomous mast motion planning for planetary exploration rovers," *Journal of Field Robotics*, vol. 37, no. 5, pp. 812–829, 2020.

[7] T. Sasaki, K. Otsu, R. Thakker, S. Haesaert, and A.-a. Agha-mohammadi, "Where to map? iterative rover-copter path planning for mars exploration," *IEEE RA-L*, 2020.

[8] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *CVPR 2004.*, vol. 1, pp. I–I, Ieee, 2004.

[9] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *IJRR*, 2015.

[10] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *IROS*, pp. 298–304, IEEE, 2015.

[11] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[12] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *JFR*, 2019.

[13] M. Azkarate, L. Gerdes, L. Joudrier, and C. J. Pérez-del Pulgar, "A gnc architecture for planetary rovers with autonomous navigation," in *ICRA*, pp. 3003–3009, IEEE, 2020.

[14] J. Jin and W. Chung, "Obstacle avoidance of two-wheel differential robots considering the uncertainty of robot motion on the basis of encoder odometry information," *Sensors*, vol. 19, no. 2, p. 289, 2019.

[15] A. Lazanas and J.-C. Latombe, "Motion planning with uncertainty: a landmark approach," *Art. intelligence*, 1995.

[16] M. Castillo-Lopez, P. Ludivig, S. A. Sajadi-Alamdari, J. L. Sanchez-Lopez, M. A. Olivares-Mendez, and H. Voos, "A real-time approach for chance-constrained motion planning with dynamic obstacles," *IEEE RA-L*, vol. 5, no. 2, pp. 3620–3625, 2020.

[17] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE TRO*, 2011.

[18] J. P. Gonzalez and A. Stentz, "Planning with uncertainty in position an optimal and efficient planner," in *2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 2435–2442, IEEE, 2005.

[19] M. Mizuno and T. Kubota, "A new path planning architecture to consider motion uncertainty in natural environment," in *ICRA*, 2020.

[20] J. A. Sethian, *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, vol. 3. Cambridge university press, 1999.

[21] S. Garrido, L. Moreno, F. Martín, and D. Alvarez, "Fast marching subjected to a vector field–path planning method for mars rovers," *Expert Systems with Applications*, vol. 78, pp. 334–346, 2017.

[22] C. J. Pérez-del Pulgar, J. Sánchez, A. Sánchez, M. Azkarate, and G. Visentin, "Path planning for reconfigurable rovers in planetary exploration," in *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 1453–1458, IEEE, 2017.

[23] C. H. Chiang, P. J. Chiang, J. C.-C. Fei, and J. S. Liu, "A comparative study of implementing fast marching method and a* search for mobile robot path planning in grid environment: Effect of map resolution," in *2007 IEEE Workshop on Advanced Robotics and Its Social Impacts*, pp. 1–6, IEEE, 2007.

[24] I. Noreen, A. Khan, Z. Habib, *et al.*, "Optimal path planning using rrt* based approaches: a survey and future directions," *Int. J. Adv. Comput. Sci. Appl*, vol. 7, no. 11, pp. 97–107, 2016.

[25] R. Kimmel and J. A. Sethian, "Optimal algorithm for shape from shading and path planning," *Journal of Mathematical Imaging and Vision*, vol. 14, no. 3, pp. 237–244, 2001.

[26] J. A. Sethian, "Fast marching methods," *SIAM review*, vol. 41, no. 2, pp. 199–235, 1999.

[27] Y. Liu and R. Bucknall, "The angle guidance path planning algorithms for unmanned surface vehicle formations by using the fast marching method," *Applied Ocean Research*, vol. 59, pp. 327–344, 2016.

[28] J. V. Gómez, A. Lumbier, S. Garrido, and L. Moreno, "Planning robot formations with fast marching square including uncertainty conditions," *Robotics and Aut. Systems*, 2013.

[29] M. Winter, S. Rubio, R. Lancaster, C. Barclay, N. Silva, B. Nye, and L. Bora, "Detailed description of the high-level autonomy functionalities developed for the exomars rover," in *14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, 2017.

[30] P. Ludivig, A. Calzada-Diaz, M. A. Olivares Mendez, H. Voos, and J. Lamamy, "Building a piece of the moon: Construction of two indoor lunar analogue environments," in *IAC*, 2020.

[31] G. Borgefors, "Distance transformations in digital images," *Comp. vision, graphics, and image processing*, 1986.

[32] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä, and L. E. Meester, *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer Science & Business Media, 2005.

[33] L. Gerdes, M. Azkarate, J. R. Sánchez-Ibáñez, L. Joudrier, and C. J. Perez-del Pulgar, "Efficient autonomous navigation for planetary rovers with limited resources," *JFR*, 2020.

[34] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," tech. rep., Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, 1992.