Towards Learned Traversability for Robot Navigation: From Underfoot to the Far Field

Andrew Howard, Michael Turmon, Larry Matthies,
Benyang Tang, Anelia Angelova
firstname.lastname@jpl.nasa.gov
Jet Propulsion Laboratory
California Institute of Technology, Pasadena, CA, 91109

Eric Mjolsness
emj@uci.edu
Department of Computer Science
University of California, Irvine, CA, 92697

Abstract

Autonomous off-road navigation of robotic ground vehicles has important applications on Earth and in space exploration. Progress in this domain has been retarded by the limited lookahead range of three-dimensional (3D) sensors and by the difficulty of heuristically programming systems to understand the traversability of the wide variety of terrain they can encounter. Enabling robots to learn from experience may alleviate both of these problems. We define two paradigms for this, *learning from 3D geometry* and *learning from proprioception*, and describe initial instantiations of them as developed under DARPA and NASA programs. Field test results show promise for learning traversability of vegetated terrain and learning to extend the lookahead range of the vision system.

This is a preprint of an article published in *Journal of Field Robotics*, Volume 23, Issue 11/12.
Manuscript received 1 April 2006, accepted 30 October 2006.

1 Introduction

Robotic ground vehicles for outdoor applications have achieved some remarkable successes, notably in autonomous highway following (Dickmanns, 1992; Pomerleau, 1996), planetary exploration (Bapna, 1998; Biesiadecki, 2005; Leger, 2005; Maimone, 2006), and off-road navigation on Earth (Lacaze, 2002; Bodt, 2004; Krotkov, 2006). Nevertheless, major challenges remain to enable reliable, high-speed, autonomous navigation in a wide variety of complex, off-road terrain. 3D perception of terrain geometry with imaging range sensors is the mainstay of off-road driving systems. However, the stopping distance at high speed exceeds the effective lookahead distance of existing range sensors. Moreover, sensing only

terrain geometry fails to reveal mechanical properties of terrain that are critical to assessing its traversability, such as potential for slippage, sinkage, and the degree of compliance of potential obstacles. Rovers in the Mars Exploration Rover (MER) mission have stuck in sand dunes and experienced significant downhill slippage in the vicinity of large rock hazards. Earth-based off-road robots today have very limited ability to discriminate traversable vegetation from nontraversable vegetation or rough ground. It is impossible today to preprogram a system with knowledge of these properties for all types of terrain and weather conditions that might be encountered. The 2005 DARPA Grand Challenge robot race, despite its impressive success, faced few of these issues, since the route was largely or completely on smooth, hard, relatively low-slip surfaces with sparse obstacles and no dense, vegetated ground cover on the route itself.

Learning may alleviate these limitations. In particular, 3D geometric properties of obstacle versus drivable terrain are often correlated with terrain appearance (e.g., color and texture) in two-dimensional imagery. A close-range 3D terrain analysis could then produce training data sufficient to estimate the traversability of terrain beyond 3D sensing range based only on its appearance in imagery. We call this *learning from 3D geometry* (Lf3D). In principle, information about mechanical properties of terrain is available from low-level sensor feedback as a robot drives over the terrain, for example from contact switches on bumpers, slip measurements produced by wheel encoders and other sensors, and roughness measurements produced by gyros and accelerometers in the robot's inertial measurement unit (IMU). Recording associations between such low-level traversability feedback and visual appearance may allow prediction of these mechanical properties from visual appearance alone; we call this *learning from proprioception* (LfP).

Learning-related methods have a long, extensive history of use for image classification and robot road-following (Pomerleau, 1989, for example), but work in the paradigms described here is quite limited. LfP has been addressed recently in formulations aimed at estimating where the ground surface lies under vegetation, and closely related work (Wellington, 2005). For navigation on Mars, we are currently investigating a form of propriceptive learning that models the relationship between wheel slip, surface type and slope (Angelova, 2006a; Angelova, 2006b; Angelova, 2006c). Research in the vein of Lf3D has been done by several other teams in the DARPA-funded Learning Applied to Ground Robotics (LAGR) program, including Georgia Tech (Kim, 2006), Applied Perception, and SRI. We also mention Lf3D results (Sofman, 2006) which use laser ranging, rather than stereo, to produce ground truth for classification using visual appearance.

This paper outlines some key issues, approaches, and initial results for learning for off-road navigation, which we developed in the context of the LAGR program and the NASA-funded Mars Technology Program (MTP). Both use wheeled robotic vehicles with stereo vision as the primary 3D sensor, augmented by an IMU, wheel encoders, and in LAGR, GPS; they also use similar software architectures for autonomous navigation (Figure 1). Section 2 outlines these architectures and how they need to change to address Lf3D and LfP. Sections 3 and 4 present initial results of our work on Lf3D and LfP, one aimed at learning about vegetation and the other aimed at learning about obstacle compliance. Our work to date necessarily stresses simple methods with real-time performance, due to the demonstration-oriented nature of

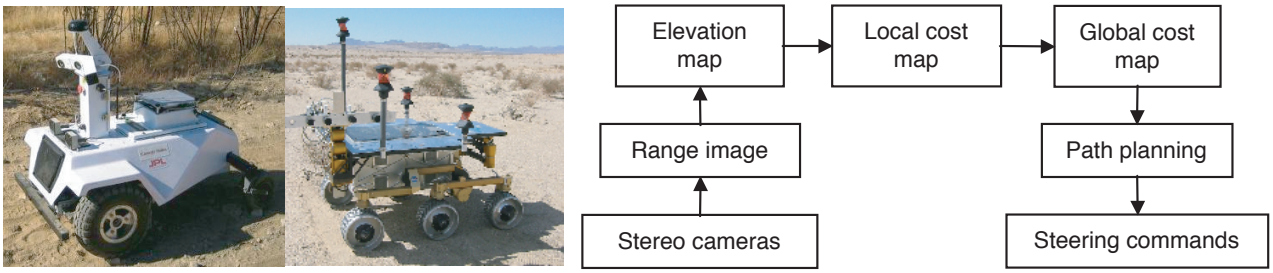

Figure 1: LAGR robot (left), Rocky 8 robot (center), and a simple view of their baseline navigation software architecture (right). Both robots are just over 1 meter long.

the LAGR and MTP programs; nevertheless, the results justify the value of our approaches and their potential to evolve to more sophisticated methods.

2 Architectures and issues

The baseline navigation software architecture in both the LAGR and MTP programs operates roughly as follows (Figure 1, right panel). Stereo image pairs are processed into range imagery, which is converted to local elevation maps on a ground plane grid with cells roughly 20 cm square covering 5–10 m in front of the vehicle, depending on camera height and resolution. The image and the map are the two basic coordinate systems used, but only pixels with nonzero stereo disparity can be placed into the map. Geometry-based traversability analysis heuristics are used to produce local, grid-based, "traversability cost" maps over the local map area, with a real number representing traversability in each map cell. The local elevation and cost maps are accumulated in a global map as the robot drives. Path planning algorithms for local obstacle avoidance and global route planning are applied to the global map; the resulting path is used to derive steering commands sent to the motor controllers.

This description illustrates both the source of the myopia and the lack of in-depth terrain understanding of traditional systems: (1) the extent of the elevation map is limited to the distance at which stereo (or ladar) get range data with acceptable resolution on the ground plane, and (2) the local map encodes only elevation, possibly enhanced with terrain class information derived from color or other properties of the image or range data, but with at best only crude prior knowledge of mechanical properties of each terrain class. Architectures similar to this have dominated DARPA, Army, and NASA robotic vehicle programs to date, though details in each box vary (Stentz, 1995; NRC, 2002; Lacaze, 2002; Maimone, 2006; Krotkov, 2006).

Figure 2 schematically illustrates the proprioceptive, appearance, and stereo information available to the robot in image and map coordinates, and how this information relates to Lf3D, LfP, and richer local map representations. We divide the scene into four zones — underfoot, near-field, mid-field, and far-field — and use s to indicate the three-dimensional position of a pixel or map cell. The variables in the left column of the figure indicate which sources of information are available about locations in the corresponding zone, reasoning as follows.

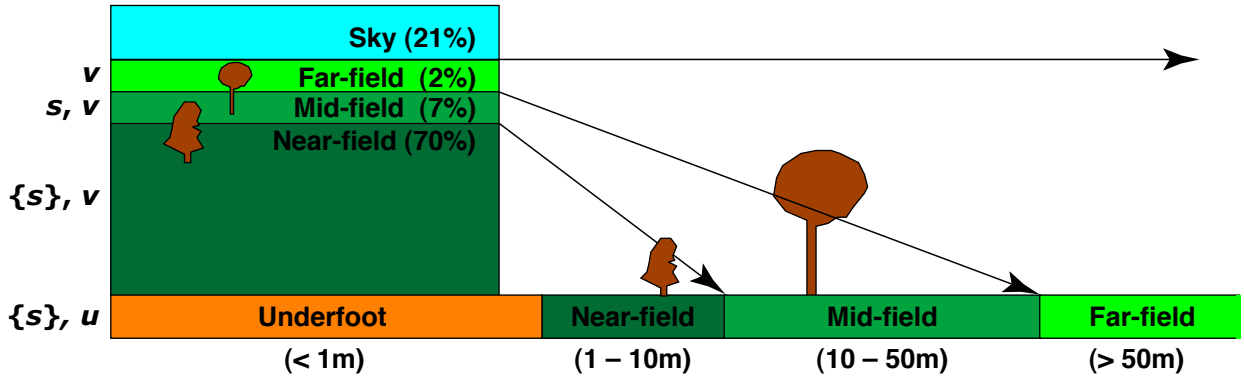

Figure 2: Typical information zones from proprioception and stereo (image space, left; map space, right), with specific numbers for the LAGR robot. The type of information present in each zone is shown in abbreviated form on the far left. See text for discussion.

Underfoot, the robot has sensors (accelerometers, gyroscopes, wheel encoders, and bumpers) that provide a proprioceptive feature vector u for the terrain beneath the robot. Additionally, the terrain geometry underfoot is known because it is present in previous maps; however, visual appearance is not available. In the *near-field*, stereo vision gets range data of sufficient density and accuracy to build a gridded local elevation map, where the grid spacing is set by the robot's size. The near-field is distinguished by the property that a large enough set of pixel sites $S = \{s\}$ lands in one map cell to collect meaningful height or roughness statistics at the scale of the robot's footprint. For example, roughness can be measured by the standard deviation of the height component of $s \in S$. In the near-field, a color and texture feature vector (collectively, "visual appearance" v) is also available for insertion into the map. In the *mid-field*, range data and visual appearance are still available. However, the per-pixel range data samples the ground too sparsely to obtain elevation statistics within a map cell (robot footprint). Schematically, as denoted in the far left of Figure 2, we have range s but not range statistics $\{s\}$. The *far-field* region is beyond the range of stereo vision (it has zero disparity), so only visual appearance is available.

Image pixels relate nonlinearly to ground areas, magnifying the importance of the mid-field and far-field to long-range planning. To make things concrete, in LAGR, the near-field is about 70% of the image, the mid-field is 7%, and the far-field is 2%. However, on the ground plane, the near-field covers about 1–10 m, the mid-field from 10–50 m, and the far-field from 50 m to infinity (right side of Figure 2).

Given this view, our problem can be cast as transferring knowledge between the adjacent distance regimes in Figure 2: (1) between underfoot and near-field (proprioception vs. appearance plus rich geometry), (2) between near-field and mid-field (appearance plus rich geometry vs. appearance plus poor geometry), and (3) between mid-field and far-field (appearance plus poor geometry vs. appearance only). Learning will extend the effective lookahead distance of the sensors by using the learned correlations to ascribe properties sensed proprioceptively or geometrically in the closer zones to regions sensed just by appearance or weaker geometric perception in the more distant zones. The same obstacle classes, and sometimes the same obstacle, will be present across all zones. Our ultimate goal is to jointly

estimate terrain traversability across zones, unifying the Lf3D and LfP concepts, and encompassing slippage, sinkage, and obstacle compliance in the notion of traversability.

2.1 Proxies and learned estimators of traversability

Traversability T is a random variable associated with a certain site s, either a pixel in the scene or a cell in the map. When T_s is associated with a pixel, it must be placed in the map to affect the route planner (Figure 1, right panel); for more on this, see the end of Section 4. T_s always takes values in the unit interval, but depending on context, we may take it to be binary (e.g., bumper hits) or real-valued (e.g., wheel slip). In making the link to path planning, it may be helpful to define T_s as the probability that the robot can successfully move out of a map cell s after deciding to do so. We could imagine a physics-based simulation that would determine this *exit probability* given vehicle and terrain parameters. Accumulating this T over a path would then yield the cumulative probability of a successful sequence of moves.

Lacking such a model, we view T as a random variable to be estimated from correlated information, where the estimator is in turn learned from training data. To learn traversability in a given zone, our strategy is to use high-quality input examples (typically, from a zone nearer the robot, as in Figure 2) to produce training labels $\tilde{T}$, which serve as proxies for the unknown T. This labeled data can be regarded as produced by a (noisy) membership "oracle" (Valiant, 1984). The proxy labels are given to a learning algorithm which trains a regression model $\hat{T}(\cdot)$ that approximates $\tilde{T}$. The regression model is then used to drive the robot.

In Lf3D, terrain geometry, measured through local elevation statistics like roughness and slope, is used to provide the proxy $\tilde{T}$, which is estimated using appearance information (normalized color in the work reported here). In LfP, the proprioceptive inputs (e.g., bumper hits and slip) are used to generate the proxy $\tilde{T}$, which is then estimated using the available appearance and geometry information from stereo images. Other approaches also fit into this framework. For example, backtracking bumped objects through past frames can be viewed as using prior appearances of the object to compute the proxy $\tilde{T}$. Fundamentally, we construct a proxy for traversability using higher-quality training data, accumulate a training set, and then select a regressor that is a function of lower-quality data at greater range. The next sections show how this idea is used in two different extrapolation schemes.

3 Learning near-field traversability from proprioception

In the LAGR program, we are using the LfP paradigm to address the key problem of learning about traversability of vegetation. For robots in general, the bumper, IMU, and slip measurements ultimately will all be important in assessing traversability underfoot. In practice, for the robot and terrain used in the LAGR program to date, the bumper provides most of the information, so we currently take the proxy $\tilde{T}$ to be a 0/1 quantity. Operationally, we can gather samples of $\tilde{T}$ by recording the geometric and visual characteristics $(\{s\}, v)$

of objects we can and cannot push through. In principle, each bumper hit provides several frames of prior presentations of the offending object which can be used as training data (Kim, 2006). However, due to limitations of localization, especially under conditions of partial slip, bumper hits are a very sparse source of data. Also, because bumper hits temporarily disable the LAGR robot, gathering nontraversable examples is expensive.

Furthermore, a technical problem of *blame attribution* arises because roughly six map cells are overlapped by the bumper at any time, so the nontraversable samples are contaminated with data from traversable cells. Heuristics alone may prove sufficient to narrow down blame to one cell, or a constrained clustering approach may be needed to separate these two classes. In these experiments, we have sidestepped the blame attribution problem by obtaining training data from hand-labeled image sequences: a human identifies sets of traversable and untraversable map cells.

3.1 Terrain representation

Elevation maps *per se* do not adequately capture the geometry of vegetated and forested terrain. Three-dimensional voxel density representations have been used successfully with range data from ladar (Lacaze, 2002). We are experimenting with such a representation for range data from stereo vision. The space around the robot is represented by a regular three-dimensional grid of 20 cm×20 cm×10 cm high voxels (Figure 3, top left). Intuitively, we expect that only low-density voxels will be penetrable. The voxel density grid is constructed from range images by ray-tracing: for each voxel, we record both the number of *passes* (rays that intersect the voxel) and the number of *hits* (rays that terminate in the voxel). The per-voxel density ρ equals the ratio of hits to passes. Since the ground may be non-planar, we also identify a *ground voxel* g in each voxel column; we assume that this voxel represents the surface of support for a robot traversing this column. The ground voxel is determined using a simple heuristic that locates the lowest voxel whose density exceeds some preset threshold. Although calculating it is relatively complex, in practice the density estimate is robust and rich in information.

Each map cell s has an above-ground density column $[\rho_{g(s)+1}\ \rho_{g(s)+2}\ \cdots\ \rho_{32}]$. For simplicity, we have worked with these features only: ρ^* (the maximum density), i^* (the height of ρ^*), ρ^{**} (the next-largest density), and i^{**} (its height). We have used ρ^* and ρ^{**} below, but ρ^* and i^* provide similar performance when used together. The average color within a map cell would also be a good feature, but we have not used this in classifications yet.

3.2 Learning algorithm

Initially, we wanted to validate the use of the density features and to replace our existing hand-coded, geometry-based traversability cost heuristic with a learned value. In this offline context, training time is not an issue so we use a support vector machine (SVM) classifier. We used a radial basis function kernel, with the SVM hyperparameters estimated by cross-validation. The training data consisted of 2000 traversable and 2000 nontraversable

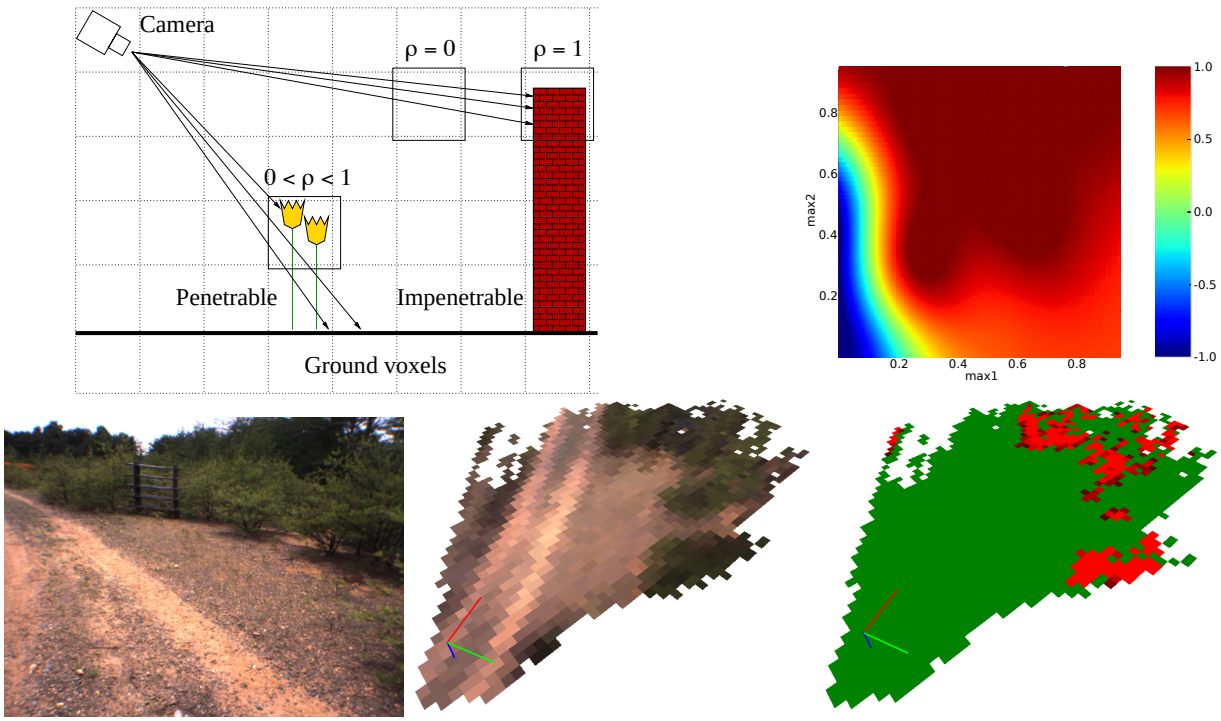

Figure 3: Learning from proprioception. Top left: schematic illustrating the voxel density map representation. Top right: learned cost lookup table (LUT) as a function of $\rho*$ and $\rho**$. Below left: sample camera image. Below center: image projected onto a local map with each map cell colored with the mean of all pixels projecting into it. Below right: cost map computed from voxel densities; green is traversable, red is not.

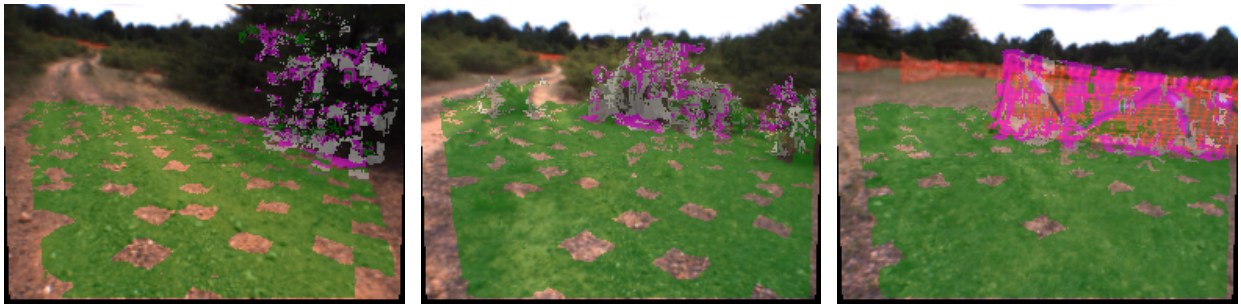

Figure 4: Backprojected LfP results for three frames of a LAGR trial. Pixels corresponding to traversable map cells are shown with a green overlay; pixels corresponding to non-traversable cells are shown in purple and gray. An orange snow fence is also in the scene.

examples, and the resulting model has 784 support vectors (SVs). The large number of SVs for a relatively modest two-dimensional problem indicates a considerable degree of overlap between the classes (which is borne out in scatter plots of the data). Tests were performed on an independent image sequence which contains roughly 2000 examples. We achieved a classification error rate of 14% on the test set, again indicative of strong class overlap from these limited features.

Classification is done at frame rates of 2–5 Hz, so SVM query time would be prohibitive. We therefore coded the SVM into a lookup table (LUT) for speed and simplicity, but a reduced-set SVM would be another alternative (Schölkopf, 1999). The continuous output of the SVM (Figure 3, right) is turned into a traversability measure through a simple linear function.

The results of SVM classification and the LUT are shown in Figure 4. The learned classifier is able to distinguish beteen traversable areas (green), and nontraversable areas (purple); intermediate values are also shown (grays). This algorithm was employed in LAGR test 5, where it was successful in distinguishing between grassy meadows (traversable), undergrowth (moderately traversable) and tree trunks (nontraversable). Unfortunately, the design of the LAGR tests to date is such that the potential of this approach has not been fully explored. The test courses have been almost entirely "binary," consisting of close cropped grass or unvegetated terrain (traversable) and tall bushes and trees (nontraversable). At no point has the robot been required to push through low vegetation in order to reach the goal. Under these conditions, simpler approaches (such as those based on pure elevation) may suffice.

4 Learning mid and far-field traversability from near-field 3D geometry

To address another goal of the LAGR program, we are using the Lf3D paradigm to extend near-field range-based proxies $\tilde{T}$ to mid-field and far-field traversability estimates $\hat{T}$. Here $\tilde{T}$ is a function of the heights of all pixels landing in a $(20\,\mathrm{cm})^2$ map cell. When at least ten pixels land in one cell, their average height δ_z above a nominal ground plane becomes resolvable: a large value indicates rough ground or obstacles. We compute a traversability proxy $\tilde{T} = f(\delta_z)$ for the cell, which is associated with the visual appearance v of all pixels mapping into that cell, thus providing a training set $\mathcal{T}$ of $(v, \tilde{T})$ pairs. We use this $\mathcal{T}$ to select an extrapolating function $\hat{T} = \hat{T}(v)$ from visual appearance to traversability.

We currently use two appearance-based features: the normalized R and G components of the RGB color; i.e., $v_1 = R/(R+G+B)$, $v_2 = G/(R+G+B)$. These features are chosen to provide some degree of robustness to variable lighting conditions and shadows. We also have choice for the amount of training data we use: at one extreme, we can train and extrapolate within a single frame only; at the other extreme, we can train over many hundreds or thousands of frames, and use the learned regressor over all subsequent frames. Given that the current feature set is relatively weak, we have mainly pursued the former approach. We thus assume that appearance and traversability are well-correlated with a single image, but do not assume that they are well-correlated over time.

For the single-frame case, speedy training and evaluation are required, prompting the re-

duction of $\mathcal{T}$ to a parameterized model. Below, we consider two approaches: unsupervised k-means clustering followed by regression, and supervised discriminant analysis with mixtures of Gaussians (MoG); a close variant of the first was used in LAGR test 7.

4.1 Unsupervised k-means regression

The geometry-based proxy is itself heuristic, so we might prefer to use $\tilde{T}$ somewhat weakly. We had success with unsupervised clustering of the input pixel appearance, followed by deducing the per-cluster traversability from the average proxy value within each cluster. That is, we discard the $\tilde{T}$ labels within $\mathcal{T}$ and perform a k-means clustering with $K = 4$. The traversability estimate is a weighted average of per-cluster traversabilities

$$\hat{T}(v) = \sum_{k=1}^{K} \tilde{T}_k \exp(-\|v - \mu_k\|^2/2\lambda^2) \Big/ \sum_{k=1}^{K} \exp(-\|v - \mu_k\|^2/2\lambda^2),$$

where $\tilde{T}_k$ is the average traversability proxy value per cluster, μ_k is the kth cluster center, and $\|\cdot\|$ is Euclidean norm. This can be viewed as nearest neighbors regression which uses a k-means data compression step to allow fast evaluation of $\hat{T}(v)$ at classification time.

We have used this method in frame-by-frame on-line learning with results that are illustrated, for three frames, in Figure 5. For each frame, the images on the bottom row show the rectified RGB image, the elevation training data, and the k-means regression result. Note that the training data has a somewhat 'blocky' structure, due to the fact that this data is back-projected from a 2D map (i.e., for each pixel in the image, we determine the corresponding cell in a 2D map and assign to the pixel the δ_z value for that cell). In the results image, very light and very dark pixels have been thresholded, which has the effect of removing both the sky (very bright) and some of the foreground bushes (very dark).

The top row of plots for each frame are generated in the feature space, and show training data and k-means cluster centers (annotated with δ_z), the alternate Mixture of Gaussians results (discussed in the next section), and the final k-means regression function. In the left-hand plots, the coherence of the training data indicates that the appearance-based clusters do indeed capture the traversability structure ("data compression" without too much lossiness). Comparing the right-hand plots across the three different frames, one can also observe that the relationship between appearance and traversability does indeed change over time (in the first two frames, redish pixels are traversable; in the final frame, they are nontraversable).

Figure 7 (left) shows the learning error rates as a function of frame number over a single trial. Four curves are shown: training and test error with four classes, and training and test error with eight classes; all curves depict the RMS error between the regressed and measured elevation with $\lambda = 0.1$. The test data was generated by dividing each frame into bands of 32 columns, and using alternating bands for training and testing.

Two key features should be noted. First, the results are insensitive to the number of classes, suggesting that the limiting factor on training error is not the learning algorithm itself, but rather the relatively weak set of features (normalized R and G). Second, the training and test error rates track quite closely (up to some offset). This suggests that we may use the

training error as a measure of reliability for the regressor, and selectively ignore the k-means traversability predictions on frames where the error is large.

4.2 Supervised MoG-based discriminants

It may be preferable to constrain the cluster membership a priori (using $\tilde{T}$ up front) rather than extracting clusters after the fact. At the expense of some reliance on a prior rule about association based on $\tilde{T}$, we may extract more stable and homogeneous clusters. We have experimented with three approaches: introducing $\tilde{T}$-based cannot-link constraints into k-means (Wagstaff, 2001), stratifying the cluster memberships according to $\tilde{T}$ within the Expectation-Maximization (EM) algorithm in a semi-supervised framework (McLachlan, 2000, sec. 2.20) and adopting a two-class discriminant-based approach with populations determined by $\tilde{T}$. We describe the last approach, which is simple and effective for the problems we have seen.

In the discriminant method, rough thresholds are used to form sets of traversable and non-traversable examples: $\mathcal{T}_0 = \{(v, \tilde{T}) : \tilde{T} \leq \tau_0\}$, $\mathcal{T}_1 = \{(v, \tilde{T}) : \tilde{T} > \tau_1\}$. We selected $\tau_0 = 0.1\,\mathrm{m}$ and $\tau_1 = 0.2\,\mathrm{m}$: obstacles lower than τ_0 are very likely traversable, those higher than τ_1 are very likely not, and we remain agnostic about those in between. Two separate MoGs $p_0(v)$ and $p_1(v)$ are fit to the two training sets, and we declare a pixel traversable if $p_1(v)/p_0(v)$ exceeds a threshold, which is set with reference to error rates on the training set.

We have used $K = 3$ component, full-covariance Gaussian mixtures to parameterize each of the two distributions, and fit the parameters by maximum-likelihood using the EM algorithm. The results are similar for $2 \leq K \leq 6$. Figure 6 shows results for this method when used in a frame-by-frame mode. We found that using a full three-dimensional RGB feature v gave superior results to normalized color. The explanation may be that the full covariance structure in the mixtures can accommodate the spread caused by varying illumination. On the other hand, the illumination-stretched clumps do not mesh well with the implicit spherical assumption of k-means. We project the three-dimensional mixtures down into the $R-G$ vs. $G-B$ plane for purposes of visualization.

In each of the frame-by-frame results of Figure 6, there is reasonably good separation between the two classes. The classes sometimes have internal structure that would not be well-captured by a single Gaussian. The test error achieved is 6%. Training time for $N = 1000$, three-dimensional data, and $K = 3$ is about 40 ms in an unoptimized code. Evaluation time for 5000 pixels (about 9% of a 192×256 pixel image) is less than 10 ms, which easily permits training and evaluation at our path-planning rates of 2–5 Hz.

4.3 Putting traversability in the map

Note that both approaches classify terrain in the *image* space: each pixel is assigned a traversability estimate $\hat{T}$. To use this result for navigation, these values must be projected into the map. There are two issues: how to combine traversability estimates and proxies, and how to determine the 3D location of image pixels. For data fusion, we currently

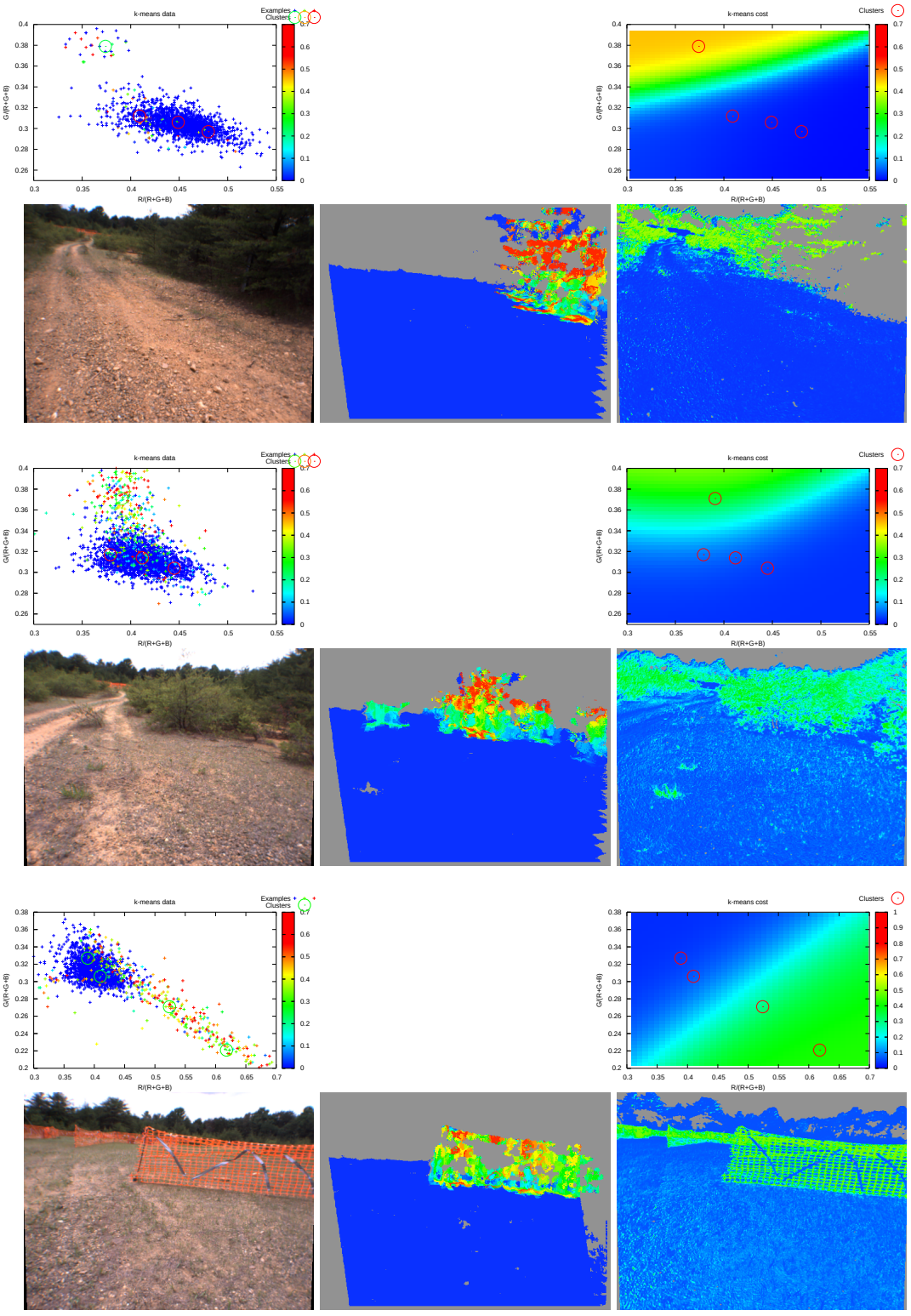

Figure 5: Learned traversability for three frames, each illustrated via two rows of plots. *Upper row:* k-means clusters and elevation-coded scatterplot (left); learned regression model (right). *Lower row:* rectified RGB image (left); image-plane elevation training data (center); learned k-means regression (right).

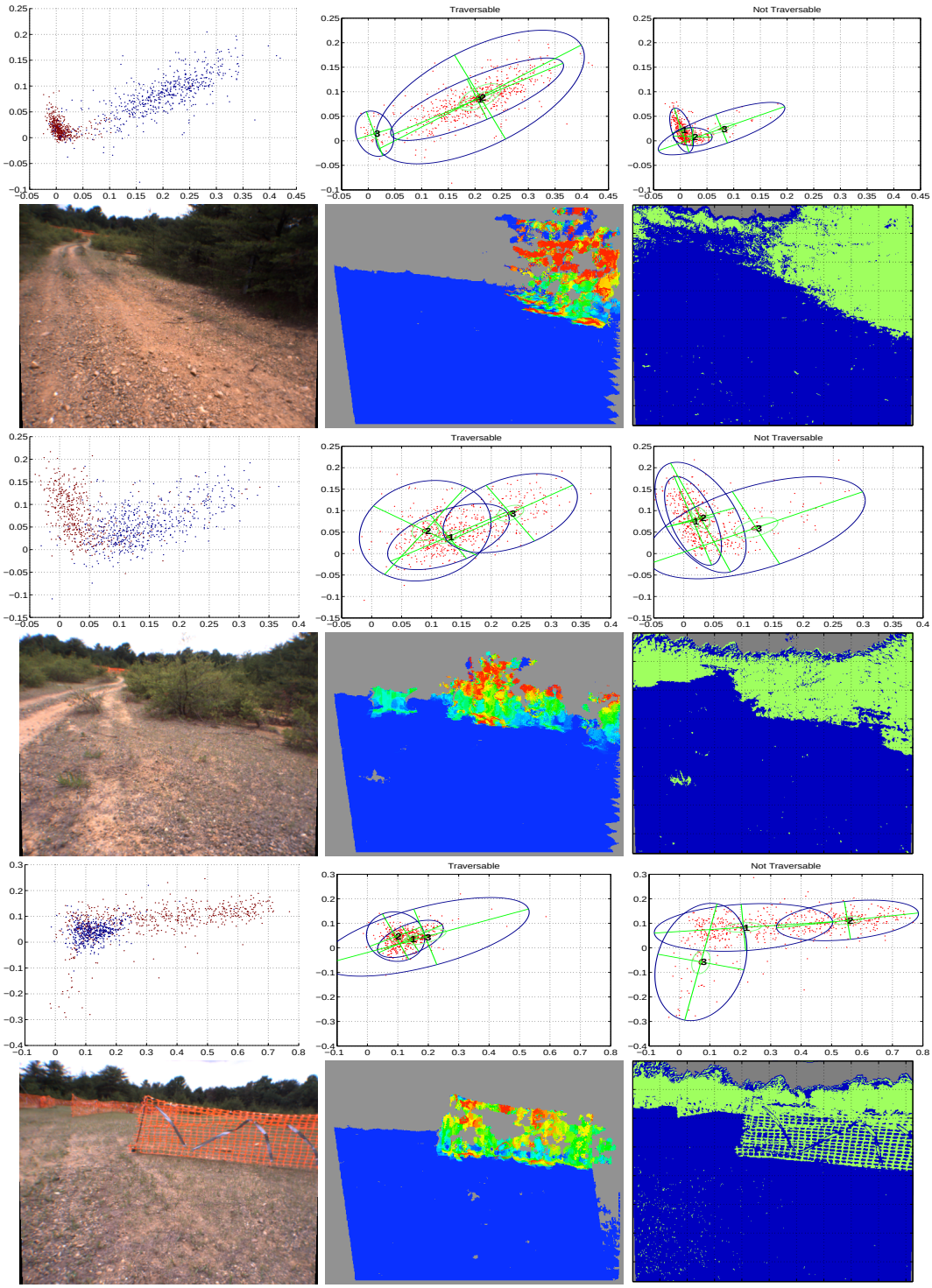

Figure 6: Learned traversability as in Figure 5, but for supervised MoG classifier. *Upper row:* Training set, with blue for traversable and red for not (left); mixture models for $p_0(v)$ (center) and $p_1(v)$ (right). These plots are projections of RGB features into R − G (abscissa) and G−B (ordinate) coordinates. *Lower row:* rectified RGB image (left); image-plane elevation training data (center); learned classification (right).

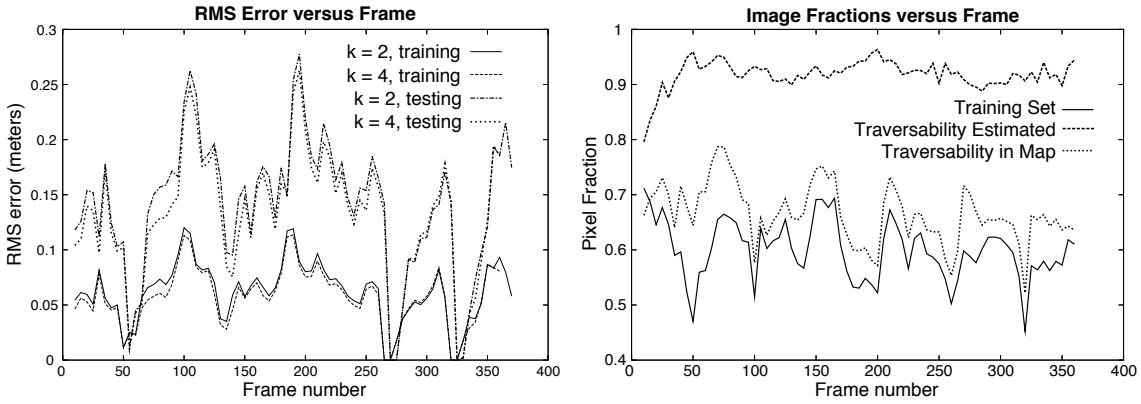

Figure 7: Learning across many frames. *Left:* RMS error on the training set for the k-means regressor ($k = 2$, $k = 4$), and on a test set disjoint from the training set. Error is insensitive to k. *Right:* Fraction of image pixels in the training set (solid line), fraction for which a traversability classification is known (dashed line), and fraction for which both classification and range are available (dotted line). Many sites have known traversability, but cannot be placed into the map.

allow traversability proxies derived from geometry (the near-field training set) to override traversability estimates inferred from appearance (the mid- and far-field query set). To project pixels from the image into the map, when the pixel has a non-zero disparity (near- and mid-field), 3D locations are computed by triangulation. Because of range uncertainty, this leads to maps with more blur with increasing range; at present, this is unavoidable. When disparity is zero (far-field), pixels can in principle be projected onto a nominal ground plane; currently, we ignore these pixels.

Figure 7 (right) shows the relative contribution of these three pixel classes, plotted as a function of frame number over a single trial. From a training set containing around 50% of the image pixels, the k-means algorithm is able to regress over 90% of the image. Of these pixels, however, only 60% have range data — the others cannot be projected into the map. Indeed, the overall improvement is modest when measured in the image. Because the learned pixels are farther away, this small number of mapped traversabilities translates to a significant improvement in the effective sensor range (between 50–100%).

Nevertheless, given the huge difference between the number of pixels regressed and the number of regressed values projected into the map, it is clear that the map-based navigation paradigm cannot fully exploit the results of image-based learning.

4.4 LAGR Test 7 results

A close variant of the k-means Lf3D algorithm described above was used in Test 7 of the LAGR program. This test was carefully designed such that near-sighted behavior would lead the robot into a maze-like area of scrubby brush, while a far-sighted robot would take the obvious (and much shorter) path to the goal. Figure 8 shows the view from the start of the

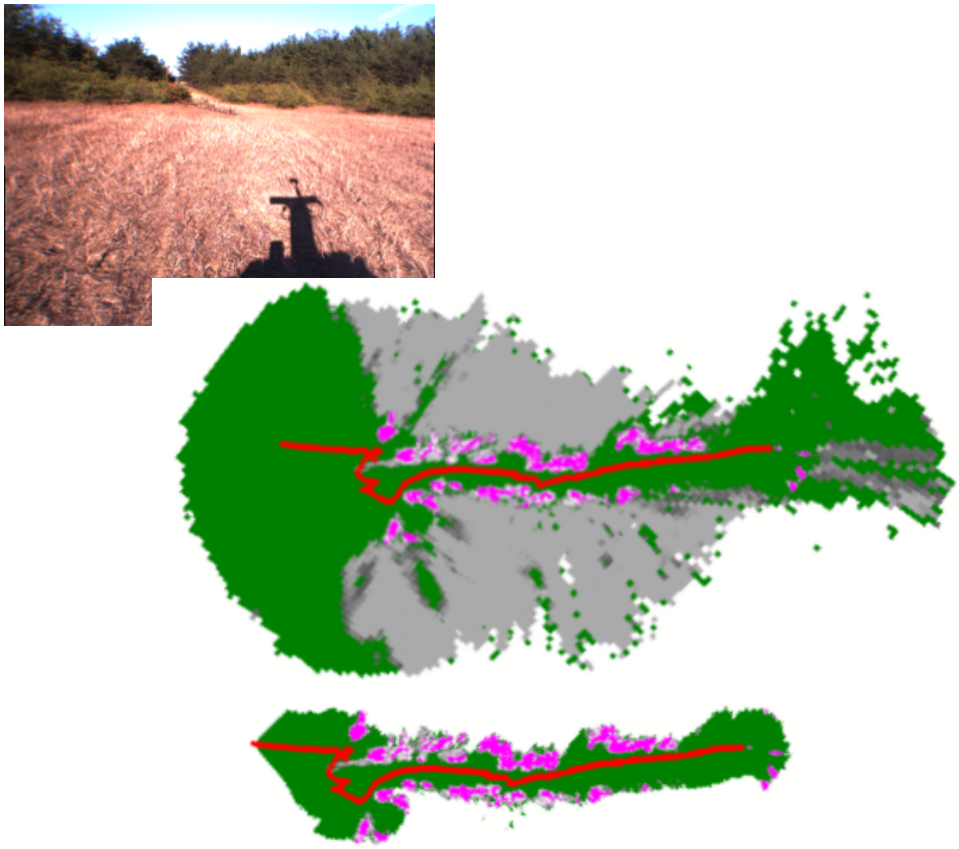

Figure 8: Top: LAGR Test 7, robot view from the start location. The obvious path leads to the goal at the top of the rise; a shortest-distance path takes the robot into the scrubby terrain on the left. Middle: Cost map generated over a successful trial using Lf3D. Green cells are traversable, purple are marked lethal, gray are intermediate. The red line denotes the path of the robot. Bottom: Corresponding map generated using geometry data only.

test course, along with the maps generated with and without Lf3D. Clearly, Lf3D extends the effective range significantly: the first map shows the incorrect route (though scrubby terrain) as a cul-de-sac, whereas the non-Lf3D map provides no information. Unfortunately, these maps also highlight one of the weaknesses of the approach as currently implemented. In order to learn that green bushes are nontraversable, the robot must first acquire some examples to train on. Since there are no bushes at the start of the course, the robot drove to within a few meters of the first rank of bushes before recognizing them as obstacles, turning around, and taking the correct route to the goal.

5 Discussion

In Section 2, we laid out a conceptual framework for learning to extrapolate traversability knowledge from underfoot to the far field. This progression exploits correlations among sensor modalities, using rich sensor inputs closer to the robot to enhance the interpretation

of poorer sensor data far from the robot. Sections 3 and 4 showed some initial steps toward instantiating that framework and described how these steps have been tested on the LAGR platform. There is still a very long way to go to fully develop this framework and establish its value experimentally, but we believe that our results to date justify pursuing this path.

A number of thorny system and infrastructure-related problems arose in both the LAGR and MTP programs that had to be solved before progress could be made on learning. Localization and data registration are key issues that impact any effort to attempt to learn by associating perceptions at one point in time with experience at another point in time, such as in LfP. Localization problems we encountered include wheel slip and GPS jumps that corrupted position estimates, time-stamping latencies that introduced attitude errors between IMU coordinate frames and image pose stamps, and stereo camera calibration errors that caused map misalignments even with perfect vehicle state knowledge. Improving solutions to these problems dramatically helped to come to grips with the learning issues; we will not elaborate on those solutions here, and only note that this was a major effort in itself.

Characteristics of stereo vision as a range sensor also have important implications for off-road perception on Earth. Specifically, stereo vision currently fails to produce range data for sparse, nearby vegetation, and it does not "penetrate" vegetation even to the limited degree that ladar does (Matthies, 2003). This put some limits what could be done with the voxel density-based terrain representation we used for LfP. This situation will be improved as real-time stereo vision progresses to higher resolution cameras and to correlation algorithms that are more tolerant of range discontinuities. Although we remain convinced that learning the traversability of different types of vegetation is a key open problem for off-road robotics, such terrain has not been the focus of the LAGR program to date, so our efforts in this direction have been constrained by the need to address other priorities.

For navigation on Mars, we are currently investigating a different form of propriceptive learning that models the relationship between wheel slip, surface type, and slope (Angelova, 2006a; Angelova, 2006b; Angelova, 2006c). Slip learning fits into the proprioceptive learning framework of Section 2: At training time, the terrain geometry and appearance of pixels within a map cell (measured from stereo imagery) is correlated to the traversability proxy, the robot's slip as it traverses the cell. At query time, slope and appearance alone are used to estimate slip. Of course, since sandy, muddy, and other sorts of slippery terrain exist in Earth-based outdoor applications as well, it would be desirable to integrate this type of learning into the robots described in this paper.

Lf3D has proven to be quite successful at extrapolating traversability information beyond the range of the local map for the kind of terrain we have faced so far in the LAGR program. Nevertheless, there are still key open issues in how to use the extrapolated information effectively for path planning. As range increases, the number of image pixels per map cell, for fixed size map cells, still decreases rapidly, so different terrain representations may be appropriate for planning than Cartesian maps with fixed cell size. A number of other possibilities exist, including obvious candidates that have been explored in the past, like multi-resolution Cartesian maps. Also, we have only scratched the surface on visual features that could be used in Lf3D-like strategies; key topics for future work include exploring

texture features and designing features that are invariant to changes in illumination and range — not to mention season. The present work sidesteps these invariance issues by using on-line learning and classification, but this entails forgetting useful past examples.

Other areas for future work include blame attribution, traversability confidence assessment, joint estimation of traversability across all range regimes, and strategic navigation, that is, choosing to push an obstacle across a regime boundary to gain appearance or geometric information about it.

Acknowledgements

The research described here was carried out by the Jet Propulsion Laboratory, California Institute of Technology, with funding from the DARPA LAGR and NASA MTP programs. The authors thank Nathan Koenig for data collection and the rest of the JPL LAGR team.

References

Angelova, A., Matthies, L., Helmick, D., Sibley, G., & Perona, P. (2006a). Learning to predict slip for ground robots. In *IEEE International Conf. Robotics and Automation*.

Angelova, A., Matthies, L., Helmick, D., & Perona, P. (2006b). Slip prediction using visual information. In *Robotics: Science and Systems Conference*.

Angelova, A., Matthies, L., Helmick, D., & Perona, P. (2006c). Learning and prediction of slip from visual information. Technical report to appear in *Journal of Field Robotics*.

Bapna, D., Rollins, E., Murphy, J., Maimone, M., Whittaker, W., & Wettergreen, D. (1998). The Atacama desert trek: Outcomes. In *IEEE International Conf. Robotics and Automation*.

Biesiadecki, J., Baumgartner, E., Bonitz, R., Cooper, B., Hartman, F. & Leger, P. (2005). Mars Exploration Rover surface operations: Driving Opportunity at Meridiani Planum. In *IEEE Conf. Systems, Man, and Cybernetics*.

Bodt, B. & Camden, R. (2004). Technology readiness level six and autonomous mobility. In *Proc. SPIE Vol. 5422: Unmanned Ground Vehicle Technology VI*, 302–313.

Dickmanns, E. and Mysliwetz, B. (1992). Recursive 3-D road and relative ego state recognition. *IEEE Trans. Patt. Analysis and Mach. Intell.*, 14(2):199–213.

Kim, D., Sun, J., Oh, S., Rehg, J., and Bobick, A. (2006). Traversability classification using unsupervised on-line visual learning for outdoor robot navigation. In *IEEE International Conf. Robotics and Automation*.

Krotkov, E., Fish, S., Jackel, L., McBride, B., Pershbacher, M., & Pippine, J. (2006). The DARPA PerceptOR evaluation experiments. *Autonomous Robots* (in press).

Lacaze, A., Murphy, K., & DelGiorno, M. (2002). Autonomous mobility for the Demo III experimental unmanned vehicles. In *AUVSI Conf. on Unmanned Vehicles*.

Leger, P., Trebi-Ollennu, A., Wright, J., Maxwell, S., Bonitz, R., & Biesiadecki, J. (2005). Mars Exploration Rover surface operations: Driving Spirit at Gusev Crater. In *IEEE Intl. Conf. Systems, Man, and Cybernetics*.

Maimone, M., Biesiadecki, J., Tunstel, E., Cheng, Y., & Leger, C. (2006). Surface navigation and mobility intelligence on the Mars Exploration Rovers. In *Intelligence for Space Robotics*, TSI Press, Albuquerque, NM.

Matthies, L., Bergh, C., Castano, A., Macedo, J., & Manduchi, R. (2003). Obstacle detection in foliage with ladar and radar. In *Proc. 11th International Symposium of Robotics Research*, Siena, Italy.

McLachlan, G., Peel, D. (2000). *Finite Mixture Models*. Wiley.

NRC, 2002. *Technology Development for Army Unmanned Ground Vehicles*. The National Academies Press.

Pomerleau, D. (1989). ALVINN: An autonomous land vehicle in a neural network. In *Proc. Conf. Neural Information Processing Systems 1*, 305–313. Morgan-Kaufmann.

Pomerleau, D., Jochem, T. (1996). Rapidly adapting machine vision for automated vehicle steering. *IEEE Expert: Special Issue on Intelligent Systems and their Applications*, 11(2):19–27.

Schölkopf, C., Mika, S., Burges, C., Knirsch, Ph., Müller, K., Rätsch, G., & Smola, A. (1999). Input space versus feature space in kernel-based methods. *IEEE Trans. Neural Networks*, 10(5):1000–1017.

Sofman, B., Bagnell, J., Stentz, A., & Vandapel, N. (2006). Terrain classification from aerial data to support ground vehicle navigation. Tech. report CMU-RI-TR-05-39, Robotics Institute, Carnegie Mellon University.

Stentz, A. & Hebert, M. (1995). A complete navigation system for goal acquisition in unknown environments. In *Proc. IEEE/RSJ International Conference on Intelligent Robotic Systems (IROS)*.

Valiant, L. G. (1984). A theory of the learnable. *Comm. Assoc. Comput. Mach.*, 27(11):1134–1142.

Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S. (2001). Constrained K-means clustering with background knowledge. In *Proc. Intl. Conf. Machine Learning*.

Wellington, C., Courville, A., & Stentz, A. (2005). Interacting Markov Random Fields for simultaneous terrain modeling and obstacle detection. In *Robotics: Science and Systems Conference*.