Probabilistic Semantic Mapping with a Virtual Sensor for Building/Nature detection

Martin Persson*[1], Tom Duckett**, Christoffer Valgren*[1] and Achim Lilienthal*

*Centre of Applied Autonomous Sensor Systems
Department of Technology
Örebro University, Sweden
martin.persson@tech.oru.se
christoffer.wahlgren@tech.oru.se
achim@lilienthals.de

**Department of Computing and Informatics
University of Lincoln
Lincoln, UK
tduckett@lincoln.ac.uk

Abstract— **In human-robot communication it is often important to relate robot sensor readings to concepts used by humans. We believe that access to semantic maps will make it possible for robots to better communicate information to a human operator and vice versa. The main contribution of this paper is a method that fuses data from different sensor modalities, range sensors and vision sensors are considered, to create a probabilistic semantic map of an outdoor environment. The method combines a learned virtual sensor (understood as one or several physical sensors with a dedicated signal processing unit for recognition of real world concepts) for building detection with a standard occupancy map. The virtual sensor is applied on a mobile robot, combining classifications of sub-images from a panoramic view with spatial information (location and orientation of the robot) giving the likely locations of buildings. This information is combined with an occupancy map to calculate a probabilistic semantic map. Our experiments with an outdoor mobile robot show that the method produces semantic maps with correct labeling and an evident distinction between 'building' objects from 'nature' objects.**

I. Introduction

The use of human concepts is very important in human-robot communication. Skubic *et al.* [13] discuss the benefits of human spatial concepts (which they call 'linguistic spatial descriptions') for different types of robot control, and point out that these descriptions are especially important for novice robot users. To enable human operators to interact with mobile robots in, e.g., task planning, or to allow the system to use data from external sources, e.g., GIS, it is necessary for the robot to be able to relate its sensor readings to human spatial concepts. One way to achieve this is to construct semantic maps, where objects in the map are labelled with human concepts.

The number of publications on semantic mapping is still quite limited. Most publications relate to mapping of indoor environments and only a few consider the problem that the robot itself extracts the semantic labels for the map. The combination of SLAM and semantic information was proposed by Dellaert and Bruemmer [3]. Extracting planes from 3D laser range data has been used to achieve semantic scene interpretations of indoor environments [10] in the form of floors, walls, doors, etc. Mozos *et al.* [8], [9] semantically label indoor environments as corridors, rooms, doorways, etc. using a classifier trained with AdaBoost. Range data from laser scanners are the main input and in addition some features extracted from a vision sensor are used. In human augmented mapping [17] a person follows a mobile robot during a tour in, e.g., a domestic environment and gives the robot information about different locations. Galindo *et al.* [5] present a method to describe an indoor environment with two hierarchies based on spatial and semantic information. Anchoring establishes links between the spatial and semantic information and the result can be classified as a hybrid metric-topological-semantic map.

Wolf and Sukhatme [19] describe outdoor semantic mapping using laser scanners and supervised learning with HMM and SVM. One map is based on the activity caused by passing objects of different sizes. Using this information roads and sidewalks can be distinguished from each other. A second type of map is based on the roughness of the terrain and is intended to be used for path planning. Closely related work concerns detection of driveable areas for mobile robots using vision [2], [6], [14]. These works do not primarily build maps but use the information for road localisation.

In our previous work [11] we introduced a virtual sensor that can be used to facilitate human-robot communication. A virtual sensor is understood as one or several physical sensors with a dedicated signal processing unit for recognition of real world concepts. As an example of a virtual sensor, we described a virtual sensor for building detection using methods for classification of views as buildings or nature based on vision. The purpose was to detect one very distinctive type of object that is often used by humans, for example, in textual description of route directions. The method was based on learning a mapping from a set of generic features to a particular concept.

The main contribution of this paper is a method that fuses data from different sensor modalities, such as range sensors and vision sensors, to create a semantic map of an outdoor environment. The method combines a learned virtual sensor for building detection with a standard occupancy map. The result is a semantic map with two object classes; 'buildings'

[1] Supported by The Swedish Defence Material Administration

and 'nonbuildings'. The training set of nonbuildings consisted of nature images and therefore also the word *nature* is used instead of *nonbuildings* in this paper. The virtual sensor uses different types of visual features selected by the boosting algorithm AdaBoost. The pose information from the mobile robot is combined with the output from the virtual sensor to give the direction to buildings. These directions are used to update the occupancy grid map with semantic information.

The paper is organised as follows. An overview of the suggested approach is given in Section II. Section III describes virtual planar cameras that give the input images to the virtual sensor. These are constructed from images acquired by an omni-directional vision system mounted on a mobile robot. Section IV describes the virtual sensor with its set of features from which weak classifiers are calculated and how these are used by an AdaBoost classifier. The algorithm to calculate probabilistic semantic maps is presented in Section V. Experiments are described in Section VI and an evaluation of the results is given in Section VII. Finally, conclusions and future work are found in Section VIII.

II. Suggested Approach

We use an omni-directional camera mounted on a mobile robot, which gives a 360° view of the surroundings. From the omni-directional image N planar views or sub-images are created with a horizontal field-of-view of $\Delta°$ (the values of N and Δ are provided in Table II). The sub-images are fed into a learned virtual sensor for building detection. We use AdaBoost for training a classifier that classifies close range monocular grey scale images into 'buildings' and 'nature' [11].

Based on the result from the virtual sensor we create local maps for building objects and nature objects. A local map is built for each robot position where images have been acquired. The local maps are then fused into a global probabilistic semantic map. For clarity we discuss a few issues concerning the process:

1) The robot has to be able to determine its pose (position and orientation) for each point.
2) Objects that should be included in the semantic map are given.
3) The result from the virtual sensor applies for the whole sub-image.

The positioning issue can be handled in different ways, e.g., by using SLAM (simultaneous localisation and mapping) or GPS. In our case we use differential GPS and odometry to compute the robot poses along the trajectory.

The requirement on the availability of map objects means that a standard occupancy map that includes the objects that should be labelled is available. An occupancy map can be built using a laser scanner or objects could be detected using stereo vision. Throughout this paper, *object* is understood as a connected component in a binarized occupancy grid.

The virtual sensor that we use classifies a complete view into one of two classes; nature and building. This means that all objects within the view are assumed to belong to the same class. To focus the attention on the main objects in the view, probabilities assigned for single objects are adjusted according to their proportions of the view. It is further assumed that the objects are large enough in height to contribute to the classification by the virtual sensor.

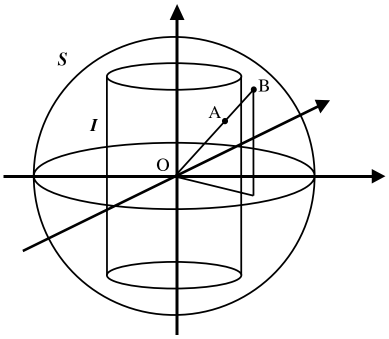

Fig. 1. Projection of the unwrapped image I onto the sphere S.

III. Virtual Planar Cameras

In our previous work, we used a digital camera mounted on a pan-and-tilt head to create a panoramic view with about 280° field-of-view horizontally [11]. A sweep of the pan-and-tilt head required a few seconds, making the method time-consuming. In order to acquire a 360° field-of-view panoramic image in one single shot, we now use an omni-directional camera. As an additional benefit, the centre of the image plane is now the same throughout the whole panoramic view. The omni-directional camera is fitted on a Pioneer P3-AT from ActivMedia. The camera itself is a standard consumer-grade SLR digital camera (Canon EOS350D, 8 megapixels). On top of the lens, a curved mirror from 0-360.com is mounted. This camera-mirror combination produces omni-directional images that can be unwrapped into high-resolution *spherical images* by a polar-to-Cartesian conversion.

From a large spherical image I, we extract smaller sub-images that appear as if they were acquired by a regular, finite projective camera. To do this, we project the image twice. The image I is folded into a cylinder and projected onto the inside of S as if there was a lightbulb placed in the common centre of the cylinder and sphere (extending the ray from the centre O to the intersection with I at A, we want to determine the intersection with S at B, see Figure 1). The projection on a sub-image plane Π is then easily found by normal perspective projection. Figure 2 shows an example of the omni-directional image, the unwrapped image, and some planar images. For further details on projections, see for example [7].

IV. Virtual Sensor

The virtual sensor that we use for building detection in outdoor environments is described in detail in [11]. The virtual sensor is based on a method that combines different types of features such as edge orientation, grey level clustering, and corners into a system with high classification rate. The method was applied on a mobile robot as a virtual sensor for

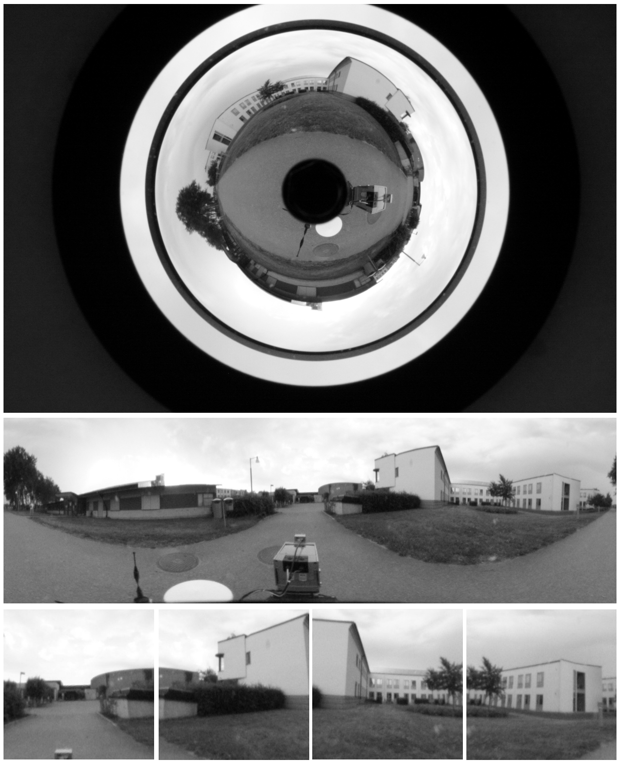

Fig. 2. The upper part of the figure is an omni-directional image from the experiment, the middle part is an unwrapped version of the same image, and the lower part shows some flat images extracted from the unwrapped image.

building detection in an outdoor environment and is expected to be extendable to other classes, such as windows and doors. AdaBoost is used for learning a classifier that classifies close range monocular grey scale images into 'buildings' and 'nature'. AdaBoost has the ability to select the best so-called weak classifiers and produces a strong classifier as a linear combination of the weak classifiers. This section shortly describes the features that are used, the classifier AdaBoost and some previous results using the virtual sensor.

A. Feature Extraction

We select a large number of image features that are assumed to capture the properties of man-made structures. These features can be divided into three groups. The first type of features is derived from edge orientation in order to calculate the relative content of vertical and horizontal edges. The second type of features combines the edges into more complex structures such as corners. The third type of features uses grey level clusters based on the observation that buildings often contain surfaces with constant grey level. The particular set of features was selected with regard to a virtual sensor for building detection. In general, i.e., as a base for other virtual sensors, an even more generic set of features would have to be used. In total 24 features are extracted and all features except two are normalised in order to avoid scaling problems. In the following we give a short overview of the features. For a more detailed description see [11].

1) Edge Orientation: After edge detection with Canny's edge detector [1] and line extraction[1] in the edge image, the absolute values of the lines' orientations and the line lengths are used to calculate the features. These features are based on different histograms of the line orientation, with various number of bins and weighted or selected based on the line lengths. The objective is to capture the frequency of vertical and horizontal edges in relation to other orientations.

2) Edge Combinations: Building facades often contain right-angled corners at connections of vertical and horizontal edges and rectangles, e.g. doors and windows. In order to capture these properties the lines extracted from the edge image are combined to form right-angled corners. The lines and corners found are then combined in order to detect rectangles. Examples of features of this class are the number of right-angled corners and rectangles, their relation to each other and their relation to the number of detected edges.

3) Grey Level Clusters: Buildings are often characterised by large homogeneous areas in their facades, while nature images typically show larger variation. Other areas in images, however, can also be homogeneous, for example, roads, lawns, water and sky. This group of features is based on grey level clusters. For a feature that works *globally* in the image we use an equally spaced 25-bin grey level histogram, normalised by the image size and sum up the largest bins. To find *local* areas with homogeneous grey levels we search for the largest 4-connected areas with the same grey levels as used for the 25-bin grey level histogram. The features are based on different sums of the largest values and areas, and normalised with the image size.

B. AdaBoost

AdaBoost is the abbreviation for adaptive boosting. It was developed by Freund and Schapire [4] and has been used in diverse applications, e.g., as classifiers for image retrieval [16], for ball tracking with soccer-robots [18], and to classify laser scans for learning of indoor places [8], [9].

The main purpose of AdaBoost is to produce a strong classifier by a linear combination of weak classifiers, where *weak* means that the classification rate has to be only slightly better than 0.5 (better than guessing). The principle of AdaBoost is as follows (see [12] for a formal algorithm).

The input to the algorithm is a number, N, of positive (buildings) and negative (nature) examples. The training phase is a loop. For each iteration t, the best weak classifier h_t is calculated and a distribution D_t is recalculated. The boosting process uses D_t to increase the weights of hard training examples in order to focus the weak learners on the hard examples. The general AdaBoost algorithm does not include rules on how to choose the number of iterations T of the training loop. The training process can be aborted if the distribution D_t does not change, otherwise the loop runs through a manually determined number of iterations T. Boosting is known to be not particularly prone to overfitting

[1] Implemented by Peter Kovesi, University of Western Australia, http://www.csse.uwa.edu.au/∼pk/Research/MatlabFns/

[12]. We used $T = 30$ for training and did not see any indications of overfitting when evaluating the performance of the classifier on independent tests.

To be able to handle feature arrays (as opposed to scalar values) from the histogram data, we use a minimum distance classifier, MDC. We use the distribution D_t to bias the hard training examples by including it in the calculation of a weighted mean value for the MDC prototype vector:

$$\vec{m}_{l,k,t} = \frac{\sum_{\{n=1\ldots N|y_n=k\}} \vec{f}(n,l) D_t(n)}{\sum_{\{n=1\ldots N|y_n=k\}} D_t(n)}$$

where $\vec{m}_{l,k,t}$ is the mean value array for iteration t, class k, and feature l and y_n is the class of the nth image. The features for each image are stored in $\vec{f}(n,l)$ where n is the image number. For evaluation of the MDC at iteration t, a distance value $d_{k,l}(n)$ for each class k (building and nature) is calculated as

$$d_{k,l}(n) = \left\| \vec{f}(n,l) - \vec{m}_{l,k,t} \right\|$$

and the shortest distance over all features l indicates the winning class for that feature.

C. Training and Evaluation of the Virtual Sensor

For the experiments we trained AdaBoost on a set of images taken by an ordinary consumer digital camera (Sony DSC-P92). The images were taken over a period of several months in an environment similar to our intended outdoor environment. The training image set (Set 1 in [11]) contains 40 images of each class. A hand-held digital camera was used to take the training images in order to collect images from a larger area than was practical when using a mobile robot for data collection. The images were converted to grey scale and the resolution was lowered to 240 × 240 pixels. We know from previous experiments that a scale change by a factor of 2 (both from 120 to 240 pixel side lengths and vice versa) can be handled by the virtual sensor. The resolution of the sub-images used here is 320 × 320 pixels giving a scale difference of 1.33 (240 to 320 pixels). Examples of images from the training set are shown in Figure 3. The training set does not contain images from the same area as the performed experiments presented later on in this paper.

Previous tests have shown high classification rates for the virtual sensor. In a test using the same data for training as in this paper (Set 1 in [11]), 90 images of size 240 × 240 collected by a mobile robot were used for evaluation. In this test AdaBoost achieved a classification rate of over 92% with a false positive rate of 0% and a false negative rate of 11%. This showed that it is possible to build a classifier that achieves very good results with images obtained with a different camera on our mobile robot, even though the corresponding image sets had structural differences.

V. Probabilistic Semantic Map

In the implementation presented in this paper we use occupancy grid maps. The semantic map is built assuming that we have an occupancy grid with objects of a certain minimum height. Using this grid we search for objects within view of the virtual sensor and create a local semantic map of the area around the robot. In a second step local maps are used to update a global map using a probabilistic method. The result is a global semantic map where building and nature objects can be distinguished.

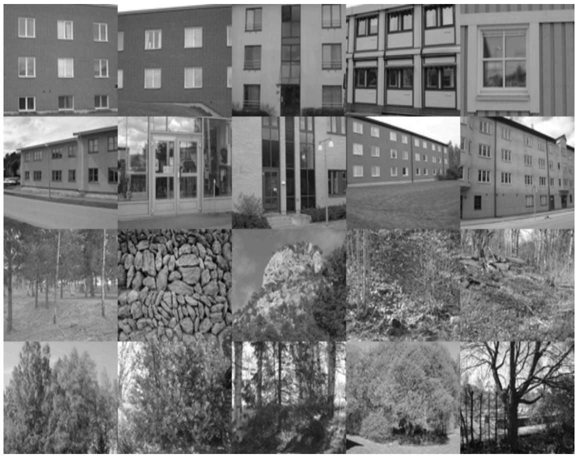

Fig. 3. Example of images used for training. The two upper rows show buildings and the two lower rows show nature.

A. Local Semantic Map

We assume that an occupancy map is supplied or built by the mobile robot. Occupancy maps can be built by different means, e.g., from stereo vision, motion stereo or laser range scans. The local grid map is a probabilistic representation of a sector in the occupancy map as seen by the robot. The sector is defined by the robot pose, the direction of the virtual planar camera (sub-image), the assumed opening angle θ of the sector and the expected maximum range of the virtual sensor, VS. The horizontal covering angles $\{\alpha_i\} = \alpha_1, \alpha_2, \ldots, \alpha_n$ of all objects within this sector are calculated. The total coverage angle is denoted α_N and is defined as

$$\alpha_N = \sum_{i=1}^{n} \alpha_i \leq \theta$$

where θ is the sector opening angle. Probabilities $P_i(\mathit{class}|\mathrm{VS}^T, \alpha_i)$ are assigned to the n objects in view (the grid cells within the sector and seen from the robot) in relation to their visible size using the following expression:

$$P_i(\mathit{class}|\mathrm{VS}^T, \alpha_i) = \frac{1}{2} + \frac{\alpha_i}{\theta}(P(\mathit{class}|\mathrm{VS}^T) - \frac{1}{2}) \qquad (1)$$

and $P(\mathit{class}|\mathrm{VS}^T)$ is the conditional probability that a view is *class* when the virtual sensor classification at time T is *class*. $P(\mathit{class}|\mathrm{VS}^T)$ gets different values depending on the output of VS and the two combinations that are interesting here are (where b indicates the class “building”):

$$P(\mathit{class}|\mathrm{VS}^T) = \begin{cases} P(\mathrm{b}|\mathrm{VS{=}b}) & > 0.5 \\ P(\neg\mathrm{b}|\mathrm{VS{=}\neg b}) & < 0.5 \end{cases}$$

Note that we give the largest objects within sight a higher probability than smaller objects (Eq. 1) because larger objects are more likely to influence the virtual sensor.

B. Global Semantic Map

In the second step we use the standard Bayes update equation (as described in, e.g., [15] p. 28) to update the global semantic map with the local map produced in the previous step. The probability that grid cell (x, y) is occupied after T sensor updates is denoted by $P(\text{occ}_{x,y}|s^1, s^2, \dots, s^T)$. Assuming that the conditional probability $P(s^{(t)}|\text{occ}_{x,y})$ is independent of $P(s^{(\tau)}|\text{occ}_{x,y})$ if $t \neq \tau$ and that the prior probability for occupancy is set to 0.5 the probability at (x, y) can be computed as:

$$P(\text{occ}_{x,y}|s^{1:T}) = 1 - \left(1 + \prod_{r=1}^{T} \frac{P(\text{occ}_{x,y}|s^{(r)})}{1-P(\text{occ}_{x,y}|s^{(r)})}\right)^{-1}$$

resulting in the update formula

$$\begin{aligned} &P(\text{occ}_{x,y}|s^{1:T}) = \\ &1 - \left(1 + \frac{P(\text{occ}_{x,y}|s^{1:T-1})}{1-P(\text{occ}_{x,y}|s^{1:T-1})} \frac{P(\text{occ}_{x,y}|s^{T})}{1-P(\text{occ}_{x,y}|s^{T})}\right)^{-1} \end{aligned} \quad (2)$$

In our case the sensor reading s^T is the output VS^T from the virtual sensor at time T and the grid cells are assigned a probability denoting whether they belong to *class*. Using these notations Eq. 2 can be rewritten as:

$$\begin{aligned} &P(\textit{class}|\text{VS}^{1:T}) = \\ &1 - \left(1 + \frac{P(\textit{class}|\text{VS}^{1:T-1})}{1-P(\textit{class}|\text{VS}^{1:T-1})} \frac{P(\textit{class}|\text{VS}^{T})}{1-P(\textit{class}|\text{VS}^{T})}\right)^{-1} \end{aligned}$$

which is the update formula used for the grid cells (the grid cell index (x, y) has been left out). The resulting global map will contain three different classes:

$$\begin{array}{lll} \textit{Building} & \text{if} & P(\textit{class}|\text{VS}^{1:T}) > 0.5 \\ \textit{Unknown} & \text{if} & P(\textit{class}|\text{VS}^{1:T}) = 0.5 \\ \textit{Nonbuilding} & \text{if} & P(\textit{class}|\text{VS}^{1:T}) < 0.5. \end{array}$$

The map is initialised with all cells set to *Unknown* (0.5) and will then be incrementally updated as the robot travels along the trajectory and evaluates the views with the VS.

VI. Experiments

A. Data Sets

Two data sets are used for the experiments described in this paper, see Table I. The sets consist of omni-images and pose information from the odometry and DGPS on the mobile robot. Each omni-image was converted into eight sub-images, where each sub-image has a resolution of 320×320 pixels and a horizontal field-of-view of 56°. This means that there is a small overlap between sub-images generated from the same omni-image. Examples of the sub-images are given in the lower part of Figure 2. The images were collected at Örebro Campus and the mobile robot trajectories are shown in Figure 4.

Using the aerial image presented in Figure 4, an occupancy map that serves as input to the semantic mapping algorithm grid map was constructed, see Figure 5. The building outlines and groups of trees around the trajectory have been marked as filled polygons by hand and the occupancy map is binary, i.e., a grid cell is either empty or occupied. A probabilistic occupancy map could have been directly used with the suggested algorithm, since the grid cells in the occupancy map can be assumed to be independent from the output of the virtual sensor.

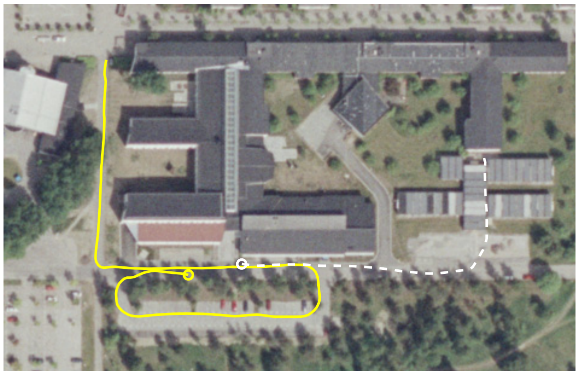

Fig. 4. The figure show the trajectories for the two data sets. Set 1 is the right trajectory (white, dashed) and Set 2 is the left trajectory (yellow, solid). The starting points are marked with a circle.

Set	Omni-images	Planar images	Length
1	88	704	146 m
2	210	1680	317 m

TABLE I
Used data sets.

B. Used Parameters

Table II lists the important parameters discussed so far. They can all be set to different values depending on the desired properties of the system. In our work we have set the last four parameters (planar camera field-of-view to grid cell size) according to the values in the table. The setting of the first three parameters (the sector opening angle θ and the probability pairs $P(\text{b}|\text{VS=b})$ and $P(\neg\text{b}|\text{VS=}\neg\text{b})$) have been

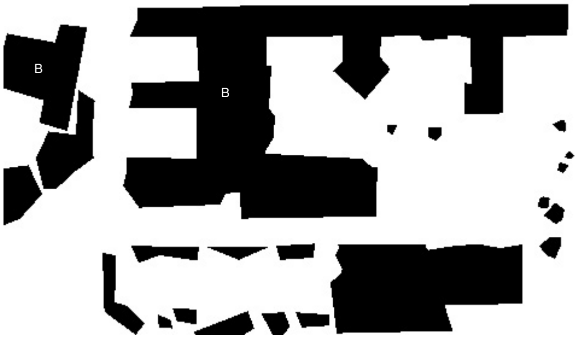

Fig. 5. The occupancy map used as input. The two building objects are marked with a 'B'. All other objects are nonbuildings. This map with the labels also serves as the ground truth in the forthcoming evaluation.

varied in order to optimize the performance of the system. It would be preferable to be able to relate the classification rate of the virtual sensor directly to $P(\text{b}|\text{VS=b})$ and $P(\neg\text{b}|\text{VS=}\neg\text{b})$. However, in reality there are a lot of views that contain a mix of buildings and vegetation that make a proper ground truth evaluation difficult. We have therefore decided to set the parameters based on the evaluation of the performance of the complete system including the virtual sensor and the map building algorithms. We use Set 1 for finding a good set of parameters and evaluate these on Set 2 in Section VII.

Parameter	Value	Description
$P(\text{b}\|\text{VS=b})$	> 0.5	Building probability
$P(\neg\text{b}\|\text{VS=}\neg\text{b})$	< 0.5	Nature (nonbuilding) probability
θ	15-56	Sector opening angle [deg]
Δ	56	Planar camera field-of-view [deg]
N	8	Number of planar views
-	50	VS maximum range [m]
-	0.5	Grid cell size [m]

TABLE II

DESCRIPTION OF PARAMETERS USED AS SETTINGS FOR THE PLANAR CAMERA VIEWS, THE SECTOR SIZE AND THE PROBABILITY MAPS.

We have in total evaluated 28 combinations of different field-of-views θ and probability pairs using Set 1 in order to find a good parameter setting. The following four θ were used: $56°, 45°, 30°$, and $15°$ and the following seven probability pairs: (0.8, 0.3), (0.8, 0.4), (0.8, 0.45), (0.9, 0.3), (0.9, 0.4), (0.9, 0.45), and (0.95, 0.48). With θ=56° we have the same field-of-view as the virtual sensor. With θ=45° there is no overlap in the local maps belonging to the same position. Using θ=30° and θ=15° we want to see how the system works when only the centre of the virtual sensor's field-of-view is used and the border parts are neglected.

One can note that $P(\text{b}|\text{VS=b})$ is always proportionally larger than $P(\neg\text{b}|\text{VS=}\neg\text{b})$. The main reason for this is that in our previous work the virtual sensor for building classification produces substantially more false negatives than false positives (see Section IV-C), which motivates the asymmetrical setting of these values.

For each parameter combination we calculate four measures:

- The true positive building detection rate, Φ_b. Number of cells correctly classified as buildings / number of covered building cells.
- The true negative detection rate, Φ_n. Number of cells correctly classified as nature / number of covered nature cells.
- The false positive rate, Φ_{fp}. Number of cells wrongly classified as buildings / number of covered nature cells.
- The false negative rate, Φ_{fn}. Number of cells wrongly classified as buildings / number of covered building cells.

We calculate the measures based on the global map and use the total detection rate $\Phi_b + \Phi_n$ as the primary selection criterion. It turns out that combination 6 (θ=56°, $P(\text{b}|\text{VS=b})$=0.9, and $P(\neg\text{b}|\text{VS=}\neg\text{b})$=0.45) and combination 12 (θ=45°, $P(\text{b}|\text{VS=b})$=0.9, and $P(\neg\text{b}|\text{VS=}\neg\text{b})$=0.45) produce the highest detection rate and also result in the lowest total false rate.

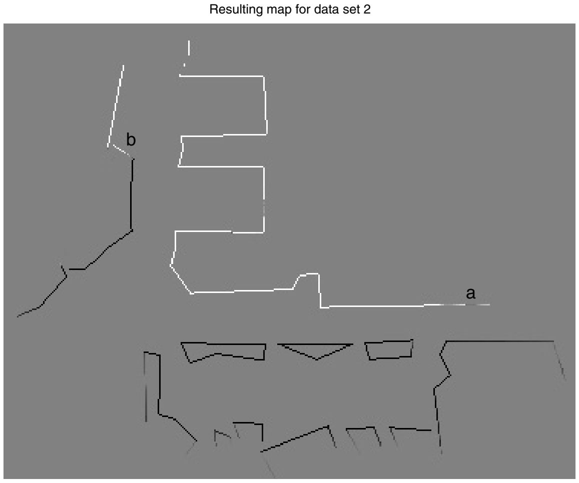

Fig. 6. The resulting map using data set 2. The outlines of both building and nature objects are correct to a large extent.

VII. RESULT

We use Set 2 to evaluate the semantic map and present the result for the two best parameter settings (combination 6 and 12) as found in the previous section.

A. Evaluation of the Nominal Maps

The final map using Set 2 (comb. 6) is presented in Figure 6. We can see that most of the outlines of the objects have been correctly labelled. Small parts that are not correct are the rightmost part of the building (marked with 'a') and a part of a grove close to the left building (marked with 'b').

Table III presents the detection rates for Set 2. The first row shows the result for combination 6 and the second row for combination 12. The evaluation was performed based on all cells in the grid map that are not equal to 0.5. We can see that the true detection rates are all equal to or higher than 96.7% and that combination 6 gives a slightly better result than combination 12 (it was the other way around for Set 1).

Test	Φ_b [%]	Φ_n [%]	Φ_{fp} [%]	Φ_{fn} [%]
6	98.3	98.7	1.3	1.7
12	96.9	98.4	1.6	3.1

TABLE III

RESULTS FOR PARAMETER COMBINATION 6 AND 12 USING DATA SET 2.

B. Robustness Test

To evaluate the robustness of the system two different Monte Carlo simulations were performed. First, the sensitivity to changes in robot pose was tested (pose noise) and second, the dependency on variations in the detection rate of the virtual sensor was evaluated (classification noise). We model the uncertainty with σ-values for the position, $\sigma_{pos} =$

2 m, and direction, $\sigma_{dir} = 5^\circ$. This position uncertainty is approximately the accuracy of standard GPS. Table IV shows the result for Monte Carlo simulations with 20 runs per test. The first two rows contain results after introducing the additional pose uncertainty. The detection rates are lower than the comparable ones presented in Table III. The total average detection rate has decreased from 98.1% to 96.3%.

The second two rows contain the result with classification noise. Here we have randomly changed 5% of the classifications (building to nature and vice versa) obtained from the virtual sensor. We can see that the result for building detection is close to the nominal case (average 97.0% compared to 97.6%), but that nature detection is clearly affected by the changed detection rates of the virtual sensor (average 81.7%). This is an effect of the assumption that building estimates are true to a higher extent and it shows that the selection of $P(\text{b}|\text{VS=b})$ and $P(\neg\text{b}|\text{VS=}\neg\text{b})$ should be carried out with this in mind.

Test	Φ_b [%]	Φ_n [%]	Φ_{fp} [%]	Φ_{fn} [%]
6 (pose unc.)	95.8±3.6	97.5±1.0	2.5±1.0	4.2±3.6
12 (pose unc.)	94.6±2.8	97.2±1.2	2.8±1.2	5.4±2.8
6 (cl. noise)	97.6±1.2	84.7±3.7	15.3±3.7	2.4±1.2
12 (cl. noise)	96.5±1.0	78.7±6.2	21.3±6.2	3.5±1.0

TABLE IV

Results for data set 2 presented with standard deviation. The first two rows show results with pose uncertainty and the second two rows show results with classification noise.

VIII. Conclusions

In this paper we have shown how a virtual sensor for pointing out buildings along a mobile robot's track can be used in the process of building a probabilistic semantic map of an outdoor environment. The presented results show that with the probabilistic mapping algorithm the uncertainty of the virtual sensor can be reduced. The method can handle the wide field-of-view of the planar camera (56°) and despite the fact that we do not know the location of the classified object in the image, an almost correct semantic map is produced. We also achieved good performance for both buildings and nonbuildings in the presence of pose uncertainty.

The benefit of using the virtual sensor with its good generalisation properties is that it produces results that are useful even though 1) the training set was quite limited (in total 80 low resolution images with side length 240 pixels), 2) we use another resolution in the sub-images (320×320 pixels), and 3) that we train on images using a standard digital camera, but the experiment images are taken using an omni-directional vision system.

A natural extension to this work would be to introduce other classes of objects. For example, drivable areas could be detected using the onboard sensor system. The map would also have to be extended to handle more than two classes. We further intend to use the semantic map to control segmentation of aerial images to find complete buildings and in this way improve the mobile robot mapping process.

References

[1] J. Canny. A computational approach for edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):279–98, Nov 1986.

[2] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2006.

[3] F. Dellaert and D. Bruemmer. Semantic SLAM for collaborative cognitive workspaces. In *AAAI Fall Symposium Series 2004: Workshop on The Interaction of Cognitive Science and Robotics: From Interfaces to Intelligence*, 2004.

[4] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[5] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. Fernández-Madrigal, and J. González. Multi-hierarchical semantic maps for mobile robotics. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 3492–3497, Edmonton, CA, 2005. Online at http://www.aass.oru.se/˜asaffio/.

[6] Y. Guo, V. Gerasimov, and G. Poulton. Vision-based drivable surface detection in autonomous ground vehicles. In *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3273–3278, Beijing, China, Oct 9-15 2006.

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[8] O. Martínez Mozos. Supervised learning of places from range data using AdaBoost. Master's thesis, University of Freiburg, Freiburg, Germany, 2004.

[9] O. Martínez Mozos, C. Stachniss, and W. Burgard. Supervised learning of places from range data using AdaBoost. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1742–1747, Barcelona, Spain, April 2005.

[10] A. Nüchter, H. Surmann, K. Lingemann, and J. Hertzberg. Semantic scene analysis of scanned 3D indoor environments. In *Proceedings of the 8th International Fall Workshop Vision, Modeling, and Visualization 2003*, pages 215 – 222, Munich, Germany, Nov 2003. IOS Press.

[11] M. Persson, T. Duckett, and A. Lilienthal. Virtual sensor for building detection by an outdoor mobile robot. In *Proceedings of the IROS 2006 workshop: From Sensors to Human Spatial Concepts*, pages 21–26, Beijing, China, Oct 2006.

[12] R. E. Schapire. A brief introduction to boosting. In *Proc. of the Sixteenth Int. Joint Conf. on Artificial Intelligence*, 1999.

[13] M. Skubic, P. Matsakis, G. Chronis, and J. Keller. Generating multi-level linguistic spatial descriptions from range sensor readings using the histogram of forces. *Autonomous Robots*, 14(1):51–69, Jan 2003.

[14] D. Song, H. N. Lee, J. Yi, and A. Levandowski. Vision-based motion planning for an autonomous motorcycle on ill-structured road. In *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3279–3286, Beijing, China, Oct 9-15 2006.

[15] S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Henning, T. Hofmann, M. Krell, and T. Schmidt. Map learning and high-speed navigation in RHINO. In D. Kortenkamp, R. P. Bonasso, and R. Murphy, editors, *Artificial intelligence and mobile robots: case studies of successful robot systems*, pages 21–52. AAAI Press / The MIT Press, 1998.

[16] K. Tieu and P. Viola. Boosting image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, South Carolina, June 2000.

[17] E. A. Topp and H. I. Christensen. Topological modelling for human augmented mapping. In *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2257–2263, Beijing, China, Oct 9-15 2006.

[18] A. Treptow, A. Masselli, and A. Zell. Real-time object tracking for soccer-robots without color information. In *European Conf. on Mobile Robotics (ECMR 2003)*, Radziejowice, Poland, 2003.

[19] D. F. Wolf and G. S. Sukhatme. Semantic mapping using mobile robots. *Accepted for publication in the IEEE Transactions on Robotics*, 2006.