Presented at 7th International Symposium
on Unmanned Untethered Submersible
Technology, June 1989.

Terrain Modeling for Autonomous Underwater Navigation[1]

Martial Hebert
The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh PA15213

Abstract

The main task of perception for autonomous vehicles is to build a representation of the observed environment in order to carry out a mission. In particular, terrain modeling, that is modeling the geometry of the environment observed by the vehicle's sensors, is crucial for autonomous underwater exploration. The purpose of this work is to analyze the components of the terrain modeling task, to investigate the algorithms and representations for this task, and to evaluate them in the context of real applications. Terrain representation is an issue that is of interest in many areas of mobile robotics, such as land vehicles, planetary explorers, etc. This paper surveys some of the ideas developed in those areas and their relevance to the underwater navigation problem. Terrain modeling is divided into three parts: structuring sensor data, extracting features, *and* merging and updating terrain models.

1 Introduction

An autonomous vehicle needs an internal representation of its environment. In the case of an underwater vehicle, a large part of the environment is the bottom terrain and discrete objects above it. A major part of the internal representation is therefore the geometric representation of the observed terrain. In this paper we review representations and algorithms developed for the representation of natural terrain and identify the issues to be investigated. The techniques discussed here come from all the fields of mobile robots for natural environments: underwater robots, land vehicles, and planetary explorers.

We view the terrain representation as logically divided into four layers (Fig. 1):

1. *Data:* The data layer is the only part of the map that is directly computed from sensor data. The algorithm that computes the data layer from sensor data must be carefully designed to include: interpolation between sparse data points to produce a dense map, uncertainty computation from a model of sensor noise, and explicit detection of occluded regions. Each cell of the data layer contains a data point and an uncertainty value. We always assume that the uncertainty can be represented by its variance σ. This information is not available everywhere due to the limited fields of view of the sensors and the possible self-occlusions of the terrain. In those regions, the data layer contains either a flag indicating that a location is outside of the field of view, or the best available estimate of the data if the point is occluded. Furthermore, additional data points may be stored at a given cell if one elevation is not sufficient to describe the local surface (such as in the case of overhanging objects).

2. *Local attributes:* Local attributes, such as local curvatures and surface normals are associated with each element of the data layer. These attributes are descriptors of the local shape of the terrain

[1]This research was sponsored in part by the Defense Advanced Research Projects Agency, DoD, through ARPA Order 5351, monitored by the US Army Engineer Topographic Laboratories under contract DACA76-85-C-0003, by the National Science Foundation contract DCR-8604199, by the Digital Equipment Corporation External Research Program, and by NASA grant NAGW-1175. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agencies.

around each map point. Even though a number of attributes could be computed, we will consider only the attributes that are needed for extracting terrain features.

3. *Features:* Local attributes are used to extract features, that is parts of the terrain that are of special interest such as high curvature points or sharp height discontinuities. Features are represented both by a list of attributes (location, shape, etc.), but also by the area they occupy on the base grid.

4. *Symbols:* If more knowledge about the environment is available, groups of features may be grouped into objects to which a semantic label can be attached. Building this layer requires specific interpretation techniques that depends on the type of knowledge to be used. We will not consider this aspect of the terrain representation in this paper.

We present the possible basic representations for the data layer in Section 2. The types of features used in terrain representation and the corresponding feature extraction algorithms are presented in Section 3. Once the terrain map representation is defined, algorithms have to be designed to *update* the map, that is to add new sensor data to an existing terrain map. This involves matching terrain representations to compute the their relative locations and the regions that are common to the maps, and merging maps into a composite map. The algorithms for terrain map updating are described in Section 4.

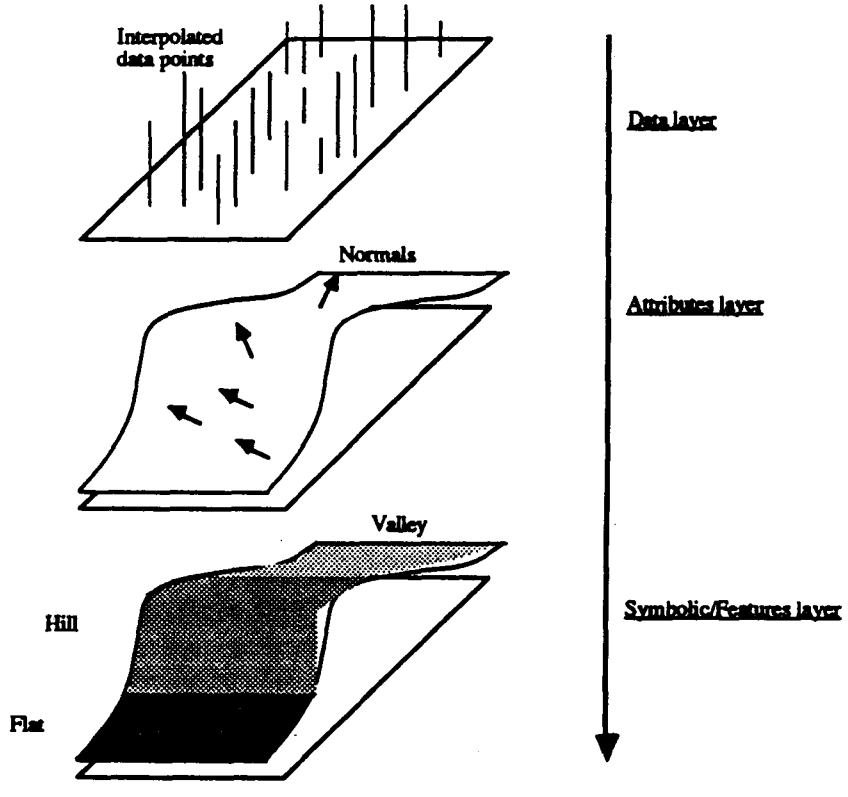

Figure 1: Levels of Representation

2 Basic data representations

The first problem to be solved is to organize the data coming from sensors into a basic representation so that it can be easily accessed, updated, and processed to extract relevant features of the environment. The

choice of the structure, or basic representation, is crucial since it determines the performance of all the other components that will be built upon it. We can divide the possible basic representations into three classes: *elevation maps*, *discrete surfaces*, and *probability maps*.

2.1 Elevation maps

Elevation maps are the most popular representation. In its simplest incarnation, an elevation map is a discrete map, each cell of which contains the elevation of the terrain. The map may be maintained at different resolutions, and may contain other attributes such as local slope or curvature. The main difficulty in building elevation maps is that the sensor data is initially sparse in the plane of the map thus it must be converted to a dense and regularly sampled set of points. The most common approach is to project the data points in the map and to interpolate across them. To get the best estimate of the elevation, a smoothing interpolator should be used, such as in [4]. The main problem with using an elevation map is that it depends on a resolution, the size of one cell, and on the direction of the reference plane of the map, as a result changing one of those two parameters requires recomputing the entire map. This occurs when two maps with different reference planes have to be merged, for example. A more recent technique [8] somewhat alleviates this problem by directly computing the elevation at each cell of the map instead of projecting the data points. In addition it provides a value for the uncertainty at each map point as well as an explicit representation of occluded regions. Due to their ease of use, the elevation maps have been used extensively as basic representations for robot path planning over rough terrain, either directly [4,2], or after feature extraction [17]. The two main issues in elevation map building are:

- *Random access:* Information must be retrievable from the map independently of the current position of the vehicle and sensor. Among other things, this implies that the map can be computed with respect to an arbitrary coordinate system and within a specific region of the world.
- *Uncertainty:* An uncertainty model is usually available in sensor space. It is a non trivial task to convert the sensor model to uncertainty values in map space.

2.2 Discrete surfaces

Discrete surfaces are graphs of irregularly spaced measured points that are embedded in the underlying observed continuous surface. Although several approaches can be used to build discrete surfaces, the most popular is by far the Delaunay triangulation which guarantees some properties of the graph such as equivalence between nearest-neighbor and neighbor in the graph. This representation has been used mainly in cases in which only very sparse data is available, or the spatial distribution of the measured data points is hard to predict. The domains of application include both underwater exploration [14] and terrestrial terrain mapping. Discrete surfaces are compact representations since they essentially do not contain anything more than the data points, they are intrinsic in that they are independent of a particular resolution or a particular reference system (as opposed to the other grid-based representations). Just like elevation maps, discrete surfaces can be used to interpolate a continuous surface through the data points. They are not usually the preferred representation, however, because they are difficult both to access and to modify.

2.3 Probability maps

A probability map is a general representation in which each cell of a discrete 2- or 3-D grid contains the probability that a surface point is present within the cell. High probability regions of the map therefore

indicate the presence of a surface in this region. The probabilities are derived from a model of the sensor. This type of representation has been applied both to data from indoor environments [13] and to underwater data [18]. Probability maps have several advantages: they are easy to manipulate, they can explicitly take into account a detailed sensor noise model, they allow for multiple sensors to be used simultaneously [11], and they require very little preprocessing of the sensor data. They are especially useful in the case of data from very low resolution sensors such as ultrasonic sensors. The main drawback, at least in the case of 3-D maps, is the size of the map and, consequently, the high computational cost of updating such a map. In addition, it is difficult to extract features from a probability map.

3 Feature extraction

By feature extraction we mean the identification of parts of the environment that are relevant to the mission at hand, *e.g.* discrete objects and their description, or local shape descriptors of a terrain. There are two ways to approach the feature extraction problem. We can either extract features from the sensor data, *e.g.* from a range image and include them in the terrain maps, or we can extract features directly from the basic representation of the terrain. The former approach is currently the most popular since it can make efficient use of existing feature extraction operators developed in the field of range data processing for example. The second approach, however, is very attractive since it computes the features in which we are really interested, that is the terrain features. Furthermore, this is consistent with the proposed layering of the map representation (Fig. 1) since the features are computed from the intermediate local attribute layer of the map. This is therefore a more natural approach since the relevant terrain features are defined in terms of terrain shapes, not in terms of their appearance in sensor data. This has also an important consequence: If changes in the sensor characteristics are needed, or if multiple sensors are used, we still use the same feature extractors independently of the sensors (Fig. 2), thus the considerable investment of developing feature extraction algorithms for a large number of possible sensors.

The first approach to feature extraction is to use local differential properties of the surface, surface normals and curvature, to extract regions or points of interest. This is a very natural idea since in theory those local properties entirely describe the shape of the surface. Several techniques have been developed to compute normals and curvatures both from regular discrete grids [16,1] and from discrete surfaces [14]. Surface normals and curvatures can be used to derive global features such as lines of curvatures, or surface patches of smooth curvature variations. Those features can be grouped into larger features with more semantic content. One example is to use the continuity of the surface normals and curvature to extract prominent objects. Another example is the extraction of regions of interest (ridges, valleys, peaks, etc.) form sonar data [3]. This approach to feature extraction is attractive since it the underlying theory is well defined, and the implementation of the curvature computations is straightforward. However, the approach relies on the surfaces being mathematically well-behaved (*i.e.* smooth), and on the data being accurate so that local high-order derivatives of the surface can be computed. This assumption may not hold when dealing with textured surfaces or with low-resolution sensors.

The second approach is to label the features as topographic events in the terrain such as hills or valleys that can describe any type of terrain [12]. A general topographic feature extraction algorithm is based on an analysis of the shape of the contour lines of the terrain: Given an elevation h_{ref}, the part of the terrain that is such that $h > h_{ref}$ is decomposed into several connected regions, each of which is bounded by a contour line. As h_{ref} varies, we obtain a set of cross-sections of the terrain, each of which is a set of separate connected regions. The regions are related from one cross-section to another in that the regions at h_1 are included into the regions at $h_2 < h_1$. Based on this inclusion relationship, the regions are organized in a tree, the topographic tree, that describe the topographic structure of the terrain. The regions at the nodes of the tree are the features. The nodes that are closer to the root of

the tree are the large scale features of the terrain, while the leaves of the tree are the finer features that can be detected. Therefore, this type of features is meaningful only in relation to a given scale. This technique is especially successful in extracting large-scale features such as large valleys or ridges that are otherwise difficult to segment out. Topographic features are typically used in higher level planning and as basic tokens for matching multiple terrain maps. Topographic features have been used to describe bottom undersea surface and open terrain for land navigation.

The third approach attempts to segment the observed surfaces into patches that can be closely approximated by simple mathematical surfaces such as planes, quadrics, or superquadrics. Several classical image processing techniques can be used such as region growing. This approach is often combined with the first approach in that differential properties, usually curvatures, are used to provide an initial segmentation of the surface that is refined by a region grouping algorithm [1]. This approach is attractive mainly because the resulting description is easy to manipulate since it is in terms of simple algebraic surfaces. However, a lot of information maybe lost in the process by oversimplifying the shape of the observed surfaces. One way to reduce this problem is to use high-order primitive surfaces such as superquadrics [15] that can describe more closely natural terrain and natural objects. This has been used for describing natural scenes for an autonomous land vehicle [7].

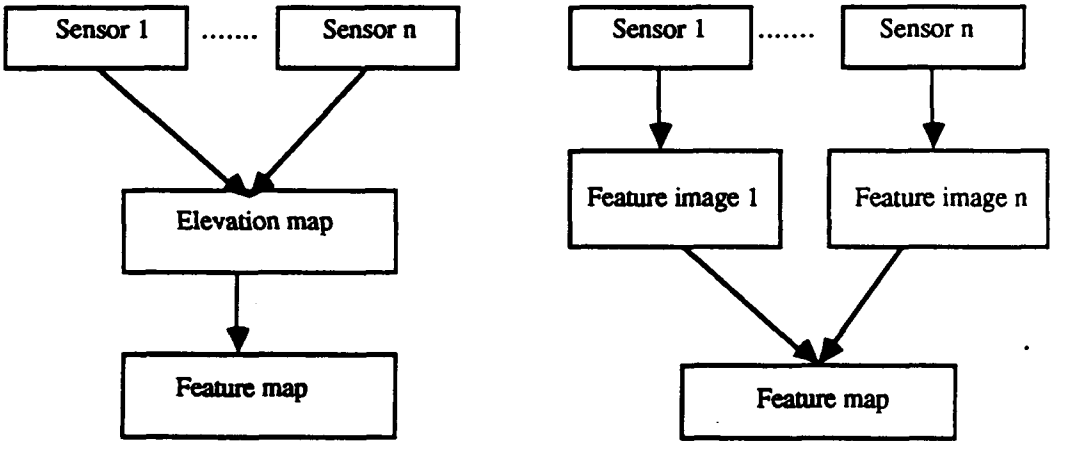

Figure 2: Two Approaches to Feature Extraction

The three approaches to feature extraction give us the basic tools for building the feature layer of a terrain representation. There are however several research issues that remain to be solved in order to build a robust terrain representation:

- *Identify the relevant features:* Many of the features that we can currently extract are tuned to particular applications. A set of features that constitutes a generic representation of terrains should be defined. The architecture of the set of features should be articulated around the large topographic features ("global" features) and the more local differential features.

- *Represent uncertainty:* The location and detection of the features is uncertain due to the nature of sensor data. Techniques for estimating and representing the uncertainties on the features must be developed. Although standard techniques can be used in the case of well defined geometric features (*e.g.* points corresponding to local maxima of the curvatures), it is more difficult to represent the uncertainty on "fuzzy" features such as the regions computed from the topographic tree.

- *Select the Scales:* There is a notion of scale associated with the features such as the level of the topographic tree or the amount of smoothing that is used in computing the differential features.

The choice of a proper scale at which features should be detected is guided by the application. For example, long-range planning requires large scale features while planning for manipulation requires finer features, that is on the order of the size of the effector. We will investigate techniques for incorporating an explicit notion of scale in the feature extractors.

4 Updating

A terrain map is not a static representation, it must be updated each time new sensor data is available. This involves a two-stage process: Identifying the common parts between the current map and the new data set, or *matching*, and adding the new information to the current map, or *merging*. The matching stage is also used to compute the location of the new data set with respect to the reference frame of the current map. Even though a significant amount has been generated in the areas of matching and merging in the context of aerial imagery and indoor mobile robots, little has been done in the case of unstructured terrain. The reason is mainly the lack of distinctive features upon which the matching algorithms can be based. In most cases this is solved by assuming that accurate positioning sensing is available, thus suppressing the need for a matching step since the transformation between successive sensor views is accurately known [4,5]. This assumption does not hold in many cases, especially since accurate positioning is needed in all six degrees of freedom in order to guarantee accurate merging of the maps at all time. In those cases, novel matching techniques are needed.

The main step in updating maps is the matching of maps, that is the identification of common parts between maps and the use of this information to estimate the displacement between the maps. We divide the matching into two stages corresponding to the data and feature layers of the basic representation. The first stage is the matching between sets of features identified in the maps to be merged. The algorithms use a tree search strategy to match compatible features. The initial hypothesis are computed based on bounds on the expected transformation and on similarity measures between features. The search is pruned by using consistency constraints, in particular by thresholding the error on the transformation computed from the matches at each stage of the search. Those algorithms are ad-hoc in the sense that they depend on particular types of features (*e.g.* regions vs. edges). This becomes a bottleneck when it comes to computing the final transformation since different features have different mathematical descriptions (*e.g.* regions moments vs. line equation). The problem is complicated by the fact that many different types of features (*e.g.* arbitrarily shaped regions, edges, points, etc.) may have to be used simultaneously. The result of this matching is used in three ways: First, it provides an initial rough estimate of the displacement between the two maps. Second, it gives an indication of which parts of the maps may overlap in the merged resulting map. Third, it produces a list of common features that must be merged in the final map.

The second stage uses only the data layer and computes a finer estimate of the transformation by minimizing an energy function that measures the distance between the maps over the potentially overlapping parts identified in the first stage. The second stage is needed because the feature matching in general does not provide enough accuracy on the resulting transformation. Furthermore, since natural environments lack distinctive features, it may be more natural to try to match maps without going through a feature extraction step, that is to match the maps by using directly the data layer. An instance of such an approach is presented in [10] in which the transformation between two elevation maps is computed by minimizing $E = \Sigma(h_1(x,y) - h_2(T,x,y))^2$ where $h_1(x,y)$ is the elevation at (x,y) in the first map, and $h_2(T,x,y)$ is the elevation at the same point computed from the second map transformed by T. Once the matching stage is completed, common parts need to be merged to produce the final composite map. At the data level, the most natural approach to merging is to combine at each point (x,y) the two values $(h_1(x,y), \sigma_1(x,y))$ and $(h_2(x,y), \sigma_2(x,y))$ using the maximum likelihood rule. This is optimal assuming that the uncertainties exhibit Gaussian distributions. Once the data layer is updated, the local attributes of the points that lie

within the overlapping parts must be recomputed, since there is no easy way to combine the attribute values from different sources in most cases. Finally, the features that are identified as common features are merged by estimating the best composite feature. The strategy for actually computing the merged features depend on their type. A theoretical background for such a minimization approach showing that it leads to an optimal estimate in a maximum likelihood sense is presented in [19]. Preliminary results [10] show that this matching technique gives accurate estimates of the transformation, that is as accurate as the map itself. Those results are obtained in cases in which only a few maps are merged and the initial estimate of the displacement between maps is close to the real value. By contrast, if many maps are to be handled, we need to have an estimate of the uncertainty on the computed transformation in order to evaluate the quality of a merged map and in order to discard inaccurate maps. If the initial estimate of the transformation is too far from the true value, we may not be able to converge. A similar approach can be used with probability maps and has been reported both in the 2-D [6] and 3-D [18] cases. Three problems remain to be solved with this approach:

- *Uncertainty on the transformation:* The uncertainty of the transformation computed from the iconic matching algorithm must be estimated. The uncertainty, modeled as a Gaussian distribution of the components of the transformation, should be based on the shape of the final error E_{min}. The uncertainty is needed for rejecting incorrect matches, a situation which may occur if the maps are too far away from each other to be matched correctly, and to have a measure of the quality of the merged map that is used to discard parts of the map as the vehicle travels.

- *Convergence properties:* Good models for the convergence of the minimization of E are not available at this point. The convergence properties of E must be analyzed both theoretically and analytically. This will give us an estimation of the limitations of the method.

- *Inclusion of the uncertainties in the criterion function:* Algorithms are needed to fold the uncertainty on the map points into the criterion function E. This is needed in a general terrain mapping system since some sensors may produce maps with very uncertain regions that should be taken into consideration in the computation of E. This is of course automatic in the case of probability maps but is more difficult in the case of elevation maps.

Both approaches to matching have advantages and problems: feature matching is efficient and can work even with a very poor initial estimation of the transformation T, but it cannot lead to an accurate estimation of the T; map correlation, on the other hand, can lead to an accuracy on T on the order of the resolution of the map, but are usually computationally demanding and require an accurate initial estimate of T. The best results are obtained by using a compromise between the two approaches: feature matching to identify common parts between the two maps followed by map correlation to refine the transformation estimate. Such a compromise is described in [7] in which natural surfaces are first segmented into superquadrics that are then correlated with the new (range) data set. Another example is [9] in which the transformation computed from matching features computed from the local curvatures of surfaces is used a starting point for the correlation of the two maps. Figures 3 and 4 show the result of the map updating: Figure 3 shows the matching between the features detected on two elevation maps, the feature are shown as black points and lines overlaid on top of the elevation maps; Figure 4 shows the elevation after iconic matching and merging.

Since the uncertainty on the position of the vehicle with respect to a global reference frame increases as the vehicle travels, we must use a vehicle-centered representation in which only the parts of the terrain map that are close to the vehicle are retained. The best approach is therefore to scroll the map whenever the uncertainty on the location of points measured in the past becomes to large.

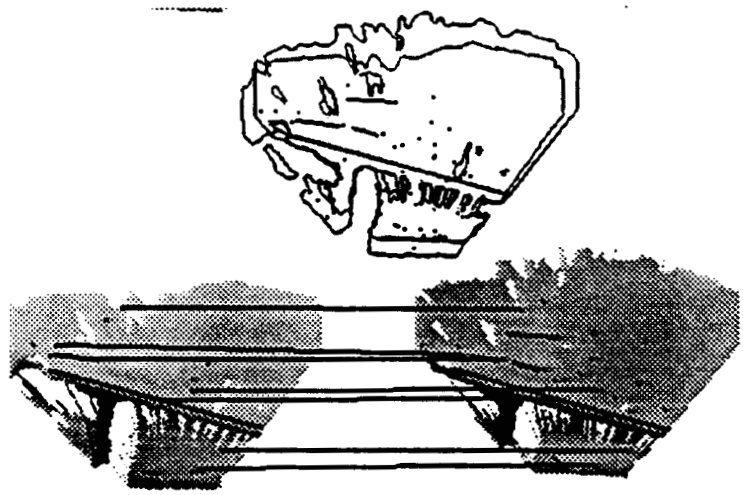

Figure 3: Matching maps using local features

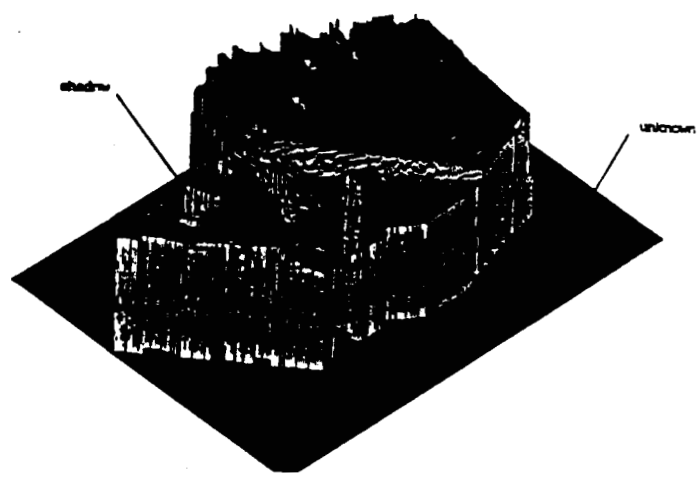

Figure 4: Composite map from feature and iconic matching

5 Conclusion

We have presented the building blocks of the data, and feature layers of a general terrain representation as well as the algorithms for building and updating maps. Existing techniques developed for autonomous mobile robots can be used to build the terrain representation. Many issues have to be investigated in order to build a robust terrain representation.

First, the notion of uncertainty must be explicitly included at all the levels of the terrain representation and must be taken into account in all the map building and updating. At the data level that involves converting the sensor uncertainty model from sensor space to map space. At the feature level that involves combining the uncertainties on the data points that compose a feature into an uncertainty value on that feature. In the updating algorithm, that involves computing an uncertainty on the displacement between maps from the uncertainties on the data points and the error on the map matching. Probability maps are ideal in this respect. They are not sufficient for a complete terrain description since they do not easily allow for feature extraction.

Second, most of the existing algorithms depend heavily on a particular type of sensor or application. It is important to extend those algorithms into generic algorithms. For instance, a generic map matching algorithm would be able to handle a variety of geometric features by combining the existing ad-hoc algorithms.

References

[1] P. Besl. *Surfaces in Early Range Understanding*. PhD thesis, Electrical Engineering and Computer Science Dept., University of Michigan, 1986.

[2] C. Caillas and M. Hebert and E. Krotkov and I.S. Kweon and T. Kanade. Methods for Identifying Footfall Positions for a Legged Robot. In *Proc. IEEE International Workshop on Intelligent Robots Systems*, 1989.

[3] J.M. Cushieri and M.Hebert. Sonar Applications for Underwater Vision. In *Proc. 11th Energy-Sources Technology Conference and Exhibition*, New-Orleans, 1988.

[4] M.J. Daily, J.G. Harris, and K. Reiser. An Operational Perception System for Cross-Country Navigation. In *Proc. Image Understanding Workshop*, Cambridge, 1988.

[5] R.T. Dunlay and S.B. Seida. Parallel Off-Road Perception Processing on the Autonomous Land Vehicle. In *Proc. SPIE Mobile Robots III*, Cambridge, MA, 1988.

[6] A. Elfes. *Occupancy Grids as a Spatial Representation for Mobile Robot Mapping and Navigation*. PhD thesis, Electrical and Computer Engineering Dept./Robotics Institute, Carnegie-Mellon University, 1989.

[7] M.A. Fischler and R.C. Bolles. Image Understanding Research at SRI International. In *Proc. Image Understanding Workshop*, 1988.

[8] M. Hebert, C. Caillas, E. Krotkov, I. Kweon, and T. Kanade. Terrain Mapping for a Roving Planetary Explorer. In *Proc. IEEE Robotics and Automation*, May 1989.

[9] M. Hebert, I. Kweon, and T. Kanade. *3-D Vision Techniques for Autonomous Vehicles*. Technical Report CMU-RI-TR-88-12, The Robotics Institute, Carnegie-Mellon University, 1988.

[10] I. Kweon, M. Hebert, and T. Kanade. Perception for Rough Terrain Navigation. In *Proc. SPIE Mobile Robots III*, Cambridge, MA, 1988.

[11] L. Matthies and A. Elfes. Integration of Sonar and Stereo Range Data Using a Grid-based Representation. In *Proc. IEEE Robotics and Automation Conference*, Philadelphia, April 1988.

[12] J.C. Maxwell. On Hills and Dales. *London, Edinburgh, and Dublin Philosophical Magazine*, 40, December 1870.

[13] H.P. Moravec and A. Elfes. High-Resolution Maps from Wide Angle Sonar. In *Proc. IEEE International Conference on Robotics and Automation*, St. Louis, 1985.

[14] D.J. Orser and M. Roche. The Extraction of Topographic Features in Support of Autonomous Underwater Vehicle Navigation. In *Proc. Fifth International Symposium on Unmanned Untethered Submersible*, University of New Hampshire, 1987.

[15] A.P. Pentland. Perceptual Organization and Representation of Natural Form. *Artificial Intelligence*, 28, 1986.

[16] J. Ponce and M. Brady. *Toward a Surface Primal Sketch*, chapter II, pages 195–241. Kluwer Academic Publishers, 1987.

[17] T. Stentz. *The NAVLAB System for Mobile Robot Navigation.* PhD thesis, Carnegie-mellon University, Fall 1988.

[18] W.K. Stewart. A Non-Deterministic Approach to 3-D Modeling Underwater. In *Proc. Fifth International Symposium on Unmanned Untethered Submersible*, University of New Hampshire, 1987.

[19] R. Szeliski. Estimating Motion from Sparse Range Data without Correspondance. In *International Conf. on Computer Vision*, Tarpon Springs, Florida, December 1988.