A Trail-Following Robot Which Uses Appearance and Structural Cues

Christopher Rasmussen, Yan Lu and Mehmet Kocamaz

Abstract We describe a wheeled robotic system which navigates along outdoor "trails" intended for hikers and bikers. Through a combination of appearance and structural cues derived from stereo omnidirectional color cameras and a tiltable laser range-finder, the system is able to detect and track rough paths despite widely varying tread material, border vegetation, and illumination conditions. The approaching trail region is efficiently segmented in a top-down fashion based on color, brightness, and/or height contrast with flanking areas, and a differential motion planner searches for maximally-safe paths within that region according to several criteria. When the trail tracker's confidence drops the robot slows down to allow a more detailed search, and when it senses a dangerous situation due to excessive slope, dense trailside obstacles, or visual trail segmentation failure, it stops entirely to acquire and analyze a ladar-derived point cloud in order to reset the tracker. Our system's ability to negotiate a variety of challenging trail types over long distances is demonstrated through a number of live runs through different terrain and in different weather conditions.

1 Introduction

Roughly linear terrain features such as roads, hiking trails, rivers, powerlines, and pipelines are common in man-made and natural outdoor environments. Such features can be navigationally useful to unmanned ground or aerial vehicles in that they

C. Rasmussen (✉) · Y. Lu · M. Kocamaz
Department of Computer and Information Sciences, University of Delaware, Newark, NJ, DE 19716, USA
e-mail: cer@cis.udel.edu

Y. Lu
e-mail: yanlu@udel.edu

M. Kocamaz
e-mail: kocamaz@udel.edu

both “show the way” and “smooth the way”. Finding and keeping to a path by driving along it or flying above it can simplify an autonomous robot’s perceptual and motion planning tasks and mitigate hazards which occur in general cross-country navigation. The relative narrowness and continuity of such features implies a certain commonality in the framework of detection, tracking, and control, but each path type has unique appearance and structural characteristics worthy of investigation.

In this chapter we describe a robotic system (shown in Fig. 1a) for following hiking and mountain-biking trails through varied field and forest terrain. We assume that the trail is everywhere traversable with a wheeled vehicle, and also that the trail is non-branching and non-terminating, removing the necessity of intersection or dead-end detection (although our results show that the robot naturally if arbitrarily “chooses” a fork when given a choice). In essence, the task is analogous to “lane keeping” from autonomous road following, involving repeated estimation, or tracking, of the gross shape and appearance attributes of a previously-found trail.

This task echoes the first two DARPA Grand Challenges, which required vehicles to follow rough roads, but there GPS and ladar were sufficient for most successful teams [1, 2]. The DARPA Urban Challenge required more road shape estimation ability, and several teams detailed approaches using primarily vision [3] and rich structural information based on a Velodyne ladar [4]. In the DARPA LAGR program robots had stereo vision instead of ladar and were looking only for open space on their way to a GPS goal, although in constrained areas this was often coincident with path following. Among LAGR-derived work, [5, 6] stand out for explicitly looking for path-like corridors of homogeneous color or texture along the ground.

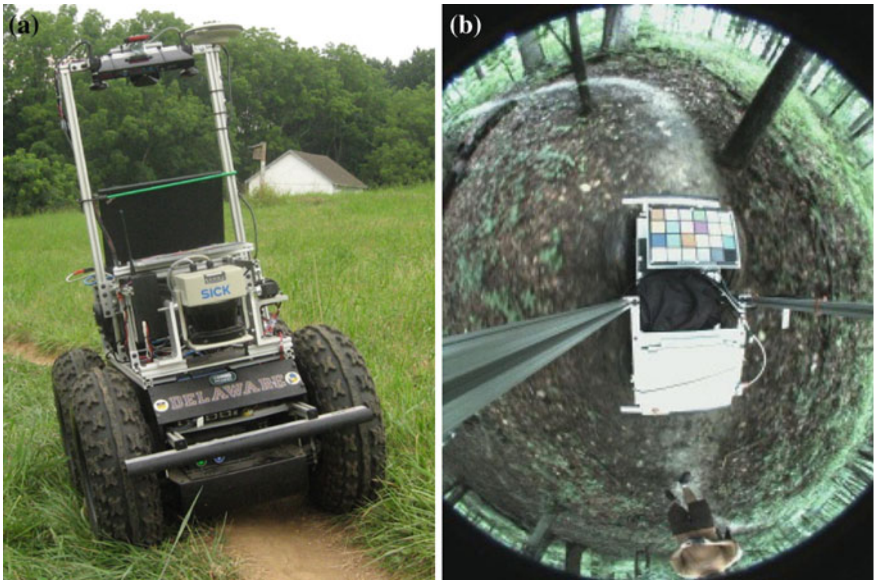


Fig. 1 **a** Robot in testing area; **b** Sample view from one omnidirectional camera

The European ELROB competitions have also required path-following skills; one robot effectively followed paths by finding "passages" among scattered trees in ladar data [7]. An approach to non-parametric trail detection and tracking using color + intensity saliency maps and agents was discussed in [8].

We reported on earlier versions of our omnidirectional trail-following system in [9, 10]. The former paper discussed a strictly monocular, *appearance*-based approach to discriminating and tracking the oncoming trail region in an image sequence, coupled with differential motion planning within the parametrized trail region while taking into account ladar-detected obstacles. [10] introduced an approach to incorporating stereo-derived scene *structure* estimates as an additional cue at the trail segmentation stage.

Here we present a fully integrated system which uses appearance and structure, not just from stereo but from ladar as well, to find and track the trail in real time. Previous iterations of the system moved at a constant speed regardless of trail or obstacle geometry; now the robot can detect loss-of-trail, excessive slope, or dangerous obstacle events to slow down and even stop in order to more deeply analyze the situation before proceeding. Finally, the differential motion planning system has been updated to lessen the likelihood of collisions while still preserving a basic impetus for forward motion. These changes have yielded vast improvements in the operational performance of the robot in many real-world situations.

2 Methods

As described in [9, 10], the trail region $\mathscr{R}$ immediately in front of the robot is approximated as a constant-width w arc of a circle with curvature κ over a fixed arc range $[d_{\min}, d_{\max}]$. The position of the robot with respect to the trail is given by its lateral offset Δx from the trail centerline and the difference θ between its heading angle and the tangent to the trail arc. Grouping these, we have the current *trail state* $\mathbf{X}$ as the 4-parameter vector $(w, \kappa, \Delta x, \theta)$.

Under the assumption that a unique trail is present in each image, it is segmented in a top-down, maximum likelihood fashion: multiple candidate regions are hypothesized and scored using a *trail likelihood* function L, and the highest-scoring region is the winner. Trail-following entails tracking the trail region over an image sequence, so we use particle filtering [11] to incorporate a prior $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ on the hypotheses which keeps them near the predicted location of the trail in the current frame as derived from the robot's dynamics, as well as setting absolute limits on every state parameter.

2.1 Appearance Likelihood

[9, 12] describe our technique for computing the color appearance likelihood of a candidate region $L_{appear}(\mathscr{R})$ based on the assumption that the trail region has a strong *color* and/or *intensity* contrast with the left and right neighboring regions $\mathscr{R}_L$ and $\mathscr{R}_R$. Briefly, following [5] we compute a small set of exemplar colors for each image using k-means clustering in CIE-Lab space and assign every pixel one of these k labels. A label histogram is computed for each candidate region and its neighbors, and the likelihood is obtained as a weighted combination of *contrast* and *homogeneity* (the entropy of the region color distribution). More details on the approach are given in [9].

In [9, 10] the color contrast is measured by the χ^2 distance between the region and its neighbors, and that measure is used here for some of our results. However, this approach can have some problems with certain scenes such as the one shown in Fig. 8, landmark 3, where several similar shades of grass are found alongside the trail. After k-means clustering the χ^2 metric treats all color clusters as equally dissimilar, meaning that two shades of green are effectively as different as a green and the brown of the actual trail. To avoid accidentally locking onto a marginally distinctive grassy strip beside the trail, we want a measure that preserves some notion of more- and less-similar colors after clustering. The earth mover's distance (EMD) [13, 14] has this property, and so we use this for contrast where noted in Table 1.

Extensive experimentation has shown this approach to trail segmentation to work on a wide range of trail types and illumination conditions without training. Nonetheless, we have found that as a practical matter camera exposure issues can cause serious problems, as with any vision algorithm run outdoors [3]. In particular, bright conditions can be very difficult because of issues with glare (i.e., oversaturation) and deep shadows. These phenomena can obliterate scene colors and make the trail impossible to see in sections, as in Fig. 2. Our cameras (Sect. 3) are in auto-exposure mode by default, but we have found that on sunny days the built-in algorithm frequently gives unsatisfactory results. To mitigate this we implemented our own proportional exposure control method which computes median intensity over a region of interest (ROI) directly in front of the robot and keeps it in a target range by adjusting the shutter speed. This results in much better contrast around the nominal trail region even if other portions of the image are under- or over-exposed.

2.2 Structure Likelihood

The color/intensity contrast between the trail region and neighboring regions depends heavily on the trail material and surrounding terrain and vegetation. While it is sufficient in many situations, when the contrast becomes too low trail tracking may deteriorate or fail entirely. An additional cue which may compensate in these situations is that of scene *structure*. Intuitively the trail region itself is expected to be

Table 1 Testing notes by day

Day	Weather	Key changes	Total km	Number of runs	Mean run (m)	Maximum run (m)
LW	Overcast, few shadows	–	0.77	7	110	310
*S*1	Overcast, few shadows	–	1.57	9	174	410
*S*2	Strong sun, deep shadow	Stop-and-scan for "danger," lowered undersaturation threshold	1.32	18	73	348
*S*3	Strong sun, deep shadow	ROI-based manual exposure control, image capture 7.5 $\rightarrow$ 10 fps	1.15	19	61	240
*S*4	Overcast, light rain at times	$\chi^2 \rightarrow$ EMD for color contrast, ladar traversability map in likelihood	1.68	7	240	800
*S*5	Scattered clouds, bright sun alternating with gray	Speed $\propto$ trail confidence, max-safety + min-hits motion planning	1.74	4	435	984

relatively smooth while off-trail regions are rougher (i.e., have higher height variance). Moreover, there is often a measurable contrast between the mean height of the trail and the mean height of regions immediately bordering it, whether due to grass, bushes, or rocks that do not exist in the trail or because a "trough" in the soil has been formed from the passage of previous hikers and bikers. More generally, we use the apparent *traversability* of a region as a proxy for the likelihood that the trail goes through it, and then linearly combine this likelihood with the appearance likelihood described above with weighting chosen based on experiments using ground-truth trail segmentations from a separate dataset. There are several sources of scene height information which we exploit:

Stereo A depth map for a subimage of every frame is generated from the robot's stereo omnidirectional cameras. We used the OCamCalib Omnidirectional Camera and Calibration Toolbox for Matlab [15] to obtain intrinsics for the two cameras. Relative extrinsics were initially estimated with manual measurements and then refined with bundle adjustment using *levmar* [16]. Following a common approach

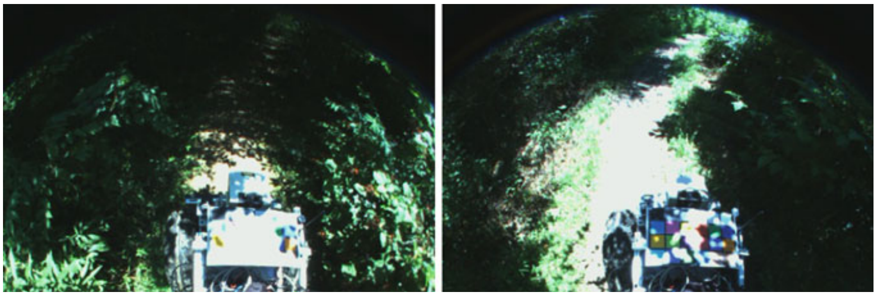


Fig. 2 Problems with oversaturation and shadows. These images are from test *S2* in the mixed section just north of landmark 4

to computing correspondences in omnidirectional imagery [17–19], we rectify the relevant portion of each omnidirectional image into a virtual perspective image such that epipolar lines are image rows; mask out the robot chassis, sensor mast, and peripheral pixels which are not imaged due to the fisheye lens; and then apply a standard pixel correspondence algorithm available in OpenCV, semi-global block matching [20]. The depth map for a sample scene is shown in Fig. 3 at left.

Next, we simplify the approach of [21], which fits planes to robot-sized chunks of a stereo-derived point cloud and combines them into a *traversability map* comprising several hazard-related factors. Full repeated plane-fitting is somewhat expensive, so we approximate it by computing the median absolute deviation (MAD) of the stereo height map over robot-sized bins. If μ_{MAD} is the mean MAD value or "badness" within a hypothesized trail region $\mathscr{R}$, then $L^{stereo}_{structure}(\mathscr{R}) = e^{-\alpha \mu_{MAD}}$. When combined with the appearance likelihood above, this formulation rewards smoother trail region hypotheses and ones which do not contain large step edges (up or down).

Tilting ladar point cloud The exact same MAD approach described for stereo can be used on any point cloud, and the tilting SICK ladar offers very accurate, very detailed point clouds when the robot stops long enough to perform a scan. Because

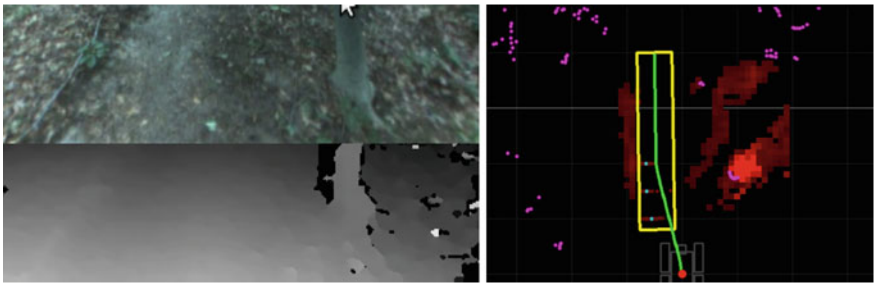


Fig. 3 (*Left*) Detail of rectified *left* camera image at landmark 13 (Fig. 8 for full image) and its stereo depth map; (*Right*) Corresponding stereo traversability map in *red*, with SICK ladar obstacles in *purple* and estimated trail region and planned robot path also indicated (grid is 1 m per square)

such point clouds are not available as the robot is moving, $L^{tilt}_{structure}$ is not part of the normal trail likelihood. When they are gathered, however (see Sect. 2.3 for an explanation of *when* full tilting ladar point clouds are created), they are used as the sole cue to infer the trail location, and the normal visual trail tracker's state is reset to the value indicated by the point cloud search. Some sample point clouds are shown in Fig. 4.

Regular ladar obstacles A traversability map of sorts can be created from the obstacles detected by the SICK ladar in its normal, level configuration as the robot travels. Because obstacles are only detected in one plane, we cannot use height variation as above. Rather, we use the simple criterion of proximity: a bin in the traversability map is incremented if it is within a robot radius of a ladar obstacle. The "badness" of a hypothesized trail region $\mathscr{R}$ is now a sum N of these colliding bins' values, and $L^{ladar}_{structure}(\mathscr{R}) = e^{-\beta N}$. Adding this component to the trail likelihood is extremely helpful because it will push trail hypotheses toward empty or less-dense regions of space even when the robot's visual system is impaired.

2.3 Motion Planning

As described in [9], our motion planner is derived from a Dubins car model [22], which accounts for differential constraints on the robot's motion in the form of a minimum turning radius and rules out reverse motion. The basic Dubins planner, which works for all start and end (x, y, θ) configurations in the absence of obstacles, is used as the kernel of a lookup-table-like approach to planning along the trail in the presence of obstacles. Briefly, given the currently estimated trail region a single *ultimate* goal pose and a set of nearer *candidate* goals are generated and planned for. Each of these plans is evaluated and possibly pruned based on their trajectories colliding with too many obstacles or leaving the trail. From the remaining plans whichever terminates closest to the ultimate goal is selected for execution.

Candidate goal poses are generated in a regular array spanning the trail region laterally in a series of *mini-lanes* and distally from just in front of the robot out to

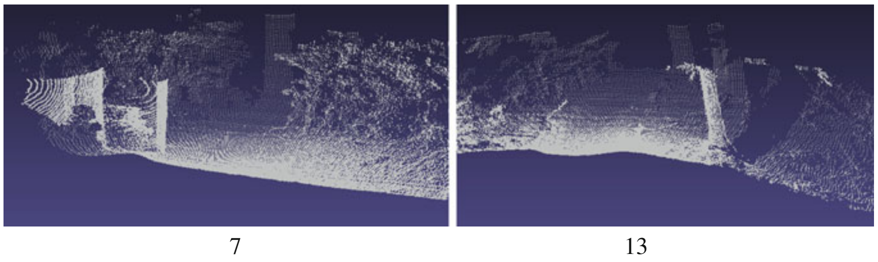


Fig. 4 Tilting ladar point cloud examples for landmarks 7 and 13 (see Fig. 8 for corresponding images). The embankment's drop-off on the *right* is clearly visible in 13

the ultimate goal, all with θ tangent to the trail. A Dubins plan is constructed from the current robot position to each candidate goal pose, and then *extended* along its mini-lane out to the ultimate goal distance. Selecting candidate goals along the same mini-lane but closer to the robot induces more aggressive lane changes in the manner of "swerves" versus "nudges" from [1] or "sharp" versus "smooth" trajectories from [23]. In this work the robot also adaptively inserts lanes which maximize clearance in order to help get through tight spaces. Some examples of plan candidates and their relation to inferred trail regions are shown Fig. 5.

In previous work [9, 10] the criterion for selecting between plans could be termed *min hits*—the path with the fewest collisions was selected. If there was a collision-free path, so much the better, but under the assumption that there is always a way forward, this heuristic would always keep the robot moving with the least amount of obstacle contact—critical for proceeding along trail sections with encroaching foliage such as landmark 5 in Fig. 8. One problem with this approach, however, is that even when there is enough room to stay well away from all obstacles, the robot may pass very close to them because there is no incentive to maximize clearance. Here we implement a two-level path evaluation technique which first ranks plans in terms of *max safety* (the best being a no-collision plan farthest from a collision) and only falls back to the previous min collisions criterion when every plan within the trail region collides.

A further safety and performance improvement can be gained by modulating the robot's speed based on the state's trail likelihood (aka "confidence") and the number of ladar hits anticipated along the robot's planned path. When the trail likelihood is above a certain threshold and the number of expected hits is 0, the robot moves at its maximum speed. As the confidence that it is accurately tracking the trail decreases and/or its expectation that it will be touching or near obstacles goes up, the robot smoothly turns its speed down to a fixed minimum to (a) Allow the trail tracker more time to find or get a better lock on the trail, (b) Allow more time for obstacle avoidance maneuvers to work if the robot is actively turning away from an approaching collision,

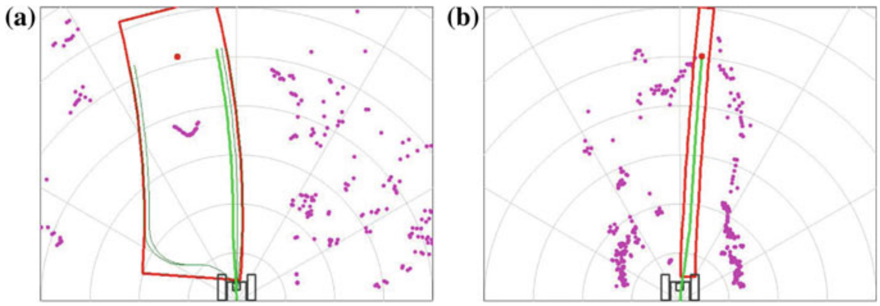


Fig. 5 Motion planning examples: Selected trajectories (*green*) and candidates (*dark green*) within the estimated trail regions. The *red* dot is the ultimate goal pose, while *purple* dots are obstacles detected by the SICK ladar. A *max safety*, no-collision plan is shown in **(a)**, while every plan collides in **(b)**, forcing a *min hits* decision

and (c) Not hit any solid obstacle as hard if a collision is unavoidable. Giving the trail tracker more time is critical at very sharp turns such as landmark 11 where the tracker may lag the angle of the quickly-curving trail, or when the exposure control is working to obtain better contrast in difficult light conditions.

Finally, as a last line of defense the robot will stop completely and perform a full tilt ladar scan when it detects "danger" in form of (a) very large roll or pitch angle which might cause the robot to roll over, or (b) too many collisions in the min collision plan. By performing a search for the trail in the traversability map generated from the full point cloud (as detailed above), the robot can correct mistracking caused by a confused vision system or prevent further travel down a steep slope.

3 Equipment

The primary sensors used for the results in this chapter are two Point Grey Flea2 color cameras and a SICK LMS 291 ladar. Each camera is mounted about 1.15 m off the ground, pointed straight down and rotated so that the longer axis of its CCD is oriented in the direction of vehicle travel. The baseline between them is roughly 0.2 m. The cameras are fitted with omnidirectional Fujinon FE185C046HA-1 lenses which provide a field of view (FOV) of 180° along the vehicle Z axis and 145° along the X axis. In all of these experiments the cameras were set for auto-white balance; where noted they were either in auto-exposure mode or had their exposure manually controlled as described in Sec. 2.1. All images were captured at 640×480 and downsampled as noted for different vision modules.

The SICK ladar is mounted on the robot about 0.6 m off the ground, 0.4 m in front of the robot center, and facing forward with a sweep plane parallel (by default) to the XZ (i.e., ground) plane. Its FOV is 180° and the maximum range is set to 8 m. Its tilt angle is controlled via a Dynamixel EX-106 high-torque servo, enabling the capture of point clouds when the robot is stationary. All point clouds used in this paper were gathered over a tilt range of $[+15, -45]$ degs., with the servo moving at a rate of 20°. / s and the SICK scanning at 50 Hz.

The robot used is a Segway RMP 400 with four-wheel differential steering. The default speed for autonomous trail-following here was 0.75 m/s except where otherwise noted, and the minimum turning radius was limited to 0.75 m. 0.6 m-wide front and rear bumper switches e-stop the motors automatically when pressed with 44.5 N or more of force.

To enable real-time performance, system tasks are distributed over several onboard computers connected via a gigabit Ethernet LAN with IPC message-passing [24]. For all of the experiments described here, the robot's primary computer for image processing, tracking, and motion planning is a Lenovo W520 laptop with an Intel Core i7-2720QM CPU and 8 Gb of RAM. A second computer (a Dell Precision M2400 laptop with an Intel Core Duo T9600 2.80 GHz processor and 4 Gb of RAM) handles and logs all data coming directly from—as well as commands sent to—the front and rear Segway motors, the SICK ladar, and the GPS.

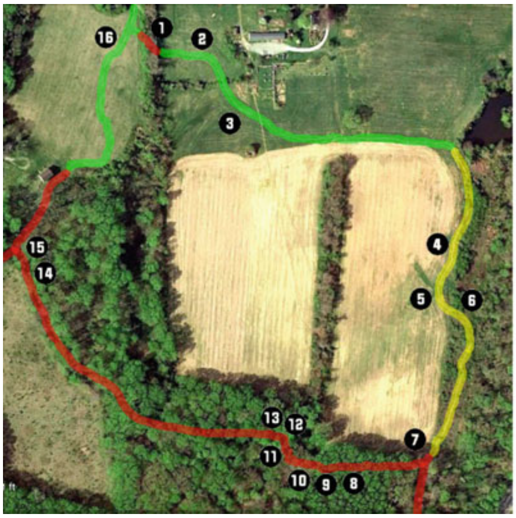
4 Experiments

All autonomous runs documented here were conducted along a combined hiking/ mountain-biking trail in a mid-Atlantic US state park which we will term *WCC*. The trail forms a ≈ 1.7 km long loop covering varied terrain which on a gross scale can be broken into three largely contiguous types:

(1) Open, grassy fields which are part of a working farm;
(2) A mixture of dense bushes and shorter trees, some overhanging; and
(3) Mature forest, some sections of which have sparse understory foliage and some which are quite dense.

As shorthand, we refer to these categories as *field* (0.6 km long), *mixed* (0.4 km), and *forest* (0.7 km), respectively. The entire loop with the types marked is shown in Fig. 6. A set of notable or difficult locations along the trail are numbered in clockwise order. A short description of each landmark is given in the table next to the trail map, and corresponding images are in Fig. 8.

Testing was conducted on six separate days, with one test occurring in late winter (*LW*) and five spanning the summer months (*S*1–*S*5). Key differences in weather conditions for each day are noted in Table 1, but seasonal variations in vegetation were also important. *LW* presented a challenge with a lack of color contrast (see Fig. 7 for some examples): the grass was dormant, making the fields predominantly brown and yellow, and the trail itself was a wet and muddy brown in many places.



#	Description	#	Description
1	Roots & clustered trees	9	Twin-tree bottleneck
2	Large trailside rocks	10	Narrow section with tall trailside foliage
3	Split for bike-jumping log	11	Sharp left turn before bridge
4	Large isolated trailside rock	12	Ramp to bridge
5	Very narrow section bordered by tall grass	13	Steep embankment after bridge
6	Right trailside sapling	14	Trailside fallen branch
7	Southeast trail junction	15	West trail junction
8	Camouflaged left trailside sapling	16	North trail junction

Fig. 6 Aerial image of ≈ 1.7 km *WCC* trail loop. *field* segments are shown in *green*, *mixed* in *yellow*, and *forest* in *red* (the trail location is approximate in the lower forest area). Numbered landmarks referenced in the text are briefly described in accompanying table and pictured in Fig. 8. The area shown is ≈ 0.6 km^2

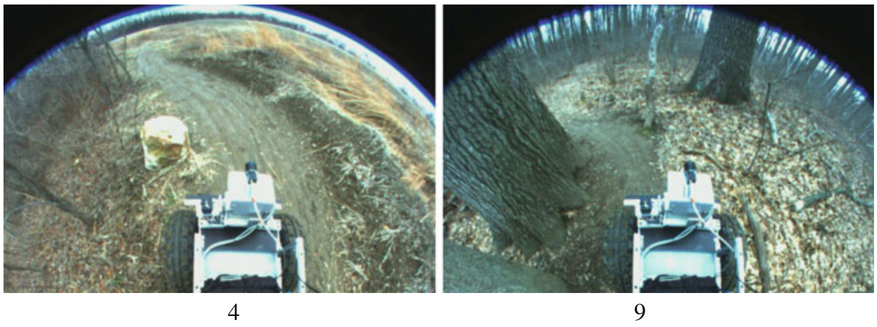


Fig. 7 Low color contrast from late winter (*LW*) at landmarks 4 and 9 on the *WCC* trail. See the summer images of the same landmarks in Fig. 8 for comparison

Trees and shrubs were largely bare and many mixed and forest sections were littered with leaves. By the time of *S*1 and *S*2, spring plant growth made for very strong color contrast for most of the field and mixed sections. This was true for some of the forest as well, but under a dense canopy much of the forest floor remains fairly brown throughout the year. Moreover, the field grass was long, providing height contrast. For *S*3–*S*5, however, the field grass had been cut short for hay.

Two small sections of the loop were not attempted because of terrain characteristics beyond the current perceptual and motion planning abilities of the robot. These were a forest segment in the north (landmark 1 in Figs. 6 and 8) and about 20 m of field just to the east of it (landmark 2). The problematic forest segment has one 2 m section of large, exposed roots and tightly spaced trees that is very difficult to negotiate even manually, and the field segment has a series of large rocks hidden in the grass right alongside the trail. Both are pictured in Fig. 8. Although the robot can track the trail through both of these sections, in the former case it cannot do technical driving that requires reasoning about balance and tire contacts or making zero-radius turns and reversing when necessary. In the latter case grass is growing so close to both sides of the trail that the robot must "brush" past it to proceed (e.g. landmark 5), and it does not detect or reason about the hidden rocks as solid obstacles which must be avoided entirely. Therefore, except for one run on day *S*2 into the rock section (shown in Fig. 8), all runs were started after the rocks.

Each run generally started immediately after the end of the previous run, but in several cases the robot was driven forward manually to get past a difficult spot or to skip a section entirely that it was not deemed ready for. This explains why the total distance traveled on each day varies and why, for example, *LW*'s total distance is not close to the loop length of 1.7 km: the brown grass contrast was so low that no field section was attempted. Time constraints on day *S2* and low motor batteries on *S3* necessitated skipping the last field section.

4.1 Results and Discussion

Top-level results are summarized in Table 1: over the course of six days of testing as the system evolved, 64 separate autonomous runs were attempted. A total of 8.23 km were traveled autonomously by the robot, with a mean run length of 129 m and a median of about 44 m. The five longest individual runs, in order, were 984, 800, 682, 504, and 410 m.

Results varied based on several factors, with the weather having surprising weight. As the table notes, days *S*2 and *S*3 were very sunny, and the average run length was quite low for those days. Of the 33 poor runs which were under 50 m in length, 24 were on these extremely bright days. Even after the ROI-based manual exposure control algorithm described in Sect. 2.1 was implemented for *S*3, the light was still an issue. A major improvement in robustness came with the incorporation of the ladar traversability map from Sect. 2.2 in the trail likelihood with *S4*. This helped the robot navigate areas like landmark 10 in Fig. 8 without being distracted by such bright patches. In general, the modification of the robot's perception and motion

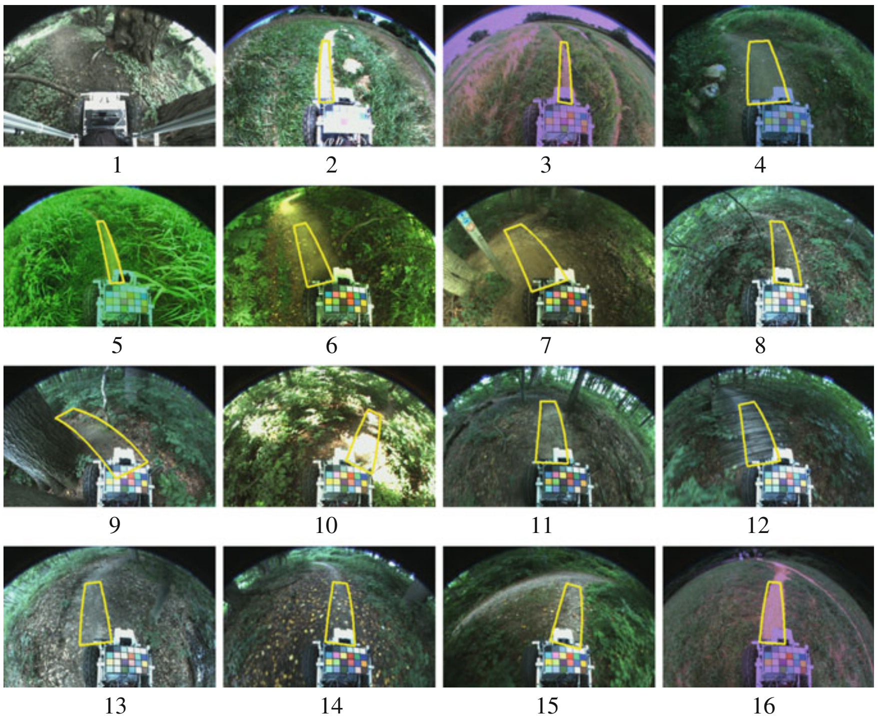


Fig. 8 Robot-view images of trail landmarks during autonomous runs, with the tracked trail region drawn in *yellow*. Landmark 1 was never attempted autonomously and thus no trail region is shown. The images for landmarks 3–16 are all from day S5

planning algorithms over the course of testing significantly improved performance. To underline this, the four longest runs overall were in *S*4 and *S*5. The mean run length for the first two days was 146 m, compared to 311 m for the last two, under similar weather conditions.

On *S*4 the robot completed the loop in seven runs, with its two longest runs that day accounting for 1.3 km or 76 % of the loop. The first of *S4*'s seven runs went from landmark 2 to landmark 6. The run ended when the robot's right front tire rubbed against a small trailside sapling on the way by, causing it to climb slightly and drop suddenly, disconnecting the camera's Firewire cable. The next run ended quickly because a USB cable was still loose. Run 3 was ended manually as the robot approached landmark 7, the trail junction (which it was able to successfully negotiate in *S*5). Run 4 began past landmark 9 and ended at landmark 11 because the turn was too sharp. Run 5 crossed the bridge but was manually terminated because the robot was very close to the embankment on the right (landmark 13). Run 6 quickly missed another sharp left turn, but run 7 took the robot to the end of the loop at landmark 16.

On *S*5 the robot completed the loop in four runs, with its two longest runs totalling 1.66 km or 98 % of the loop. Run 1 went from landmark 2 to landmark 3, where it appeared to mistrack because the trail was obscured by tall grass. Run 2 began at landmark 3, spanned most of the first field section, all of the mixed, and a difficult early portion of the forest before its bumper clipped the sapling in the center-left of the landmark 8 image in Fig. 8. Run 3 ended at landmark 9 because the robot could not maneuver between the two tightly-spaced trees, but on run 4 the robot made it across the bridge, past the embankment, and all the way to the finish line at landmark 16.

By the last two days of testing the trail-tracking system was mature enough that the types of failures observed were primarily ones of motion planning in technical and tight situations, rather than of mistracking the trail. A recurrent problem, echoed in landmarks 2, 6, 8, and 9, stems from a shortcoming in the robot's reasoning about obstacles in the two-level motion planner described in Sect. 2.3. The problem is that when the planner falls down to the *min hits* level, it is basically assuming that the obstacles it will be colliding with are all soft vegetation like grass and twigs. Seeking a least-density path through such obstacles makes sense as a strategy to stay on the trail, but the robot is unable to recognize that some obstacles like saplings and rocks, which may be mixed in with the grass, are solid and *must* be avoided.

5 Conclusion

The trail-following system presented here has been successfully tested over a variety of challenging terrain types, a range of weather conditions and seasons, and at different times of day from mid-morning to late afternoon. In its final form, no parameter changes are necessary for the robot's perceptual component to function in these different situations, nor did the robot have any *a priori* model of the characteristics of its area of operation. Based on previous work using very diverse image data sets [12, 25] and live runs in other locations [10], we believe that the system's perceptual capabilities would transfer quite well to other kinds of trails accessible to wheeled vehicles. In order to increase the overall reliability of the system, however, further improvements in the motion-planning component of the system are being incorporated, including explicit detection of certain classes of obstacles.

Despite these strengths, the possibility of dynamic parameter or sensor changes and exploitation of prior knowledge is attractive. For example, incorporation of trail map information via GPS and visual odometry along GPS-denied trail sections could be quite helpful (1) to let the robot know if it had strayed from the trail, or where the nearest trail was if "lost"; and (2) to allow for intersection anticipation and higher-level route planning over the trail network.

A scenario in which adapation would be desirable is nocturnal trail-following, as neither the color nor stereo structure information derived from the omnidirectional cameras would work in the dark without active lighting. As a dark-capable source of dense structural information for traversability map computations, we have tested a pair of non-overlapping Microsoft Kinect stereo depth cameras and found them highly useful in shady patches and early morning/twilight.

Acknowledgments The authors gratefully acknowledge the support of the National Science Foundation under award 0546410.

References

1. S. Thrun, M. Montemerlo, et al., Stanley the robot that won the DARPA grand challenge. J. Field Robot. **23**(9), 661–692 (2006)
2. C. Urmson et al., A robust approach to high-speed navigation for unrehearsed desert terrain. J. Field Robot. **23**(8), 467–508 (2006)
3. A. Huang, D. Moore, M. Antone, E. Olson, S. Teller, Multi-sensor lane finding in urban road networks, in Robotics: science and systems (Zurich, Switzerland, 2008)
4. C. Urmson et al., Autonomous driving in urban environments: Boss and the urban challenge. J. Field Robot. **25**(1) (2008)
5. M. Blas, M. Agrawal, K. Konolige, S. Aravind, Fast color/texture segmentation for outdoor robots, in *Proceedings of the international conference in intelligent robots and systems* (2008)
6. G. Grudic , J. Mulligan, Outdoor path labeling using polynomial mahalanobis distance, in Robotics: science and systems (2006)
7. C. Armbrust, T. Braun, T. Fohst, M. Proetzsch, A. Renner, B. Schafer, K. Berns, "Ravon—the robust autonomous vehicle for off-road navigation", in *IARP Workshop on Robotics for Risky Interventions & Environmental Surveillance* (2009)
8. P. Santana, N. Alves, L. Correia, and J. Barata, Swarm-based visual saliency for trail detection, in *Proceedings of the international conference in intelligent robots and systems* (2010)
9. C. Rasmussen, Y. Lu, M. Kocamaz, Trail following with omnidirectional vision, in *Proceedings of the international conference in intelligent robots and systems* (2010)
10. C. Rasmussen, Y. Lu, M. Kocamaz, Integrating stereo structure for omnidirectional trail following, in *Proceedings of the international conference in intelligent robots and systems* (2011)
11. A. Blake, M. Isard, *Active Contours* (Springer-Verlag, 1998)
12. C. Rasmussen, Y. Lu, M. Kocamaz, Appearance contrast for fast, robust trail-following, in *Proceedings of the international conference in intelligent robots and systems* (2009)

13. Y. Rubner, C. Tomasi, L. Guibas, A metric for distributions with applications to image databases, in *Proceedings of the international conference in intelligent robots and systems* (1998)
14. G. Mori, Guided model search using segmentation, in *Proceedings of the international conference in computer vision* (2005)
15. D. Scaramuzza, *Omnidirectional vision: from calibration to robot motion estimation* (Ph.D. dissertation, ETH Zurich, Switzerland, 2008)
16. M. Lourakis, levmar: Levenberg-Marquardt nonlinear least squares algorithms in C/C++ (2009), http://www.ics.forth.gr/ lourakis/levmar/. Accessed Nov 2009
17. N. Winters, J. Gaspar, G. Lacey, J. Santos-Victor, Omni-directional vision for robot navigation, in *IEEE Workshop on Omnidirectional Vision* (2000)
18. H. Koyasu, J. Miura, Y. Shirai, Realtime omnidirectional stereo for obstacle detection and tracking in dynamic environments, in *Proceedings of the international conference in intelligent robots and systems* (2001)
19. S. Lin, R. Bajcsy, High resolution catadioptric omni-directional stereo sensor for robot vision, in *Proceedings of the international conference in intelligent robotics and automation* (2003)
20. H. Hirschmuller, Stereo processing by semi-global matching and mutual information, *IEEE Trans. Pattern Anal. Mach. Intell* **25**(2), 328–341 (2008)
21. M. Maimone, C. Leger, J. Biesiadecki, Overview of the mars exploration rovers autonomous mobility and vision capabilities, in *ICRA Space Robotics Workshop* (2007)
22. S. LaValle, *Planning Algorithms* (Cambridge University Press, Cambridge, England, 2006)
23. D. Ferguson, T. Howard, M. Likhachev, Motion planning in urban environments: Part I, in *Proceedings of the international conference in intelligent robots and systems* (2008)
24. R. Simmons, Inter Process Communication (IPC) library (2012), http://www.cs.cmu.edu/~ipc. Accessed Jan 2012
25. C. Rasmussen, Shape-guided superpixel grouping for trail detection and tracking, in *Proceedings of the international conference in intelligent robots and systems* (2008)