Self-supervised Obstacle Detection for Humanoid Navigation Using Monocular Vision and Sparse Laser Data

Daniel Maier Maren Bennewitz Cyrill Stachniss

Abstract— **In this paper, we present an approach to obstacle detection for collision-free, efficient humanoid robot navigation based on monocular images and sparse laser range data. To detect arbitrary obstacles in the surroundings of the robot, we analyze 3D data points obtained from a 2D laser range finder installed in the robot's head. Relying only on this laser data, however, can be problematic. While walking, the floor close to the robot's feet is not observable by the laser sensor, which inherently increases the risk of collisions, especially in non-static scenes. Furthermore, it is time-consuming to frequently stop walking and tilting the head to obtain reliable information about close obstacles. We therefore present a technique to train obstacle detectors for images obtained from a monocular camera also located in the robot's head. The training is done online based on sparse laser data in a self-supervised fashion. Our approach projects the obstacles identified from the laser data into the camera image and learns a classifier that considers color and texture information. While the robot is walking, it then applies the learned classifiers to the images to decide which areas are traversable. As we illustrate in experiments with a real humanoid, our approach enables the robot to reliably avoid obstacles during navigation. Furthermore, the results show that our technique leads to significantly more efficient navigation compared to extracting obstacles solely based on 3D laser range data acquired while the robot is standing at certain intervals.**

I. INTRODUCTION

Autonomous navigation with humanoid robots is still a challenging task. First, humanoids have only limited payload capabilities, which means that compact and light-weight sensors have to be used. Typically, this directly affects the possible precision and update rates of their sensors. While walking, the robot's observations are typically highly affected by noise due to the shaking behavior of humanoids. Second, depending on the placement of the individual sensors on the robot, the area in front of the robot's feet may not be observable while walking which raises the question of whether the robot can safely continue walking without colliding with unanticipated objects.

In this paper, we present an approach to obstacle detection for collision-free and efficient humanoid robot navigation based on monocular images and sparse laser range data. To detect arbitrary obstacles, we interpret sparse 3D laser data obtained from a 2D Hokuyo laser range finder installed in the robot's head (see Fig. 1). Obstacles close to the robot's feet are out of the laser scanner's field of view while walking, which inherently increases the risk of collisions. Thus, the robot regularly needs to stop, adopt a scanning position, and tilt its head to obtain reliable distance information about objects in the vicinity. This is time-consuming and leads to inefficient navigation.

All authors are with the Department of Computer Science, University of Freiburg, Germany. This work has been supported by the German Research Foundation (DFG) under contract number SFB/TR-8 and within the Research Training Group 1103 as well as by Microsoft Research, Redmond. Their support is gratefully acknowledged.

Fig. 1. Left: Humanoid robot Nao equipped with a Hokuyo laser scanner in its head. Middle: Navigating robot. Right: Corresponding image taken by the robot's onboard camera together with traversability labels estimated by our approach (green refers to traversable, red to non-traversable areas). This image is best viewed in color.

We therefore present a technique to train obstacle detectors based on the sparse laser data to interpret the images obtained from a monocular camera installed in the robot's head. Our approach projects detected objects from the range scans into the camera image and learns a classifier that considers color and texture information in a self-supervised fashion. While the robot is walking, it then applies the learned classifiers to the current camera image to decide which areas are traversable. Using this classification, the robot updates a local 2D occupancy grid map of the environment which it uses for path planning. Note that our approach does not require a learning phase before navigating – learning is done online in a self-supervised fashion. Our approach can be seen as an extension to the vision-based obstacle detection system proposed by Dahlkamp *et al.* [4]. In contrast to our method, their approach carries out a classification based on color only, whereas we additionally use texture information and consider the neighborhood relations between nearby areas in an image. Furthermore, we integrate the vision information over time to plan paths for the humanoid.

The experiments carried out with our humanoid robot illustrate that our approach enables the robot to reliably avoid obstacles while walking in the environment. The field of view of the robot is increased and it can detect dynamic obstacles in the scene. We furthermore present results demonstrating that using our technique, the robot reaches its goals significantly faster than with an approach that extracts obstacles solely based on 3D laser data acquired while standing at certain intervals.

II. RELATED WORK

We first discuss collision-avoidance techniques for humanoid robots. Several approaches only consider static obstacles while choosing actions leading the robot towards the goal [9], [10] or use an external tracking system to compute the position of objects blocking the robot's way [19], [13].

Michel *et al.* [12] apply a model-based technique to track individual objects in monocular images during walking. They have to manually initialize the objects before tracking starts. Cupec *et al.* [3] detect objects with given shapes and colors in monocular images and determine the robot's pose relative to these objects to adjust the trajectory accordingly.

Stachniss *et al.* [17] presented an approach to learn accurate 2D grid maps of large environments with a humanoid equipped with a laser scanner located in the neck. Such a map was subsequently used by Faber *et al.* [6] for humanoid localization and path planning in 2D. During navigation, the robot carries out a potential field approach to avoid obstacles sensed with the laser and ultrasound sensors located at the height of the robot's hip. Obstacles smaller than this height are not detected. Tellez *et al.* [20] use two laser scanners mounted on the robot's feet. The authors use 2D laser data to construct a 2D occupancy grid map which they use for path planning.

Chestnutt *et al.* [2] use 3D laser data acquired with a constantly sweeping scanner mounted at a pan-tilt unit on the humanoid's hip. The authors first extract planes to identify traversable areas and obstacles. Subsequently, they plan footstep paths in a height map of the environment. Such a setup can only be used on robots with a significantly larger payload than our Nao humanoid. Gutmann *et al.* [8] construct a 2.5D height map given stereo data and additionally use a 3D occupancy grid map to plan complex actions for the robot leading towards the goal.

Li *et al.* [11] proposed a vision-based obstacle avoidance approach for the RoboCup domain. The authors assume known shapes of the obstacle, i.e., the other field players. They use gradient features learned from training images and, also, apply a color-based classifier and data from ultrasound sensors to determine obstacle positions. Their approach relies on a specific color coding further simplifying the problem.

Plagemann *et al.* [15] and Michels *et al.* [14] presented approaches to estimate depth from monocular images. These approaches apply regression techniques to learn a mapping from the feature space (pixel columns and edge-based features [15], texture [14]) to distance and yield impressive results. However, both require a prior learning phase before the classification can start whereas our approach performs a self-supervised learning during navigation.

Ulrich and Nourbakhsh *et al.* [21] proposed to learn color histograms of pixels corresponding to the floor. The histograms are learned while initially steering a wheeled robot equipped with a monocular camera through the environment and learning about the free-space.

Fazl-Ersi and Tsotsos [7] use stereo images to produce dense information about floor and obstacles in the images. They classify regions of neighboring pixels with similar visual properties and consider their distances form the ground plane using planar homography. Subsequently, they learn color-based models for the floor and obstacles. There exist further techniques relying on stereo data to detect *moving* obstacles, i.e., walking people, for which special detectors are trained (e.g., [5]). These approaches are, however, computationally highly demanding.

Our approach is most closely related to the one presented by Dahlkamp *et al.* [4]. They also use 3D laser data to learn a vision-based obstacle classifier. In contrast to their work, we combine different information, i.e., color and texture, to distinguish between obstacles and the floor. We then apply probabilistic relaxation labeling [16], [18] for considering dependencies between nearby areas in an image. Finally, we construct a local grid used for path planning in which the vision-based obstacle information is integrated over time.

III. THE HUMANOID ROBOT NAO

The humanoid robot Nao is 58 cm tall, weighs 4.8 kg and has 25 degrees of freedom. Aldebaran Robotics developed in cooperation with our lab a laser head for this type of robot. Thus, our humanoid is equipped with a Hokuyo URG-04LX laser range finder mounted in the modified head, in addition to the default sensors such as cameras. See Fig. 1 for an illustration. While the measurements of this laser sensor are relatively noisy, it is small and lightweight and allows a field of view of 240° with a resolution of 0.33°. In our implementation, we use the top camera in the robot's head. The camera's diagonal field of view is 58°.

IV. OBSTACLE DETECTION USING VISION AND SPARSE LASER DATA

For collision-free navigation, it is important that the robot can sense its surroundings and distinguish traversable from non-traversable areas. In this section, we describe our approach to obstacle detection using monocular images and sparse laser data which is the main contribution of this paper. The robot continuously receives 2D range data from the laser sensor in its head with a frequency of approx. 10 Hz. In order to obtain 3D data, the robot has to stop walking and to slowly tilt its head. In this way, a sweeping laser line is obtained and the robot's surroundings are scanned. The goal is now to use the continuous flow of image data together with the rather seldom obtained 3D range data to navigate without collisions. This is done by learning classifiers using the laser data to automatically generate training data. Fig. 2 illustrates an overview of our system.

A. Classification of Laser Data

The first task is to identify obstacles in the laser range data. To achieve that we analyze the traversability of the area around the robot in a two-step procedure: (i) identify the ground plane and (ii) label areas as obstacles which show a significant difference in height to the ground plane.

For the first step, we insert the 3D end points measured by the laser scanner into a 2D grid structure (xy plane) and

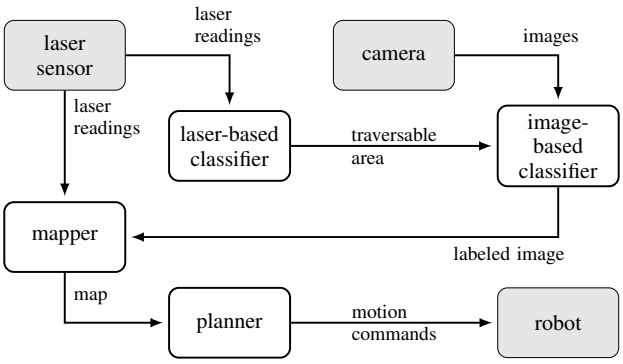

Fig. 2. Overview of the proposed system.

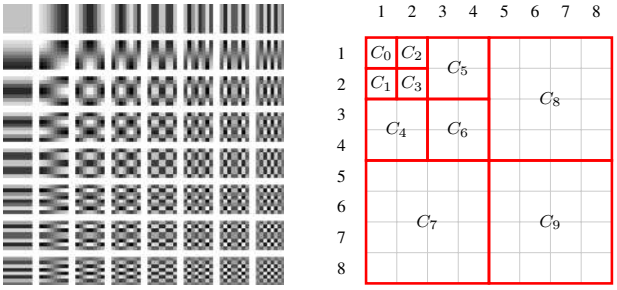

Fig. 3. Left: Basis functions of the 2D DCT for a 8×8 image. Right: Scheme for texture feature extraction using DCT coefficients. The illustration shows the matrix D from DCT coefficients for an 8×8 image.

compute the mean elevation for each cell. Under the assumption that the robot stands on the ground plane, we perform a region growing. This procedure expands a region by adding neighboring grid cells to the region if they show similar elevation than the current cell under consideration. After convergence, the ground plane can be obtained by means of principal component analysis based on the 3D laser points that fall into the grid cells belonging to the ground plane according to the region growing procedure. By ignoring the Eigenvector that corresponds to the smallest Eigenvalue (which should have a value close to zero), we obtain the ground plane. Subsequently, we perform a further assignment of the grid cells in the elevation map to the parametric representation of the ground plane. This is important since the region growing algorithm will not reach all cells due to gaps in the map resulting from the sparse laser data. All areas that show a deviation from the ground plane that is not compatible with the walking capabilities of the robot, are labeled as not traversable and all others as traversable.

In case the robot would be able to constantly obtain full 3D range scans while walking, it would be sufficient to navigate solely based on the this representation. However, since obtaining an 3D scan requires the robot to stop and is therefore time-consuming, 3D data can only be acquired seldom during navigation. Therefore, we use the camera of the humanoid as the main sensor to estimate the traversability while walking.

B. Classification of Image Data

The goal is to estimate from the camera images which parts of the robot's surrounding are traversable so that obtaining time-consuming 3D scans can be avoided as much as possible. We achieve that by relating the image data to an initial 3D scan of the environment and learn classifiers to estimate the traversability in a self-supervised fashion.

1) Training Data: For training our classifiers, we need camera images together with the information which pixels in the image correspond to traversable and which ones to not traversable parts of the environment. The idea of our approach is to use the laser data whenever it is available to train the image-based classifiers. This is done in a self-supervised approach by assigning to each pixel a traversability label based on the projection of the classified laser data into the image.

2) Traversability Estimation based on Color Information: In most indoor environments in which humanoid robots are operating, color information provides a good estimate about traversability. To be less sensitive to illumination changes, we operate only in the HSV color space, i.e., we use the hue and saturation values.

Based on the training data, which provides for every pixel of an image the corresponding traversability information, we learn a distribution of color values for each traversability class. A natural way of modeling such distributions from digital images are histograms. Each bin corresponds to a color and, thus, such a histogram is obtained by simply considering the individual pixels and updating the bins accordingly. After counting the color occurrences, we smooth the bins with a Gaussian kernel and normalize the histogram.

Once the histograms are generated, we can determine the likelihood that an intensity value of a pixel is generated by the individual classes. Let t be the variable indicating traversability, and i_h and i_s the intensity values of the hue- and saturation-channels, respectively. If we assume a uniform distribution of $P(t)$, $P(i_h)$, and $P(i_s)$ and independence of i_h and i_s, we can evaluate the likelihood of traversability for each pixel as

$$\begin{aligned} P(t \mid i_h, i_s) &= P(i_h \mid t, i_s) P(t \mid i_s) P(i_h \mid i_s)^{-1} && (1) \\ &= P(i_h \mid t, i_s) P(i_s \mid t) P(t) \\ &\quad \cdot P(i_h)^{-1} P(i_s)^{-1} && (2) \\ &\propto P(i_h \mid t) P(i_s \mid t). && (3) \end{aligned}$$

3) Texture-based Classification: Besides color information, texture is also a source of information that can be used for the classification of the image data. Therefore, we use the same training data as before and seek to exploit also texture information for determining the traversability label.

One feature for describing texture is the discrete cosine transformation (DCT) [1]. For an input image, the DCT computes a set of coefficients which can be regarded as weights of a set of two-dimensional basis functions. Each basis function is an image constructed from a two-dimensional cosine function with a different frequency. As an illustration, the basis functions for a 8×8 image are shown in the left

image of Fig. 3. The DCT transforms an input image into an image of the same size where every pixel corresponds to a DCT coefficient representing the amount of presence of a certain frequency in the original image. The frequencies increase horizontally to the right and vertically to the bottom. Therefore, the lower-right part of the transformed image contains information about the high frequency content of the image. One can observe that by considering only a small subset of the coefficients, mainly the low to mid frequency parts, an input image can already be reconstructed surprisingly well.

In the following, we describe how to use the DCT to learn a traversability classification based on texture information. Since texture information is not available per pixel, we divide the input image's hue-channel into overlapping patches of size 16×16 computed at the fixed distance of 8 pixels in vertical and horizontal direction. Each patch is assigned a traversability label t, based on the percentage of labeled pixels inside the patch using the classified laser data. If more than a certain percentage θ_P (which we chose as 90%) of the pixels in that image patch are labeled as traversable, we label it as example for traversable texture. Analogously, if more than θ_P percent of the pixels in the patch are labeled non-traversable, we assign the label non-traversable to the patch. If neither condition holds, for example at the boundaries of obstacles, the patch is not used for self-supervised learning. From the labeled image patches, we compute a feature vector f_{DCT} based on the DCT transform of the patch.

The feature vector f_{DCT} is computed as follows. Let P be such a patch of size 16×16 and D be the DCT of P. Let C_i represent the set of all the DCT coefficients in the corresponding marked region of D, according to Fig. 3. For example C_0 is the DCT coefficient located at $D_{1,1}$ and C_4 is the set of the DCT coefficients located at $D_{1,3}, D_{1,4}, D_{2,3}$, and $D_{2,4}$, etc. Further, let M_i and V_i be the average and the variance over all coefficients in C_i, respectively. We then define f_{DCT} as

$$f_{\text{DCT}} = (M_0, M_1, M_2, M_3, V_4, V_5, \ldots, V_{12}). \tag{4}$$

Using this form of feature computation, we represent the visually significant low frequency coefficients directly and accumulate the less significant high frequency components by their variance. From these feature vectors, together with the corresponding traversability label for the patches, we train a support vector machine (SVM). The SVM learns a function $p_T : \mathbb{R}^{12} \mapsto [0, 1]$, where $p_T(f_{\text{DCT}})$ is the likelihood that the feature vector f_{DCT} represents traversable area.

To predict the traversability of regions in an image, we first extract image patches with their corresponding feature vectors. Then we evaluate p_T for all these patches. As the patches overlap, we assign to all pixels (x, y) of the image the average over all $p_T(x, y)$ obtained from the patches containing (x, y).

C. Smoothing via Relaxation Labeling

A classification based on the individual classifiers above is not perfect. Often small spurious classification errors exist which can actually prevent the robot from collision-free navigation using the techniques presented above. In both approaches above, we ignored the dependencies between nearby areas. One way of taking neighborhood information into account and to combine both classifiers is probabilistic relaxation labeling as proposed by [16].

Probabilistic relaxation labeling works as follows. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph consisting of nodes $\mathcal{V} = \{v_1, \ldots, v_N\}$ and edges $\mathcal{E}$ between pairs of nodes. In our setting, the nodes correspond to the small rectangular image patches and the edges describe their neighborhood relations.

Let furthermore $\mathcal{T}$ be the set of possible labels, in our case traversable and non-traversable. We assume that each node v_i stores a probability distribution about its label, represented by a histogram P_i. Each bin $p_i(t)$ of that histogram stores the probability that the node v_i has the label t. For two classes, P_i can efficiently be represented by a binary random variable.

For each node v_i, the neighborhood $\mathcal{N}(v_i) \subset \mathcal{V}$ refers to the nodes v_j that are connected to v_i via an edge. In our case, we assume an eight-connected graph of neighborhood relations. That means that each local area only influences its 8 neighbors. Each neighborhood relation is represented by two values. One describes the compatibility between the labels of both nodes and the other one represents the influence between the nodes. The term $\mathcal{R} = \{r_{ij}(t, t') \mid v_j \in \mathcal{N}(v_i)\}$ defines the compatibility coefficients between the label t of node v_i and the label t' of v_j. Finally, $\mathcal{C} = \{c_{ij} \mid v_j \in \mathcal{N}(v_i)\}$ is the set of weights indicating the influence of node v_j on node v_i.

Given an initial estimation for the probability distribution over traversability labels $p_i^{(0)}(t)$ for the node v_i, the probabilistic relaxation method iteratively computes estimates $p_i^{(k)}(t)$, $k = 1, 2, \ldots$, based on the initial $p_i^{(0)}(t)$, the compatibility coefficients $\mathcal{R}$, and the weights $\mathcal{C}$ in the form

$$p_i^{(k+1)}(t) = \frac{p_i^{(k)}(t)\left[1 + q_i^{(k)}(t)\right]}{\sum_{t' \in \mathcal{T}} p_i^{(k)}(t')\left[1 + q_i^{(k)}(t')\right]}, \tag{5}$$

where

$$q_i^{(k)}(t) = \sum_{j=1}^{8} c_{ij} \left[\sum_{t' \in \mathcal{T}} r_{ij}(t, t') p_j^{(k)}(t')\right]. \tag{6}$$

The compatibility coefficients $r_{ij}(t, t')$ take values between -1 and 1. A value $r_{ij}(t, t')$ close to -1 indicates that the label t' is unlikely at the node v_j given that the node v_i has label t. Values close to 1 indicate the opposite.

Probabilistic relaxation provides a framework for smoothing but does not specify how the compatibility coefficients are computed. In our work, we apply the coefficients as defined by Yamamoto [22]

$$r_{ij}(t, t') = \begin{cases} \frac{1}{1 - p_i(t)}\left(1 - \frac{p_i(t)}{p_{ij}(t \mid t')}\right) & \text{if } p_i(t) < p_{ij}(t \mid t') \\ \frac{p_{ij}(t \mid t')}{p_i(t)} - 1 & \text{otherwise,} \end{cases}$$

where $p_{ij}(t \mid t')$ is the conditional probability that node v_i has label t given that node $v_j \in \mathcal{N}(v_i)$ has label t' (which

Fig. 4. Left: scene from the robot's view, 2nd left: top view, 3rd left: scene changed while navigating, right: labeled image from the robot's camera.

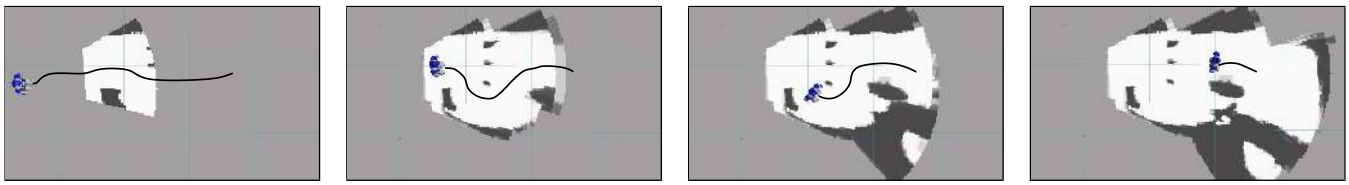

Fig. 5. Maps and planned trajectories of the robot while navigating. Left: initially built map (corresponds to the 1st and 2nd image in Fig. 4), 2nd left: map after new obstacle has been detected (corresponds to the 3rd and 4th image in Fig. 4), 3rd and 4th: updated map while approaching the goal.

we determined by counting given the training data). The initial probabilities $p^{(0)}_{(x,y)}(t)$ are obtained by averaging over the outputs from the individual classifiers described above. Each of the weights c_{ij} is initialized with the value $\frac{1}{8}$, indicating that all the eight neighbors v_j of node v_i are equally important. After termination, we obtain the final classification as an image I_p, where $I_p(x, y)$ represents the probability that pixel (x, y) is traversable.

D. Re-training the Classifiers

Obviously, the learned classifiers need to be re-trained whenever the appearance of the scene changes. Such a change can be detected by monitoring the color/texture feature distributions over time. In our current implementation, we follow a heuristic approach that triggers the re-learning in case the histogram correlation between the current color histogram computed over the whole image and the one obtained during the previous learning step shows a value lower than 0.5. This is clearly a heuristic but appears to work well in our experiments.

E. Map Update and Motion Planning

For locally planning the motion of the robot, we use a 2D occupancy grid map to combine the laser data as well as traversability information from the classified camera data.

To integrate the traversability information from the camera images, we first compute a homography H induced by the floor plane between the camera's image plane and a virtual camera's image plane which is looking perpendicular at the floor plane from a far distance. This allows us to construct a bird's eye view from the robot's camera image. By applying the homography H to the labeled image I_p and by using bilinear interpolation, the traversability information from the image I_p is mapped to the coordinate frame of the occupancy map. The rest of the occupancy grid update is straightforward. An example of such a map is shown in Fig 5.

For planning the robot's motion, we apply the A*-algorithm based on this 2D map. To drag the robot away from obstacles, we efficiently compute for each cell the distance to the closest obstacle using the Euclidean distance transform. If the distance for a cell is larger than one robot's radius plus a safety margin, the cell is not considered as traversable during planning. The optimal collision-free path to a goal location leads through the remaining cells and is computed by A*.

V. EXPERIMENTS

The experimental evaluation of our approach is designed to show that our robot can detect obstacles using its self-supervised image classifiers and thereby reduce the number of 3D range scans that need to be acquired. We evaluate the accuracy of our system and show that our approach allows the robot to navigate faster to the desired goal location.

A. Obstacle Avoidance

The first experiment illustrates the functionality of our visual obstacle avoidance system. We placed obstacles on the floor in our lab and let the robot navigate through the scene. The robot first took a 3D range scan to train its classifiers, and then started navigating and updating its map based on the visual input. In the example shown in Figs. 4 and 5, we placed an obstacle in front of the robot after it started to navigate. The left image of Fig. 4 shows the initial scene, the target location was close to the second (red) humanoid robot. The second image shows a top view of the partial scene at the time when the robot was taking the 3D scan. The third image shows the same scene after placing the ball in the way of the robot. The right image of Fig. 4 shows a corresponding labeled image recorded by the robot. In addition to that, Fig. 5 illustrates the updated grid map and the trajectories planned by the Nao robot at the different points in time.

B. Classification Accuracy

Fig. 1 and Fig. 6 show qualitative classification results achieved in different environments. To evaluate the accuracy of the image-based classifiers, we set up two different

Fig. 6. Two examples of obtained classification. Left: external camera view for reference, right: classified onboard camera image (best viewed in color).

scenarios on two different floor surfaces and placed various obstacles on the ground. The robot's task was to navigate through the scene. First, the robot took one 3D range scan, trained its classifiers, and then used only the camera to map its surroundings and to plan a trajectory to the given goal location. We repeated the experiments 12 times. In case the robot detected substantial changes in the appearance of the scene, re-training was triggered automatically based on the histogram correlation as explained in Sec. IV-D. The images in Fig. 7 illustrate an example in which the re-training is carried out.

For the evaluation, we saved an image every 10 seconds while the robot was navigating and manually labeled each pixel in these images. Whenever the probability for a pixel corresponding to an obstacle was bigger than 0.5, we counted it as obstacle and as free space otherwise. We then compared these results to the manual labels. The obtained accuracy in terms of confusion matrices is shown in Table I. As can be seen, the probability that a pixel corresponding to an obstacle was classified as free space lies between 1% and 4%.

We also tested the influence of noise, induced by shaking movements of the robot, on the classification rates. To better control the noise, we applied a motion blur filter to a set of images. We observed that the classification rates for obstacles decreased from approximatively 0.98 to 0.91, compared to the original data set without blur, while the floor detection rates remained unchanged. During this experiment, we chose the parameters of the blur filter to add stronger blur than the worst case we observed in practice with our humanoid robot.

C. Improved Robot Navigation

Three claims for using the approach presented in this paper are made: First, the robot can better observe the area in front of its feet which is the most important part during navigation. Second, the robot can move faster overall since it needs to stop less often to acquire a 3D range scan to check for obstacles. Third, it can instantly react to dynamic obstacles in the scene.

TABLE I

EVALUATION OF THE IMAGE CLASSIFIERS. CONFUSION MATRICES FOR ALL CLASSIFIERS DURING TWO EXPERIMENTS.

	experiment 1		experiment 2	
	TEXTURE CLASSIFIER			
	estimated as		estimated as	
true class	obstacle	floor	obstacle	floor
obstacle	0.84	0.16	0.99	0.01
floor	0.12	0.88	0.10	0.89
	COLOR CLASSIFIER			
	estimated as		estimated as	
true class	obstacle	floor	obstacle	floor
obstacle	0.97	0.03	0.99	0.01
floor	0.20	0.80	0.04	0.96
	COMBINED APPROACH			
	estimated as		estimated as	
true class	obstacle	floor	obstacle	floor
obstacle	0.96	0.04	0.99	0.01
floor	0.07	0.93	0.07	0.93

Fig. 7. Left: scene during training (robot outside the cameras field of view). Middle: new but similar-looking objects do not trigger re-training. Right: Adding the carpet is interpreted as a substantial change and re-training is triggered.

The first claim does not require experimental support. Based on the geometry of the robot and its sensors, the closest sweep line of the laser is 0.84 m away from the robot's feet while walking. Using the camera in the robot's head, this distance reduces to 0.45 m.

To support the second claim, we compared the overall travel time for our Nao robot using only laser data with the proposed method. Without vision, the robot has to record a 3D laser scan every 1.3 m to 1.4 m since this is the distance in which the floor can be observed in sufficient detail with the Hokuyo scanner[1]. Note that such 3D scans are needed for navigation if no vision-based classifiers are used. Constructing a consistent 3D model while walking and continuously nodding, however, is challenging and was not possible in our setup. The reason is the minimal overlap in consecutive 2D laser scans while nodding in combination with the shaking movement of the robot.

In this set of experiments, the task of the robot was to travel through an initially empty corridor. We first uniformly

[1]For larger distances, typical floors (wood or PVC) provide poor measurements with the Hokuyo due to the inclination angle. A travel distance of 1.3 m between 3D scans was used in our navigation system before implementing the approach presented here.

TABLE II

TRAVEL TIME WITH AND WITHOUT OUR VISION-BASED SYSTEM.

technique	travel time (5 runs)					avg.
3D laser only	219s	136s	208s	135s	135s	167s
laser & vision	136s	94s	120s	96s	87s	107s

sampled the robot's goal location, the number of obstacles (from 1 to 3) and their positions. After placing the obstacles in the scene, we measured the time it took the robot to navigate through the corridor with and without our vision-based system. The experiment was repeated five times. The timings for each experiment are depicted in Table II and, as can be seen, our approach requires on average 107 s compared to 167 s. We also carried out a paired two sample t-test with a 0.999 confidence level. The test shows that this result is statistically significant ($t_{value} = 5.692 >$ t-table$(\text{conf}{=}0.999; \text{DoF}{=}8) = 4.501$).

Using the camera data, the robot can furthermore react more quickly to dynamic changes in the scene (third claim). Our current implementation runs with a frequency of 4 Hz and thus the robot can react every 250 ms based on new camera data. Using solely the 3D laser data, the robot only updates its model after traveling for 1.3 m. Obviously, one can increase the frequency in which 3D scans are recorded. This, however, leads to significantly higher travel times.

D. Limitation of our Approach

Finally, we discuss limitation of our approach. Obstacles looking identical or very similar (same color and same texture) to the ground will prevent the system from learning robust classifiers to distinguish ground from obstacles. One way to detect this, is to classify the labeled training images directly after learning the classifiers. In case of large errors on the training data, the robot can switch back to the strategy of using only 3D range scans.

Furthermore, moving obstacles in the environment during the acquisition of the 3D range scan impose a problem to the training of the image classifiers. In this case, the training image and the classified range data do not correspond. However, this situation would also cause problems if navigation was based solely on 3D laser data. In future work, we want to investigate how these effects can be reduced, e.g., by identifying areas where obstacles have moved and discard them as training data. Note that dynamic obstacles during *navigation* are not problematic to the approach.

VI. CONCLUSIONS

In this paper, we presented an approach to combine sparse laser data and visual information for obstacle avoidance on a humanoid robot. Our method allows the robot to train classifiers for detecting obstacles in the camera images in a self-supervised fashion. Based on this information, the robot can navigate more efficiently and avoid obstacles. Our approach provides the humanoid with a better field of view, leads to a reduced travel time, and allows to deal with changes in the scene.

ACKNOWLEDGMENTS

The authors would like to acknowledge Armin Hornung and Christoph Sprunk for their help in the context of humanoid robot navigation and Andreas Ess for fruitful discussions on the topic.

REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transfom. *IEEE Transactions on Computers*, 23(1):90–93, 1974.

[2] J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. Kuffner, and S. Kagami. Biped navigation in rough environments using on-board sensing. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

[3] R. Cupec, G. Schmidt, and O. Lorch. Experiments in vision-guided robot walking in a structured scenario. In *Proc. of the IEEE Int. Symp. on Industrial Electronics*, 2005.

[4] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. In *Proc. of Robotics: Science and Systems (RSS)*, 2006.

[5] A. Ess, B. Leibe, K. Schindler, and L. van Gool. Moving obstacle detection in highly dynamic scenes. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2009.

[6] F. Faber, M. Bennewitz, C. Eppner, A. Goeroeg, A. Gonsior, D. Joho, M. Schreiber, and S. Behnke. The humanoid museum tour guide Robotinho. In *Proc. of the 18th IEEE Int. Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2009.

[7] E. Fazl-Ersi and J.K. Tsotsos. Region classification for robust floor detection in indoor environments. In *Proc. of the Int. Conf. on Image Analysis and Recognition (ICIAR)*, 2009.

[8] J.-S. Gutmann, M. Fukuchi, and M. Fujita. 3D perception and environment map generation for humanoid robot navigation. *Int. Journal of Robotics Research (IJRR)*, 27(10):1117–1134, 2008.

[9] A. Hornung, M. Bennewitz, and H. Strasdat. Efficient vision-based navigation – Learning about the influence of motion blur. *Autonomous Robots*, 29(2), 2010.

[10] J. Ido, Y. Shimizu, Y. Matsumoto, and T. Ogasawara. Indoor navigation for a humanoid robot using a view sequence. *Int. Journal of Robotics Research (IJRR)*, 28(2):315–325, 2009.

[11] X. Li, S. Zhang, and M. Sridharan. Vision-based safe local motion on a humanoid robot. In *Workshop on Humanoid Soccer Robots*, 2009.

[12] P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade. GPU-accelerated real-time 3D tracking for humanoid locomotion and stair climbing. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

[13] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade. Vision-guided humanoid footstep planning for dynamic environments. In *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2005.

[14] J. Michels, A. Saxena, and A.Y. Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2005.

[15] C. Plagemann, C. Stachniss, J. Hess, F. Endres, and N. Franklin. A nonparametric learning approach to range sensing from omnidirectional vision. *Robotics & Autonomous Systems*, 58:762–772, 2010.

[16] A. Rosenfel, R.A. Hummel, and S.W. Zucker. Scene labeling by relaxation operations. *IEEE Trans. Systems. Man. Cybernet*, 6(6):420–433, 1976.

[17] C. Stachniss, M. Bennewitz, G. Grisetti, S. Behnke, and W. Burgard. How to learn accurate grid maps with a humanoid. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2008.

[18] C. Stachniss, O. Martínez-Mozos, A. Rottmann, and W. Burgard. Semantic labeling of places. In *Proc. of the Int. Symp. of Robotics Research (ISRR)*, San Francisco, CA, USA, 2005.

[19] M. Stilman, K. Nishiwaki, S. Kagami, and J. Kuffner. Planning and executing navigation among movable obstacles. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2006.

[20] R. Tellez, F. Ferro, D. Mora, D. Pinyol, and D. Faconti. Autonomous humanoid navigation using laser and odometry data. In *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2008.

[21] I. Ulrich and I. Nourbakhsh. Appearance-based obstacle detection with monocular color vision. In *Proc. of the National Conf. on Artificial Intelligence (AAAI)*, 2006.

[22] H. Yamamoto. A method of deriving compatibility coefficents for relaxation operators. *Compt. Graph. Image Processing*, 10, 1979.