Journal of Zhejiang University SCIENCE A
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Robust water hazard detection for autonomous off-road navigation*

Tuo-zhong YAO†, Zhi-yu XIANG†‡, Ji-lin LIU

(*Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China*)

†E-mail: {thomasyao, xiangzy}@zju.edu.cn

Received Mar. 18, 2008; Revision accepted July 11, 2008; Crosschecked Dec. 26, 2008

Abstract: Existing water hazard detection methods usually fail when the features of water surfaces are greatly changed by the surroundings, e.g., by a change in illumination. This paper proposes a novel algorithm to robustly detect different kinds of water hazards for autonomous navigation. Our algorithm combines traditional machine learning and image segmentation and uses only digital cameras, which are usually affordable, as the visual sensors. Active learning is used for automatically dealing with problems caused by the selection, labeling and classification of large numbers of training sets. Mean-shift based image segmentation is used to refine the final classification. Our experimental results show that our new algorithm can accurately detect not only 'common' water hazards, which usually have the features of both high brightness and low texture, but also 'special' water hazards that may have lots of ripples or low brightness.

Key words: Water hazard detection, Active learning, Adaboost, Mean-shift
doi:10.1631/jzus.A0820198 **Document code:** A **CLC number:** TP317.4

INTRODUCTION

Robust water hazard detection is very important for autonomous off-road navigation. Traversing through deep water bodies is likely to damage non-watertight unmanned ground vehicles (UGVs). Different kinds of water hazards in complex outdoor environments make water detection particularly challenging.

Few studies have been carried out in this new domain. The National Institute of Standards and Technology, USA (NIST) proposed combining information from a laser range finder and a color camera to detect water (Hong *et al.*, 2002). The Jet Propulsion Laboratory, USA (JPL) compared the performance of a short-wave infrared camera (SWIR), a mid-wave infrared camera (MWIR), and radar in finding water hazards (Matthies *et al.*, 2003). Recently, JPL tried to detect water hazards by extracting and combining several features of water such as color and texture (Rankin *et al.*, 2004), while PercepTek Robotics, USA (PTR) developed a method based on adjusting the polarization angles of the camera lens (Sarwal *et al.*, 2004). In (Yao *et al.*, 2007), we presented a new method to extract reflection regions of water using stereo vision systems and to detect integrated water regions by combining multiple features of brightness, texture, and reflection. Generally, these water detection methods can be classified into two types: those using normal digital cameras and those using special and expensive sensors, such as infrared cameras or radar. The first type usually makes use of traditional image processing using simple feature extractions of color or texture to detect water hazards. The second type can offer some additional useful information such as the measurement of temperature because of the special capabilities of the sensors.

Water hazards usually have the features of both high brightness and low texture (Fig.1a) and we define these as 'common' water hazards. However, sometimes the brightness of water changes because the illumination varies (Fig.1b) or because of the occurrence of ripples on the surface (Fig.1c) caused

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 60505017 and 60534070) and the Natural Science Foundation of Zhejiang Province, China (No. 2005C14008)

by wind. These are defined as 'special' water hazards. The first type of method usually performs well in detecting 'common' water hazards, but it is quite susceptible to variation in the surroundings. Thus, it will probably suffer instability or even invalidation in detecting 'special' water hazards. The second type of method may provide some additional information for detecting the 'special' water hazards, but it is also not robust and is usually too expensive and thus not widely used.

(a) (b) (c)

Fig.1 (a) 'Common' water hazards; (b) and (c) 'Special' water hazards

To improve the performance of our previous detection method in (Yao *et al.*, 2007), we present a new water hazard detection algorithm that integrates traditional machine learning and image segmentation with inexpensive color cameras as the only sensors. This paper is organized as follows. Section 2 introduces the scheme of the novel algorithm and Section 3 describes the algorithm in detail. In Section 4, qualitative and quantitative experimental results and analysis are given. Conclusions and suggestions for future work are summarized in Section 5.

SCHEME OF OUR ALGORITHM

A flowchart of the water hazard detection algorithm is shown in Fig.2. Active learning is very useful in solving the problem of labeling large datasets in outdoor robotics applications (Dima, 2006). The steps shown in boxes constitute the active learning algorithm. Here, active learning is used for automatic sample selection, labeling, and classification. Only a small number of unlabeled samples, those that can improve the performance of classification, will be automatically selected and labeled. In this way, the impractical and laborious manual labeling of all the unlabeled samples in the training library can be avoided. The number of labeled samples selected for training is only a small proportion of the training library, thereby reducing the training time. Then, by using a conventional Adaboost algorithm, a high performance classifier can be obtained. Finally, we use a mean-shift based image segmentation algorithm for classification refinements. Each collected test image is not only classified but also processed by automatic sample selection and labeling. Thus, the boundary decision of the classifier can be modified and the classification performance gradually improved, giving our classifier the ability of self-improvement.

DETAILS OF THE ALGORITHM

Data pre-processing

Before starting active learning, several steps should be carried out for data initialization: manual sample labeling, feature extraction, and feature compression.

1. Manual sample labeling

Some representative images in the training library are manually selected and labeled. These include not only images of different kinds of water

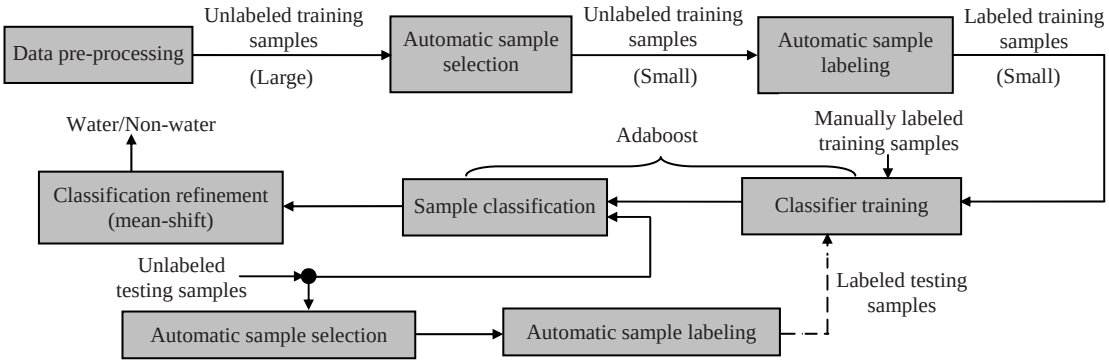


Fig.2 Flowchart of the water hazard detection algorithm

hazards for sample diversity but also images with no water hazards. In the manual labeling process, regions of sky in the images should be labeled as 'water regions' at first because their features are very similar to those of common water hazards. Otherwise, incorrect boundary decisions would be made during classifier training leading to poor performance of the final classification. Representative images from the training library and the manually labeled results are shown in Fig.3 (see p.791).

2. Feature extraction

Each image is divided into 8×8 blocks and transformed into YUV color space for feature extraction. In order to extract color features, means and standard deviations of Y, U and V channels of each 8×8 block are calculated, and then six color features are acquired, which represent the entire color information in each block. To extract texture features from each block, the Gabor filter (Forsyth and Ponce, 2002) is applied in orientations of 0°, 30°, 60°, 90°, 120°, 150°, and 180°. The Gabor filter formula is

$$G(x, y, f, \theta, \sigma) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(x')^2+(y')^2}{2\sigma^2}\right) \cdot \sin[2\pi f(x'\cos\theta + y'\sin\theta)], \quad (1)$$

$$x' = x\sin\theta + y\cos\theta,\ y' = x\cos\theta - y\sin\theta,$$

where f is the central frequency of the Gabor filter, θ is the filtering orientation, σ is the bandwidth of the Gaussian function (here, σ is empirically set as 0.01), x and y are 2D coordinates of the image, and $G(x, y, f, \theta, \sigma)$ represents the filtering coefficients.

$G(x, y, f, \theta, \sigma)$ independently convolves with Y, U and V matrices of each block. Then means and standard deviations of the convolving results F_1, F_2 and F_3 are acquired, as shown in Eq.(2):

$$\begin{cases} F_1(x,y) = \| Y(x,y) * G(x,y,f,\theta,\sigma) \|, \\ F_2(x,y) = \| U(x,y) * G(x,y,f,\theta,\sigma) \|, \\ F_3(x,y) = \| V(x,y) * G(x,y,f,\theta,\sigma) \|. \end{cases} \quad (2)$$

Finally, we can acquire 42 texture features which represent the entire text information of each block.

The color and texture features are combined and a 1×48 feature vector is built up (referred to as 'sample' in the following text) that represents the entire feature information of each 8×8 block.

3. Feature compression

The dimension of the feature vector extracted from each 8×8 block is obviously too high and dimension compression is needed to avoid the 'curse of dimensionality'. Compared with traditional principal component analysis (PCA), nonlinear principal component analysis (NLPCA) (Scholz and Vigario, 2002) makes use of a self-organizing neural network to construct an 'auto-encoder' whose inputs to and outputs from the network are the same. Through network training, the linear components can be sufficiently acquired from the hidden layers of the network. Through subsequent experiments we found that quite similar classification results were acquired when compressing the feature vector to 2, 3, 4, or 5 dimensions. Thus, we compress each feature vector to two dimensions by NLPCA to accelerate the operation.

Automatic sample selection

The function of unlabeled data filtering (UDF) (Dima *et al.*, 2004) here is automatic sample selection referred to as 'filtering'. The detailed process is as follows:

Step 1: Project the manually labeled samples into a 2D feature library after NLPCA.

Step 2: Add each unlabeled sample to the feature library, which is empty at first. Then calculate the likelihood of each sample in the feature library and reorder these likelihoods by ranking the values from small to large. If the likelihood of the newly added sample is included in the first 5% of ranked values, it means that it is probably projected into the 'sparse' space (Fig.4) and it may provide useful information for future classifier construction. So this new sample will be kept in the feature library. Otherwise, the new sample is probably projected into the 'dense' region (Fig.4). It contains redundant information and will be discarded.

Step 3: The UDF algorithm is not completed until all unlabeled samples have completed Step 2. Finally, some useful unlabeled samples will be added to the feature library.

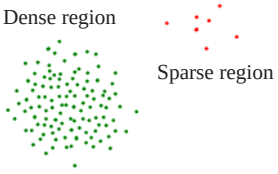


Fig.4 Unlabeled samples projected into a 2D feature library

In Step 2 of the UDF algorithm, kernel density estimation (KDE) (Gray and Moore, 2003) is used for calculating the likelihood $p(\boldsymbol{x}_i)$ of each feature vector $\boldsymbol{x}_i$. The density estimation at the point $\boldsymbol{x}_i$ is

$$p(\boldsymbol{x}_i)=\frac{1}{N-1}\sum_{j=1,i\neq j}^{N}K\left(\frac{|\boldsymbol{x}_i-\boldsymbol{x}_j|}{h_i}\right),\ i=1,2,...,N,\quad(3)$$

where N is the whole number of feature vectors in the feature library, K represents the Gaussian kernel function, and h_i is the bandwidth of each corresponding feature vector.

The bandwidth h_i selection impacts $p(\boldsymbol{x}_i)$ seriously. Here likelihood cross validation (LCV) (Nguyen *et al.*, 2001) is used for bandwidth selection. Also, a *k*d-tree data structure (Wald and Havran, 2006) is constructed before computing $p(\boldsymbol{x}_i)$ to speed up the calculation of KDE.

Automatic sample labeling

The unlabeled samples added to the feature library by UDF should be labeled automatically and correctly; otherwise, it will seriously affect the performance of the classifier. Here, a semi-supervised expectation-maximization (EM) based algorithm (SSEM) (Nigam, 2001) is proposed for both sample labeling and classification, and is composed of the following steps:

Step 1: Build an initial naïve Bayes classifier from the labeled samples in the feature library and use maximum a posteriori (MAP) to do the parameter estimation.

Step 2: Loop while classifier parameters improve:

(i) E-step: Use the current classifier to estimate the probability of each unlabeled sample.

(ii) M-step: Re-estimate the classifier given the estimated likelihood of each sample and do parameter estimation by MAP.

Fig.5 (see p.791) shows some unlabeled images from the training library. The color blocks in each image correspond to the feature vectors that are selected for the feature library after UDF. Among these samples blue and red blocks are labeled as water and non-water samples, respectively.

Classifier training and sample classification

Adaboost (Dima, 2004) is a well known data classification algorithm that aims to maximize the separation margin between two classes with the only requirement being that a so-called 'weak classifier' (a learning algorithm that can perform better than a random one) is available. The standard Adaboost is shown in Fig.6.

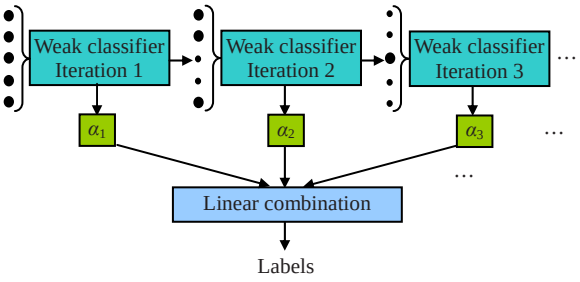


Fig.6 Standard form of Adaboost
Larger black points represent the larger weights

First, some of the samples are chosen for training a weak classifier. Each training sample is initially assigned the same weight that represents the likelihood of being selected into the training sets. In each iteration, if a sample is classified incorrectly, its weight will become bigger (larger black points represent the larger weights in Fig.6). Thus, the Adaboost algorithm can concentrate on those samples that are difficult to classify. The final classification results are the weighted average of outputs from all classifiers. In this way, we need use only a weak classifier to finally acquire a high performance classification.

The classification results obtained using the Adaboost algorithm are shown in Fig.7 (see p.791). The blue regions represent the detected water hazards.

Self-improvement mechanism

In the practical implementation, each testing image is not classified for water hazard detection but processed by active learning for classification improvement. The two processes are independent. Sample classification can be done independently by the previously trained classifier. Simultaneously, newly added testing samples provide useful information by active learning. However, active learning is time consuming, preventing real-time detection. In our experiments, we assume that only if the percentage of newly added testing samples in the feature library is higher than 5% (and therefore will noticeably improve the classification results) and the robot is in an idle state (e.g., the robot is in maintenance after a long period of activity), will active learning be carried out for the feature library and the classifier be updated in the next operating period.

Classification refinements

The water regions acquired after Adaboost are usually not integrated, so we use mean-shift (Comaniciu and Meer, 2002) based image segmentation to try to acquire more complete water regions and to discard the inaccurate regions. We define the mean-shift vector $\boldsymbol{M}(\boldsymbol{x})$ at point $\boldsymbol{x}$ as

$$\boldsymbol{M}(\boldsymbol{x}) = \frac{\sum_{i=1}^{N} \boldsymbol{x}_i K(\boldsymbol{x} - \boldsymbol{x}_i)}{\sum_{i=1}^{N} K(\boldsymbol{x} - \boldsymbol{x}_i)} - \boldsymbol{x}, \tag{4}$$

where $\boldsymbol{x}_i$ is the pixel of the image and K is the kernel function. Then, in each iteration move $\boldsymbol{x}$ in the direction of $\boldsymbol{M}(\boldsymbol{x})$ until the optimal matching position in the image is found. Here we use the Epanechnikov function for the kernel density estimation of $K(\boldsymbol{x})$.

Before image segmentation, some pre-processing should be done. The original testing image is transformed according to

$$\begin{cases} I = R\cos\theta + B\sin\theta, \\ \theta = \dfrac{1}{2}\arctan\left(\dfrac{2b}{a-c}\right), \end{cases} \tag{5}$$

where I is the transformed image, a and c are deviations of color channels R and B respectively, and b is the covariance of R and B. This color transformation uses only two color channels R and B, and filters unnecessary color information from the original testing images using the strong relationship between the color channels in RGB space. Thus, it can better support the subsequent image segmentation.

After appropriate parameter configuration [the primary parameter—bandwidth in the Epanechnikov kernel function—can also be selected by likelihood cross validation (LCV) (Nguyen *et al.*, 2001)], the image is segmented into hundreds of regions of different sizes (Fig.8). Then we calculate the percentage of pixels that have been classified as water in each region and define a region as a water hazard if the percentage is higher than 60%. Simultaneously, sky regions should be discarded. This can be carried out using the height constraint in the post-processing step (e.g., by defining the top 50 rows of images as possible sky regions). Another more robust way is by using a seed-expansion algorithm. First, we search through several lines on top of the image and find the pixels that have the feature of high brightness as the seeds for subsequent four-neighbor direction expansion. If there is a high brightness pixel in the neighborhood of the seed, this pixel will be defined as a new seed for the next expansion. If not, the expansion process stops. After all the expansions have stopped, the expanded regions of high brightness at the top of the image will be considered as possible sky regions. Thus, we can acquire more accurate water regions by discarding these sky regions (Fig.9, see p.791).

(a) (b) (c)

Fig.8 (a)~(c) showing mean-shift based image segmentation results

EXPERIMENTS

Devices

The experiment was carried out on a Pioneer platform from ActiveMedia^{TM}, as shown in Fig.10. A Canon^{TM} VC-C4 pan-tilt-zoom CCD camera was used with an image resolution of 320×240 pixels.

Fig.10 Mobile robot—Pioneer 4

Training step

In our training library there were 500 different kinds of water hazard images. Each image was divided into 8×8 blocks giving a total of 1008 samples per image. Firstly, 30 representative images were selected for manual labeling. Then in the training process,

Fig.3 Representative images selected (top) and artificially labeled (bottom)

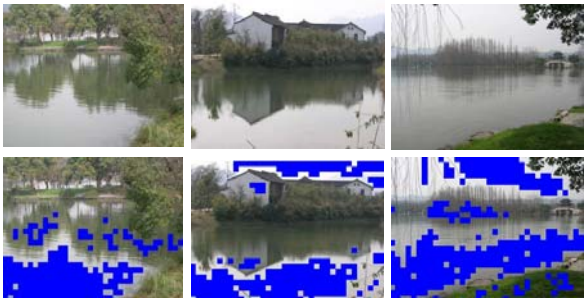
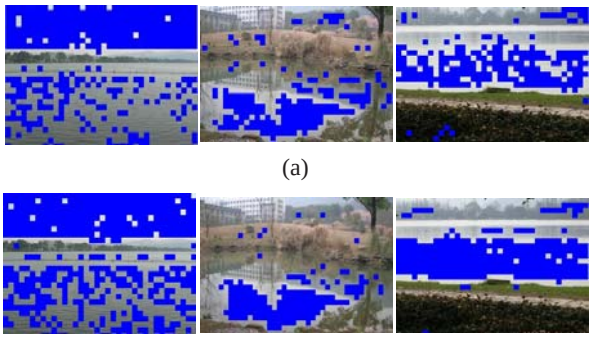


Fig.7 Classification results obtained by using standard Adaboost



(a)

(b)

Fig.11 Comparison between 'traditional learning' (a) and active learning (b)

Fig.14 Detection results from different types of water hazards

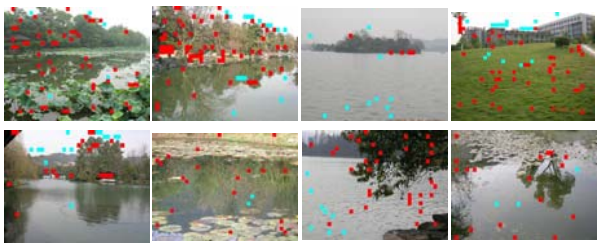
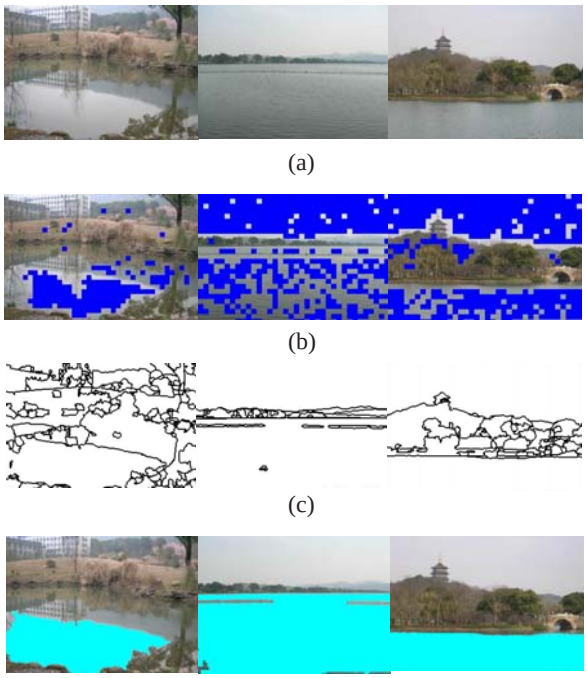


Fig.5 Automatic sample labeling by SSEM in unlabeled images

Blue and red blocks are labeled as water and non-water samples, respectively

Fig.9 Final refined classification results



(a)

(b)

(c)

(d)

Fig.13 The results in each step of the algorithm

(a) Raw testing images; (b) Classification results after using Adaboost; (c) Mean-shift based image segmentation results of (b); (d) Final detection results after classification refinements

Fig.15 Detection results from regions that contain no water

automatic sample selection and labeling was implemented for the remaining 470 images. Finally, from 473760 samples of 470 images a total of 18377 samples were added to the feature library for classifier training. In the automatic labeling process only 1599 samples were classified incorrectly and the rate of correct automatic sample labeling was 0.913, showing that the accuracy of automatic sample labeling was satisfactory. Fig.11a executes only 'data preprocessing' and 'classifier training and sample classification', and we call this method 'traditional learning'; Fig.11b uses active learning that combines automatic sample selection and labeling. The comparison shows that the classification results were improved by active learning, suggesting that the effect of the incorrect labeling is insensitive and that the new samples added in by UDF actually improve the classification. The receiver operating characteristic (ROC) curves proving the superiority of the active learning are shown in Fig.12.

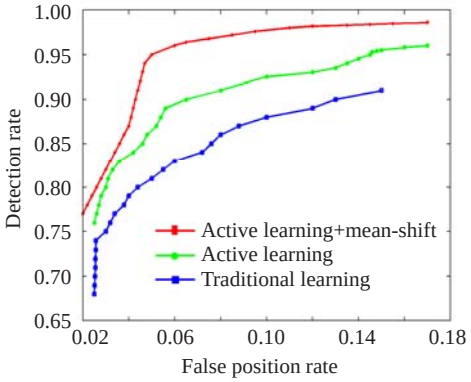


Fig.12 Receiver operating characteristic curves of the novel water detection algorithm

Testing step

In the testing process, testing images were collected at 5 Hz and processed in real time when the robot was moving. The algorithm was run under Linux on a computer with a 2.8 GHz Pentium IV CPU and 1 GB primary memory, and the average time cost of processing each image was 0.147 s. Fig.13 (see p.791) shows the results in each step of our detection algorithm.

Other results

In Fig.14 (see p.791), some other detection results are listed for different types of water hazards. Based on these results, we can conclude that our method outperforms previous methods in detecting different kinds of water hazards. Our algorithm is more robust in detecting 'special' water hazards. Fig.15 (see p.791) also shows that our method is still robust in detecting regions that contain no water. Also, ROC curves in Fig.12 show that image segmentation based refinement efficiently improves the performance of the detection.

Water regions that reflect land or trees rather than sky are common. To solve this problem, we proposed a stereo vision based detection algorithm (Yao *et al.*, 2007) that can successfully detect these types of water regions.

CONCLUSION AND FUTURE WORK

In this paper, a novel algorithm is proposed for use in autonomous off-road navigation, which integrates traditional machine learning and image segmentation to robustly detect different kinds of water hazards. By using active learning, labeling a large number of training samples was avoided and the final classification results were successfully refined by mean-shift based image segmentation. Moreover, the classifier was designed to be capable of self-improvement using a loop of active learning from testing samples. The experimental results showed that our proposed algorithm can detect both 'common' and 'special' water hazards more robustly than previous methods and without using expensive sensors.

The next step will be to combine this method with our previous method (Yao *et al.*, 2007) to achieve more accurate detection that can acquire more integrated water hazards.

References

Comaniciu, D., Meer, P., 2002. Mean-shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, **24**(5):603-619. [doi:10.1109/34.1000236]

Dima, C., 2004. Classifier Fusion for Outdoor Obstacle Detection. IEEE Int. Conf. on Robotics and Automation, **1**:665-671.

Dima, C., 2006. Active Learning for Outdoor Perception. PhD Thesis, the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA.

Dima, C., Hebert, M., Stentz, A., 2004. Enabling Learning from Large Datasets: Applying Active Learning to Mobile Robotics. Proc. Int. Conf. on Robotics and Automation, **1**:108-114.

Forsyth, D., Ponce, J., 2002. Computer Vision: A Modern Approach. Professional Technical Reference, Prentice Hall.

Gray, A.G., Moore, A.W., 2003. Rapid Evaluation of Multiple Density Models. Proc. 9th Int. Workshop on Artificial Intelligence and Statistics.

Hong, T., Chang, T., Rasmussen, C., Shneier, M., 2002. Feature Detection and Tracking for Mobile Robots Using a Combination of Radar and Color Images. IEEE Proc. on Robotics and Automation, p.4340-4345.

Matthies, L., Bellutta, P., McHenry, M., 2003. Detecting Water Hazards for Autonomous Off-road Navigation. SPIE, **5083**:231-242. [doi:10.1117/12.496942]

Nguyen, N., Milanfar, P., Golub, G., 2001. Efficient generalized cross-validation with application parametric image restoration and resolution enhancement. *IEEE Trans. Image Processing*, **10**(9):1299-1308. [doi:10.1109/83.941854]

Nigam, K., 2001. Using Unlabeled Data to Improve Text Classification. PhD Thesis, Carnegie Mellon University, Pittsburgh, PA, USA.

Rankin, A., Matthies, L., Huertas, A., 2004. Daytime Water Detection by Fusing Multiple Cues for Autonomous Off-road Navigation. Proc. 24th Army Science Conf., p.177-184.

Sarwal, A., Nett, J., Simon, D., 2004. Detection of Small Water-bodies. Technical Report, PercepTek Robotics, USA.

Scholz, M., Vigario, R., 2002. Nonlinear PCA: A New Hierarchical Approach. Proc. European Symp. on Artificial Neural Networks, p.439-444.

Wald, I., Havran, V., 2006. On Building Fast kd-trees for Ray Tracing, and on Doing That in $O(N\log N)$. IEEE Symp. on Interactive Ray Tracing, p.61-70. [doi:10.1109/RT.2006.280216]

Yao, T.Z., Xiang, Z.Y., Liu, J.L., Xu, D., 2007. Multi-feature Fusion Based Outdoor Water Hazards Detection. Proc. IEEE Conf. on Mechatronics and Automation, p.652-656. [doi:10.1109/ICMA.2007.4303620]