A Laser Based Rut Detection and Following System for Autonomous Ground Vehicles

Camilo Ordonez∗
camilor@eng.fsu.edu
Oscar Y. Chuy Jr.∗
chuy@eng.fsu.edu
Emmanuel G. Collins Jr.∗
ecollins@eng.fsu.edu
Xiuwen Liu†
liux@cs.fsu.edu
Center for Intelligent Systems, Control and Robotics
*Department of Mechanical Engineering
Florida A&M-Florida State University
†Department of Computer Science
Florida State University
Tallahassee, Florida 32310

Abstract

An important off road driving rule is to keep the vehicle wheels in existing ruts when possible. Rut following has the following benefits: 1) it prevents the ruts from serving as obstacles that can lead to undesirable vehicle vibrations and even vehicle instability at high speeds; 2) it improves vehicle safety on turns by utilizing the extra lateral force provided by the ruts to reduce lateral slippage and guide the vehicle through its path; 3) it improves the vehicle energy efficiency by reducing the energy wasted on compacting the ground; 4) it increases vehicle traction when traversing soft terrains such as mud, sand or snow. This paper first presents a set of field experiments that illustrate the improved energy efficiency and traction obtained by rut following. Then, a laser based rut detection and following system is proposed so that autonomous ground vehicles can benefit from the application of this off road driving rule. The proposed system utilizes a path planning algorithm to aid in the rut detection process and an extended Kalman filter to recursively estimate the parameters of a local model of the rut being followed. Experimental evaluation on a Pioneer 3-AT robot shows that the proposed system is capable of detecting and following ruts in a variety of scenarios.
1 Introduction

Autonomous ground vehicles (AGVs) are increasingly being considered and used for challenging outdoor applications. These tasks include polar exploration, fire fighting, agricultural applications, search and rescue, as well as military missions. In these outdoor applications, ruts are usually formed in soft terrains like mud, sand, and snow as a result of habitual passage of wheeled vehicles over the same area. Fig. 1 shows a typical set of ruts formed by the traversal of manned vehicles on off-road trails.

![Figure 1: Typical off road ruts created by manned vehicles](image)

(a) Ruts along a low curvature trail, (b) Ruts at the bottom of a side slope.

Through experience expert off road drivers have realized that ruts can offer both great help and great danger to a vehicle (Baker, 2008; Blevins, 2007). When a vehicle is not performing rut following, the ruts essentially serve as obstacles which can cause undesirable vehicle vibrations and at high speeds can lead to vehicle instability, including vehicle tip over. However, when traversing soft and slippery terrains, rut following can improve vehicle safety on turns and slopes by utilizing the extra lateral force provided by the ruts to reduce lateral slippage and guide the vehicle through the desired path (Allen, 2002; Baker, 2008; Blevins, 2007; Thurman, 2004). On soft terrains ruts improve vehicle performance by reducing the energy wasted on compacting the ground as the vehicle traverses over the terrain (Muro and O’Brien, 2004; Ordonez et al., 2009a). In the same vein, the more compacted terrain associated with ruts can lead to improved vehicle traction. Hence, an AGV can use rut detection and following to improve its efficiency and safety on missions involving traversal of challenging outdoor terrains.

Besides the benefits of rut following already explained, proper rut detection and following can be applied in diverse applications. Rut detection can signal the presence of vehicles in the area, and also can help in the guidance of loose convoy operations. In forestry, rut detection and following can minimize the impact of heavy machinery on the ground. In agriculture, rut following can help automate seed planting and harvesting. A rut detection system can be used as a robot sinkage measurement system, which is key in the prediction of high-centering situations. Automatic rut detection
can also be employed to determine the coefficient of rolling resistance (Saarilahti and Anttila, 1999) (a vital parameter in robot dynamic models), and in general can be used to learn different properties of the terrain being traversed.

Initial research on rut detection (Laurent et al., 1997; Ping et al., 2000) focused on surface inspection for paved roads. In particular, (Laurent et al., 1997) presented a system to measure the depth of ruts in the pavement and (Ping et al., 2000) introduced a methodology to reduce the size of rut data sets in pavement management applications. Then, a rut detection method for mobile robots traversing on soft dirt was presented in (Ordonez and Collins, 2008). The approaches of (Laurent et al., 1997; Ordonez and Collins, 2008) were based on the detection of ruts using single laser scans without considering the spatial continuity of the ruts and did not address the rut following problem. An improved rut detection and following system that incorporated the spatial continuity of the ruts by modeling the ruts locally using second order polynomials was presented in (Ordonez et al., 2009a). However, the approach of (Ordonez et al., 2009a) did not make use of the spatio-temporal coherence that exists between the detections from consecutive laser scans while the vehicle is in motion. The research of (Ordonez et al., 2009b) incorporates the spatio-temporal coherence between measurements by using a rut detection and tracking module based on an extended Kalman filter (EKF) that recursively estimates the parameters of the ruts (tracking) and uses these estimates to improve the detection of the ruts for subsequent laser scans. In addition, the EKF also generates smooth state estimates of the relative position and orientation (i.e., the ego-state) of the vehicle with respect to the ruts, which are the inputs to the steering control system used to follow the ruts. However, the work presented in (Ordonez et al., 2009b) included only simulation results.

Work related to the recursive estimation of the rut model parameters involves the estimation of road center lines and road lanes. The vast majority of road tracking approaches rely on vision systems that detect the lane markings in the roads and then utilize filtering techniques such as Kalman or particle filtering to recursively estimate the parameters of the road center lines (Dickmanns and Mysliwetz, 1992; Khosla, 2002; Kim, 2006; Redmill et al., 2001; Zhou et al., 2006). Since vision based systems are easily affected by illumination changes and road appearance (e.g. changes from paved surfaces to dirt), other researchers have used laser range finders as the main sensor to detect and track road boundaries (Cremean and Murray, 2006; Kirchner and Heinrich, 1998; Wijesoma et al., 2004).

Additional research that is related to rut detection is the development of a seed row localization method using machine vision to assist in the guidance of a seed drill (Leemand and Destain, 2006). This system was tested in agricultural environments and was limited to straight seed rows. In (Reina et al., 2008) a vision based system is designed to detect the rut (i.e., the trace) left by a robot during its traverse on sandy terrain. The system utilizes the orientation of the detected rut to estimate the robot slip angle.

Furthermore, an evaluation of different models to predict rut formation is presented in (Saarilahti and Anttila, 1999).
The different models show that there is a correlation between rut depth and rolling resistance. However, this work did not present a methodology for actually detecting the ruts.

The main contributions of this paper are the inclusion of a new rut detection method based on a path planner, which is used to detect a set of ruts (from multiple candidates) that point in the general direction of the goal and to provide the initial state estimates required by the rut tracking and the steering control system. In addition, the paper presents an experimental evaluation of the rut tracking module and steering controller originally proposed and simulated in (Ordonez et al., 2009b).

The remainder of the paper is organized as follows. Section 2 contains a series of motivational experiments with two different robotic platforms and two different terrains. In addition, it outlines the proposed approach. Sections 3 and 4 respectively present detailed descriptions of the proposed approaches for rut detection and rut following. Section 5 introduces the experimental setup and shows experimental results. Finally, Section 6 presents concluding remarks, including a discussion of future research.

**Nomenclature**

- **A** process Jacobian
- **a** \(= (a_x, a_y)\) vector in the direction of initial rut segment
- **B** body fixed frame
- **B'** rut detection frame
- **b_c** body clearance, cm
- **c** \((n_1, n_2)\) cost between grid cells \(n_1\) and \(n_2\)
- **\(\zeta\)** minimum grid cost value
- **d** \((n_1, n_2)\) Euclidean distance between grid cells \(n_1\) and \(n_2\), cm
- **d** \((p_v, R_0^*)\) Euclidean distance between the robot’s current position \(p_v\) and the rut’s initial segment \(R_0^*\), cm
- **\(e_i^2\)** sum of the squared error, cm\(^2\)
- **\(e_{\text{min}}^2\)** minimum squared error, cm\(^2\)
- **\(e_v\)** velocity error, m/s
- **\(e_{xt}\)** normalized cross track error
- **\(f_r\)** rolling resistance, N
- **\(G\)** rut grid reference frame
- **\(g_w\)** grid width, cm
- **\(G\)** rut grid
- **\(G(n)\)** value of grid cell \(n\)
- **\(\mathcal{G}\)** filtered rut grid
- **\(H\)** measurement Jacobian
- **\(K\)** Kalman gain
- **\(k_1\)** steering control gain, s\(^{-1}\)
- **\(k_2\)** steering control gain, s\(^{-1}\)
- **\(l^*\)** path length of optimal rut, grid cells
- **\(\ell\)** laser lookahead distance, cm
\( N \) inertial frame
\( \mathbf{n} \) grid cell with coordinates \((i, j)\)
\( n_p \) number of support cells of rut parallel to the optimal rut
\( n^* \) number of support cells of optimal rut
\( \mathbf{P} \) error covariance
\( \mathbf{P}^{-k} \) a priori estimate error covariance
\( \mathbf{P}_k \) a posteriori estimate error covariance
\( \mathcal{P}^* \) path of minimum cost
\( P(\cdot) \) probability
\( p(\cdot) \) probability density function
\( p_c \) power consumption, W
\( \mathbf{p}_{r} \) a point with coordinates \((x_r, y_r)\) in the rut being followed expressed in the rut frame \( R \)
\( \mathbf{p}_{v} \) the robot’s position \((x_v, y_v)\) in the inertial frame \( N \)
\( Q \) process noise covariance
\( \mathbf{q} \) process state vector
\( \hat{\mathbf{q}}^{-k} \) a priori state estimate
\( \mathbf{q}_0 \) initial state vector
\( \mathbf{R} \) measurement noise covariance
\( R \) rut frame
\( r \) grid resolution, cm
\( r_c \) critical turn radius, cm
\( r_w \) distance between left and right ruts, cm
\( \mathcal{R}^* \) optimal rut
\( \overline{\mathcal{R}}^* \) piecewise approximation of the optimal rut
\( \mathcal{R}^*_{0} \) initial segment of the optimal rut
\( \mathcal{R}^*_{f} \) end segment of the optimal rut
\( S \) sensor frame
\( \mathbf{S}^{(m)}(k, \ell) \) structuring element of size \((2m + 1) \times (2m + 1)\) centered at \((k, \ell)\)
\( S \) rut detection output set
\( S_1 \) set of positive rut samples
\( S_2 \) set of negative rut samples
\( S_r \) search region for ruts
\( S_{\text{test}} \) rut detection testing set
\( S_{\text{train}} \) rut detection training set
\( s \) arc length, cm
\( \overline{T}_i \) average rut template of quadrant \( i \)
\( t_s \) simulated time of convergence, s
\( t_w \) tire width, cm
\( t_x \) experimental time of convergence, s
\( v \) measurement noise
\( v_c \) robot commanded speed, cm/s
\( v_l \) vehicle length, cm
\( v_v \) robot speed, cm/s
\( v_{v,\text{max}} \) maximum robot speed, cm/s
\( v_w \) vehicle track width, cm
\( w \) process noise
\( w_j \) pattern class
\( x_m \) intermediate variable Eqn. (17), cm
\( y_b \) lateral distance from vehicle axis \( x_B \) to the rut center, cm
\( y_d \) desired lateral offset, cm
\( y_f \) lateral offset between the vehicle and the rut, cm
\( y_{f,0} \) initial lateral offset between the vehicle and the rut, cm
\( y_{v,d} \) desired \( y \) value of the vehicle’s kinematic center in frame \( N \), cm
2 MOTIVATIONAL EXPERIMENTS AND PROPOSED APPROACH

This section begins with a series of experiments on two robotic platforms of different scales and operating on different terrains that provide quantitative verification of two of the benefits of rut following: increased energy efficiency and increased traction. The section concludes with a brief outline of the proposed approach for rut detection and following.

2.1 Experiments Illustrating the Importance of Rut Detection and Following

To show the increase in energy efficiency and traction obtained by rut following three controlled experiments were performed using the Pioneer 3-AT traversing sand and the eXperimental Unmanned Vehicle (XUV) moving in mud (see Fig. 2). In these experiments energy efficiency is measured by power consumption. Traction is assumed to be proportional to the absence of sliding and slipping and is measured by the velocity tracking error, which is small.
when no slipping or sliding occur. The main objective was to compare the two robot performance metrics (power consumption and velocity tracking) when the robot traverses a predetermined path where ruts are not present against subsequent traversals of the same path following the created ruts. It is important to note that these motivational experiments show the relevance of rut following for off road robot navigation, but did not use the proposed rut detection and following algorithm. Instead the robots achieved rut following by following a set of preassigned waypoints. Because the Pioneer 3-AT used only odometry for localization, the experiments involving this vehicle were performed on short and straight ruts (approximately 6m in length) and the vehicle was carefully placed and aligned at the starting point of the ruts. In contrast, the XUV employed more accurate localization based on differential GPS and a high cost IMU. Hence, the XUV experiments were performed with substantially longer ruts (over 40m in length).

First, a Pioneer 3-AT robotic platform was commanded to follow a set of ruts over sandy terrain at 0.8m/s. Six trials were performed; the first run was used as a baseline because it corresponds to the no-rut case (i.e., the robot is beginning the first creation of ruts). Fig. 3 shows a comparison of the power consumption for the first (no ruts) pass and the sixth pass. Notice that by following the ruts, there is an average reduction in power consumption of 18.3%. Furthermore, the experiments revealed that as early as the second pass, there is an average reduction in power consumption of 17.9%.

A second experiment was performed on mud with the XUV robotic platform. The robot was commanded to follow a
set of waypoints along a straight line at a speed of 2.23m/s. This experiment showed that by following the ruts, there was a reduction in power consumption of 12.6% for the second pass.

A third experiment was performed on mud with the XUV robotic platform. The robot was commanded to follow a set of waypoints along a curved path at 4.92m/s and three trials were performed. Fig. 4 shows the robot velocity profiles for the first and third run. Notice how on the first run, when there were no ruts, the vehicle was not capable of generating enough torque to track the commanded speed. This caused the motor to stall and the vehicle was not able to complete its mission. On the contrary, in the 3rd trial the robot was able to complete its mission by using the ruts created during the first two passes. The velocity tracking error reduced from 46.2% for the first run to 19.3% for the third run. It is also worth mentioning that the robot finished the mission successfully on the second pass and exhibited a velocity tracking error of 20%.

In the above experimental results it is clear that rut following improved the vehicle performance. This is important from a practical stand point because it means that a robot in the field can benefit from detecting and following ruts, even those that are freshly formed.
2.2 Outline of the Proposed Approach

Fig. 5 presents a schematic of the proposed approach. Notice that the system is divided into two primary subsystems:
1) a reactive control system to generate low level control commands needed to place the robot wheels in the ruts, and
2) a deliberative planning system that selects the best rut to follow among a set of possible candidates based on a predefined cost function. Note that the optimal rut determined by the deliberative system is the primary information needed to initialize the reactive system. Sections 3 and 4 detail the components and subcomponents of the proposed approach.

3 RUT DETECTION FOR THE REACTIVE AND DELIBERATIVE SYSTEMS

This section describes in detail the proposed approach for rut detection, which is performed at different levels of abstraction depending on whether it is being used by the reactive or by the deliberative system. As shown in Fig. 5, the reactive system takes as inputs laser readings at short range (< 1m) and is in charge of generating fine control commands to place the robot wheels in the ruts. Therefore, it requires an efficient rut detection method to determine the center of the rut being followed, which is achieved by performing rut detection using short range sensing in conjunction with a rut tracking module. On the other hand, the deliberative system takes as its input a map of the terrain surrounding the robot (for the Pioneer 3-AT experiments a 6m × 6m area), which is generated using midrange laser sensing; it also requires a more elaborate rut detection method to provide the system with the ability to determine the optimal rut to follow and decide whether or not this rut deserves to be followed. This section starts with a description of the low level rut detection module for the reactive system and then builds upon this result to develop a high level rut detection algorithm.

Figure 5: Schematic of the proposed approach to rut detection and following.
module for the deliberative system.

3.1 Low Level Rut Detection for the Reactive System

This section describes the low level rut detection system in detail. However, it is important to remember that, as shown in Fig. 5, this system works in parallel with the rut tracking module described later in Section 4.1.

In this research, it is assumed that the AGV is equipped with a laser range finder that observes the terrain in front of the vehicle and relies on the coordinate systems illustrated in Fig. 6: the inertial system $N$, the sensor frame $S$, the vehicle frame $B$, and the rut detection frame $B'$, which is coincident with the the vehicle kinematic center $B$ and has the $x_B'$ axis oriented with the robot and the $z_B'$ axis perpendicular to the terrain. For rut detection, the algorithm starts by transforming the laser scans from sensor coordinates to the $B'$ frame, which is a convenient transformation because it compensates for the vehicle roll and pitch. In addition, the laser scans are sampled at equally spaced points along the $y_B'$ axis.

Since ruts shapes are terrain and vehicle dependent, it is desired to develop experimental models of traversable ruts (ruts that do not violate body clearance and have a width similar to that of the vehicle tire) that can be used for rut detection. Notice that these models can be generated offline on the terrains of interest and/or they can be updated online by using a laser sensor to observe the ruts being created by the robot as it traverses the terrain. In this paper, we obtain the experimental models offline, using laser data from ruts created by a vehicle prior to the mission. The vehicle used to conduct experiments in this research is a Pioneer 3-AT robot with a body clearance $b_c$ of 8cm and a tire width $t_w$ of 10cm. In the following discussion, we assume that ruts with depths in the range $[0.4b_c, 0.8b_c]$ and widths in the range $[t_w, 1.5t_w]$ are traversable and form what is referred to here as the Acceptable Region, shown in Fig. 7.

In order to generate the experimental models of the ruts, a set $S_1$ of one hundred rut cross sections was manually selected with the vehicle having relative orientations with respect to the ruts. In particular, 20 rut cross sections were chosen for each of the following orientations: $-20^\circ$, $-10^\circ$, $0^\circ$, $10^\circ$, and $20^\circ$. Different orientations were used because

![Figure 6: Coordinate systems.](image)
the shape of the rut changes depending on the relative angle between the vehicle and the rut. Each rut sample in \( S_1 \) contains 31 points equally spaced in their horizontal coordinates with 1 cm resolution and is chosen such that the center of the rut corresponds with the center of the sample. Fig. 8 describes the depth and width of each of the rut samples in \( S_1 \).

One could use all of the rut samples in \( S_1 \) as the rut templates for rut detection. However, this number is large, which can lead to slow online implementation. Hence, this data set was used to construct a small set of rut templates for rut detection. The Acceptable Region for the Pioneer 3-AT was divided uniformly into 4 quadrants as shown in Fig. 8. Then a set of average templates \( \{ T_i : i = 1, 2, 3, 4 \} \) was constructed for the quadrants as shown in Fig. 9. However, it is expected that a larger vehicle may encounter a wider variety of rut shapes, and therefore may require a larger number of templates, which can be generated by using a template scaling procedure such as the one used in (Ordonez et al., 2009b).

In the rut detection process, the closeness between the 31 laser points in a window of 30 cm (which is the width of the templates) and each rut template \( T_i \) is determined by computing the sum of the squared error \( e_i^2 \). Then, \( e_{\min}^2 = \min\{e_1^2, e_2^2, e_3^2, e_4^2\} \) is used as the feature to estimate the posterior probabilities \( P(w_j|e_{\min}^2) \) for \( j = 1, 2 \), where \( w_1 \) corresponds to the class “No Rut” and \( w_2 \) corresponds to the class “Rut”. Bayes’ theorem yields

\[
P(w_j|e_{\min}^2) = \frac{p(e_{\min}^2|w_j)P(w_j)}{\sum_{j=1}^{2} p(e_{\min}^2|w_j)P(w_j)}, \quad j = 1, 2,
\]

where \( P(w_j) \) is the prior probability of each class and it is assumed that \( P(w_1) = P(w_2) = 0.5 \); the likelihoods \( p(e_{\min}^2|w_j) \) are estimated using the maximum likelihood approach (Duda et al., 2001) and a training set \( S_{train} \) that consists of the 100 positive rut samples of \( S_1 \) and a set \( S_2 \) of 100 negative samples obtained from terrain surrounding the ruts (i.e., \( S_{train} = S_1 \cup S_2 \)). A separate testing set \( S_{test} \) containing 100 ruts samples from the Acceptable Region.
and 100 negative samples was used to test the rut detection approach, yielding a detection rate of 87% and a false alarm rate of 9%.

To better illustrate the rut detection process, Fig. 10 shows the posterior probability $P(Rut|e_{min}^2)$ estimation for each point of a laser scan obtained from a set of ruts in front of the vehicle. The two probability peaks in Fig. 10(b) coincide with the location of the ruts.

### 3.2 High Level Rut Detection for the Deliberative System

The high level rut detection module builds upon the probabilistic method employed by the low level rut detection described in Section 3.1. However, the high level rut detection adds the following features to provide the system with the ability to engage the reactive control system in the case that a suitable rut is found: 1) gridding of the local map that contains the ruts, 2) determination of the rut with the minimum cost, 3) evaluation of the suitability of the optimal
rut for traversal, and 4) initialization of the reactive system for rut following. These features are detailed below.

3.2.1 Map Gridding for Rut Detection

A rut grid $G$ with a size of $602\text{cm} \times 602\text{cm}$ and a resolution $r$ of $2\text{cm}$ is constructed around the robot. As shown in Fig. 11, the rut grid is aligned with the inertial coordinate system $N$ and has its own reference frame $G$ attached at the top left corner. $G(n)$ represents the value of the grid cell $n$ with coordinates $(i, j)$.

In the current setup it is possible to restrict the rut grid to only the front portion of the robot’s workspace, which would reduce the computational load of the high level rut detection algorithm. However, here we choose to include the back area of the workspace because as mentioned in Section 6, future research involves the incorporation of replanning strategies that can benefit from rut information contained in this part of the workspace (e.g., the location of a section of the optimal rut).

The rut grid is used to define a traversability cost map and is constructed as follows. First, each laser scan is passed through the low level rut detection module, which returns a set $S = \{(x_j, y_j, P_j) : j = 1, ..., n\}$, where $n$ is the number of laser points in a scan, $x_j$ and $y_j$ correspond to the $x_N$ and $y_N$ coordinates of a laser point in inertial coordinates and $P_j = P(\text{Rut}|e_{\text{min}}^2)$ is the corresponding posterior probability computed using (1). Finally, the terrain points $(x_j, y_j)$ are mapped onto the rut grid and their corresponding probabilities $P_j$ are used to update the cost of the corresponding grid cell using Algorithm 1.

![Figure 10](image_url)

**Figure 10:** (a) Laser data containing two ruts; (b) The corresponding probability estimates of $P(\text{Rut}|e_{\text{min}}^2)$.
Algorithm 1 Rut Grid Update

**Input:** A grid $\mathcal{G}$ with resolution $r$ and width $g_w$, the robot position $p_r = (x_r, y_r)$ with respect to the inertial frame $N$, the set $S = \{(x_j, y_j, P_j) : j = 1, \ldots, n\}$ of laser points $(x_j, y_j)$ with corresponding posterior probabilities $P_j$, a probability threshold $\gamma_1$, and a minimum grid cost value $c^\ast$.

**Output:** Updated Rut Grid $\mathcal{G}$

```plaintext
for $j = 1$ to $n$ do
    if $P_j \geq \gamma_1$ then
        $i \leftarrow \text{int}(\frac{y_r + \frac{g_w}{2} - y_j}{r})$, where int$(x)$ approximates $x$ to the nearest integer.
        $j \leftarrow \text{int}(\frac{-x_r + \frac{g_w}{2} + x_j}{r})$
        $n \leftarrow (i, j)$
        $\mathcal{G}(n) \leftarrow c^\ast$
    end if
end for
return $\mathcal{G}$
```

Figure 11: Rut grid and coordinate systems.

Once the grid has been updated, a filter is employed to eliminate some outliers and join narrow breaks in the ruts. The filter consists of a set of morphological operations applied to the grid $\mathcal{G}$ employing $(2m + 1) \times (2m + 1)$ squared structuring elements $S^{(m)}$ with ones in all of their components (Gonzalez and Woods, 2002; Wilson, 1996). First a closing operation (•) using a $5 \times 5$ structuring element $S^{(2)}$ is employed to join narrow breaks that may exist in the ruts. Then an opening operation (◦) using a $3 \times 3$ structuring element $S^{(1)}$ is conducted on the resultant grid to eliminate small outliers. The filtering process can be summarized as follows:

$$
\overline{\mathcal{G}} = (\mathcal{G} \bullet S^{(2)}) \circ S^{(1)}
$$ (2)

### 3.2.2 Determination of the Minimum Cost Rut

Once a rut grid is available to the robot, the path of minimum cost $P^\ast$ from the start position (i.e., the current location of the robot) to a goal position (i.e., a waypoint or a final destination) is found using the path planning algorithm $A^\ast$ (Hart et al., 1968; Choset et al., 2005).
$A^*$ is an algorithm for efficiently finding cost-minimal paths from a start to a goal node in a graph (Koenig and Likhachev, 2006) (e.g., the graph induced by the rut grid). In order to perform an efficient search on the graph, $A^*$ utilizes heuristics that hypothesize the cost from a graph node to the goal node. In addition, the path returned by $A^*$ is optimal when the heuristic is optimistic (i.e., it always returns a value that is less or equal to the cost of the shortest path from the current node to the goal node (Choset et al., 2005)). The efficiency, optimality, and the ability to work on a grid make $A^*$ a good candidate for the task of finding the optimal rut from the current robot position to the desired goal.

In the current implementation, $A^*$ employs sixteen-point connectivity between grid cells and uses the Euclidean distance as a heuristic to estimate the cost from a given grid cell $n$ to the goal. In addition, the cost between the current grid cell $n_1$ and a neighboring cell $n_2$ is given by

$$c(n_1, n_2) = d(n_1, n_2) + d(n_1, n_2) \cdot (G(n_1) + \alpha),$$

where $d(n_1, n_2)$ is the Euclidean distance between the cells $n_1$ and $n_2$, $G(n_1)$ is the local cost of the rut grid $G$ at cell $n_1$, and $\alpha$ is a term used to penalize leaving a rut. The path returned by $A^*$ is optimal with respect to (3) and consists of a sequence of grid cells from start to goal. Therefore, the optimal path sequence generated by $A^*$ can be expressed as

$$P^* = \{n_0, n_1, ..., n_n\},$$

where $n_0$ is the cell corresponding to the start position and $n_n$ is the cell corresponding to the goal.

The effects of the penalty term $\alpha$ on the solutions returned by the planner are illustrated in Figs. 12 through 14. Fig. 12 shows that by using the penalty term $\alpha$ it is possible to eliminate or minimize situations in which the planner returns solutions that visit isolated outliers. Fig. 13 shows that the use of $\alpha$ causes the planner to prefer ruts with no gaps over broken ruts. Furthermore, Fig. 14 shows that $\alpha$ causes the planner to neglect very short ruts.

As shown in Fig. 13, the optimal rut sequence $R^* \subseteq P^*$. In particular, referring to (4), $R^*$ is defined such that it maintains all the cells $n_i \in P^*$ that satisfy $G(n_i) = c$. $R^*$ is then approximated by $\overline{R^*}$ in the least squares sense using a piecewise cubic spline approximation parameterized in terms of arc length.

### 3.2.3 Evaluation of the Suitability of the Optimal Rut

Once the optimal rut has been found and modeled, the algorithm proceeds to evaluate if the reactive rut following system should be engaged by evaluating the suitability of the optimal rut for inclusion in the vehicle path plan. The
optimal rut is considered suitable if the following three criteria are satisfied:

**Criterion 1.** The optimal rut has a significant length in comparison to the total path length from start to goal.

**Criterion 2.** There exists a rut that is parallel to the optimal rut and the distance between the two ruts is such that the vehicle is able to move with all wheels in the two ruts.

**Criterion 3.** The end of the optimal rut points in the general direction of the goal.
Figure 13: (a) Planner selects left rut regardless of gap ($\alpha = 0$), (b) Planner selects right rut to avoid gap of left rut ($\alpha = 20$)

Below, we develop quantifications of these three criteria.

At this point, it is assumed that $\mathcal{R}^*$, the approximation to the optimal rut expressed in inertial coordinates has been found. However, it is necessary to determine whether $\mathcal{R}^*$ corresponds to a right or to a left rut, a problem that the proposed algorithm solves by hypothesizing ideal locations (based on the vehicle width) of possible parallel ruts located to the right and left of $\mathcal{R}^*$. These ideal ruts are then mapped onto the rut grid and a search for “support cells” (i.e., grid cells with low cost that are in the proximity of the center of the ideal ruts) is conducted along each of the ruts.
Figure 14: (a) Path includes short rut ($\alpha = 0$), and (b) Planner avoids the short rut ($\alpha = 20$).

by using a $\left(2m + 1\right) \times \left(2m + 1\right)$ squared structuring element $S_{k, \ell}^{(m)}$, centered at $(k, \ell)$ and formally defined as

$$S_{k, \ell}^{(m)} \triangleq \{(i, j) : i \in \{k - m, k - m + 1, \ldots, k + m\}, j \in \{\ell - m, \ell - m + 1, \ldots, \ell + m\}\}. \quad (5)$$

By considering the points $(k, \ell)$ that are in the center of the ruts $S_{k, \ell}^{(m)}$ is used to identify support cells. A cell $n = (k, \ell)$ is formally defined to be a support cell if there exists $(i, j) \in S_{k, \ell}^{(m)}$ such that $G(n) = \mathbb{C}$. 
Fig. 15 illustrates two structuring elements centered at two cells located at the center of a potential rut. Notice that the lower cell location is marked as a support cell because there are grid cells with low cost (solid black) that are contained in the $5 \times 5$ structuring element while the upper cell is not a support cell. Furthermore, Fig. 16 presents the support cells found for the optimal and ideal right and left ruts. Notice that only the left rut contains enough support cells to be considered parallel to the optimal rut. Therefore, at this point it is possible to label the optimal rut as a right rut.

Now, let $n^*$ be the number of support cells of the optimal rut, $n_p$ the number of support cells of the rut parallel to the optimal rut, $l^*$ the path length of $R^*$ expressed in number of grid cells, and $\gamma_2$ and $\gamma_3$ threshold values. Then Criterion 1 and Criterion 2 listed at the beginning of this section can be quantified as follows:

**Criterion 1.**  
$$100 \frac{n^*}{l^*} \geq \gamma_2$$  
(6)

**Criterion 2.**  
$$100 \frac{n_p}{n^*} \geq \gamma_3$$  
(7)

Finally, Criterion 3 states that the optimal rut should drive the vehicle in the general direction of the goal. To evaluate Criterion 3, it is necessary to evaluate if $R^*$ points in the general direction of the goal. As shown in Fig. 17, the rut orientation is here estimated by finding the rut’s “end segment” ($R^*_f$), which is a linear approximation of the portion of the optimal rut that is closest to the goal and covers a path length equivalent to one lookahead distance. (As explained in Section 4.2.2, in the current implementation, the laser plane intersects the ground plane at a lookahead distance...
of 42.82cm.) The rut end segment is translated to the center of the ruts and it is then used to construct a rectangular region named the Goal Region (see Fig. 17), which delimits an area of possible goals that will trigger the reactive rut following system (i.e., if the goal is inside the Goal Region, Criterion 3 is satisfied). The dimensions of the Goal Region are chosen proportional to the vehicle dimensions as $\beta v_w \times \lambda v_l$, where $v_w$ and $v_l$ are respectively the vehicle’s track width and the vehicle’s length.

As described above, in the current implementation, the deliberative systems looks for ruts that have a separation similar to the vehicle’s track width. In addition, the current approach aims to place the right side wheels on the right rut and the left side wheels on the left rut. However, in many situations where prior knowledge of the upcoming terrain is available (e.g., a left or a right hand turn), off road drivers purposely place only one set of wheels in the ruts such that they can still benefit from the extra lateral support provided by the ruts and at same time minimize the amount of turning on the rutted area when the vehicle needs to leave the ruts. This situation is clarified in Fig. 18, where a vehicle with prior knowledge of a left hand turn coming ahead in the road, purposely chooses to place the right side wheels on the left rut. It is important to note that by using a single rut instead of a set of ruts, it is possible to eliminate the constraint of using only ruts created by vehicles of similar track width. In addition, this minimizes some of the downsides of rut following (e.g., difficulty in altering a path to avoid obstacles or change lanes).

### 3.2.4 Initialization of Reactive System for Following Suitable Ruts

If the solution returned by the path planner meets the three criteria of Section 3.2.3, the algorithm proceeds to initialize and engage the reactive system to follow the detected ruts. As explained in Section 4, the reactive system requires initial values for three state variables: the relative angle between the vehicle and the rut ($\theta_{vr,0}$), the rut curvature $\kappa_0$, and the relative offset between the vehicle and the rut ($y_{f,0}$).

To find the initial state $q_0 = [\theta_{vr,0}, \kappa_0, y_{f,0}]$, a similar approach as the one used to find the rut’s end segment $R_f^*$ is...
employed. However, in this case, as shown in Fig. 19, the initial rut segment ($R_0^*$) is used and not the rut end segment. Once ($R_0^*$) has been obtained, $q_0$ is computed using

$$
\begin{bmatrix}
\theta_{vr,0} \\
\kappa_0 \\
y_{f,0}
\end{bmatrix} = \begin{bmatrix}
\theta_{v,0} - \tan(a_y/a_x) \\
0 \\
d(p_v, R_0^*)
\end{bmatrix},
$$

(8)

where $\theta_{v,0}$ is the heading of the vehicle at the current position $p_v = (x_v, y_v)$, $A = (a_x, a_y)$ is a vector in the direction of $R_0^*$, and $d(p_v, R_0^*)$ is the Euclidean distance between the robot’s current position $p_v$ and $R_0^*$. Fig. 19 illustrates the initial position and orientation of the vehicle axis $(x_B, y_B)$ with respect to the rut axis $(x_R, y_R)$.
4 RUT FOLLOWING FOR THE REACTIVE SYSTEM

This section presents the rut following approach, which is composed of a rut tracking module that keeps state estimates of the relative position and orientation of the vehicle with respect to the rut and a steering control law that generates control commands for the robot to place the wheels in the ruts.

4.1 Rut Tracking Module

In this paper we propose an improvement to the previous approach to rut tracking presented in (Ordonez et al., 2009a) by incorporating an EKF that recursively estimates the parameters of the ruts (curvature) and also generates smooth estimates of the vehicle state with respect to the rut (orientation and lateral offset), which is advantageous compared to the approach of (Ordonez et al., 2009a) because these states can be used directly for the steering control as shown in Section 4.2. It is important to note that one of the strengths of the reactive system is that it employs only these relative (vehicle-rut) state estimates from the EKF and not the vehicle’s global state (with the exception of the vehicle’s initial state), which can be difficult to accurately estimate.

4.1.1 Local Modeling of the Relative Position and Orientation of the Rut and Vehicle

Motivated by the work of (Cremean and Murray, 2006), which models the road center line using heading and curvature, we model the rut locally as a curve of curvature $\kappa$ using frame $R$, a frame that moves with the vehicle and illustrated in Fig. 20. In order to fully describe the rut relatively to the vehicle it is therefore necessary to develop expressions for the rut curvature ($\kappa$) and the relative position ($y_f$) and orientation ($\theta_{vr}$) of the vehicle with respect to the rut.

As shown in Fig. 20, the $x_R$ axis is always tangent to the rut and the $y_R$ axis passes at each instant through the

![Figure 20: Rut frame coordinates used for rut local modeling.](image)
kinematic center of the vehicle $B$. In frame $R$, the position of a point $p_r$ in the rut as a function of arc-length ($s$) is given by

$$x_r(s) = \int_0^s \cos(\theta(\tau)) \, d\tau,$$

$$y_r(s) = \int_0^s \sin(\theta(\tau)) \, d\tau,$$

$$\theta(s) = \kappa s,$$

where $\theta$ is the orientation relative to the $x_R$ axis of the tangent vector to the curve at point $p_r$. Let us also define $\theta_r$ as the orientation of the $x_R$ axis with respect to the $x_N$ axis, $\theta_v$ as the orientation of the robot’s $x_B$ axis with respect to the $x_N$ axis, and $\theta_{vr}$ as the angle between the $x_B$ and $x_R$ axis. As the vehicle moves with linear velocity $v_v$ and angular velocity $\omega_v = \frac{d\theta_v}{dt}$, the evolution of $\theta_r$ can be derived from (11) and is given by

$$\dot{\theta}_r = \dot{\theta} = \kappa \dot{s} = \kappa v_v \cos(\theta_{vr})$$

In a similar way, the evolution of $\theta_{vr}$ and $y_f$ are computed using

$$\dot{\theta}_{vr} = \omega_v - v_v \sin(\theta_{vr}) \kappa,$$

$$\dot{y}_f = v_v \sin(\theta_{vr}).$$

Assuming that the evolution of the curvature is driven by white and Gaussian noise, after discretizing (13) and (14), using the backward Euler rule and sampling time $\delta_t$, the process model can be expressed as

$$\begin{bmatrix}
\theta_{vr,k} \\
\kappa_k \\
y_{f,k}
\end{bmatrix}
= \begin{bmatrix}
\theta_{vr,k-1} - \kappa_{k-1} v_v \cos(\theta_{vr,k-1}) \delta_t \\
\kappa_{k-1} \\
y_{f,k-1} + v_v \sin(\theta_{vr,k-1}) \delta_t
\end{bmatrix}
+ \begin{bmatrix}
1 \\
0 \\
0
\end{bmatrix} \delta \theta_{v,k-1} + w_{k-1},$$

(15)
where $\delta \theta_{v,k-1}$ is the model input (the commanded change in vehicle heading) and $w$ represents the process noise which is assumed white with normal probability distribution with zero mean, and covariance $Q (p(w) \sim N(0, Q))$.

The measurement model corresponds to the lateral distance $y_b$ from the vehicle $x_B$ axis to the rut center, which is located at the intersection of the laser plane $\Pi_1$ and the rut (see Fig. 20). The actual process employed to obtain the sensor measurements $y_b$ is detailed in Section 4.1.3. Using geometry and small angle approximations, it is possible to express $y_b$ as

$$y_{b,k} = -\sin(\theta_{vr,k})x_m + \frac{1}{2} \kappa x_m^2 \cos(\theta_{vr}) - y_{f,k} \cos(\theta_{vr}) + v_k,$$

where $v$ is white noise with normal probability distribution ($p(v) \sim N(0, R)$). As shown in Fig. 20, $x_m$ is a function of the state $q_k = [\theta_{vr,k}, \kappa_k, y_{f,k}]^T$ and the lookahead distance $\ell$ of the laser and satisfies

$$\frac{1}{2} x_m^2 \kappa_k \sin(\theta_{vr,k}) + \cos(\theta_{vr,k}) x_m - (\ell + y_{f,k} \sin(\theta_{vr,k})) = 0,$$

where (17) is obtained as a result of a coordinate transformation from the rut frame $R$ to the vehicle frame $B$.

### 4.1.2 Estimation of the Relative Position of the Rut and Vehicle Using an Extended Kalman Filter

In order to incorporate the spatio-temporal coherence between rut measurements, here we propose to use a tracking module based on an extended Kalman filter (EKF) that recursively estimates the parameters of the ruts (i.e., tracks the ruts) and then uses these estimates to improve the detection of the ruts for subsequent laser scans. In addition, the Kalman filter also generates smooth state estimates of the relative position and orientation (ego-state) of the vehicle with respect to the ruts, which are the inputs to the steering control system used to follow the ruts (see Fig. 5).

In compact form, we can rewrite (15) and (16) as

$$q_k = f(q_{k-1}, \delta \theta_{v,k-1}) + w_{k-1},$$

$$y_{b,k} = h(q_k) + v_k,$$

where

$$Q = \begin{bmatrix} Q \end{bmatrix},$$

$$R = \begin{bmatrix} R \end{bmatrix}.$$
where $q_k$ is the state of the process to be estimated, and $f(\cdot)$ and $h(\cdot)$ are nonlinear functions of the states and the model input and are given by

$$f(q_{k-1}, \delta \theta_{v,k-1}) = [f_1(q_{k-1}, \delta \theta_{v,k-1}), f_2(q_{k-1}, \delta \theta_{v,k-1}), f_3(q_{k-1}, \delta \theta_{v,k-1})]^T,$$

(20)

where

$$f_1(q_{k-1}, \delta \theta_{v,k-1}) = \theta_{vr,k-1} - \kappa_{k-1} v \cos(\theta_{vr,k-1}) \delta_t + \delta \theta_{v,k-1},$$

(21)

$$f_2(q_{k-1}, \delta \theta_{v,k-1}) = \kappa_{k-1},$$

(22)

$$f_3(q_{k-1}, \delta \theta_{v,k-1}) = y_{f,k-1} + v \sin(\theta_{vr,k-1}) \delta_t,$$

(23)

and

$$h(q_k) = -\sin(\theta_{vr,k}) x_m + \frac{1}{2} \kappa_k x_m^2 \cos(\theta_{vr,k}) - y_{f,k} \cos(\theta_{vr,k}).$$

(24)

In the following discussion, we adopt the notation of (Welch and Bishop, 2004), where $\hat{q}_{k-1}$ is our a priori state estimate at step $k$ given knowledge of the process prior to step $k$, $P_{k-1}$ is the a priori estimate error covariance and $P_k$ is the a posteriori estimate error covariance. The time update equations of the EKF are then given by

$$\hat{q}_{k} = f(q_{k-1}, \delta \theta_{v,k-1}),$$

(25)

$$P_{k} = A_k P_{k-1} A_k^T + Q.$$  

(26)

Equations (25) and (26) project the state and covariance estimates from the previous time step $k - 1$ to the current time step $k$. $f(\cdot)$ is given by (20), $Q$ is the process noise covariance, and $A_k$ is the process Jacobian at step $k$, which is computed using
\[ A_{k[i,j]} = \frac{\partial f_{i,j}}{\partial q_{i,j}}(\hat{q}_{k-1}, \delta \theta_{v,k-1}). \]  

(27)

Once a measurement \( y_{b,k} \) is obtained, the state and the covariance estimates are corrected using

\[ K_k = P_k^{-1}H_k^T(H_kP_k^{-1}H_k^T + R)^{-1}, \]

(28)

\[ \hat{q}_k = \hat{q}_k^- + K_k(y_{b,k} - h(\hat{q}_k^-)), \]

(29)

\[ P_k = (I - K_kH_k)P_k^-, \]

(30)

where \( h(\cdot) \) is given by (24), \( R \) is the measurement covariance, \( K_k \) is the Kalman gain and \( H_k \) is the measurement Jacobian, which is computed as

\[ H_{k[i,j]} = \frac{\partial f_{i,j}}{\partial q_{i,j}}(\hat{q}_k^-). \]

(31)

4.1.3 Sensor Measurement and Reduced Search Region for Rut Detection

In order to improve the efficiency and robustness of the rut detection and tracking algorithm, only a small region of the laser scan is analyzed for ruts. The search region is selected based on the distribution of the measurement prediction, which we assume follows a Gaussian distribution after the linearization process (Negenborn, 2003) and is given by

\[ p(y_{b,k}) = N(h(\hat{q}_k^-), H_kP_k^-H_k^T + R). \]

(32)

A confidence interval can then be defined around the predicted rut location \( h(\hat{q}_k^-) \). However, in this work, we use a search region of equal size (30cm) as the rut templates described in Section 3.1 and centered at \( h(\hat{q}_k^-) \).

As shown in Fig. 21, the search region contains 31 points with coordinates \( (y_{b,i}, P_i) \), for \( i = 1, 2, \ldots, 31 \), where \( y_{b,i} \) correspond to the distance between the \( x_B \) axis of the vehicle and the rut center (see Fig. 20) and \( P_i \) is the estimate of \( P(RUT|y_{b,i}) \), which represents the probability that the point at a distance \( y_{b,i} \) from the \( x_B \) axis is the center of the rut. The probability estimates \( P_i \) are computed using the low level rut detection algorithm explained in Section
3.1. Finally, the reported sensor measurement corresponds to the mean value of the distribution formed by the points within the search region and is obtained using

\[ y_b = \sum_{i=1}^{31} y_{b,i} P_i. \] (33)

This sensor measurement is then used to compute the filter innovation (29). If \( P(Rut|y_{b,i}) \leq \gamma_4 \) for \( i \in \{1, 2, \ldots, 31\} \), where \( \gamma_4 \) is a predefined threshold, then no measurement is used and the filter uses the prediction without the update step.

### 4.2 Rut Following (Steering Control)

This section details the development and the procedure followed to tune the steering control law that guides the vehicle towards the desired rut.

#### 4.2.1 Development of a Control Law for Steering Control

The proposed steering control is an adaptation of the controller proposed in (Thrun et al., 2006). The controller of (Thrun et al., 2006) was developed for an Ackerman steered vehicle. Here, we approximate the vehicle kinematics using a differential drive model and include a second speed varying gain \( k_2 \), which provides flexibility in the tuning of the controller as required for rut following.
As explained in Section 4.1.2, the EKF continuously generates estimates of the lateral offset \( y_f \) and the relative angle between the vehicle and the rut \( \theta_{vr} \). Here, the proposed steering controller is in charge of taking \( y_f \) and \( \theta_{vr} \) as inputs and then generates control commands for the robot to follow the ruts. In order for the robot to follow the ruts, \( \theta_{vr} \) should be driven to zero and the lateral offset \( y_f \) should be driven to a desired offset \( y_d = \frac{v_w + t_w}{2} \), where \( v_w \) is the width of the robot and \( t_w \) is the width of the tire. To achieve this, a desired angle for the vehicle \( \theta_{v,d} \) is computed using the nonlinear steering control law,

\[
\theta_{v,d} = \theta_r + \arctan \left( \frac{k_1(y_d - y_f)}{v_v} \right),
\]

where \( \theta_r \) is the angle of the rut with respect to the global frame \( N \), \( v_v \) is the robot velocity, and \( k_1 \) is a gain that controls the rate of convergence towards the desired offset. Based on (Thrun et al., 2006), it is possible to show that if the robot follows the desired angle \( \theta_{v,d} \), it will converge to the desired offset. To show this, first note that the rate of change of the lateral offset \( \dot{y}_f = \frac{dy_f}{dt} \) is given by

\[
\dot{y}_f = v_v \sin(\theta_{vr}),
\]

Substituting (34) into (35) yields

\[
\frac{dy_f}{dt} = v_v \sin \left( \arctan \left( \frac{k_1(y_d - y_f)}{v_v} \right) \right),
\]

which is equivalent to

\[
\frac{dy_f}{dt} = \frac{k_1(y_d - y_f)}{\sqrt{1 + \left( \frac{k_1(y_d - y_f)}{v_v} \right)^2}}.
\]

For small cross track errors, (37) can be approximated by

\[
\frac{dy_f}{dt} \approx k_1(y_d - y_f).
\]

Hence,
\[ y_f(t) = \eta e^{-k_1 t} + y_d, \]  

where \( \eta \) is a constant. From (39), one can see that the offset converges exponentially to the desired value at a rate controlled by the gain constant \( k_1 \).

In the proposed control approach, the desired angle \( \theta_{v,d} \) is then tracked using the proportional control law,

\[ \omega_v = k_2 (\theta_{v,d} - \theta_v) = k_2 \left( \theta_{vr} - \arctan \left( \frac{k_1 (y_d - y_f)}{v_v} \right) \right), \]  

where \( \omega_v \) is the desired angular velocity for the robot and \( k_2 \) is a speed dependent gain, selected as explained in 4.2.2. Notice that (40) takes as inputs the state estimates generated by the EKF. In order to avoid abrupt changes in the heading of the vehicle, saturation limits are imposed on (40), which leads to the final steering control scheme

\[
\omega_c = \begin{cases} 
\omega_v, & -\omega_{v,max} \leq \omega_v \leq \omega_{v,max} \\
\omega_{v,max}, & \omega_v > \omega_{v,max} \\
-\omega_{v,max}, & \omega_v < -\omega_{v,max}
\end{cases}
\]  

where \( \omega_{v,max} \) is the maximum angular rate that would be commanded to the vehicle, and \( \omega_c \) represents the commanded angular velocity.

As shown in Section 5, the proposed proportional control law yielded good experimental tracking results. However, for more aggressive driving maneuvers, especially in soft terrains, good tracking may require the use of derivative and integral terms in the control law.

4.2.2 Tuning of the Control Law

Tuning of the controller begins by determining the expected speeds of operation for the Pioneer 3AT. For rut following, according to (Blevins, 2007), the recommended speeds of operation for a Landrover LR3 vehicle are in the range of 1 - 10mph, which corresponds to speeds in the range of 0.2 - 1.54BodyLengths/s and map to speeds in the range of 0.10 - 0.77m/s for the Pioneer 3-AT.

In the current configuration, the laser plane intersects the ground plane at a lookahead distance of 42.82cm. In addition,
as explained in Section 4.1.1, small angle approximations are assumed while deriving the measurement model used for the Kalman filter. Therefore, for the ruts the system is designed to follow it is assumed that the maximum orientation change in a lookahead distance \( l \) is \( \Delta \theta_r \approx 15^\circ \). Thus, the robot should be able to achieve a turn radius of \( r_c = \frac{l}{\Delta \theta_r} = 163.52 \text{cm} \), at any given speed in the range 0.10 - 0.77 cm/s. This critical turn radius and the maximum speed \( v_{e,\text{max}} = 77 \text{cm/s} \) are used to set the saturation limits \( (\omega_{e,\text{max}} = 0.47 \text{rad/s}) \) in the control law (41).

In our implementation, the controller gain \( k_1 \) was set to 0.2 s\(^{-1} \) and \( k_2 \) was designed as a function of speed as follows:

As discussed before the maximum expected change in the orientation of the rut in one lookahead distance is \( \Delta \theta_r = 15^\circ \). Therefore, \( k_2 \) should satisfy

\[
\omega_v = k_2 \Delta \theta_r = \frac{v_{e,\text{max}}}{r_c},
\]

(42)

where \( v_{e,\text{max}} = 77 \text{cm/s} \). Therefore, at maximum speed

\[
k_2 = v_{e,\text{max}} \left( \frac{180}{r_c 15 \pi} \right) = 77 \left( \frac{180}{163.52 15 \pi} \right) = 1.79 \text{s}^{-1}.
\]

(43)

Experimentally a gain \( k_2 = 0.5 \text{s}^{-1} \) was found to produce good results for a vehicle speed of 0.1 m/s. From this result and (43), the gain \( k_2 \) was chosen as

\[
k_2 = 0.0193(v_v - 10) + 0.5,
\]

(44)

where \( v_v \) is in cm/s.

In the likely event that there is an initial lateral offset and a nonzero relative orientation between the vehicle and the rut, it is desirable to have an algorithm that will result in the robot approaching the ruts at small approach angles. By doing this, it is possible to avoid overshooting the ruts and to minimize the amount of turning in the rutted area. A general scenario is illustrated in Fig. 22, where the robot has an initial offset \( y_{f,0} \) and an initial orientation relative to the rut \( \theta_{vr,0} \). Fig. 22 also shows the approach angle to the ruts \( \theta_a \) and the desired offset \( y_d \).

In the following discussion the approach angles generated by the proposed steering controller are analyzed for different initial conditions and the speed range of operation. Fig. 23 shows a set of phase portraits \( (\theta_{vr} \text{ vs } y_f/y_d) \) for robot velocities \( v_v \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.77\text{m/s}\} \) and initial robot conditions which correspond to normalized (by \( y_d \))
lateral offsets $y_{f,0} \in \{-4, -3, -2, 4, 5, 6\}$ and relative orientations $\theta_{vr,0} \in \{-40^\circ, -20^\circ, 0^\circ, 20^\circ, 40^\circ\}$. As shown in Fig. 23, the robot converges to the desired state ($\theta_{vr} = 0, y_f/y_d = 1$) for all initial conditions and for all velocities. In addition, as seen from the phase portraits, the proposed controller with the selected values of gains $k_1$ and $k_2$ leads to small approach angles ($< 32^\circ$) and there is no overshoot of the ruts. It is important to note that the results presented in Fig. 23 constitute only forward simulations of the vehicle’s path using the proposed steering controller assuming perfect sensor measurements and initial conditions in the expected range of operation for the actual vehicle. However, if the initial conditions are chosen outside the range of operation (e.g., a very large lateral offset), the actual vehicle may diverge from the desired state because the ruts would not be visible for a prolonged amount of time, which could cause the Kalman filter to diverge.

5 EXPERIMENTAL PLATFORM AND EXPERIMENTAL RESULTS

In the following experimental evaluation of the proposed approach, we start by evaluating the rut tracking module of the reactive algorithm and the steering controller under different conditions including: S-shape ruts, broken ruts, shallow ruts, and ruts that are not directly in front of the vehicle. Finally, we perform an experiment which requires the usage of both the reactive and deliberative systems.

All experiments were conducted on a Pioneer 3-AT robotic platform equipped with a laser range finder URG-04LX (Okubo et al., 2009), which has an angular resolution of 0.36°, a scanning angle of 240°, and a detection range of 0.02m-4m. The laser readings are taken at 5 Hz and the laser is mounted in a custom build tilt platform, which allows the robot to collect the mid range data required by the deliberative system. In order to obtain ground truth data during the experiments an external positioning system was designed based on a SICK laser (Ye and Borenstein, 2002) that tracks the position of a cylindrical shape mounted on top of the robot with an accuracy of ±1.13cm. Fig. 24 shows a picture of the Pioneer 3AT, the tilt platform and the external positioning system.

The experimental evaluation was performed on soft dirt. It is important to note that the ruts created in this terrain type
Figure 23: Phase portraits for different velocities (a) \( v = 0.1 \text{ m/s} \), (b) \( v = 0.2 \text{ m/s} \), (c) \( v = 0.3 \text{ m/s} \), (d) \( v = 0.4 \text{ m/s} \), (e) \( v = 0.5 \text{ m/s} \), and (f) \( v = 0.77 \text{ m/s} \).

Figure 24: Experimental platform: Pioneer 3-AT, tilt platform, and external positioning system.

are structured similarly to the ruts typically encountered in off road trails as illustrated in Fig. 1. The evaluation of the algorithm on less structured ruts and different terrains will be considered in the future. In order to create the ruts used in this experimental evaluation, the following procedure was used. In each case the terrain was wetted and the robot was teleoperated for two or more runs following the same path. This procedure was enough to create the shallow ruts
used in the scenarios described in Sections 5.1.4 and 5.2.1. For the rest of the experimental scenarios, which contain
deeper ruts similar to those typically created on terrains with high moisture content such as mud or snow, the final ruts
were created using a single wheel attached to a shaft, which was rolled and pressed manually against the terrain for
two or more passes.

5.1 Evaluation of Reactive System

In experiments 1 through 4, the initial state (curvature of the rut \( \kappa_0 \), initial lateral offset \( y_{f,0} \), and relative orientation of the vehicle with respect to the rut \( \theta_{v,r,0} \)) is assumed to be known. These experiments are used to evaluate the rut tracking performance of the proposed approach. It is important to note that the experiments involved ruts of low curvature, which enabled verification that the wheels were in the ruts (i.e., the orientation of the robot was adequate) by using the video footage from the experiments and also visually observing the tread marks left by the robot as a result of its traversal.

The performance of the proposed algorithm is measured using the RMS value of the normalized cross track error \( e_{xt} \), which in these experiments, is computed using the data obtained by the ground truth system. The cross track error is normalized by the tire width \( t_w \) and it is computed as

\[
e_{xt}(x_i) = \frac{1}{nL_w} \sqrt{\sum_{i=0}^{n} (y_{v,d}(x_i) - y_v(x_i))^2},
\]

where, \( y_{v,d}(x_i) \) is the desired \( y \) value of the kinematic center of the vehicle at \( x_i \) and \( y_v(x_i) \) is the actual \( y \) value of the kinematic center of the vehicle at \( x_i \) as obtained by the ground truth system. The cross track error is computed at equally spaced increments of \( x \) (every 1cm).

5.1.1 S-shape rut with outliers

This scenario is used to show the robustness of the proposed approach against outliers. In particular, the scenario contains three ruts that are considered outliers. The rut length is 400cm, the rut depth is 6cm and the minimum turn radius is 71cm. Fig. 25 shows a set of snapshots obtained from the actual robot run and Fig. 26 shows the desired and actual paths for the robot kinematic center along with the outliers, and a summary of the cross track errors.
5.1.2 Scenario with broken ruts

This scenario shows the ability of the system to handle scenarios with broken ruts. Fig. 27 shows a set of snapshots from the actual run. In this scenario, the rut length is 365cm, the rut depth is 5-8cm and the left and right ruts disappear for a length of 64cm, which is equivalent to 1.28 times the body length of the vehicle. Fig. 28 shows the desired and actual path for the robot kinematic center obtained from the external positioning system and summarizes the cross track errors.
5.1.3 Scenarios with initial position, heading offsets and different speeds

These scenarios serve two purposes. First, they are used to show that the robot is capable of following a set of ruts despite initial lateral and heading offsets with respect to the ruts. In addition, these experiments are used to verify the correct tuning of the steering controller.

In the first experiment, the robot starts with a normalized lateral offset $y_{f,0} = 4$. The lateral offset was normalized by the desired offset $y_{d}$, which is measured from the right rut. The vehicle speed was set to 0.2m/s and the orientation between the vehicle and the rut was set to $0^\circ$. Fig. 29 shows a set of snapshots from the actual run on the robot and Fig. 30 shows a comparison of the simulated and experimental phase portraits. Notice that as expected, there are some differences between the two due to unmodeled effects in the simulation like tire-ground interactions. However, in both cases the robot converges to the desired state with no overshoot and at similar angles of approach. The simulated angle of approach $\theta_s$ was $-15.55^\circ$ and the experimental angle of approach $\theta_x$ was $-17.32^\circ$. The simulated time of
convergence \( t_s \) to \( y_d \) with a tolerance band of 2% was 11.8s and the experimental time of convergence \( t_x \) was 14s.

In the second experiment, the robot starts with a normalized lateral offset \( y_{f,0} = 4.6 \) and an initial relative orientation \( \theta_{vr,0} = 40^\circ \). The vehicle speed was set to 0.3m/s. Fig. 31 shows a set of snapshots from the actual run on the robot and Fig. 32 shows a comparison of the simulated and experimental phase portraits. In this scenario, \( \theta_s = -10.25^\circ \) and \( \theta_x = -11.23^\circ \). The convergence times are \( t_s = 14.8s \) and \( t_x = 16s \). Please see a video demonstration at: http://dl.dropbox.com/u/5214481/NonZeroInitialConditions.wmv
5.1.4 Scenario with shallow ruts

This scenario is used to show that the proposed system is able to track shallow ruts, the scenario corresponds to an S-shaped rut with a length of 240cm, minimum turn radius of 61cm and 3cm in depth. Fig. 33 shows both the desired and actual path of the robot kinematic center, and summarizes the cross track errors.

5.2 Evaluation of Deliberative System

Contrary to experiments 1 through 4, which assumed known values for the initial conditions of the EKF, in these scenarios the high level rut detection system described in Section 3.2 is used to automatically generate initial conditions for the filter. The first scenario shows the ability of the deliberative system to estimate the robot’s initial position and orientation relative to the rut. In the second scenario, the complete rut detection and following approach is tested using a scenario with multiple ruts.
5.2.1 Estimation of Initial Conditions

This scenario corresponds to the same S-shaped shallow ruts used to evaluate the tracking performance in Section 5.1.4. However, the robot has an initial orientation relative to the rut $\theta_{vr,0} = 25^\circ$ and an initial lateral offset $y_{f,0} = -8.5\text{cm}$. In this experiment, as shown in Fig. 34, the deliberative system was able to identify the optimal rut and determine that it pointed in the general direction of the goal. In addition, it estimated the initial conditions as $\theta_{vr,0} = 24.58^\circ$ and $y_{f,0} = -7.3\text{cm}$.

5.2.2 Determination of the Optimal, Suitable Rut

In this scenario, as shown in the snapshots from the actual experiment (Fig. 35) and in the rut grid of Fig. 36, the robot is faced with multiple sets of ruts. However, the deliberative system is able to compute the optimal rut and determine that it points in the general direction of the goal. Fig. 36 presents the optimal rut as found by the planner and the estimate of the initial lateral offset $y_{f,0}$. Finally, Fig. 37 shows the vehicle path and summarizes the cross track errors.

Please see a video demonstration at: http://dl.dropbox.com/u/5214481/Multipleruts.wmv

6 CONCLUSIONS AND FUTURE WORK

The main contributions of the paper are the inclusion of a new rut detection method based on a path planner and the experimental validation of the proposed rut detection and following approach on diverse scenarios. The paper presents and evaluates algorithms that employ a laser range finder to detect, model, and use the ruts in the terrain to guide the vehicle to a desired goal. In addition, the paper presents a set of motivational experiments on different robotic
The proposed $A^*$ based path planning algorithm is used to select the optimal rut to follow among several possible platforms (a large and a small scale robot) and terrains, which show that the power consumption of the robot can be minimized and the traction increased by following existing ruts.

Figure 35: Snapshots of the robot following a chosen set of ruts among different candidates.

Figure 36: Rut grid with multiple ruts.

Figure 37: Scenario with multiple ruts: desired and actual path followed by the kinematic center of the robot (cross track errors: min = 0.008, avg = 0.162, max = 0.337).
candidates and to initialize a reactive system that employs an extended Kalman filter to recursively estimate the parameters of a local model of the rut being followed. Experimental results were conducted on a Pioneer 3-AT robot and showed that the proposed system was able to handle scenarios with S-shaped ruts, broken ruts, shallow ruts, ruts that are not directly in front of the vehicle, and scenarios with multiple ruts.

Future work will involve the incorporation of replanning strategies (e.g., D* (Stentz, 1994)), which would enable the vehicle to switch from one rut to another in the middle of a mission and could also help in situations where the EKF might diverge. This will require the ability to obtain accurate vehicle state estimates (global position and orientation). It is also important to develop more advanced motion planners to simultaneously consider mission critical components such as the reduced energy cost of driving in a rut, vehicle safety, obstacle avoidance, and fuel-optimality.

In addition, as explained in Section 3.2.3, we would like to expand the current system to handle scenarios where the vehicle can make use of single ruts instead of sets of ruts. It is also be important to investigate a vision based approach to rut detection because it would provide long range information to complement the current local information obtained with the laser range finder and would open the possibility of detecting ruts based on textural differences.

Finally, it would be beneficial to evaluate the performance of the proposed algorithm in vehicles with steered wheels because a self aligning torque, due to the ruts, is generated on the wheels and therefore the steering control algorithm should be designed to take advantage of these natural dynamics.

Acknowledgment

Prepared through collaborative participation in the Robotics Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD 19-01-2-0012. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. This work is also partially funded by NSF grant DMS-0713012.

References


