Terrain Drivability Analysis in 3D Laser Range Data for Autonomous Robot Navigation in Unstructured Environments

Frank Neuhaus, Denis Dillenberger, Johannes Pellenz, Dietrich Paulus
Active Vision Group, University of Koblenz-Landau
56070 Koblenz, Germany
{fneuhaus,dillenberger,pellenz,paulus}@uni-koblenz.de

Abstract

Three-dimensional laser range finders provide autonomous systems with vast amounts of information. However, autonomous robots navigating in unstructured environments are usually not interested in every geometric detail of their surroundings. Instead, they require real-time information about the location of obstacles and the condition of drivable areas.

In this paper, we first present grid-based algorithms for classifying regions as either drivable or not. In a subsequent step, drivable regions are further examined using a novel algorithm which determines the local terrain roughness. This information can be used by a path planning algorithm to decide whether to prefer a rough, muddy area, or a plain street, which would not be possible using binary drivability information only.

1 Introduction

The development of autonomous, mobile systems is a current research topic in both signal processing and artificial intelligence. So far, research focused mostly on structured areas, such as the inside of buildings, or operations in well-known domains as it is done in factory automatization. The ongoing development now enables using robots in rough terrain as well. This was demonstrated recently in the well known DARPA Grand Challenge in 2004 and 2005, as well as in the DARPA Urban Challenge in 2007 [3].

In both Grand Challenges, the participants mostly acquired terrain drivability information using cameras, two-dimensional laser range finders, or a combination of both [3]. The advent of three-dimensional laser range finders such as the *Velodyne HDL-64E S2*[1] in the Urban Challenge yielded a much richer, thorough picture of the environment.

Our goal is to get all terrain drivability information with only one 3D laser range finder in real-time.

[1] See: http://www.velodyne.com/lidar

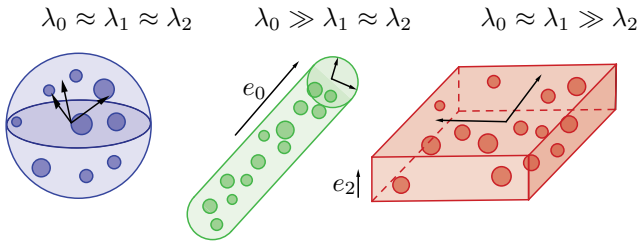

Figure 1. Three cases of point cloud analysis

1.1 Sensor description

The *Velodyne HDL-64E S2* is a high resolution 3D laser range finder delivering a data rate of up to 1.8 million distance measurements per second, scanning a vertical field-of-view of about 28°. It was developed particularly with regard to autonomous vehicle navigation, mapping, exploration as well as industrial automatization. The sensor has a head containing a total of 64 lasers which spins at a user-programmable frequency between 5 and 15 Hz while the built-in lasers continually perform measurements. We accumulate all measurements of one revolution into one point cloud and subsequently perform an analysis of the latter.

2 Algorithms

2.1 PCA

The principal component analysis (PCA) of the covariance matrix of a three-dimensional point cloud (which is in this specific case in fact equivalent to the eigendecomposition) yields three eigenvectors with corresponding eigenvalues. The latter indicate the variance of the point cloud along the corresponding axes, and can be used to determine the shape of the point cloud. Figure 1 shows the three different cases that can occur: If all eigenvalues are roughly equal, the point cloud is more or less unstructured without any specific principal direction. If one of them is significantly higher than the remaining two, the shape of the point cloud is cylindrical and the eigenvector corresponding to the highest eigenvalue defines the direc-

tion of the cylinder. If one of the eigenvalues is very low in comparison to the other two, the point cloud is more or less a plane, and the eigenvector corresponding to the lowest eigenvalue defines the normal of the plane.

The above mentioned methodology only yields good results for reasonably small point clouds. Those containing multiple planes/cylinders or other complex structures simply end up in the case where all three eigenvalues are large. In order to gain *local* information about a large point cloud such as the one delivered by our sensor, it first needs to be subdivided into smaller chunks, which can then be analyzed with the help of the PCA.

2.2 Grid-based PCA

There is a multitude of possibilities to subdivide the point cloud. We use a simple 2D grid, centered around the origin of the sensor. Other researchers such as [1] used 3D grids.

The problem with these statically sized cloud-fragments is that even though the amount of information that is being delivered by the *Velodyne HDL-64E S2* is immense, the ever increasing spread of sensor's field of view makes information about far-away ground surfaces quite sparse. Selecting a small cell-size for the grid results in many distant cells being completely empty, simply because not a single laser ray endpoint happens to be inside of them. Selecting a large cell size allows larger regions to be considered in the PCA step. However it causes problems in nearby regions: If a small obstacle is inside of a cell, the whole cell has to be classified as undrivable. Subsequent path planning algorithms operating exclusively on this grid would have to keep significantly more distance to obstacles than really needed. In some cases, such as narrow streets, where precise navigation is required, this methodology can even impede path-planning altogether, simply because almost everything has to be marked as undrivable. What is really needed is an algorithm that combines the advantages of both small and large cells.

2.3 Hierarchical PCA

The Hierarchical PCA is an algorithm we developed to recursively subdivide the point cloud. We use it on a 2D grid, however the extension to 3D is straightforward. The cell size is chosen to be relatively small.

The initial input of the algorithm is the whole grid. In each step, it performs a PCA analysis on the points in the considered region. If the point cloud contained in the region is not 'flat' enough, it splits the region in two pieces and recursively calls itself on the two resulting fragments. The splitting simply occurs along the longest axis of the region. If the input region can no longer be subdivided (i. e. 1×1), and it is still not flat enough, it is considered an obstacle.

To quantify flatness, we first define the *local height disturbance* h as:

$$h = \lambda_k \quad \text{with} \quad k = \underset{i \in \{0,1,2\}}{\operatorname{argmax}} |\mathbf{e}_i^{\mathrm{T}} \mathbf{z}| \tag{1}$$

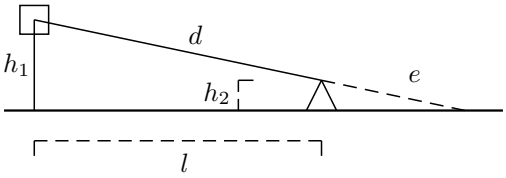

Figure 2. Geometry of distance disturbances

where $\mathbf{z}$ is a vector pointing up, $\mathbf{e}_i$ are the eigenvectors, and λ_i the eigenvalues of the point cloud in the current region. A region is considered flat, if h is below some threshold. This definition makes sure, that even sloped but flat areas get low values of h.

2.4 Roughness analysis

Having distinguished between drivable and undrivable areas, one issue remains: Even though a region may technically be drivable, it may still be desirable to prefer one region over another. An example of this happening is when robot is driving on a dirt-track with acres left and right of this track (see Figure 4). Obviously, one expects the robot to continue it's way on the dirt-track instead of driving through the mud of the acres. Apart from this specific case, there is a multitude of other situations where a fine-grained terrain analysis is beneficial.

One may be tempted to use simply the local height disturbance as a measure for the roughness of the terrain at that spot. Unfortunately it turns out that due to the large sensor noise relative to the height of the bumps that are to be detected, it is infeasible to distinguish rough from flat areas properly.

As [2] have already noticed, a much better indicator of roughness is the local *distance* disturbance: Small bumps and dents in the ground cause large changes in distance of adjacent laser measurements because of the grazing angles with which the laser rays hit the ground. This can be seen in Figure 2: The box represents the laser range finder, mounted at a height h_1. The robot is standing in front of a bumpy area. The bumps are assumed to have a fixed height h_2 in the considered region. Now each ray will either hit a bump or barely graze above one, hitting the ground behind. The distance difference e between a measurement that has hit the ground, and one that has hit the bump is relatively large compared to the height of the bump. In addition to that, it linearily increases as the distance to the bump grows. This can be seen in the following equation for e, which can be derived using the intersecting lines theorem:

$$e = d \cdot \frac{h_2}{h_1 - h_2} \tag{2}$$

Note how d scales up the remaining term, which in fact describes the intensity of the bump.

2.5 Computing the local terrain roughness

The *Velodyne HDL-64E S2* provides 64 distance time series per revolution: one for each laser in the sensor-head. We consider the time series of each of the 64 lasers seperately. If one laser is not able to measure a distance at a certain spot, a very low distance value will be reported. In a first fast preprocessing step, we eliminate these erroneous measurements. This is done by looping over each series, overwriting every faulty measurement with the last valid value. More sophisticated – but also computationally more expensive – approaches could be applied at this point. However, they are not really necessary because typically only individual points are corrupted. In the next step, a heavily smoothed copy of this data is made. We use a simple mean filter (size nine) – again mostly because of speed and because superior low-pass filters such as Gaussian or even edge-preserving smoothing filters did not show any significant improvement. Finally, the unfiltered version is subtracted from the low-pass filtered version forming a high-pass filtered version of the data. Now, the distance deviation δ_i from the smoothed version is known for each laser measurement. For each grid-cell containing points, we compute the variance of distance deviations $\hat{\sigma}_\delta^2$ of all n points in that cell, using

$$\hat{\sigma}_\delta^2 = \frac{1}{n}\sum_{i=1}^{n}\delta_i^2 - \left(\frac{1}{n}\sum_{i=1}^{n}\delta_i\right)^2 \tag{3}$$

However, $\hat{\sigma}_\delta^2$ is really a sum of the inherent sensor noise σ_L^2 and the actual distance variance σ_δ^2. Since we are only interested in the latter, and since we assume the sensor noise to be additive, we eliminate the sensor noise by computing σ_δ^2 as $\max\{0, \hat{\sigma}_\delta^2 - \sigma_L^2\}$.

This measure is not yet independent to the distance, as we have motivated in equation (2): It will be too low in areas near the robot, or too high for areas which are far away. So, in a final step we obtain the local terrain roughness r as we eliminate the previously computed measure from the influence of the distance:

$$r = \frac{\sigma_\delta^2}{d_{\text{Cell}}^2} \tag{4}$$

where d_{Cell} is the distance of the laser to the respective grid-cell. The roughness r can now be used to determine the drivability of the terrain: High values of r correspond to high terrain roughness, low values to low roughness. Now a robot could, for example, try to find a path, where the sum of all r's in the crossed cells is minimal.

3 Results

3.1 Test setup

The *Velodyne HDL-64E S2* was mounted onto a (human-driven) car using a custom aluminum frame, along with a number of sensors including cameras, an inertial measurement unit, a GPS receiver, etc. (see Figure 3). Though, for this specific experiment, *only* data from the laser range finder was used.

Figure 3. Test setup

3.2 Experiments

Using the previously described setup, a total of approximately 4 hours of data, with a size of roughly 40 GB, was collected, i. e. the full raw sensor data was written to disk and can now be played back in real-time. Thus the exact conditions at the time of recording can be replicated for testing purposes. A wide range of different surface conditions including concrete, grass and muddy areas were captured.

One common scenario of semi-structured environments is depicted in Figure 4. The right-hand side image was taken by our car's front camera. At the time of recording, the car was driving on a dirt-track. A barn can be seen to the front/right of the vehicle. Muddy acres are located directly left and right of the vehicle. The results of our algorithms for this scenario, are depicted in the left-hand side image of Figure 4: Every grid-cell is rendered as a small quadriliteral. They are oriented and located to approximate optimally the points contained in the respective grid-cells. One can see that the barn with the adjacent hedges is entirely marked as undrivable. This is visualized by red boxes. The color of the quadliterals is the result of the roughness analysis algorithm which was presented in section 2.4. The colors range from green (low roughness) to orange (high roughness). Note how the green color of the dirt-track successfully indicates that it is a lot more desirable to drive on the dirt-track than on the acres left and right of the vehicle.

The result of the hierarchical PCA that was introduced in subsection 2.3 can be seen in Figure 5 and 6. Figure 5 shows the non-empty grid-cells in a typical scenario. It is clearly visible that distant cells often do not contain any laser-endpoints. Figure 6 shows how the hierarchical PCA deals with this issue. Multiple equally flat cells are merged for the analysis: Small holes in the data coverage no longer represent a problem for the path-planner. However, since the algorithm performs the maximum subdivision in areas close to obstacles, the issue remains in those areas. This can be seen on the right in Figure 5 (there is a small hole next to the obstacle). Fortunately, these cases do not represent a problem for path planning.

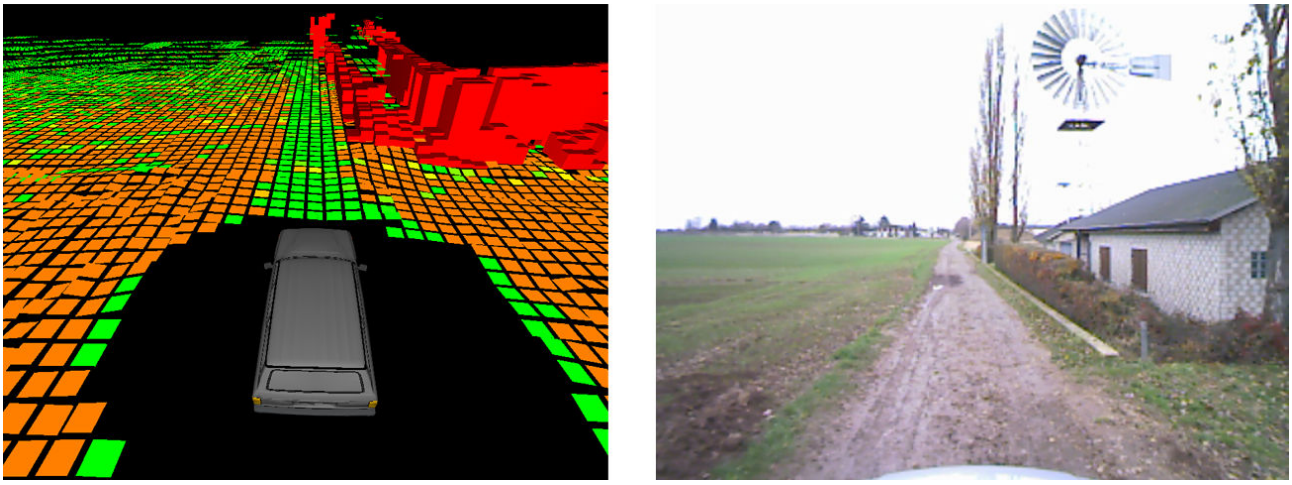

Figure 4. Left: 3D visualization of terrain roughness analysis Right: Camera image of the scenario

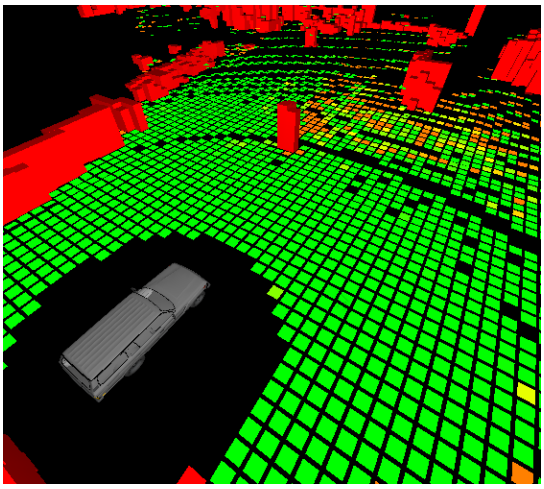

Figure 5. without hierarchical PCA

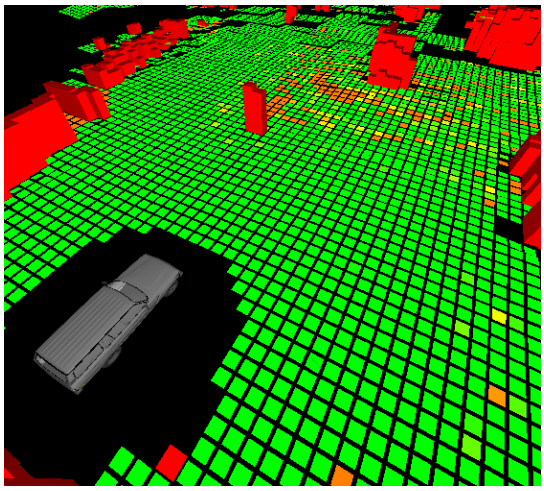

Figure 6. with hierarchical PCA

Both algorithms run together in real-time (15 Hz) on a normal laptop with an Intel Core2Duo processor.

4 Future work

In the future, we will work on fusing camera information with the data we have extracted from the laser. The color and texture of each terrain cell can be determined and may be used to complement classification in some cases. Another subject of research is the utilization of neighbourhood information during the classification of grid-cells, as well as the implementation of the actual path-planning algorithms, operating on our classification results.

References

[1] D. H. Jean-Francois Lalonde, Nicolas Vandapel and M. Hebert. *Natural terrain classification using three-dimensional ladar data for ground robot mobility*, 23(1):839 – 861, November 2006.

[2] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Hähnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun. Junior: The stanford entry in the urban challenge. *J. Field Robotics*, 25(9):569–597, 2008.

[3] S. Thrun. Winning the darpa grand challenge: A robot race through the mojave desert. In *ASE '06: Proceedings of the 21st IEEE/ACM International Conference on Automated Software Engineering*, page 11, Washington, DC, USA, 2006. IEEE Computer Society.

Acknowledgement

This work was partially funded by Wehrtechnische Dienststelle 51 (WTD), Koblenz and by V&R Vision & Robotics GmbH, Koblenz, Germany.