Appearance-Based Traversability Classification in Monocular Images Using Iterative Ground Plane Estimation

Daniel Maier Maren Bennewitz

Abstract— **In this paper, we present an approach to traversability classification solely based on monocular images and odometry estimates. We iteratively estimate the ground plane by detecting and matching features. Since the features are only sparse in the images and do not lead to dense information about traversability, we present a technique to train appearance-based floor detectors. In this way, we achieve a dense classification of the image data. Our approach trains the classifiers online in a self-supervised fashion from the ground plane estimation. During robot navigation, the classifiers are automatically updated and applied to the image stream to decide which areas are traversable. From this information, the robot can compute a two-dimensional occupancy grid map of the environment and use it for planning collision-free paths. As we illustrate in thorough experiments with a real humanoid, the classification results of our approach are highly accurate and the resulting occupancy map enables the robot to reliably avoid obstacles during navigation. Our appearance-based classifiers can also be used to augment stereo or RGBD-data in close ranges where these sensors cannot provide any depth information.**

I. Introduction

Collision-free navigation is one of the most essential capabilities of autonomous robots. Typically, they use distance data, i.e., from laser range finders or stereo cameras to detect obstacles. Recently, depth cameras such as the Microsoft Kinect or Asus Xtion have become available on the consumer market and have been used for navigation with mobile robots (e.g., [1]). However, monocular cameras are still the typical sensor, light-weight robots are equipped with.

In this paper, we extend our previous work on obstacle detection that operated on monocular images and sparse laser data [2]. In that approach, the robot trained visual classifiers for obstacles and the floor based on classified distance data, which lead to highly accurate image labeling and traversability classification results. Since not all humanoid robots possess a laser range finder whose data can be used to acquire training data, we developed a new approach that relies solely on monocular images and odometry data. The approach does not make use of any distance data. Our method finds sparse features on the floor plane from geometric information in an iterative fashion and deploys these features to train appearance-based classifiers. The classifiers are subsequently applied to densely label the camera images during walking. From the labeled pixels, we learn a traversability map of the robot's surroundings in form of a 2D occupancy grid map in which a collision-free path can be computed. The learned classifiers are constantly updated during navigation. In this way, the robot can deal with changing ground appearance and reliably identify the traversable floor.

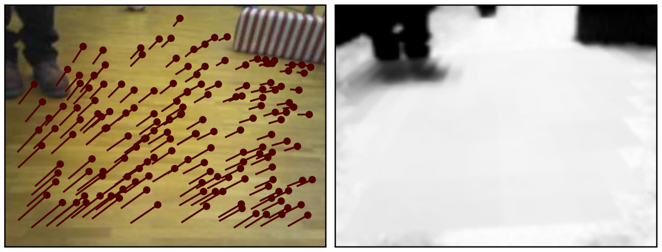
Fig. 1. Left: Image captured by the camera of a walking humanoid robot with flow vectors of detected features on the estimated ground plane. Given the floor features, we train visual classifiers to densely label patches of subsequent images as being traversable or not. Right: Resulting traversability probabilities for the successive camera image as estimated by our approach (the brighter the color, the higher the probability of being traversable).

The left image of Fig.1 shows an image acquired from the onboard camera of a walking robot. The circles show the location of detected features on the floor plane. Due to the motion of the camera, the locations of the features in the image change from one image to another as indicated by the flow vectors. Our approach iteratively estimates the ground plane based on the detected flow. From the floor features, we then train visual classifiers to label the traversable area in successive camera images (right image of Fig.1).

Several approaches for estimating traversability from a moving monocular camera have been proposed [3, 4, 5, 6]. Most of them try to recover structural information about the environment from the camera motion. These methods often rely on simplifying assumptions such accurate knowledge of the camera or smooth motions that do not hold for walking humanoid robots. Further popular approaches relying only on feature correspondences (e.g., [7]) are unable to cope with rotations when the translation is small and hence, are only of limited use for humanoid navigation. In contrast to that, our technique provides a dense labeling of the images from which the traversable areas can be inferred, while the robot is walking, standing, or turning on the spot.

As the experiments with a Nao humanoid demonstrate, our approach achieves high classification rates and leads to an accurate, dense labeling of the camera images. The resulting occupancy grid map allows for collision-free navigation of the robot.

II. Related Work

Several techniques have been presented for obstacle detection based on monocular image data. Broadly, these approaches can be categorized according to whether they operate directly on the appearance level or try to recover structural information from a moving camera. Our approach can be considered as a hybrid technique as it acquires training data from structural information but reaches its traversability decision based on appearance. We first discuss related work focusing on structure from motion techniques and then discuss appearance-based approaches.

Einhorn *et al.* [3] recover the scene's geometry by tracking features in consecutive images from the robot's onboard camera. Reconstructing the scene requires accurate pose estimates of the camera, which the authors obtain from odometry and filtering. This approach is applicable for wheeled robots. Humanoids, typically provide only poor odometry information.

Wang *et al.* [8] and Braillon *et al.* [4] presented approaches that segment a camera image into different regions and apply a given model of floor motion to these regions. A region is considered as traversable if it can be matched in the consecutive image at the computed location based on the assumed floor motion. These methods also rely on accurate odometry data to estimate the floor model.

Kim and Kim [5] use dense optical flow to compute the plane normals for small image regions. Traversability is estimated from the normals' consistencies with the estimated normal of the ground plane. Obtaining accurate dense optical flow for images from a walking humanoid's onboard camera is difficult to obtain even with a state-of-the-art approach [9] as we found out in experiments. Further approaches based on optical flow to detect obstacles make certain assumptions about the environment that are not generally applicable. For example, Low and Wyeth [6] presented a method that relies on the assumption that optical flow only occurs at obstacles. Hence, the approach is unsuitable for environments with textured floor such as parquet.

Pretto *et al.* [10] presented a framework for visual odometry based on specifically designed feature detectors that can cope with effects of motion blur typically occurring during humanoid walking. The detected features are sparse in the image and hence not sufficient for reliable obstacle detection.

Common to all these approaches is that they rely on information from a moving camera. Appearance-based approaches, on the other hand, classify the camera image using only visual information. For instance, Ulrich and Nourbakhsh [11] use color histograms to model traversable areas. They learn the model by retrospectively updating a histograms over the color of the floor from a robot's camera while manually steering the robot through the free space.

Michels *et al.* [12] presented an approach to estimating depth from a single monocular image. The approach applies a regression techniques to learn a mapping from the feature space to distance. The drawback is that the approach requires a prior training period whereas our approach carries out automatic learning while navigating.

Newcombe *et al.* [13] presented *DTAM*, a system for monocular camera-based tracking and dense reconstruction and showed impressive results. The system reconstructs a scene by minimizing a photometric cost function, computed from multiple images. The approach relies on a static world assumption and on brightness consistency. Further, it requires a special initialization stage to construct an initial environment model. This is because an accurate model is needed for tracking and subsequent mapping. It is questionable how well the system performs in case of rapid exploration (i.e., quick movements), as is typically the case in mobile robotics.

Previously, we presented an approach where a humanoid constantly adopts a scanning position and tilts its head to acquire 3D range data from a laser scanner on top of the head [2]. Afterwards the robot trains visual classifiers for obstacles and the floor based on classified laser data projected to the camera images. While this technique leads to highly reliable classification results, a drawback is that collecting the 3D data is time-consuming. Furthermore, not all robots possess a laser scanner that can be used to acquire the training data. Therefore, we present a new approach that uses only monocular images and odometry information from which dense information about traversability is estimated.

III. Preliminaries

In this section, we briefly describe the mathematical background [14] and notations used in this paper. Readers familiar with multi-view geometry can skip this section. Our approach learns appearance-based classifiers from features on the floor plane. We identify these features using an iterative technique based on homography estimation.

A homography is an invertible, projective mapping between two planes. Here, we consider the special case that corresponding points $\mathbf{x} \leftrightarrow \mathbf{x}'$ in two images of a plane $\boldsymbol{\pi}$ in 3D are mapped to each other by a homography $H \in \mathbb{R}^{3\times3}$. This means, that $\mathbf{x}$ and $\mathbf{x}'$ represent the same point on the plane $\boldsymbol{\pi}$ in 3D but in different camera views. We call H the homography induced by $\boldsymbol{\pi}$ and its defining equation is

$$\tilde{\mathbf{x}} = H\,\tilde{\mathbf{x}}', \tag{1}$$

where $\tilde{\mathbf{x}}$ indicates that $\mathbf{x}$ is expressed in homogeneous coordinates.

A homogeneous representation of a point $\mathbf{x} = (x, y)^T$ in 2D can be obtained by adding an additional coordinate with value 1, i.e., $\tilde{\mathbf{x}} = (x, y, 1)^T$. The same principle is easily extended to higher dimensions. It is important to note that there exists an equivalence class of homogeneous coordinates that all represent the same point. Two homogeneous points are equivalent if they differ only by a scale factor $\lambda \in \mathbb{R}\backslash\{0\}$. Thus, from a homogeneous representation $\tilde{\mathbf{x}} = (x_1, x_2, x_3)^T$, we can go back to the original representation $\mathbf{x} = (x, y)^T$ by defining $x = \frac{x_1}{x_3}$ and $y = \frac{x_2}{x_3}$. The advantage of homogeneous coordinates is that all affine transformations can be expressed by a single matrix multiplication.

For instance, for a point in 3D given in homogeneous coordinates $\tilde{\mathbf{X}}$, its homogeneous image point is given by

$$\tilde{\mathbf{x}} = M\ [R \mid t]\,\tilde{\mathbf{X}}, \tag{2}$$

where $R \in \mathbb{R}^{3\times3}$ is a rotation matrix and $t \in \mathbb{R}^{3\times1}$ is a translation vector. $[R \mid t]$ is the extrinsic camera matrix transforming points from the world coordinate system to the camera coordinate system. The matrix $M \in \mathbb{R}^{3\times3}$ is the

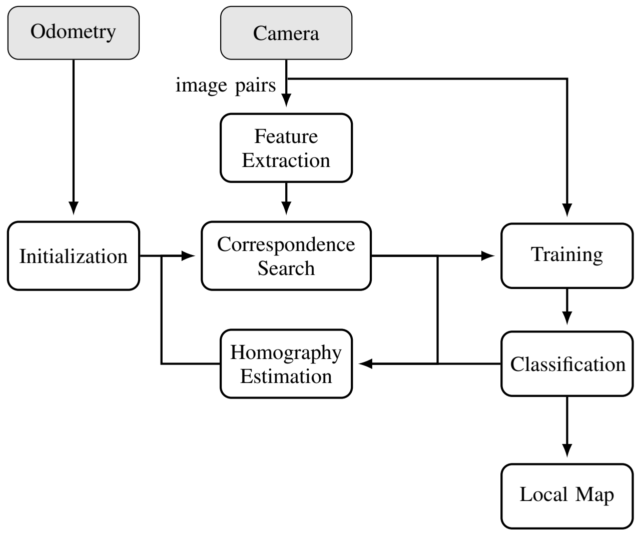

Fig. 2. Overview of the proposed system. We compute corresponding features in image pairs by iteratively estimating the ground homography and matching corresponding features. Features identified as belonging to the floor are then used to train appearance-based visual classifiers. These classifiers are applied to estimate dense traversable areas in the images and to guide the homography estimation in subsequent image pairs.

intrinsic matrix of the camera and projects points to the image plane. Here, we assume that the standard pinhole model for M is applicable and the camera images are undistorted. In the following, we abandon specific indication of homogeneous coordinates for sake of comprehensibility.

IV. Approach

Fig. 2 gives an overview of our system. We extract features that are matched between image pairs and used to iteratively estimate the floor homography. The homography is initialized from odometry to guide the first matching step. Once the correspondences on the floor plane are established, we interpret the sparse feature points as training examples for the appearance of the floor and train visual classifiers. Subsequently, the classifiers are applied to assign traversability labels to successive images, which we then use to construct a local map of the scene around the robot for navigation.

V. Geometric Floor Estimation

For identifying points on the floor, we match features that are compatible with the floor homography in image pairs captured by a camera while the robot is moving. One of the biggest problems with extracting and matching features between images from a moving camera is the occurrence of motion blur. Especially humanoid robots often move in a jerky way that induces blur effects in the camera image. Hence, we first introduce our technique to acquire steady images that further have the maximum possible baseline.

A. Identification of Steady Images While Walking

To identify phases in which the camera is steady, we exploit the specific movement of humanoid robots, which often sway sideways during walking. When the direction of the swaying changes, the body has to undergo a short phase with zero velocity before it sways the other way. Our algorithm tries to detect these phases and extracts the camera images at that moment. A further advantage of this approach is that the recorded camera images have the maximum possible baseline during two foot steps which facilitates computing the planar homography.

To detect the corresponding phases, we compute the lateral inclination angle of the camera relative to gravity using an integrated inertial measurement unit. We use a delayed running average filter with a window length of 0.1 s on the angle measurements to smooth the data. Then, we identify the steady points as the extrema of the smoothed function. As a result, we are able to identify camera images with little motion blur and a relatively large baseline that we use for geometric floor estimation as described next.

B. Feature Extraction and Association for Floor Estimation

1) General Approach: To identify the floor plane we use an iterative approach. We consider image pairs from which we detect and match features that are compatible with the homography H induced by the ground plane. Thus, we search for correspondences $\mathbf{x}_i, \leftrightarrow \mathbf{x}_i'$ in the images I_j, I_{j+1} with

$$\|\mathbf{x}_i' - H\,\mathbf{x}_i\| < \xi, \tag{3}$$

for a threshold ξ. The homography H can in turn be re-estimated from the feature pairs that correspond to the floor by optimization. In more detail, we consider successive steady images I_j and I_{j+1} as identified in Sec. V-A in which we extract SURF features [15]. Let the detected features in I_j be f_j and the features from I_{j+1} be f_{j+1}. We seek the set $S = \{(\mathbf{x}_i, \mathbf{x}_i')\}$ of correspondences with $\mathbf{x}_i \in f_j$ and $\mathbf{x}_i' \in f_{j+1}$, representing the floor in I_j and I_{j+1}, respectively. At each iteration k of our algorithm, we update the estimate for the floor homography $H^{(k-1)}$ from the current correspondences $S^{(k-1)}$ to yield $H^{(k)}$ and then refine the correspondences $S^{(k-1)}$ from $H^{(k)}$ to yield $S^{(k)}$.

2) Correspondence Search: To find the set of correspondences $S^{(k)}$, we consider the homography $H^{(k)}$ and the 64-dimensional SURF descriptor. We define the set of candidates C_i for a feature $\mathbf{x}_i \in f_j$ as

$$C_i = \{\mathbf{y} \in f_{j+1} \mid \left\|\mathbf{y} - H^{(k)}\,\mathbf{x}_i\right\| < \xi\} \tag{4}$$

according the homography $H^{(k)}$, where ξ is the search radius. As correspondence $\mathbf{x}_i' \in C_i$ to $\mathbf{x}_i$, we select the visually most similar candidate feature according to the SURF descriptor $d(\cdot)$, i.e.,

$$\mathbf{x}_i' = \operatorname*{arg\,min}_{\mathbf{y} \in C_i} \|d(\mathbf{y}) - d(\mathbf{x}_i)\|. \tag{5}$$

We also apply the common constraint that the ratio between the descriptor distance of the best match $\mathbf{x}_i'$ to the descriptor distance of the next best match is above a threshold ρ. Further, we perform the mutual consistency check, i.e., we demand that $\mathbf{x}_i$ is also within the l most similar correspondences for $\mathbf{x}_i'$ in terms of the descriptor distance. $S^{(k)}$ is then the set of all pairs $(\mathbf{x}_i, \mathbf{x}_i')$ fulfilling these constraints. In each iteration, we reduce the search radius ξ. In this way, we reduce the ambiguities in the feature matching process because the number of candidates $|C_i|$ shrinks with ξ.

3) Homography Estimation: To estimate $H^{(k+1)}$, we apply a standard non-linear optimization to $S^{(k)}$. The non-linear optimization finds a least-squares solution for H to the problem $\mathbf{x}'_i = H\,\mathbf{x}_i$, i.e., it minimizes

$$\sum_{i=1}^{n} \mathbf{x}'_i - H\,\mathbf{x}_i, \tag{6}$$

where $n = \left|S^{(k)}\right|$. For the minimization we use the numeric Levenberg-Marquardt algorithm.

4) Initialization: To initialize the algorithm, we consider the homography $H^{(0)}$ computed from the odometry as

$$H^{(0)} = M\,[\,e_1^{j+1}\;e_2^{j+1}\;e_4^{j+1}\,]\,[\,e_1^{j}\quad e_2^{j}\quad e_4^{j}\,]\,M^{-1} \tag{7}$$

where M is the camera's intrinsic matrix and e_i^j are the i-th columns of the camera's extrinsic matrix at time j. The e_3 have been dropped due to the projective nature of homographies [16, Ch. 11]. We then initialize $S^{(0)}$ from $H^{(0)}$ as in *2)*. To include prior knowledge into the estimation of $H^{(1)}$, we further demand that the features $\mathbf{x}_i$ in $S^{(0)}$ are only sampled from parts of the image that are labeled as floor by our appearance-based classifiers. This constraint is waived in all subsequent iterations to be able to adapt to changing floor appearances but substantially improves the initialization. We then robustly estimate $H^{(1)}$ from the remaining correspondences using RANSAC. Afterwards, we proceed with the iterative refinement.

5) Iterative Refinement: We iterate steps *2)* and *3)* until the number of correspondences n have converged or a maximum number of iterations is reached. Typically, five to ten iterations are sufficient. In this way, we obtain a set of corresponding features on the floor plane $\mathbf{x}_{\text{floor}} \leftrightarrow \mathbf{x}'_{\text{floor}}$ in the two subsequent images I_j and I_{j+1}. These features are then used to train visual classifiers as described in the following section. Fig. 1 illustrates the result of our algorithm. The left image shows the first image I_j of the pair (I_j, I_{j+1}) along with the detected floor points $\mathbf{x}_{\text{floor}}$ and the corresponding flow vectors, i.e., the displacement $\mathbf{x}'_{\text{floor}} - \mathbf{x}_{\text{floor}}$. As can be seen, our algorithm reliably extracts features corresponding to the floor.

VI. Appearance-based Traversability Estimation

The set of floor points $\mathbf{x}_{\text{floor}}$ is typically sparse in the image as illustrated in Fig. 1. However, for collision-free navigation, dense information is required. We therefore developed an appearance-based approach to estimate traversability from images. We model it as an one-class classification problem since we have accurate information about features on the floor but no reliable information about obstacles.

We first train a texture-based and a color-based classifier using the floor point correspondences $\mathbf{x}_{\text{floor}}$ from the image I_j, as explained in the next paragraph. These classifiers are then used to label all successive images until new training data is available. Note that all images are labeled, not only the steady images used for ground plane estimation (see Sec. V-A). This is necessary to be able to react to changes in the environment and update the map in a timely manner as well as to label images while standing or turning.

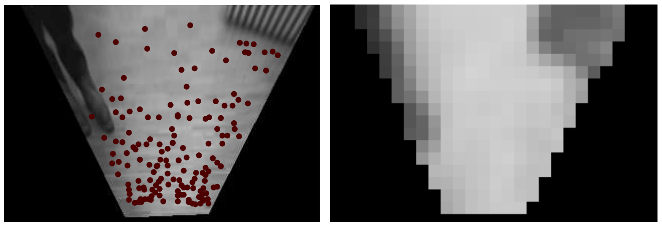
Fig. 3. Virtual downwards-looking camera as described in Sec. VI-A. The images correspond to the situation depicted in Fig. 1. The dots in the left image indicate the detected floor points $\mathbf{x}_{\text{floor}}$. The right image shows the traversability estimation based on texture (the brighter the color, the higher the probability of being traversable).

A. Texture-Based Classification

The texture classifier labels image patches according to their representation in the frequency-domain. In particular, we employ the discrete cosine transformation (DCT) to extract texture information [17]. 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 corresponding to different frequencies. Hence, the coefficients represent the amount of presence of certain frequencies in the image. Since the DCT is not invariant to perspective, we do not operate on the camera image directly but project it to a virtual downwards-looking camera. This projection is computed by applying a homography induced by the floor plane between the camera and a virtual camera looking perpendicular at the floor at a fixed height of 2 m and 1 m in front of the real camera. The homography is computed from odometry, similar to Eq. 7. The left image in Fig. 3 shows an example of such a top-down view. Note that this step works also with inaccurate odometry information: Translational errors have no impact on the classification accuracy, and we observed the classifier to be robust to rotational errors up to 30°. From the projected image, we extract rectangular patches of size 16×16. To these patches, we apply the DCT and keep only the 13 coefficients corresponding to the lowest frequencies in the image patch for dimensionality reduction and generalization. The extraction of these features is described in more detail in our previous publication [2]. For training, we extract the patches from the projected camera image at the location of the features $\mathbf{x}_{\text{floor}}$, compute the DCT-based descriptor and store it in a KD-tree. For classification, we project an image to the top-down view and also align the orientation of the top-down view with the orientation of the projected training image. In this way, we compensate for the rotational invariance of the DCT-based descriptor. Subsequently, we extract 16×16 patches at a fixed distance of 8 pixels, compute the DCT-based descriptor and determine a traversability probability. Here, we use the distance of the patch's DCT-based feature vector y to its nearest neighbor y' in the KD-tree and define the probability of y being traversable as $\exp\left(-\frac{\|y-y'\|}{\sigma}\right)$.

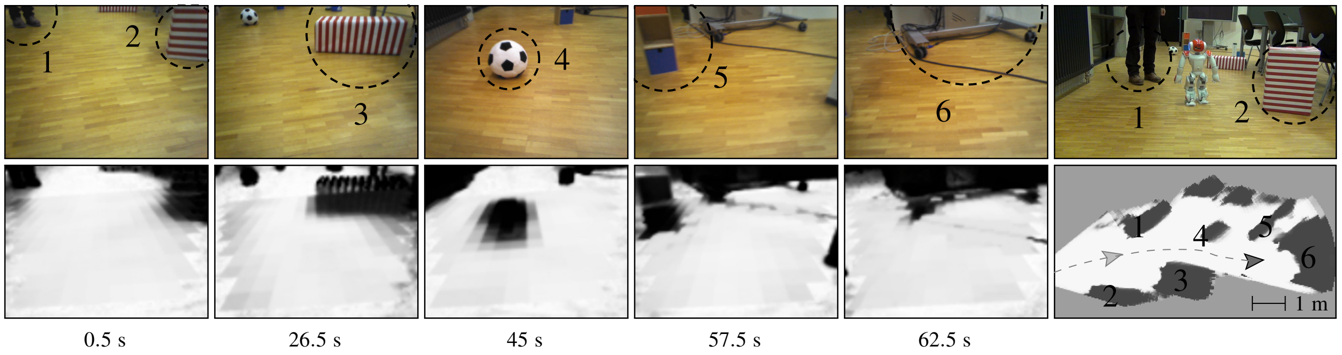

Fig. 4. Qualitative evaluation of traversability estimates. The columns show the robot's onboard camera images and the corresponding traversability estimates from our approach along with the corresponding timestamps. The last column depicts an overview of the scene and the learned occupancy grid (where obstacles appear dark, traversable area white, and unknown gray). Note that the varying coarseness within the labeled images stems from combining two different classifiers with different resolutions. The numbers in the map correspond to the obstacles marked in the camera images. The light arrow indicates the robot's pose as shown in the overview picture. The dashed line illustrates the robot's trajectory. The map is shown at the time the robot reached its goal location (dark arrow).

Hence, we apply the exponential probability density function with scale σ, which controls the steepness of the function. We average the probabilities for the pixels according to the overlapping patches. The right image in Fig. 3 shows an example classification result in terms of traversability probabilities.

B. Color-Based Classification

Due to the projection to the virtual downwards-looking camera, some parts of the original image cannot be labeled. This is demonstrated in Fig. 3 where a part of the original camera image shown in Fig. 1 is missing due to the projection. Also, the regions close to the margins of the visible area are not suitable for texture-based classification. Therefore, we train a second classifier based on color to achieve a classification of the whole image.

We compute the average HSV color in a 4×4 neighborhood around the floor points $\mathbf{x}_{\text{floor}}$ and store them in a KD-tree. For classification, we again divide the image into 4×4 patches and assign traversability probabilities according to the distance to the nearest neighbor as described above.

For the final classification, we combine the prediction from both classifiers by a weighted average. For considering dependencies between nearby areas and smoothing, we apply probabilistic relaxation labeling [2, 18]. Fig. 1 shows an example for a final classification result.

C. Using Traversability Information for Navigation

For planning the motion of the robot, we maintain a 2D occupancy grid map [19] to integrate the traversability information from the classified camera data over time. Each cell of the map represents the probability of being traversable. To integrate the traversability information from the camera, we project the labeled camera image to a virtual top-down looking camera as described in Sec. VI-A. By using bilinear interpolation, the traversability information from the camera image is mapped to the coordinate frame of an occupancy map. We update each cell using standard occupancy grid map updating. An example of such a map is shown in Fig 4 (right image). Afterwards, the map is used for planning collision-free motions by applying A*.

VII. Experiments

We conducted a series of experiments on a Nao (V4) humanoid robot to demonstrate that our approach is able to reliably learn about traversability using only odometry information and monocular images. In the experiments, we used the top camera in the robot's head. The camera's diagonal field of view is 72.6° and provides images with a resolution of 320×240. For all experiments, we weighted the prediction of the texture-based classifier with 0.9 and the color-based classifier with 0.1. The values were experimentally determined. We further initialized the radius ξ, in which candidate corresponding features are searched, to the length of the expected flow according the homography. This way we allow smaller deviations for features in the background which move only little between two images.

A. Qualitative Results on Traversability Estimation

In the first experiment, we placed various obstacles on the floor in our lab. The robot walked through the scene to a designated goal location. Fig. 4 shows classification results obtained during this experiment. The columns show the camera image and the traversability estimates, ordered chronologically from left to right. The last column shows an overview picture of the scene as well as the state of the learned occupancy grid map at the time when the robot reached the goal location. As one can see, our approach results in correctly labeled images and the constructed grid map can be used for collision-free navigation.

B. Adapting to Changing Ground Appearance

The following experiment is designed to illustrate that our approach can automatically adopt to new environments, specifically to changing ground appearance. To illustrate this, the robot started to navigate on a parquet floor. Distant, we placed a PVC floor coating that was visually dissimilar from the parquet. Fig. 5 shows resulting classification results as well as the learned grid map in this experiment. The

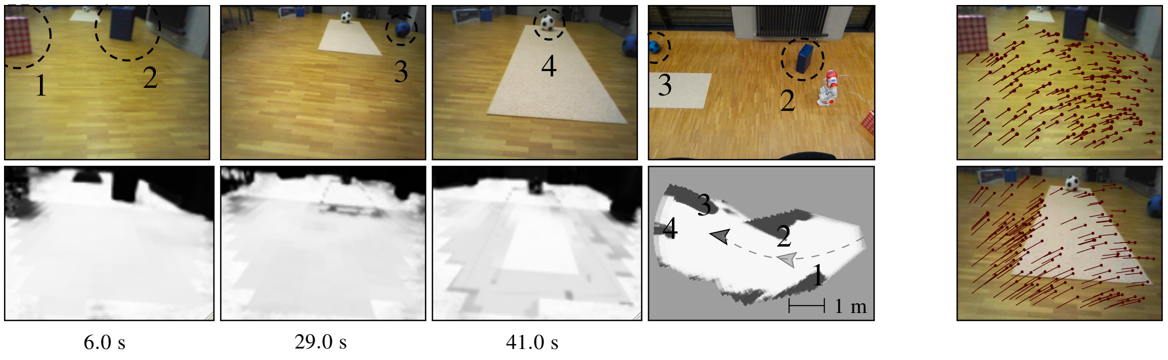

Fig. 5. The images illustrate that our approach automatically adapts to changing ground appearance. The columns show the robot's onboard camera images at different timestamps along with the traversability estimates, while the robot approaches a floor with different appearance. Both floors are correctly identified as traversable. The fourth column depicts an overview and the constructed grid map. The numbers in the map correspond to the obstacles marked in the camera images. The bright arrow indicates the robot's pose in the overview picture. The map is shown at the time the robot reached its goal location (dark arrow). The last column shows the detected feature points and flow vectors for the camera images in column 1 and 3, respectively.

robot immediately labeled the parquet as traversable (1st column) but considered the PVC floor as obstacle due to its dissimilarity from the parquet and only few feature matches for the PVC floor in the background. As soon as the robot got closer (2nd column) and hence, could detect features on the PVC floor for training the classifiers, the PVC as well as the parquet parquet were labeled as traversable (3rd column). As can be seen, the resulting grid map (4th column) correctly represents the area corresponding to the PVC floor as traversable. The last column shows the detected feature points and corresponding flow vectors for the camera images in the 1st and 3rd column, respectively.

C. Classification Accuracy

To quantitatively evaluate our approach, we compared the traversability estimates for the camera images assigned by our algorithm to manually assigned ground truth labels for the experiments described in Sec. VII-A (experiment 1) and Sec. VII-B (experiment 2), respectively. To compare the binary manual labels of the images with the traversability probabilities provided by our approach, we considered each pixel in the latter as representing floor area, if its traversability probability was above 0.5 and as obstacle otherwise. Then, we counted the number of pixels coinciding with the manual labels. Table I presents the classification accuracy. As can be seen, the classification rates are highly accurate. The probability that a pixel corresponding to an obstacle was classified as traversable space lies between 6% and 12%. Note that in practice, the number of potentially dangerous false classification is even lower because most false classifications occurred in the background of the scene. In these regions, the texture-based classifier has no information due to the projection of the image to the virtual camera (see Sec. VI-A) and our approach relies only on color information. As soon as the robot gets closer to these obstacles, they appear in the projected image and the texture classifier can be used as well to identify the traversable floor.

D. Turning on the Spot

In the next experiment, we demonstrate that our technique is able to classify camera images even if the robot is turning on the spot. Many structure-from-motion based approaches (e.g., [7]) fail in these situations since it is not possible to compute structural information for rotations around the optical center of the camera. In contrast to that, our approach yields correct classification results also for rotations by using the previously learned visual classifiers. After turning, the robot then automatically re-trains the classifiers when it starts to move. Fig. 6 shows results from the traversability estimation while the robot turned counter-clockwise by circa 100°. As can be seen, the classification results allow the robot to detect obstacles in the scene and will improve even further once the robot starts navigating and updating the classifiers.

TABLE I
TRAVERSABILITY CLASSIFICATION ACCURACY.

	experiment 1		experiment 2	
	estimated as		estimated as	
true class	obstacle	floor	obstacle	floor
obstacle	0.88	0.12	0.94	0.06
floor	0.03	0.97	0.06	0.94

E. Dealing with Moving Obstacles

In a final experiment, we illustrate the ability of our approach to deal with dynamics in the environment. While the robot was walking, we rolled a soccer ball several times into the robot's field of view. Fig. 7 shows results for the traversability estimation as well as the corresponding camera images. As can be seen, the images are robustly labeled and the traversability estimation leads to collision-free navigation of the robot.

F. Remarks

If obstacles look (in terms of the texture features and color) very similar to the floor, appearance-based classification will fail. The same may hold for textureless floor, for example a homogeneously black floor, since it will not generate any interest points. Furthermore, we assume that the floor plane

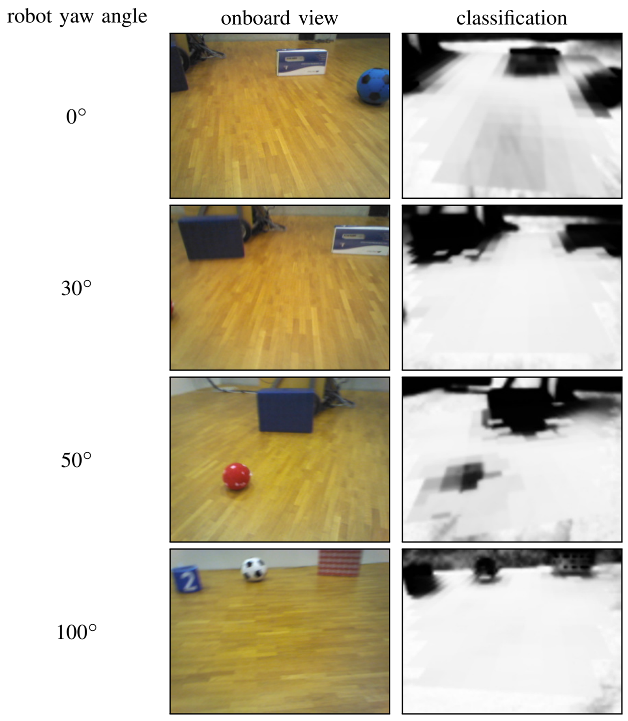

Fig. 6. Obtained classification results while the robot turns on the spot. The rows show the robot's onboard camera images and the traversability estimates for different rotation angles.

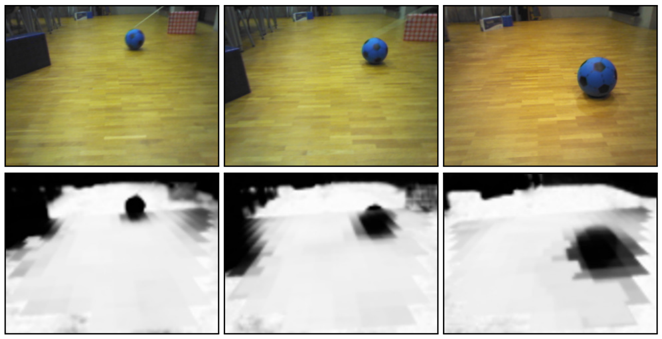

Fig. 7. We achieve reliable traversability estimates also in presence of moving obstacles. We rolled a soccer ball multiple times in front of the robot's field of view while navigating. The top row shows the camera images and the bottom row the corresponding traversability estimates.

is dominant (with respect to the number of features) in the first image pair. Afterwards, the floor can be identified by considering the appearance model in the homography estimation, even in the presence of larger obstacles. Concerning the computation time, we found that labeling images takes 0.1 s on a single core of a standard i7 PC. Training the classifiers takes 2.0 s, of which the majority is required for the iterative feature matching. For real-time navigation, we thus cannot re-train from every image pair, but are still able to label all steady camera images, that we obtained at a rate of 2 Hz on a Nao humanoid.

VIII. Conclusions

In this paper, we presented an approach to estimate traversable areas in the surroundings of a humanoid robot solely based on monocular images and odometry information. We apply an iterative approach to estimate the floor homography by detecting and matching compatible features. To get dense information about traversability given the sparse floor features, we developed a technique to train appearance-based classifiers based on color and texture. Using the classifiers, which are automatically learned online, the robot labels its camera images and updates a map of the environment while walking, standing, and turning. As we showed in practical experiments with a Nao humanoid, the achieved classification rates are highly accurate. The labeled images lead to safe navigation and the robot constantly updates the classifiers so as to adapt to changing ground appearance. The proposed appearance-based classifiers can also be used in combination with RGBD-sensors that do not provide depth data at close ranges. In these cases, monocular image data can be analyzed using the learned classifiers.

References

[1] J. Biswas and M. Veloso. Depth camera based localization and navigation for indoor mobile robots. In *RSS 2011 Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2011.

[2] D. Maier, M. Bennewitz, and C. Stachniss. Self-supervised Obstacle Detection for Humanoid Navigation Using Monocular Vision and Sparse Laser Data. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

[3] E. Einhorn, C. Schröter, and H.-M. Gross. Monocular scene reconstruction for reliable obstacle detection and robot navigation. In *European Conf. on Mobile Robots (ECMR)*, 2009.

[4] C. Braillon, C. Pradalier, J. Crowley, and C. Laugier. Real-time moving obstacle detection using optical flow models. In *IEEE Intelligent Vehicles Symp.*, 2006.

[5] Y.-G. Kim and H. Kim. Layered ground floor detection for vision-based mobile robot navigation. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.

[6] T. Low and G. Wyeth. Obstacle detection using optical flow. In *Aust. Conf. on Robotics and Automation (ACRA)*, 2005.

[7] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE Int. Symp. on Mixed and Augmented Reality (ISMAR'07)*, 2007.

[8] H. Wang, K. Yuan, W. Zou, and Y. Peng. Real-time obstacle detection with a single camera. In *IEEE Int. Conf. on Industrial Technology (ICIT)*, 2005.

[9] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by GPU-accelerated large displacement optical flow. In *European Conference on Computer Vision (ECCV)*, Crete, Greece, 2010.

[10] A. Pretto, E. Menegatti, M. Bennewitz, W. Burgard, and E. Pagello. A visual odometry framework robust to motion blur. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009.

[11] I. Ulrich and I. Nourbakhsh. Appearance-based obstacle detection with monocular color vision. In *National Conf. on Artificial Intelligence (AAAI)*, 2000.

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

[13] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *IEEE Int. Conf. on Computer Vision (ICCV)*, 2011.

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

[15] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *European Conf. on Computer Vision (ECCV)*, 2006.

[16] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, Cambridge, MA, 2008.

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

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

[19] H. P. Moravec and A. E. Elfes. High resolution maps from wide angle sonar. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1985.