TRAJECTORY CONTROL FOR AUTONOMOUS PLANETARY ROVERS

Jan Filip[1], **Martin Azkarate**[2], **and Gianfranco Visentin**[2]

[1] *Czech Technical University in Prague, Faculty of Electrical Engineering, Czech Republic, E-mail:jan.filip2@gmail.com*

[2] *Automation and Robotics Section, European Space Agency, ESA, Noordwijk, The Netherlands*

ABSTRACT

In this paper, a modified pure-pursuit path following algorithm is presented. The proposed modification utilizes piecewise linear path representation and reduction of lookahead distance based on tracking error. A key safety requirement in application to planetary rovers is to keep the rover within a safety corridor of the path. The main shortcoming of the original algorithm, cutting corners during turns, is addressed through the proposed modifications.

Presented simulation results show a reduction of path tracking error of up to 15% compared to the original algorithm while respecting the safety corridor.

Key words: Path following, trajectory control, pure-pursuit, planetary rover locomotion control.

1. INTRODUCTION

Path following is a critical control system for automated operation of remote planetary missions. With large time-delays in extraterrestrial exploration scenarios, direct control of the rover is not feasible. Therefore, the rover must be capable of executing higher-level commands, such as reaching a given goal position, to carry out the science interest of the mission safely and efficiently.

A path is comprised of ordered points in the operational Cartesian space (called waypoints) which the robot should visit. The task of path following consists of steering the vehicle along the path to the goal position. A particularly important aspect of path following in planetary exploration is the safety of the rover. The path is usually provided by a higher-level layer of the navigation subsystem, either from an automated path planner or by a human operator, and can be assumed traversable up to certain lateral deviation from it. This gives rise to the notion of nominal path and its safety corridor. The nominal path is given as piecewise-linear interconnection of the waypoints, where each line connecting two consecutive waypoints defines a path segment. The safety corridor of each segment is bounded by two lines parallel to the path segment line in a fixed distance. Tracking error is defined as the lateral deviation of the rover position from the nearest segment line.

The purpose of this paper is to introduce a modified geometry-based path following algorithm. The algorithm builds upon the *pure-pursuit* follower of [Cou92], which is a popular path following solution, often considered as a reference. The pure-pursuit algorithm was used for trajectory control of cars by successful teams both in the DARPA Grand [UAB+05] and Urban Challenges [PH08, BIS09], as well as for planetary rover trajectory control [HRC+06]. A comprehensive summary of the algorithm's applications can be found in [MMM16].

The introduced *c-pursuit*, standing for conservative pursuit, modifies the existing algorithm to add a guarantee of path following bounded within a predefined safety corridor along the path, which is a critical safety feature for planetary exploration missions. The paper presents analysis, simulation results and experimental verification with rover prototypes in Mars-like terrain. Additionally, we present design guidelines for adjusting the controller for similar platforms with respect to their geometric constraints. The proposed controller is tuned using a single parameter with an intuitive geometric interpretation, allowing for a quick adoption to similar platforms.

2. METHODOLOGY

This section describes the used methodology, namely the representation of the path, the model of the rover, the concepts of geometry-based path following and the metrics used in the evaluation of path following results.

2.1. Path Representation

We selected to represent the path as a polyline, a series of line segments interconnecting the waypoints. This representation allows to define the path either by user-selected points or using grid-based planner nodes directly, without any additional processing needed. This format is

user-intuitive and highly efficient in terms of computations. The step of generating a smooth interpolating trajectory, which is commonly done in other path control algorithms, is not necessary. Instead, the path following algorithm utilizes the safety corridor along the path. The vehicle is allowed to deviate from the nominal path to execute turns, resulting in smooth transitions to the consecutive path segments while respecting the bounds of the safety corridor. The path following error at each control step is evaluated efficiently using the vehicle distance from the current segment line. However, the interpolation step might still be used if the waypoints are selected coarsely. Then, the path represents the piecewise-linear discretization of the smooth curve.

2.2. Kinematic Model

The proposed path following algorithm is aimed mainly at planetary exploration rovers. In this application, the nominal forward velocity is relatively low, on the order of units of cm/s. Therefore, the dynamics of the motion are negligible, and the problem can be represented using the kinematic equations only.

A typical locomotion architecture of planetary rovers considered is the triple-boogie, which commonly has six independently driven wheels. The front and rear wheels can be steered independently, enabling the rover to execute both Ackermann turns and turns on the spot. To execute a spot turn, the rover must stop first to align its wheels. While spot-turning capability is advantageous in many scenarios, such as narrow passages or dead ends, Ackermann turns are more efficient for general purpose path tracking as the rover can execute the path without stopping, adjusting the steering angles of wheels continuously.

The rover is represented using a body-fixed frame attached to its kinematic center, aligned with the axis of the middle pair of wheels. Since the rover is moving on a surface, only two positional coordinates are independent; hence the $\mathbf{x}_\mathrm{r} = [x, y]$ coordinates are used, determining the position of the rover projected onto the horizontal plane. The heading of the rover θ is given as the angle of rotation of the body-fixed frame with respect to the world reference frame. The selection of reference frames and the steering geometry is illustrated by Fig. 1.

The Ackermann-steering geometry of the rover can be simplified by assuming a 2D kinematic bicycle model with an equivalent turn radius r_turn. Simplification is done by representing each pair of wheels by a single wheel located in the middle, with the steering angle corresponding to the radius of turn which the rover center is executing. Moreover, the rear pair of the wheels is steered symmetrically as the front pair, and so it can be omitted in the kinematic model. This way, the steering geometry of the six wheeled triple-boogie is equivalently represented using the well-known kinematic bicycle model.

While executing a circular arc, each wheel is aligned tangentially to a concentric circle (with respect to the center of the turn) passing through its contact point. During path following, the steering angle is varied. For simplification, the dynamics of the steering mechanisms are not modeled, assuming that the response of the steering is fast compared to the variation of steering angle. Such assumption is valid for this application due to the low velocity of the motion execution.

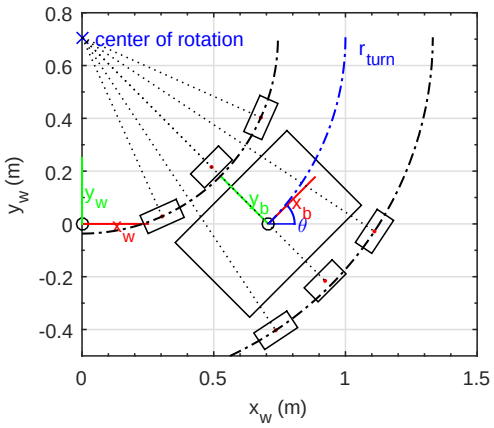


Figure 1: Reference frames and Ackermann steering geometry of the rover

2.3. Geometric-based Following

There are two main path following approaches for vehicles with nonholonomic constraints: the geometric-based and control theory based followers. The proposed algorithm for path following is based on the geometric approach and is a modification of the original *pure-pursuit* follower introduced in [Cou92]. This method uses geometric relationships between the vehicle and the path, resulting in control law for the path following problem.

A survey comparing various control theory and geometry based approaches is presented in [Sni09], from which the pure-pursuit algorithm comes as the most promising solution for slow-moving vehicles in terms of tracking performance and robustness.

Pure-pursuit type path followers are position-based and velocity-independent. The controller has no requirements on the path smoothness and is relatively robust to transient disturbances. The disadvantages arising from the neglected system dynamics become significant only at speeds several orders of magnitude higher than those considered in this application. Therefore, due to slow velocity of the rover (on the order of cm/s) and relatively fast response of the steering mechanism, the shortcomings can be neglected, making the pure-pursuit follower a viable solution.

The pure-pursuit algorithm solves the path following problem of an Ackermann-steered vehicle, producing the radius of the turn (or equivalently the turn curvature) as

the input into the locomotion control subsystem. The main steps of the algorithm are

1. Determine the position of the vehicle $\mathbf{x}_\mathrm{r}$ with respect to the nominal path.
2. Determine a lookahead point $\mathbf{x}_\mathrm{lh}$ ahead on the path.
3. Calculate the radius r_turn required to steer the vehicle from $\mathbf{x}_\mathrm{r}$ to $\mathbf{x}_\mathrm{lh}$.

This approach uses the notion of *lookahead point* and *lookahead distance*. The lookahead point is a virtual point located on the path ahead of the vehicle. The vehicle is instantaneously steered toward this virtual point. As the vehicle moves, the lookahead point also shifts forward, hence the name pure-pursuit. The lookahead point is uniquely determined by the path and the current position. A single scalar parameter is used to parametrize how far ahead is the lookahead point placed: the lookahead distance.

2.4. Performance evaluation

The immediate path tracking error is evaluated as the distance from the line of the current path segment. The performance of different algorithms or parameters setting was evaluated using two criteria. Firstly, it is required that the vehicle stays within the safety corridor, which has a binary outcome. Secondly, the root mean square tracking error is evaluated to account for both the mean and the variability of the error, penalizing larger deviations more. Lastly, the improvement of the proposed algorithm is evaluated as

$$I = 100\,\frac{e_\mathrm{RMS,A} - e_\mathrm{RMS,B}}{e_\mathrm{RMS,A}}\ (\%), \tag{2.1}$$

where e_RMS is the RMS of path tracking error of the original (A) resp. the proposed (B) algorithm.

3. PROPOSED ALGORITHM

This section describes the modifications to the original pure-pursuit algorithm described in [Cou92, Sni09].

3.1. Position along the path

To determine the vehicle position and progress along the path in each iteration, the current n-th path segment (consisting of waypoints $\mathbf{w}_{n-1}, \mathbf{w}_n$) is kept in memory, as well as the precalculated unit line vector $\mathbf{s}_n$ of n-th segment

$$d_n = \|\mathbf{w}_n - \mathbf{w}_{n-1}\|_2\,, \tag{3.1}$$

$$\mathbf{s}_n = \frac{\mathbf{w}_n - \mathbf{w}_{n-1}}{d_n}. \tag{3.2}$$

The position with respect to the path is determined as perpendicular projection $\mathbf{x}_\mathrm{i}$ of the current position $\mathbf{x}_\mathrm{r}$ into the path segment line

$$k_\mathrm{s} = (\mathbf{x}_\mathrm{r} - \mathbf{w}_{n-1}) \cdot (\mathbf{s}_n), \tag{3.3}$$

$$\mathbf{x}_\mathrm{i} = \mathbf{w}_{n-1} + k_\mathrm{s}\mathbf{s}_n. \tag{3.4}$$

To determine the progress along the segment and the transition into the subsequent one, distance k_s is compared with current segment length d_n. The distance from the current path segment line determines the tracking error ϵ

$$\epsilon = \|\mathbf{x}_\mathrm{r} - \mathbf{x}_\mathrm{i}\|_2 = |\mathbf{s}_n^\perp \cdot \mathbf{x}_\mathrm{r}|. \tag{3.5}$$

The polyline path representation allows for a simple and computationally efficient evaluation of the tracking error.

3.2. Adaptive lookahead distance

Given the position $\mathbf{x}_\mathrm{r}$, the location of the lookahead point $\mathbf{x}_\mathrm{lh}$ is determined uniquely and parametrized by the lookahead distance d_lh only. Our approach to lookahead distance adaptation was inspired by [Kel97], where the distance was *increased* by the lateral deviation ϵ from the path, in order to slow down the convergence and prevent overshooting. The aim was to use the pure-pursuit algorithm for vehicles with non-negligible motion dynamics. Lookahead distance increasing was applied in [G$^+$05] for a fast-moving all-wheel-drive military vehicle.

However, in the planetary robotics application, the rover velocity is low and fast convergence with the nominal path is the main path following objective. Therefore, lookahead distance *reduction* was investigated. In our approach, the lookahead distance modification is generalized to unify both schemes

$$d_\mathrm{lh,rd} = d_\mathrm{lh} - k_\epsilon\,\epsilon, \tag{3.6}$$

where d_lh is the initial setting of the lookahead distance and k_ϵ is the gain of the deviation penalization. In this paper, reduction ($k_\epsilon > 0$) of the lookahead distance is further investigated, in order to prevent cutting corners and to guarantee corridor-bounded path following.

With lower values of lookahead distance, the vehicle is steered more towards the current position on the nominal path rather than towards the path heading, speeding up the convergence with the path. However, this can also result in steering angle saturation and path intersection at a large heading error. While this approach may not be suitable for car-like vehicles with non-negligible motion dynamics, it is well suited for rovers capable of point-turns. In the rare cases of sharp turns where the commanded arc curvature exceeds significantly the steering limits, a point turn can be commanded to realign the rover with the heading towards the lookahead point and the path. This prevents overshooting, improves the convergence with the nominal path and helps satisfy the corridor-bounded tracking at the cost of stopping to align the wheels for spot turn. While this maneuver is useful in practice, it is not further developed in the theoretical analysis of the algorithm.

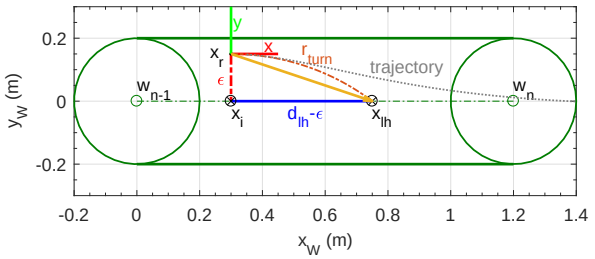


Figure 2: Lookahead point $\mathbf{x}_{\text{lh}}$ calculation at position $\mathbf{x}_{\text{r}}$ with $d_{\text{lh}} = 0.6\,\text{m}$, $k_\epsilon = 1$ and $\epsilon = 0.15\,\text{m}$

3.3. Lookahead point search

The original algorithm searched the lookahead point $\mathbf{x}_{\text{lh}}$ as a point satisfying

$$\|\mathbf{x}_{\text{lh}} - \mathbf{x}_{\text{r}}\|_2 = d_{\text{lh}} \tag{3.7}$$

and maximizing the progress along the path, by evaluating intersections of a circle with the path. Although this approach was applied previously in a planetary robotics application [HRC+06], it has a disadvantageous tendency to cut corners while executing sharper turns (angle between segment lines $\gamma > 45°$), which is a safety concern.

In the proposed adaptive lookahead approach, the lookahead point is found by moving distance $d_{\text{lh,rd}}$ of Eq. (3.6) along the path from the projected position $\mathbf{x}_{\text{i}}$. If the distance from $\mathbf{x}_{\text{i}}$ to the end of the segment is not sufficient, following segment(s) are used, summing the distance along the segment lines until $d_{\text{lh,rd}}$ is reached. The calculation of lookahead point is illustrated in Fig. 2 with $d_{\text{lh}} = 0.6\,\text{m}$ and $\epsilon = 0.15\,\text{m}$. Yellow arrow shows the vector of the desired heading towards the lookahead point. For the given heading, this translates to a circular arc r_{turn} drawn in brown color. As the vehicle would move and the lookahead point would shift along the path, this control scheme would result in the trajectory shown in gray, which gradually converges back towards the path while progressing along it.

3.4. Turn Radius Calculation

Using the current position and the lookahead point, turn radius is calculated. The radius is determined by interconnecting points $\mathbf{x}_{\text{r}}, \mathbf{x}_{\text{lh}}$ with a circular arc. The center of the circle is located on the vehicles y_{b}-axis so that the arc is tangential to the current vehicle heading θ. The calculation in the body frame is given as

$$\mathbf{x}_{\text{lh,b}} = \mathbf{R}_{z,\theta}^{\top}(\mathbf{x}_{\text{lh}} - \mathbf{x}_{\text{r}}), \tag{3.8}$$

$$r_{\text{turn}} = \frac{x_{\text{lh,b}}^2 + y_{\text{lh,b}}^2}{2y_{\text{lh,b}}}, \tag{3.9}$$

where $\mathbf{R}_{z,\theta}^{\top}$ denotes the rotation from world to body-fixed reference frame. In the original algorithm, the commanded arc curvature $\kappa = \frac{1}{r_{\text{turn}}}$ is proportional to the lateral error between the current vehicle heading vector and the lookahead point $y_{\text{lh,b}}$, since the numerator of Eq. (3.9) was the constant lookahead distance. Therefore, the pure-pursuit path follower has the properties of a P controller in terms of $y_{\text{lh,b}} \to \kappa$, as discussed in [Sni09], with the P gain being inversely proportional to the lookahead distance d_{lh}. In the proposed algorithm, the same applies, only the P gain is no longer constant, but adjusted with the tracking error ϵ.

4. ANALYSIS

The assignment of lookahead point to each position can be visualized as a vector field, where the local direction corresponds to the direction vector to the assigned lookahead point – recalling the yellow arrow of Fig. 2. The actual motion direction of the vehicle may be different as it is heading dependent. However, if the vehicle is aligned with the vector field, its instantaneous motion corresponds to the heading vector. Moreover, the vehicle is gradually aligned with the field as a result of the control. Therefore, the vector field represents the control strategy and can locally approximate the resulting trajectory.

Fig. 3a resp. 3b shows the behaviour of the original pure-pursuit algorithm for increasing angle of turn γ. While the approaches are initially similar, the difference occurs at the transition to the next segment. The original approach commands the vehicle to cut the corner, directing it outside of the safety corridor, which is not acceptable in our mission scenario. Decreasing the constant d_{lh} would reduce this problem, but at the cost of making the tracking response too sensitive to small heading changes, causing oscillatory tracking response.

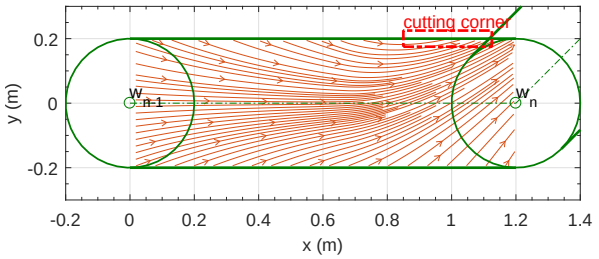
On the other hand, the proposed modification maintains the rover trajectory at all positions inside of the safety corridor, as shown in Fig. 3c. The vector field streamlines do not exit the safety corridor, and the vehicle is never commanded outside of it. The commanded direction towards the lookahead point is directed inwards the safety corridor up to

$$\gamma \leq 90° \qquad \text{and} \qquad d_{\text{lh}} \leq \frac{3}{2} d_{\text{corr}}, \tag{4.1}$$

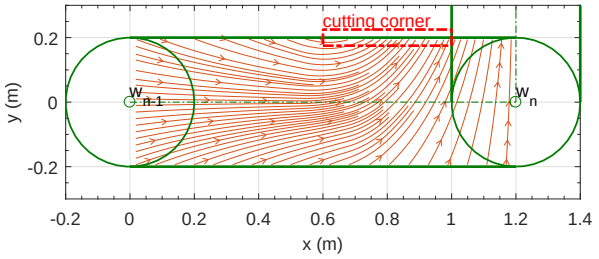
which is in the limit case shown using arrows in Fig. 3c, decomposing the distances in the same way as in Fig. 2. For this case, the direction to lookahead point is incident with the safety corridor edge. The proposed modification satisfies the requirement of corridor-bounded tracking while the original algorithm does not give such guarantee.

5. SIMULATION

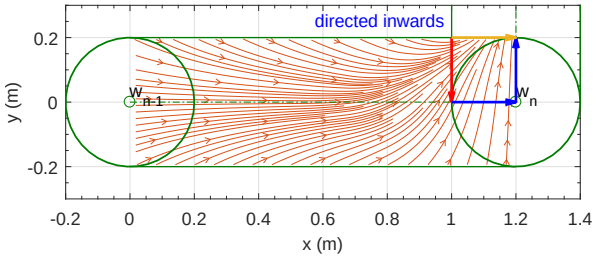
The path following algorithms were compared in simulation using kinematic bicycle model implemented in Mat-



(a) original pure-pursuit algorithm, $\gamma = 45°$ between segments



(b) original pure-pursuit algorithm, $\gamma = 90°$ between segments



(c) proposed modification, $\gamma = 90°$ between segments

Figure 3: Vector fields of direction vector to lookahead point $\mathbf{x}_{\text{lh}}$ from all possible $\mathbf{x}_{\text{r}}$ inside the safety corridor

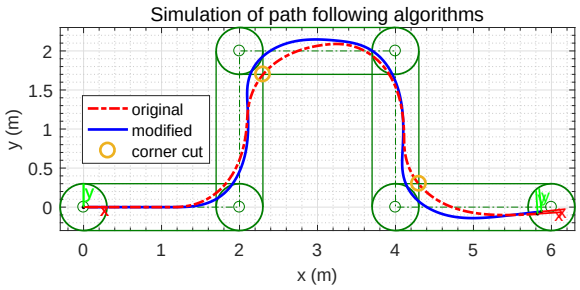
lab. As a benchmark, paths consisting of five segments with following angles γ were selected

$$\boldsymbol{\gamma} = \gamma\,[1 \quad -1 \quad -1 \quad 1]\,. \tag{5.1}$$

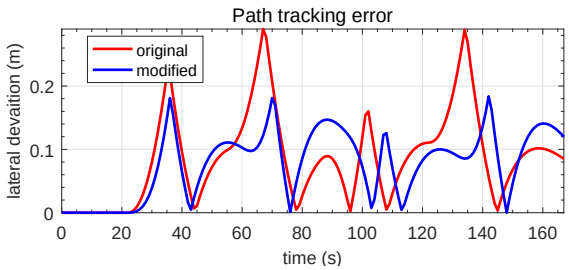
The simulation parameters are summarized in Tab. 1. An example of path with $\gamma = 90°$ is shown in Fig. 4a, emulating an obstacle avoiding maneuver. The proposed algorithm follows the nominal path with lower deviation and successfully maintains the safety corridor. On the other hand, a violation of corridor bounds occurs twice in the case of the original algorithm with the same d_{lh} lookahead distance setting. The path tracking error ϵ is depicted in Fig. 4b, indicating a lower mean error and lower peak deviation of the proposed algorithm. The two maxima of the original algorithm correspond to the cut corners. The same experiment was repeated for different values of γ and the results are summarized in Tab. 2. Overall, the performance difference becomes significant as the angle between segments increases above $\gamma > 45°$. Using the modified approach, corridor-bounded tracking is satisfied, path tracking error is reduced, and computational performance of the algorithm is improved. These results are in correspondence with the analysis of Fig. 3.

Table 1: Simulation parameters

Parameter		Set using	Value
Min. turn radius	r_{min}	given	0.6 m
Lookahead distance	d_{lh}	$\frac{3}{2}r_{\text{min}}$	0.9 m
Safety corridor width	d_{corr}	$\frac{2}{3}d_{\text{lh}}$	0.6 m
Deviation penalization	k_{d}	–	1
Path segment length		–	2 m



(a) simulated trajectories



(b) lateral error from the closest path segment

Figure 4: Comparison of path tracking algorithms, simulated on path with $90°$ between consecutive segments

Table 2: Simulation results

$\gamma(°)$	Mean ϵ (mm)		RMS ϵ (mm)		Improved
	orig.	mod.	orig.	mod.	(%)
30	29.30	27.74	37.96	35.47	6.55
45	43.79	40.36	56.63	50.72	10.43
60	58.23	52.39	75.23	64.50	14.26
90	86.19	79.06	111.39	93.87	15.73

6. TESTING

The proposed path following algorithm was integrated into the software of two rover platforms and tested in Mars-like terrain conditions. The implementation of proposed path following algorithm is available as open source in the repository [F+16].

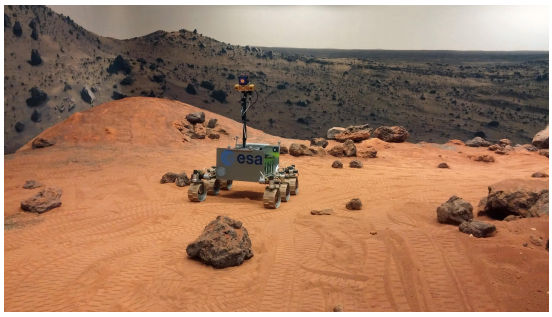


Figure 5: ExoMars Test Rover (ExoTeR) platform in the terrain of Planetary Robotics Laboratory, ESTEC

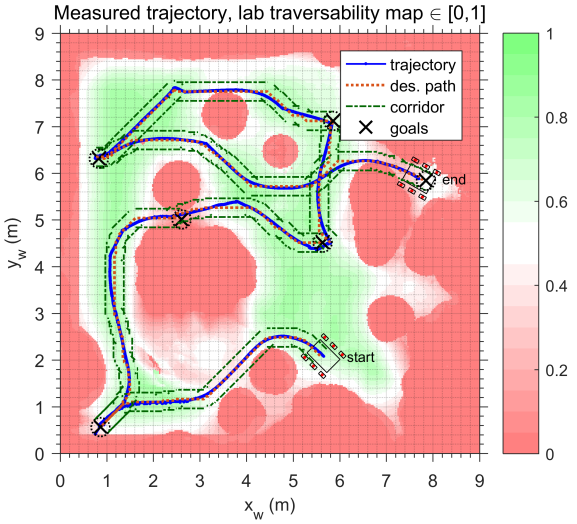


Figure 6: Measured trajectory of path following experiments with ExoTeR platform in PRL using planner-based path

6.1. Planner-based paths

The first experiment was conducted with the ExoTeR platform, which is shown in Fig. 5, in the terrain of the Planetary Robotics Laboratory in ESTEC. The path was generated onboard using AD* planner operating on the traversability map of the terrain, thus planning the path to avoid obstacles and excessive slopes. Vicon motion tracking system was used for rover localization during the experiment.

The results are depicted in Fig. 6, overlapping the path and the trajectory with the traversability map of the terrain from the top view. The modified algorithm managed to steer the rover along the nominal path in the terrain of moderate difficulty, maintaining the deviation from the nominal within the safety corridor $\pm 20\,\mathrm{cm}$. Moreover, the deviation from the nominal path was below $15\,\mathrm{cm}$ during the whole experiment, and the goal positions were reached successfully. This experiment demonstrates the performance of the proposed algorithm and integration with a grid-based planner outputting the path as a list of nodes. In the presented application, the path can be used directly with no need for further processing.

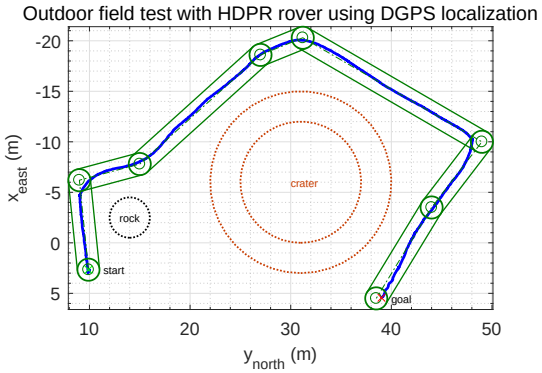


Figure 7: Measured trajectory of outdoor path following experiments with HDPR platform using user-defined GPS waypoints

6.2. User-defined waypoints

A path consisting of ordered waypoints interconnected by straight lines offers a user-intuitive interface and can be conveniently defined manually. Moreover, the notion of the safety corridor is easy to interpret, visualize and gives the operator a clear indicator of the safety of a candidate path.

Path following of user-defined GPS waypoints was demonstrated using HDPR platform [BH+16] in outdoor environment. HDPR platform is a full-sized rover. During the experiment, its velocity was set to $30\,\mathrm{cm\,s^{-1}}$ and the minimum turn radius restricted to $1\,\mathrm{m}$. The experiment was conducted at an outdoor test site with Mars-analogous terrain in the Netherlands, around the crater located at 52.215930N, 4.426796E, the same test site as depicted in [BH+16]. Test results are shown in Fig. 7. The rover was commanded to traverse the terrain around the crater using several user-defined GPS waypoints. Although the waypoints were coarsely spaced, forming a non-smooth path, the traverse was successful. The deviation from the nominal path was minimal with the only exception of turns. Turns were executed successfully, maintaining the safety corridor. Overall, the algorithm behavior was reliable and predictable, making it a highly useful tool in automating the field tests with rovers.

7. TUNING GUIDELINES

This section presents the tuning guidelines of the proposed algorithm, based both on simulation analysis and laboratory tests, to provide tips for adaptation of the algorithm on similar locomotion platforms. If Ackermann steering geometry is applicable, the minimum radius of turn is the performance constraint, and other controller parameters can be derived from its value.

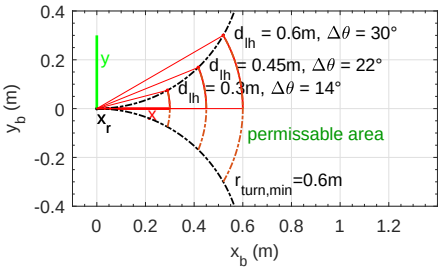


Figure 8: Minimum turn radius as the limitation on usable lookahead distance value

Fig. 8 shows the impact of the lookahead point Euclidean distance from the vehicle for minimum turn radius $r_{\text{turn,min}} \doteq 0.6\,\text{m}$. The sector (the right-hand side from the black circles) for which an executable arc exists shrinks rapidly with decreasing distance. In order to track paths varying in direction, the algorithm must allow a sufficient $\Delta\theta \approx \frac{y_{\text{lh,b}}}{x_{\text{lh,b}}}$ so that saturation of the commanded turn radius is mostly avoided. Therefore, a lookahead distance equivalent to the minimum turn radius is recommended as a good starting point for controller tuning. It allows a change of heading to the lookahead point up to 30° before reaching turn radius saturation. For lower values the response is too aggressive, causing oscillations.

Based on our simulations and experimental results, it is advisable to increase the lookahead distance slightly above the minimum setting, in the range of

$$d_{\text{lh}} \in \langle r_{\text{turn,min}},\ 1.5\, r_{\text{turn,min}} \rangle. \tag{7.1}$$

Increasing the lookahead distance slightly improves the margin before reaching the steering command saturation. Using a setting above the minimum is advisable since the effective lookahead distance is reduced using Eq. (3.6). Staying within the linear range of the steering command avoids overshooting and improves the path tracking behaviour.

On the other hand, the minimum safety corridor, which the algorithm can guarantee, scales with the selected lookahead distance. Based on the result of Eq. (4.1) and Fig. 3c, the width of the corridor should be set to

$$d_{\text{corridor}} = \tfrac{2}{3} d_{\text{lh}}, \tag{7.2}$$

which allows path tracking error up to $\pm\frac{1}{3} d_{\text{lh}}$. This setting guarantees the corridor-bounded tracking under no-slip assumption.

8. CONCLUSION

In this paper, a modification to the popular path following *pure-pursuit* algorithm was proposed. The modification utilizes piecewise-linear path representation for efficient lookahead point and tracking error calculations. A scheme for reducing the lookahead distance based on the current tracking error was developed. Using these two novel ideas, the executed trajectory is bounded within a guaranteed safety corridor along the path. The results of the modified algorithm were successfully demonstrated both in simulations and in laboratory experiments with two rover platforms in Mars-analogous terrain.

REFERENCES

[BH+16] E. Boukas, R. Hewitt, et al. Hdpr: A mobile testbed testbed for current and future rover technologies. In *13th International Symposium on Artificial Intelligence, Robotics, and Automation in Space (iSAIRAS)*, June 2016.

[BIS09] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. Springer, 2009.

[Cou92] R. Craig Coulter. Implementation of the Pure Pursuit Path Tracking Algorithm. Technical report, Carnegie Mellon University, Robotics Institute, 1992.

[F+16] J. Filip et al. Open-Source GitHub Repository: ExoTeR Rover, Waypoint Navigation. `https://github.com/exoter-rover/control-waypoint_navigation`, 2016.

[G+05] J. Giesbrecht et al. Implementation and adaptation of the pure pursuit algorithm. Technical report, Defence Research and Development Canada, 2005.

[HRC+06] D. M. Helmick, S. I. Roumeliotis, Y. Cheng, et al. Slip-compensated path following for planetary exploration rovers. *Advanced Robotics*, 20(11):1257–1280, 2006.

[Kel97] A. Kelly. *Intelligent Unmanned Ground Vehicles: Autonomous Navigation*, chapter RANGER: Feedforward Control Approach to Autonomous Navigation, page 115–121. Kluwer Academic Publishers, 1997.

[MMM16] S. Moveh, Hussein M., and M.B. Mohamad. A review of some pure-pursuit based path tracking techniques for control of autonomous vehicle. *International Journal of Computer Applications*, 2016.

[PH08] Papelis Y. Pillat R. Stein G. Patz, B. J. and D. Harper. A practical approach to robotic design for the darpa urban challenge. *Journal of Field Robotics*, 2008.

[Sni09] J. M. Snider. Automatic Steering Methods for Autonomous Automobile Path Tracking. Technical report, Carnegie Mellon University, Robotics Institute, 2009.

[UAB+05] Ch. Urmson, J. Anhalt, D Bartz, et al. A robust approach to high-speed navigation for unrehearsed desert terrain. *Journal of Field Robotics*, 2005.