Journal of Machine Learning Research 1 (2005) 1-48 Submitted 4/00; Published 10/00

Learning for Autonomous Navigation

Anelia Angelova, Andrew Howard, Larry Matthies,
Benyang Tang, Michael Turmon FIRSTNAME.LASTNAME@JPL.NASA.GOV
Jet Propulsion Laboratory
California Institute of Technology, Pasadena, CA, 91109, USA

Eric Mjolsness EMJ@ICS.UCI.EDU
Department of Information and Computer Science
University of California, Irvine, CA, 92697, USA

Editor: Jane Mulligan, Greg Grudic

Abstract

1. Introduction

Robotic ground vehicles for outdoor applications have achieved some remarkable successes, notably in autonomous highway following (Dickmanns, 1987), planetary exploration (?), and off-road navigation on Earth (?). Nevertheless, major challenges remain to enable reliable, high-speed, autonomous navigation in a wide variety of complex, off-road terrain.

3-D 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. Prospects for extending the range of 3-D sensors is strongly limited by sensor physics, eye safety of lasers, and related issues. Range sensor limitations also allow vehicles to enter large cul-de-sacs even at low speed, leading to long detours. 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 got 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 non-traversable 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.

Learning may alleviate these limitations. If 3-D geometric properties of obstacle vs. non-obstacle terrain are strongly correlated with appearance in 2-D imagery (eg. color and texture), then it may be possible to use close-range 3-D analysis to learn to predict the traversability of terrain beyond 3-D sensing range based only on its appearance in imagery. To have a simple moniker for this strategy, we call it *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, from slip measurements produced by wheel encoders and other sensors, and from roughness measurements produced by gyros and accelerometers in the robots

©2005 J. P. L. Author et al..

Figure 1: LAGR vehicle (left) and Rocky 8 (right).

inertial measurement unit (IMU). Recording associations between such low-level feedback and visual appearance may allow learning to predict mechanical properties from visual appearance alone; we call this strategy *learning from proprioception* (LfP). While clustering, neural nets, and related algorithms with a learning component have long been used for image classification in the context of terrain typing (Manduchi, 2005), (Campbell, 1997) and road-following (Rasmussen, 2002), learning specifically to assess terrain traversability for ground robots has only begun to be addressed very recently (Wellington, 2005).

2. Architectures and issues

This paper is intended both as a position paper to outline what we see as some key issues and approaches to learning for off-road navigation and as a report on initial results for those approaches. We describe work in the DARPA-funded Learning Applied to Ground Robotics (LAGR) program and the NASA-funded Mars Technology Program (MTP). Architecturally, these programs have a great deal in common, in that both use wheeled robotic vehicles with stereo vision as the primary 3-D sensor, augmented by an IMU, wheel encoders, and in LAGR, GPS; plus, they use closely related software architectures for autonomous navigation. In this section we describe these architectures and how they need to change to address Lf3D and LfP, and key engineering issues that are critical to reducing these ideas to practice.

The pipeline in Figure 1 is a simplified illustration of the baseline navigation software architecture used by both of these vehicles. 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 to 10 m in front of the vehicle, depending on camera height and resolution. Geometry-based traversability analysis heuristics are used to produce grid-based 'traversability cost' maps over the same 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 to send 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

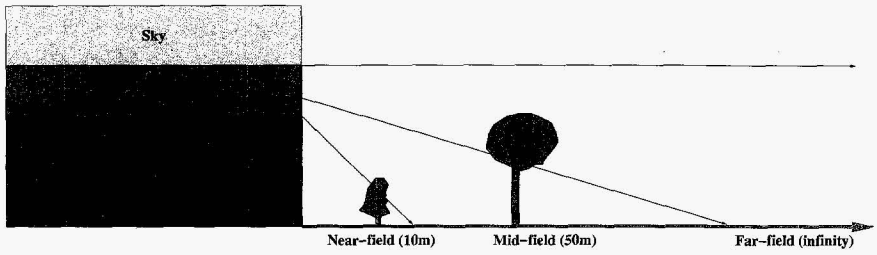

Figure 2: The learning from proprioception idea. ????

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.

Figure 2 gives a deeper look at information available from stereo vision and how this relates to Lf3D, LfP, and richer local map representations. In the image on the left, region A represents where stereo vision gets range data of sufficient resolution to build a local elevation map; color and texture information (collectively, 'appearance' is also available for insertion in the map. Region B represents where the range data, even if it is available at every image pixel, samples the ground too sparsely to create a useful elevation map; image appearance is also available. Region C is beyond the range of stereo vision (i. e. zero disparity), so only image appearance is available. Lf3D strategies as we conceive them attempt to use the association of geometry and appearance in A to learn how to estimate traversability in B and C. To make things concrete, in LAGR, A is about 68% of the image, B is 9%, and C is 23%; conversely, on the ground plane, A extends to about 10 m, B from 10 to 50 m, and C from 50 m to infinity (right side of Figure 3). Thus, Lf3D attempts to make inferences about small parts of image space (B, C) that represent very large parts of map space. Section 3 describes our initial approach to this problem.

The inset at the bottom of the image in Figure 2 represents information available from the proprioceptive sensors - the bumper, IMU, and wheel encoders. LfP adds this to the local map as the robot drives over terrain. Initially, we have used this to learn an improved traversability assessment function for regions in the local map ahead of the vehicle (ie. A), replacing a hand-coded traversability heuristic based only on geometry. Section 4 describes one version of this aimed at vegetated terrain in the LAGR program; Section 5 describes a different approach aimed at slippery terrain on Mars. Our ultimate goal is to develop a theoretical framework for estimating terrain traversability T that unifies the Lf3D and LfP concepts and encompasses slippage, sinkage, and obstacle compliance.

Since LfP associates what the robot feels under its wheels and on its bumper now with what it saw ahead of it sometime in the past, accurate robot position and heading estimation ('localization') is critical to this concept and a weakness of it. We have experienced considerable wheel slip in both LAGR and MTP, which significantly degrades localization. Visual feature tracking with the stereo cameras ('Visual Odometry' (VO) (Matthies, 1987)) can in many cases produce better short-term localization performance than the other, traditional state estimation sensors onboard; it can also be used to get an estimate of slip. We are using VO for these purposes in both programs. Another critical issue in this whole

problem domain is invariance of visual features to range, lighting, and weather conditions; to date, we have treated this only with very simple color space normalizations

Proprioceptive proxies for traversability.

Traversability T is a random variable ($\in [0,1]$) associated with a certain cell c of the map. Depending on context, we may take $T(c)$ to be binary (0/1) or real-valued. For example, we can think of $T(c)$ as the probability that the robot can successfully move out of cell s after deciding to do so. We could imagine a physics-based simulation model which would determine this 'self-transition probability' given vehicle and terrain parameters. Accumulating this T over a path would then tell us the cumulative probability of a successful sequence of moves. Of course, lacking such a model, we consider T as a random variable which is to be learned. Our learning strategy is to use the 'higher-quality' input examples to produce training labels $\tilde{T}$, which serve as proxies for the unknown T. The proxy labels are given to a learning algorithm which trains a regression model $\hat{T}$ that approximates $\tilde{T}$. The regression model is then used to drive the robot. In Lf3D, map elevation statistics $\{h\}$ are used to provide the proxy $\tilde{T}$, which is estimated using appearance information (normalized color in our case). In LfP, the proprioceptive inputs (such as bumper hits or slip) are used to generate the proxy traversability index, which is then estimated using the available appearance and geometry information from stereo images.

3. Learning of traversability from proprioception

The key goal in this work area is to learn which obstacles the robot can drive through and which are impassable. Our motivation is scenes with stands of pliant vegetation that can be pushed aside, as well as brush that is stiff enough to stop the robot. The threshold of traversability depends on many factors and must in effect be learned. As we are interested in terrain traversability, our current proprioceptive proxy $\tilde{T}$ is therefore a 0/1 quantity.

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; a constrained clustering approach may be sufficient to separate these two classes. Rather than solving the blame attribution problem directly, initially we obtained training data from hand-labeled image sequences: a human identifies a mix of traversable and untraversable map cells.

Terrain representation

Our terrain representation attempts to capture some of the mechanical properties of the environment, including the notion of *penetrable* versus *impenetrable* vegetation. The space around the robot is represented by a regular three-dimensional grid, with a *density* estimate associated with each voxel; intuitively, we expect that only low-density voxels will be penetrable (see Figure 3). The voxel density grid is constructed from range images (such as those obtained from stereo vision or LIDAR) using a simple ray-tracing algorithm: 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 voxel density ρ is given by the ratio of hits to passes. Since the ground may be non-planar, we also identify a *ground voxel* in each vertical 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.

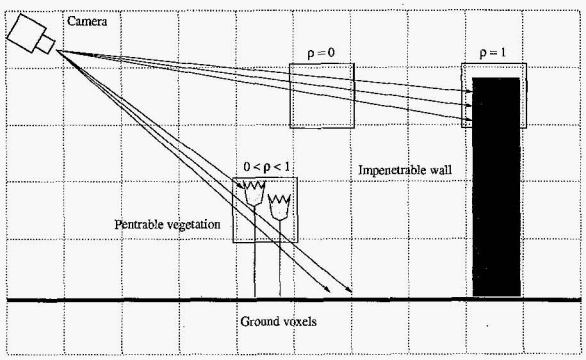

Figure 3: Side view of the voxel density map. For each voxel, we store the number of *hits* (rays that terminate in the voxel) and the number of *passes* (rays that pass through the voxel); the voxel density ρ is given by the hit/pass ratio. Three types of voxels are highlighted in this figure: empty space ($\rho = 0$), an impenetrable wall ($\rho = 1$), and penetrable vegetation ($0 < \rho < 1$).

Each map cell c has an above-ground density column $[\rho_1\, \rho_2 \ldots \rho_H]$ where H is a function of c. For simplicity we prefer to work with fixed-dimension feature vectors. Accordingly, we work with these features: ρ^*, the maximum density; ρ^{**}, the next-highest; h^*, the height of ρ^*; and h^{**}. We also use p^*, the number of passes associated with the density ρ^*, and an indication of its confidence. The average color values associated with a map cell would also be a good feature, but we have not experimented with this.

Learning algorithm

We are using Support Vector Machine (SVM) learned offline. The SVM uses a radial basis function kernel. We apply cross-validation to tune its parameters ($\gamma = 10$, $C = 1$ are the best parameters selected). The training data consists of 2000 traversible and 2000 non-traversible examples. The resulting model has 1381 support vectors. Tests are performed on an independent image sequence which contains roughly 2000 examples. We achieved a classification accuracy of 14% on the test set. We are encoding the SVM output into a lookup table (LUT) for speed of classification and simplicity, but a reduced-set SVM would be easy to substitute. SVM traversability classification results are incorporated into the planner as a cost based on the margin to the decision boundary (????). The results of SVM classification and the LUT are shown on Figure 4.

4. Learning of traversability from 3-D geometry

In this section we use appearance information from the image plane to correlate to the correspondent geometry informaion (height) in the map available at close ranges so that to predict the average height at far range, i.e. we use the geometry as a traversability proxy for learning (the Lf3D scenario). In general, in Lf3D, we use the locations of pixels to compute a traversability proxy $\tilde{T}$, which is linked to the pixel appearance u. In this case $\tilde{T}$ is a

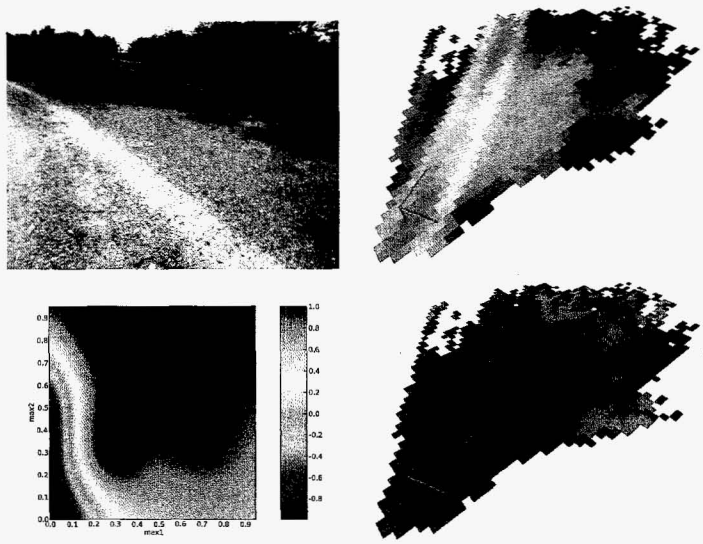

Figure 4: SVM learning results. (a) and (b) Sample camera image and its projection into a 2D local map (each cell shows the average color of all pixels that project into a voxel column). (c) Cost lookup-table generated through off-line training. (d) Cost map infered from the voxel density grid; green cells are traversable, red cells are non-traversable.

function of the heights of all pixels landing in one $(0.2\text{m})^2$ map cell. When at least 10 pixels land in one cell, their height standard deviation becomes resolvable: a large value indicates rough ground or obstacles. The standard deviation z can then be used as a heuristics to compute a traversability proxy in $[0, 1]$ for the cell, which is associated with the appearance u of all pixels mapping to that cell.

Currently, we use only two appearance-based features: the average R, G, B color within a map cell, normalized to sum to unity, leaving two degrees of freedom (the normalized R and G are used). Training data for the experiments described below was gathered from 10 consecutive frames (~4500 training examples). We consider below two approaches based on unsupervised clustering and on supervised Mixture of Gaussians (MoG).

4.1 Unsupervised K-means

As the geometry based proxy is based on heuristics (and therefore not absolutely reliable), the first approach we take is of unsupervised clustering of the input data and deducing the traversability from the height standard deviation z within each cluster. We perform a K-means clustering with $K = 5$ (Figure 5). For each datapoint, the resultant traversability cost is evaluated, as follows:

$$z(u_i) = \sum_{k=1}^{K} \bar{z}_k e^{-r_{i,k}^2/\sigma^2} / \sum_{k=1}^{K} e^{-r_{i,k}^2/\sigma^2},$$

where $\bar{z}_k$ is the average standard deviation value per cluster and $r_{i,k}$ is the distance of the point to the center of the cluster. In other words, the z value of a point is an interpolation

of the average z values in each cluster using the distances of this point to the centers of the K cluasters. A threshold for classification z_{thr} is set, so that 65% of the data is traversible. Points with $z \leq z_{thr}$ are considered as traversable, and otherwise as non-traversable. The parameter σ is selected using cross-validation. Clssification accuracy is 7%, for $\sigma = 0.5$ (we are reporting accuracy on the same sequences as the idea is to extrapolate colors corresponding to the near and mid-field to the far field in the same image).

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 EM algorithm in a semi-supervised framework (McLachlan, 2000) and adopting a two-class discriminant-based approach with populations determined by $\tilde{T}$.

In the discriminant approach, rough thresholds based on $\tilde{T}$ are used to form sets of positive and negative examples: $\mathcal{T}_0 = \{(u, z) : z \leq z_{thr}\}$, $\mathcal{T}_1 = \{(u, z) : z > z_{thr}\}$. We selected a value of z_{thr} of 0.4 so that 65% of the training data belongs to the traversable class. It is also conceivable to define a 'borderline' traversability class by using two thresholds. Two separate MoGs $p_0(u)$ and $p_1(u)$ are fit to the two training sets, and we declare a pixel traversable if $\log p_0(u)/p_1(u) < L$. Confidence is related to $\lambda p_0(u) + (1 - \lambda)p_1(u)$ where $\lambda = \Pr(T = 0)$, ($\lambda = 0.35$ for our dataset) (do we need the confidence???). We have used K-component ($K = 5$ for each class), full-covariance Gaussian mixtures to parameterize the two distributions, and fit the parameters by maximum-likelihood using the EM algorithm (Figure 5). The prior probability is set at 65% for traversible and 35% for nontraversible. The test error achieved is 6%. Training time for $N = 1000$ 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.

Note that both approaches classify terrain in the *image* space; i.e., each pixel in each frame is assigned a value for the traversability proxy $\tilde{T}$. In order to use this result for navigation, these values must then be projected back into the map. This presents us with two issues: how to combine traversability data from different sources, and how to determine the 3D location of 2D image points. For data fusion, we currently use a simple hierarchical scheme, in which traversability measures derived from geometry (the training set) take priority over traversability measures infered from appearance (the query set). Two methods are used for projecting pixels from the image into the map: when the pixel has a non-zero disparity value, 3D locations are computed from triangulation; when no disparity is available, pixels are projected onto a nominal ground plane. Both forms of projection are somewhat problematic: triangulation leads to large range errors when disparity values are small (i.e., when objects are far away), and the environment may not be planar. Thus, the question of how best to use classified images for navigation remains open.

The results of both algorithms are shown on Figure **??**. How are the results incorporated in the planner??? The main limitation of these methods at present is the limited information content of the appearance features used. Introducing local texture measures

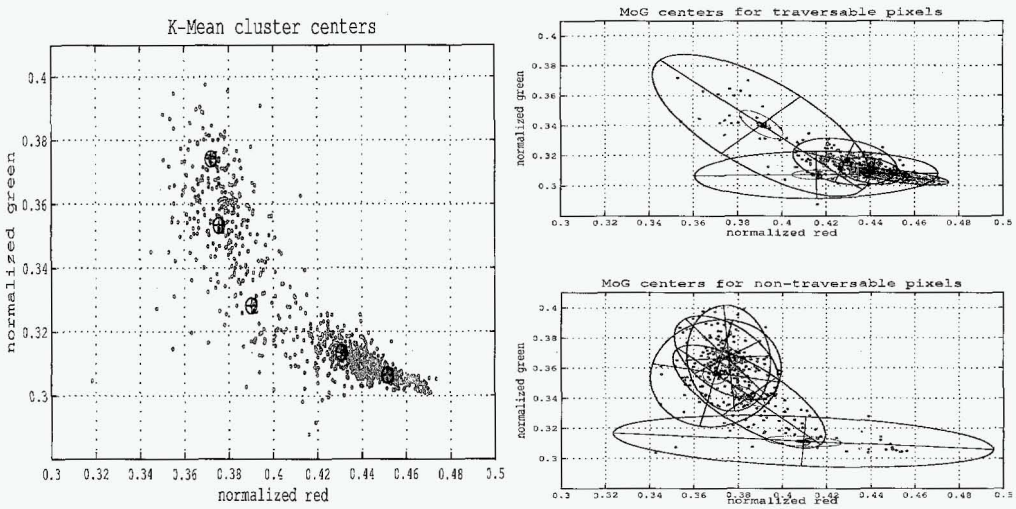

Figure 5: K-means (left) and MoG (right) algorithms. (??? how to describe this)

should substantially improve classification accuracy and robustness, at which point new algorithmic tradeoffs could be evaluated. It remains to be seen which of the two approaches would be more successful for navigation after applying learning from proprioception (for example, as described in Section 3) to learn the traversability proxy rather than using cost heuristics to estimate it.

5. Learning of slip from proprioception

Slip is a measure of lack of progress of the vehicle and can be defined as the difference between the commanded velocity (estimated by the vehicle's kinematics) and the actual velocity (estimated in our case using VO). As mentioned above, considering slip in the total traversability cost is important because the robot's mobility on certain terrains can significantly degrade, especially as the slope angle increases (Lindemann, 2005). So, it would be cost-effective to predict the amount of slip before entering a particular terrain.

Slip learning is a typical example of the abovementioned proprioceptive learning framework: information about the terrain geometry and appearance at some location is collected from the stereo imagery and is correlated to the rover's slip (a propriocetive measurement) when the rover traverses this location. Both appearance and geometry are needed, as slip is a function of terrain slopes, but also depends on the terrain type traversed (Bekker, 1969).

Slip learning framework

Slip is learned in a Mixture of Experts framework in which the terrain type is classified first using appearance information and then the slip, as a function of terrain slopes, is learned (Angelova, 2006). The rationale for doing that is: 1) one can safely assume that terrain type and appearance are independent of the terrain slopes; 2) introducing structure in the problem helps learning in very high dimensional spaces which would require a lot of training data. In this paper we describe mainly the problem of learning slip as a function

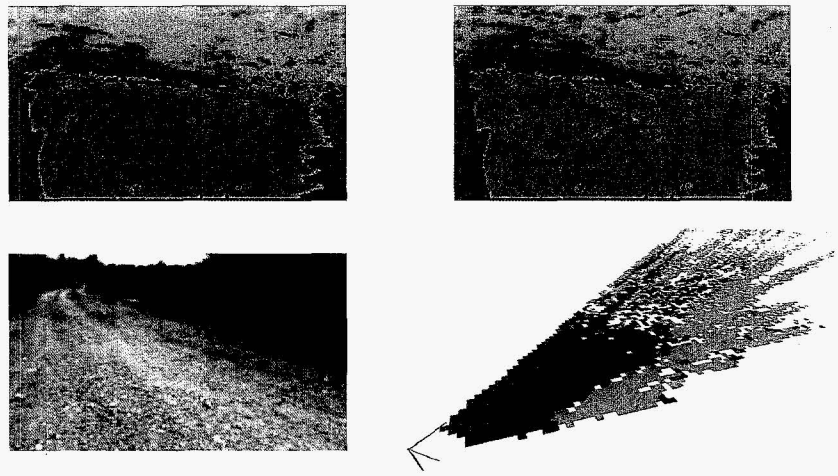

Figure 6: K-means (top left) and MoG (top right) classifications using elevation-based traversability proxy. Input image (bottom left) and the k-means results re-projected into the map (bottom right); note that this map extends out to around 45 m (the effective limit for stereo disparity). Pixels from the training set are shown in green (traversable) and red (non-traversable). Classified pixels are shown in blue (traversable) and magenta (non-traversable).

of slopes when the terrain type is known. We give also a brief outline of the terrain type classification algorithm, which is a subject of our current work.

As slip is a nonlinear function of terrain slopes, we use the Locally Weighted Projective Regression method (Vijayakumar, 2005). We prefer it to other nonlinear approximation methods, such as Neural Networks, because it can be easily extended to online learning. The slip S is approximated as follows:

$$\hat{S}(\mathbf{x}) = \sum_c^C K(\mathbf{x}, \mathbf{x}_c)(b_0^c + \sum_{i=1}^r b_i^c \langle \mathbf{p}_i^c, \mathbf{x} \rangle)$$

where $\mathbf{x}$ are the input terrain slopes, $K(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{\lambda})$ is a smoothing kernel, $\mathbf{x}_c$ are the centers of local neighborhoods, called receptive fields, where local linear regression models are evaluated, r is the number of linear projections, here $r \leq 2$. The learning proceeds by assigning receptive fields to cover populated regions of the input domain and then estimating a local linear regression (i.e. estimating the parameters b_i^c, $\mathbf{p}_i^c$) in each receptive field weighting all training point with their correspondent weights to the receptive field center $\mathbf{x}_c$, thus giving more weights to point within or closer to the receptive field (see Angelova, 2006, for details). The parameter λ, which defines the receptive field size, is selected using a validation set, and varies dependent on the dataset.

We use a 2D cell map representation (a cell is of size 0.2m x 0.2m; the map has a 15m look-ahead). The average elevation per cell is used to compute a best plane fit for a location (a 6x6 neighborhood of cells centered at the location's cell). The slope itself is decomposed into longitudinal (along the forward motion direction) and lateral (perpendicular to the

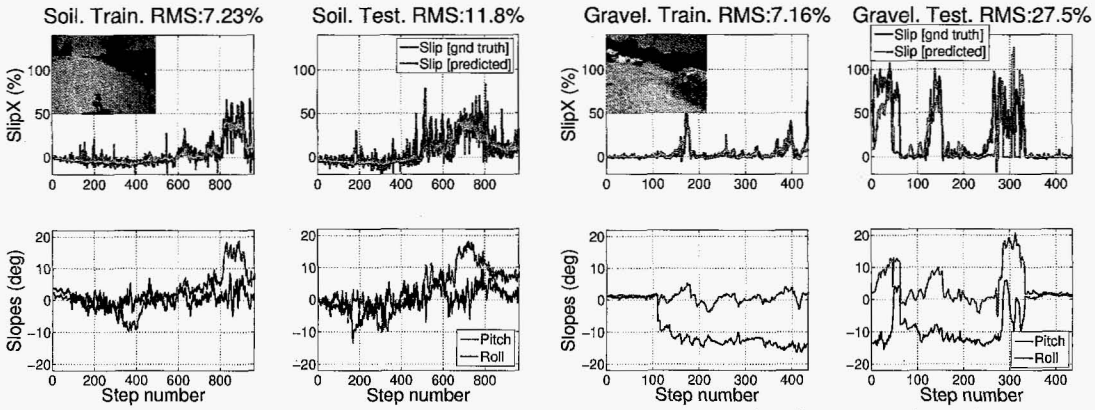

Figure 7: Learning slip as a function of slopes: soil (left), gravel (right).

Figure 8: Example terrain classification results: sand, soil, and gravel.

forward motion) components with respect to the rover's position, referred to as pitch and roll, since they correspond to the pitch and roll of the vehicle. VO is used for localization and the vehicle's attitude (received from the IMU) gives an initial gravity leveled frame to retrieve correct longitudinal and lateral slope angles from the terrain (Angelova, 2006).

Figure 7 shows learning and prediction of slip in X (along the forward motion direction) as a function of slopes for soil and gravel terrains (Angelova, 2006). Training is performed on the first portion of the traverse, testing - on the second. The average Root Means Squared (RMS) error is provided on top of each figure (slip is normalized by the average velocity per step to get the results in %). These results are very promising given the amount of noise involved and the learning scenario selected (notice that there are combination of roll and pitch angles for the gravel data which were not seen during training, which makes it much harder to generalize). Figure 8 shows preliminary terrain classification results using a texton-based approach (Varma, 2005) performed in the image plane. Despite some classification errors, the method is successful in discriminating visually similar terrains at close range which serves well the purposes of slip prediction. For now, the system is working offline, but we are exploring methods to speed up the terrain classification algorithm and integrate it into the whole navigation system.

6. Discussion

Acknowledgments

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology with funding from the DARPA Learning Applied to Ground Robotics program and from NASA Mars Technology Program. We thank the LAGR team at JPL and Nathan Koenig for data collection.

References

Angelova, A., Matthies, L., Helmick, D., Sibley, G., Perona, P., Learning to predict slip for ground robots, *submitted to ICRA*, 2006

Bekker, M., *Introduction to terrain-vehicle systems*, Univ. of Michigan Press, 1969

Dickmanns, E., Zapp, A., Autonomous high speed road vehicle guidance by computer vision, *Proc. 10th IFAC World Congress*, Munich, 1987

Campbell, N., Thomas, B., Troscianko, T., Automatic segmentation and classification of outdoor images using neural networks, *International Journal of Neural Systems*, vol. 8 (1), 1997

Lindemann, R., Voorhees., C., Mars exploration rover mobility assembly design, test and performance, *IEEE International Conference on Systems, Man and Cybernetics*, 2005

McLachlan, G., Peel, D., *Finite Mixture Models*, John Wiley & Sons, 2000

Manduchi, R., Castano, A., Talukder, A., Matthies, L., Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation, *Autonomous Robots*, 19, 2005

Matthies, L., Schafer, S., Error modeling in stereo navigation, *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 3, June, 1987

Rasmussen, C., Combining laser range, color, and texture cues for autonomous road following, *ICRA*, Washington DC, 2002

Varma, M., Zisserman, A., A statistical approach to texture classification from single images, *Int. Journal of Computer Vision*, Vol. 62, 2005

Vijayakumar, S., D'Souza, A., Schaal, S., Incremental online learning in high dimensions, *Neural computation*, 2005

Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S., Constrained K-means clustering with background knowledge, *Proc. of the Int. Conf. on Machine Learning*, 2001

Wellington, C., Courville, A., Stentz, A., Interacting Markov Random Fields for simultaneous terrain modeling and obstacle detection, *Robotics Science and Systems*, 2005