Chapter 2
Roadside Video Data Analysis Framework

This chapter introduces a general framework for roadside video data analysis. The main processing steps in the framework are described separately. It also reviews previous related work on vegetation and generic object segmentation, and lists several commonly used data processing algorithms.

2.1 Overview

Figure 2.1 depicts a general framework for roadside video data analysis, which is composed of five main steps. For a given roadside video, the data is firstly pre-processed to make it suitable for further processing in the framework. For instance, the video can be converted into a sequence of static frames and rescaled to the same resolution. An object segmentation step is then employed to segment each frame into regions of objects, from which a set of representative features is extracted for each object of interest and further used for the classification of the states of the object (e.g. low vs. high fuel load of grasses) using deep learning or other machine learning algorithms. Once all objects of interest are correctly classified, the applications can be implemented accordingly based on the goals of a specific application.

It is important to note that designing automatic systems for vegetation segmentation and classification in natural video data generally face many challenges, such as unstructured, dynamic or even unpredictable configuration of vegetation, significant changes in environmental conditions, and high dependence on data capturing settings such as camera configuration and resolution. The scene may be also overexposed, underexposed or blurred. Therefore, it is critical to take all or a part of these challenges into consideration when designing techniques for different processing steps of the whole system.

B. Verma et al., *Roadside Video Data Analysis: Deep Learning*,
Studies in Computational Intelligence 711, DOI 10.1007/978-981-10-4539-4_2

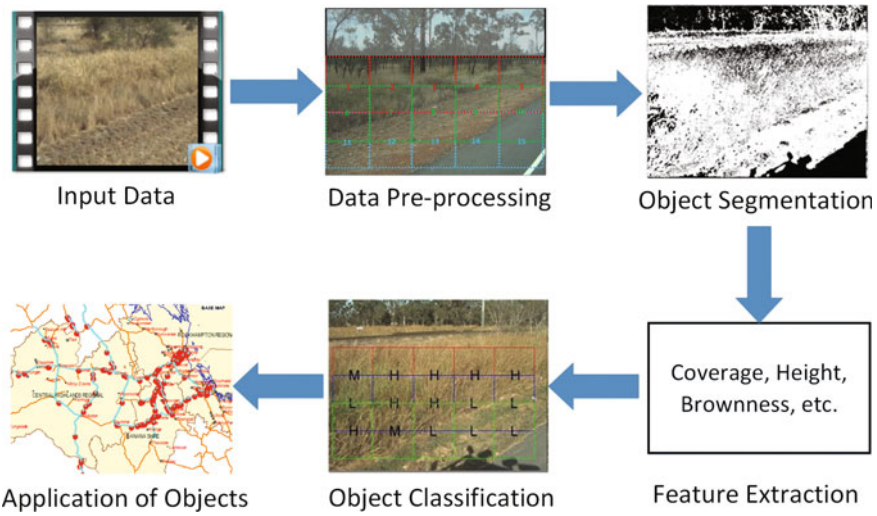

Fig. 2.1 A general framework for roadside video data analysis

2.2 Methodology

2.2.1 Pre-processing of Roadside Video Data

Data pre-processing is an important step in providing suitable types of data for the steps that follow and ensuring processing outcomes as expected. Depending on the purpose of each specific application and the nature of the video data, different types of pre-processing techniques can be employed in this step. Here, we introduce several commonly used pre-processing techniques.

(1) Video to frame conversion. Without neglecting the fact that there are many applications that demand only raw video data and directly perform processing on them to obtain the desired outcomes, it is often the case that a conversion step from video data to a sequence of frames is a pre-requisite for many applications. Extracting frames from the input video data allows performing further detailed analysis and processing on each individual frame separately, which is often vital in obtaining detailed information as required by specific applications. Figure 2.2 exemplifies the processing of extracting a series of frames from a roadside video.

(2) Color space conversion. There are many types of color spaces that have been proposed previously which have their own characteristics in the perception of the video frame content, including RGB, HSV, CIELab, $O_1O_2O_3$, etc. For robust feature extraction, it is a common step to convert a color space into another suitable space (e.g. [1]) that is more robust or invariant to effects from the environment such as shadow, illumination, and dynamic and uncontrollable

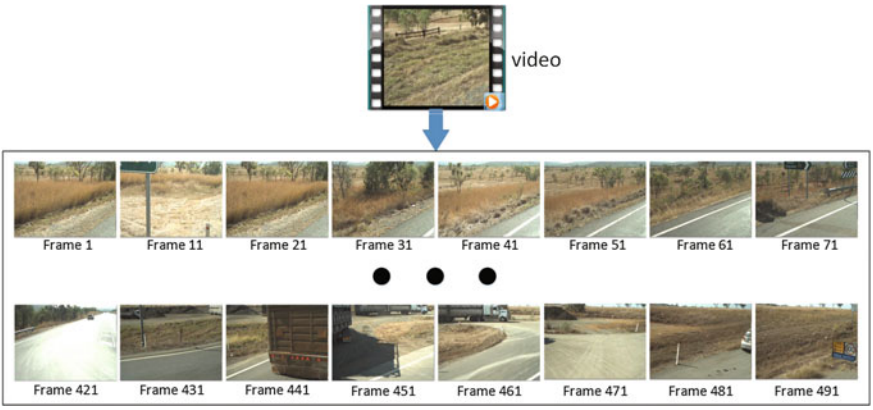

Fig. 2.2 An example of video to frame conversion. There are 500 frames in total in the example video, and every 10th frame is displayed

lighting conditions. For algorithms such as Gray-Level Co-occurrence Matrix (GLCM) and histogram equalization that require grayscale images, the original color images are needed to be converted to grayscale. RGB to gray conversion is one of the most frequently used conversion methods, as most existing color images can be represented by R, G, and B channels. Among many ways of converting RGB to grayscale, such as taking the mean value over R, G, and B, the most commonly used one is based on Eq. 2.1:

$$I = 0.2989 \times \mathrm{R} + 0.5870 \times \mathrm{G} + 0.1140 \times \mathrm{B} \tag{2.1}$$

where, R, G, and B stand for red, green and blue channels, respectively and I is the grayscale image. The advantage of this conversion is that it does not treat each color channel equally, which corresponds to the fact that humans' eyes perceive green more strongly than red and red more strongly than blue. Figure 2.3 shows an example of RGB to gray conversion.

Fig. 2.3 An example of converting an RGB image (*left*) to a *grayscale* image (*right*)

(3) Frame resizing. Due to different specifications of video recording or image capturing equipment, video frames may have large variations in the resolution of pixels. For roadside video analysis systems that have requirements of the size of input data and computational time, it becomes a necessary step to resize frames to a desired resolution. A crucial step is to choose a proper rescaled size to accommodate data with different resolutions. Normally, down-scaling the size of frames results in the loss of some information about the content, while up-scaling introduces artificially generated information, which may significant impact the performance of the system. There are also various types of resizing algorithms available for use, such as nearest-neighbor interpolation, bilinear interpolation, and bicubic interpolation, which also have influence on the quality of resized frames. Figure 2.4 shows the results of image resizing on samples from the cropped roadside object dataset.

(4) Histogram equalization. Data captured in real-world environmental conditions may be exposed to different lighting effects, such as shadow, shining, under- and over-exposure. These effects form a major challenge for the robustness of machine learning algorithms, as they may substantially change the appearance of a proportion or all parts of the scene data and lead to confusion between objects. Although many studies [2] have investigated techniques to overcome some of these effects in scene content understanding, it is a common pre-processing step to firstly perform illumination adjustments to ensure even illumination in the scene data before inputting the data into further processing steps. One of most popular approaches for handing uneven illumination is performing histogram equalization, which transforms the intensity image into one with an approximately equally distributed histogram, and thus the lighting effects are reduced.

(5) Noise removal. Image or video data is often prone to a wide range of noise, such as salt and pepper noise, that do not reflect the true intensities of real-world objects. The noise can be introduced at the data acquisition stage, transmission

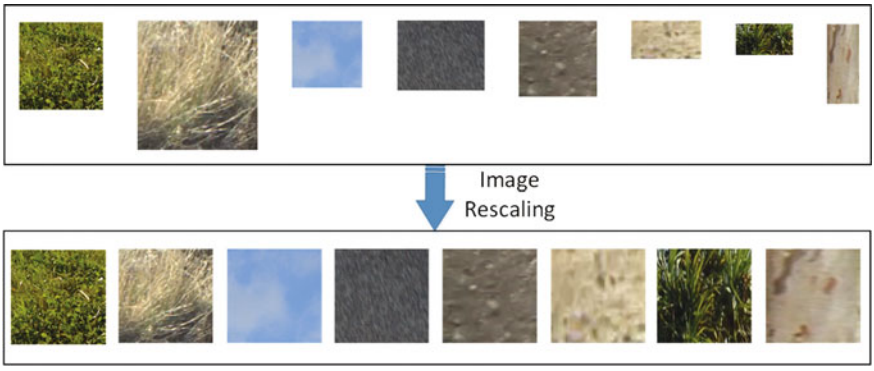

Fig. 2.4 Examples of cropped roadside regions resized into the same resolution by applying an image resizing technique

stage, or post-processing stages, depending on the method used for data creation. There are many image filtering methods for noise removal, such as the averaging filter, medium filter, Sobel filter, and Wiener filter. Figure 2.5 displays an example image with noise removed by applying a median filter.

(6) Sample region selection. Within the whole frame captured to represent a wide range of scene content and objects, in most cases, only a proportion of the frame is of great interest to the end user or a specific application. Thus, it becomes necessary to perform sample region selection to obtain the Region of Interest (ROI) from the scene that corresponds precisely to the roadside region used in the practice. The region selected is often different for different applications and this selection process can often be assisted by manual cropping by human or pre-setting the location, size, and shape of the ROI in an automatic system. Figure 2.6 shows an example of a sample region in field tests and its corresponding sample region in the captured image.

Fig. 2.5 An example of noise removal (*right*) by applying a median filter of 3 × 3 pixels to an original roadside image (*left*)

Fig. 2.6 A sampling region indicated by a *white plastic square* in field tests (*left*) and one possible corresponding sample region indicated by a *red rectangle* in the image (*right*)

2.2.2 Segmentation of Roadside Video Data into Objects

Object segmentation aims to find out the type of each possible object and the location where it is present in the video or frame data. Object segmentation is a pre-requisite step in many computer vision tasks, and it supports detailed analysis or further processing on the objects of interest. Object segmentation itself is a relatively popular research direction with numerous studies in the literature from the perspectives of scene labelling, scene parsing, image segmentation, etc. However, automatic and accurate segmentation of vegetation from natural roadside data is still a challenging task, due to substantial variations in both the unconstrained environment and the appearance of objects. The data may be accompanied with various outdoor environmental effects, such as overexposure, underexposure, shadow and sunlight reflectance. It is still a difficult task to accurately predict the type and appearance of objects that are present in a new scene, even with prior knowledge about the location, season, time, weather condition, etc. Figure 2.7 shows two roadside frames and their corresponding object segmentation results.

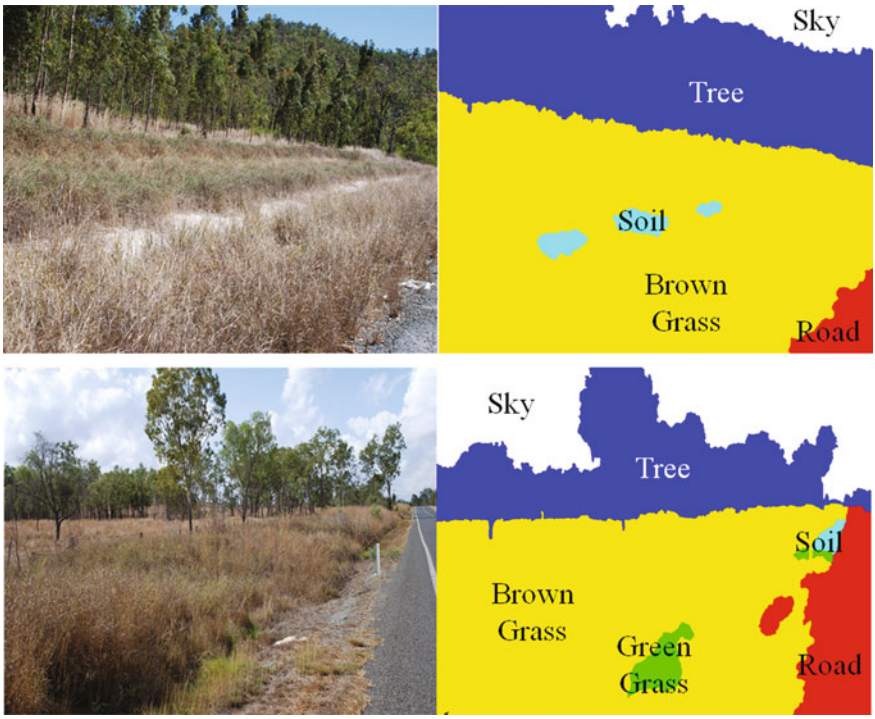

Fig. 2.7 Graphic illustration of object segmentation in roadside frames. The frames are segmented into different object categories, such as tree, sky, road, soil, brown grass, and green grass

2.2.3 Feature Extraction from Objects

To be able to recognize objects, it is often a pre-requisite to extract a set of features that can effectively represent visual characteristics of different objects. Dependent on the aim of a specific application, different types of features need to be extracted. For instance, sky can be generally represented in a blue or while color, while trees are primarily featured by a green or yellow color. It should be noted that the features can be used for both object segmentation and object classification tasks. Challenges for automatic feature extraction are changes in the color and intensity of light in different sunshine conditions and outdoor environments, as well as the lack of specific shapes and texture for most types of vegetation. Most recent studies reporting successful segmentation of vegetation have focused on features extracted from specific species of vegetation but not general vegetation.

According to the optical spectrum of data capturing equipment, features used in existing studies can be roughly divided into two categories: visible features and invisible features.

(1) Visible features reflect the shape, texture, geometry, structure and color characteristics of roadside objects such as sky, road and soil in the visible spectrum. These features are often extracted in the visible spectrum, and thus have high consistency with human eye perception. Color is one of the dominant resources that the human eyes depend on in the perception and discrimination of different types of objects. Some vegetation do not have a specific type of texture or shape, but usually can be represented by a dominant color. The usual color channels presented in vegetation regions include green, red, orange, brown and yellow, and the most popular color spaces include RGB, HIS, HSV, YUV, and CIELab. However, there are also objects that share similar color characteristics and thus cannot be easily distinguished solely using color features, specifically in complicated real-world environmental conditions. For instance, vegetation color is believed to be green in the HSV space under most environmental conditions. However, this may not be the case in scenes containing sky and with varying lighting conditions, such as the presence of shadow, shining, under- and over-exposure effects. For objects that are difficult to be discriminated by color features, other types of features such as texture, location, and geometric properties are able to provide complementary information and become crucially important for robust classification of those objects. Thus, it is advisable to fuse multiple types of features for better results in natural conditions. Examples of texture features used in the field of computer vision include Local Binary Patterns (LBPs), Gabor filter, Scale-Invariant Feature Transform (SIFT), Histograms of Oriented Gradients (HOGs), and GLCM.
(2) Invisible feature approaches extract the reflectance characteristics of vegetation in the invisible spectrum to differentiate them from other objects. It is widely known that vegetation needs to use chlorophyll to convert sunlight radiant energy from the sun into metabolic energy, which exhibits unique absorption characteristics of wavelengths. Based on this, various types of Vegetation

Indices (VIs) have been designed to characterize the differences between the spectral properties of vegetation and those of other objects on the available bands, especially for green and near infrared wavelengths. One great advantage of invisible features is that they often retain a high robustness against large variations in environmental conditions, such as illumination changes and light exposure, and thus they are suitable for achieving consistently stable performance in real-world environments. By contrast, one major drawback of invisible features is that they require specialized data capturing equipment, such as LIght Detection And Ranging (LIDAR), near-infrared cameras, and sensors. This requirement, to some extent, limits direct adoption of invisible features in a wide range of applications. It still remains a question as to how to define VIs capable of being adapted to any kind of natural conditions. Table 2.1 shows a list of different types of features used in existing studies.

2.2.4 *Classification of Roadside Objects*

Classification of objects aims to recognize the type or state of objects in roadside data, such as fuel load of grass, height of tree, content of traffic sign, and width of road. Given the feature sets extracted for each object, the key task is to design a proper machine learning algorithm that is capable of robustly predicting the state of objects of interest. Although humans are able to easily identify the state of some objects without being impacted by environmental effects such as shadows of objects, the use of machines for automatic object classification is still a challenging task. In the literature, there are a wide range of algorithms and they can be generally

Table 2.1 Typical types of features used in existing studies for video data analysis

Category	Sub-category	Feature
Visible	Color space	Lab, RGB, HSV, YUV, etc.
	Color statistics	Histogram, mean, standard deviation, max, min, variance, entropy, etc.
	Texture	LBP, SIFT, HOG, Gabor filter, GLCM, CWT, Pixel Intensity Difference (PID), etc.
	Geometry	Location, size, shape, area, centroid, eccentricity, etc.
	Motion	Optical flow
	VI	Excess Green (ExG), Excess Red (ExR), Visible Vegetation Index (VVI), Color Index of Vegetation Extraction (CIVE), etc.
Invisible	–	Normalized Difference Vegetation Index (NDVI)
	–	Near Infrared Ray (NIR)
	–	Modification of NDVI (MNDVI)
	–	Laser reflectivity

Note Vegetation Index (VI) can be either invisible or visible features

classified into supervised learning and unsupervised learning, which are briefly introduced below.

(1) Supervised learning, which normally involves the design of a suitable machine learning algorithm and finds the optimal parameters of the algorithm based on a labeled training dataset. Each sample in the dataset is composed of an input object and a desired output value, and the whole dataset is often divided into training, validation and test subsets. The parameters of the algorithm are firstly trained using the training subset, further evaluated using the validation subset, and finally applied to classify the state of objects on the test subset, producing performance measures such as prediction accuracy. The most widely used supervised learning based techniques include Artificial Neural Network (ANN), Support Vector Machine (SVM), decision tree, random forest, non-linear regression, Conditional Random Field (CRF), and nearest neighbor algorithm.
(2) Unsupervised learning, which performs predictions without the need of the training data, and directly draws inferences from the dataset with or without labelled ground truths. Unlike supervised learning where a training data subset should be prepared and pre-labelled with ground truths, unsupervised learning tries to infer a prediction function that can best describe the pattern information in the data that is labeled or unlabeled, and thus it has the advantage of not requiring labelling work which is crucially important for applications where manual annotation of ground truths is difficult or even impossible in practice. The most widely unsupervised algorithms include K-means clustering, hierarchical clustering, principal component analysis, independent component analysis, non-negative matrix factorization, and the Self-Organizing Map (SOM).

2.2.5 Applications of Classified Roadside Objects

Once roadside objects are classified, there are a lot of potential applications that can be generated that play an important role in specific areas, such as agriculture, transport, road safety, and natural disaster prevention. This section lists several examples of such applications.

(1) Traffic sign detection. Traffic signs are one of the most important signals in regulating and guiding vehicle drivers for safe driving. Automatic traffic sign detection could be useful in assisting drivers making correct driving decisions, especially in severe weather conditions and for signs that are not visible enough. It is also crucial in developing automatic guidance vehicles with smart sensing capacity that can drive automatically in various road conditions, or that can send alerts to drivers to avoid possible accidents.
(2) Fire-prone region identification. Roadside fire risk arising from roadside vegetation, such as brown grasses and trees, is a major hazard to road safety and can potentially be a contributory factor to major disasters, such as bushfires.

The current practice of transport authorities is heavily dependent on human visual checks to find fire-prone roadside regions, and they still lack effective systems to automatically identify fire-prone regions. The implementation of automatic techniques to handle this issue has become increasingly important, and the investigation of robust machine learning algorithms for roadside object classification can bring us a step closer to this goal. Figure 2.8 shows examples of roadside frames with high or low fire risk.

(3) Roadside vegetation management. Effective roadside vegetation management requires dynamically, accurately, and constantly monitoring of the growth conditions of different species of roadside vegetation. Being able to obtain vegetation species along specific road sites in a certain season can help farmers and agricultural professionals make better decisions and more effective plans on the necessary treatments to ensure health conditions of vegetation and eliminate possible obstacles, such as insects and dry weather.

(4) Roadside tree regrowth control. In some roadside regions, trees can grow progressively approaching the boundary of the road, and thus can potentially pose a hazard to road safety. It is therefore necessary to implement automatic methods to identify the conditions of these potentially dangerous trees and take appropriate actions accordingly to eliminate the potential hazard. There are generally four levels of regrowth conditions, including heavy, medium, light and zero, as shown in Fig. 2.9. A 'zero' level indicates big trees which are far

Fig. 2.8 Examples of roadside grasses with high and low fire risk

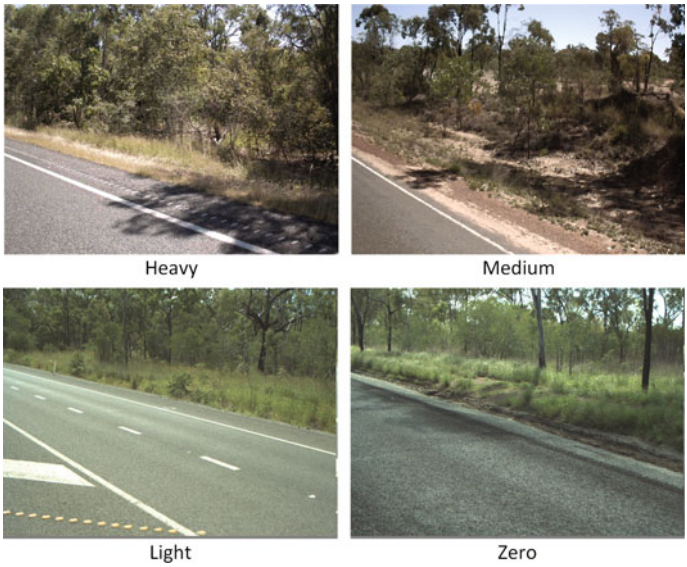

Fig. 2.9 Examples of four levels of tree regrowth conditions, including heavy, medium, light and zero

away from the road, while a 'heavy' level implies small trees and shrubs which are close to the road. Normally, trees within 10 meters are considered as being close and dangerous. The results can assist service providers in deciding the right equipment for managing the regrowth conditions at the right places. Small equipment is often required to cut small trees and shrubs, while big equipment for large trees. This is also useful for estimating the cost, as removal of large trees is expensive, while small shrubs require relatively less expense. The aim of tree growth control is to effectively handle regrowth problems, minimize road deficiency, and reduce associated liability.

2.3 Related Work

In this section, we review related work on vegetation segmentation and classification. In addition, we also briefly review existing approaches to segmenting objects from generic scenes, which can be potentially used for vegetation segmentation. It is noted that most of existing work related to vegetation segmentation are from the fields such as remote sensing [3] and ecosystems which use different types of sensors, laser scanners, radar and special types of autonomous vehicles. This section limits the focus to reviewing only those approaches that utilize ground data collected using ordinary digital cameras.

2.3.1 Vegetation Segmentation and Classification

According to the type of features used, existing studies on vegetation segmentation and classification can be roughly grouped into three categories: visible feature approach, invisible feature approach, and hybrid feature approach.

2.3.1.1 Visible Feature Approach

Visible feature approaches attempt to distinguish vegetation from other objects, such as soil, tree, sky and road, by exploring their discriminative characteristics in the visible spectrum, such as color, shape, texture, geometry and structure features. A major advantage of using visible features is that they retain high consistency with human visual perception of objects.

Color is one of the dominant information that the human eyes depend on in the perception and discrimination of different objects in real-world environments. Most vegetation are characterized primarily by a green or orange color, and thus color is one of most widely used features in existing studies on vegetation segmentation, which mainly focus on investigating the suitability of various color spaces, such as CIELab [4], YUV [5], HSV [6], and RGB [7]. However, it is still a challenging task to find a suitable color representation of vegetation in complex natural conditions. Designing color spaces that are illumination invariant or able to automatically adapt to the dynamically changing environment is still an active research direction [1].

Except for color, another popular type of visible feature is texture, which mainly reflects the appearance structure of objects and is often represented by performing wavelet filters, such as Gabor filters [8] and Continuous Wavelet Transform (CWT) [6], extracting pixel intensity distributions, such as Pixel Intensity Differences (PIDs) [4, 5] and variations in a neighborhood [9, 10], or generating spatial statistic measures [10], entropy [7], or statistical features over superpixels [11].

Table 2.2 lists typical visible approaches for vegetation segmentation in existing studies. One of the early studies on vegetation segmentation in outdoor images was presented in [12], which employed an SOM for object segmentation and then extracted color, texture, shape, size, centroid and contextual features of segmented regions for 11 object classification using a Multi-Layer Perceptron (MLP). In [7], the entropy was used as a texture feature, together with RGB color components and an SVM classifier for detecting vegetation from roadside images. The intensity differences between pixels were combined with a 3D Gaussian model of YUV channels for grass detection [5], and with L, a, and b color channels for object segmentation [4]. The motion between video frames estimated by optical flow was also employed in a pre-processing step to detect a ROI [6], from which color and texture features were extracted using a two-dimensional CWT, and also to assist vegetation detection by measuring the resistance of vegetation [13]. In [14], LBP and GLCM were combined for discriminating between dense and sparse roadside

Table 2.2 Summary of typical visible approaches for vegetation segmentation

Ref.	Color	Texture	Classifier	Object	Data	Acc. (%)
[12]	RGB, O1, O2, R − G, (R + G)/2 − B	Gabor filter, shape	SOM + MLP	Veg, sky, road, wall etc.	3751 R	61.1
						80
[7]	RGB	Entropy	RBF SVM + MO	Veg versus non-veg	270 I	95.0
[6]	RGB, HSV, YUV, CIELab	2D CWT	SVM + MO	Veg versus non-veg	270 I	96.1
[5]	YUV (3D Gaussian)	PID	Soft segmentation	Grass versus non-grass	62 I	91
[8]	O1, O2	NDVI and MNDVI	Spreading rule	Veg versus non-veg	2000 I	95
		Gabor filter			10 V	
[18]	H, S	Height of grass (ladar)	RBF SVM	Grass versus non-grass	N/A	N/A
[4]	Lab	PID	K-means clustering	Object segmentation	N/A	79
[10]	Gray	Intensity mean and variance, binary edge, neighborhood centroid	Clustering	Grass versus artificial texture	40 R	95
						90
[14]	Gray	LBP, GLCM	SVM, ANN, KNN	Dense versus sparse grass	110 I	92.7
[15]	RGB, HLS, Lab	Co-occurrence matrix	Gaussian PDF + global energy	5, 5 and 7 objects	41 I	89.9
					87 I	90.0
					100 I	86.8
[19]	RGB, Lab	Color moment	Superpixel merging	7 objects	650 I	>90
					50 I	77%

N/A not available, *Veg* vegetation, *non-veg* non vegetation, *I* image, *V* video, *R* region

grasses using majority voting over three classifiers—SVM, ANN, and K-Nearest Neighbour (KNN). In [15], RGB, HLS, and Lab color channels and co-occurrence matrix based texture features were fused for outdoor scene analysis. A set of initial seed pixels was selected based on probabilistic pixel maps which were built using a Gaussian probability density function on a selected subset of color and texture features, and pixels were then grown from the initial seeds by integrating region and boundary information in the minimization of a global energy function.

There are also many studies [16, 17] that investigated detecting or classifying crops from other objects such as soil and weeds using image or video data captured in crop fields. Most of the studies accomplished the recognition task using the green color characteristics of crops, and were often based on simplified environmental conditions rather than natural conditions and thus are not reviewed here.

Most existing visible approaches focus on a binary classification of vegetation versus non-vegetation. Although various types of color and texture features in the visible spectrum can often achieve promising performance, there is no common feature set that is widely accepted as being capable of working well in natural conditions. Most visible approaches become problematic when similar kinds of objects present in the scene like grasses and trees, and green vehicles and green grasses, and when illumination conditions change. An alternative solution is adopting features in the invisible spectrum which are capable of remaining better robustness against environmental variations.

2.3.1.2 Invisible Feature Approach

Invisible feature approaches use the spectral properties of chlorophyll-rich vegetation and their reflectance characteristics in the invisible spectrum to differentiate vegetation from other objects, or to determine their properties (e.g. passable vegetation detection in vehicle navigation [13]). It is well known that vegetation needs to use chlorophyll to convert sunlight radiant energy from the sun into metabolic energy, which exhibits unique absorption characteristics of wavelengths. From this theory, various types of VIs have been designed to highlight the differences between the spectral properties of vegetation and those of others on the available bands, especially for green and near infrared wavelengths. Compared with visible features, one prominent advantage of invisible features is that they have better robustness to environmental variations such as shadow, shining and underexposure effects.

The power of VI for vegetation classification has been demonstrated by the fact that a simple pixel-by-pixel comparison between red and Near Infrared Ray (NIR) reflectance potentially provides a powerful and robust way to detect photosynthetic vegetation [20]. In general, healthy and dense vegetation reflects high NIR and low red reflectance. Conversely, sparse and not so healthy vegetation shows low NIR reflectance but high red reflectance. The NIR has been modified to the Normalized Difference Vegetation Index (NDVI) which was successfully applied to vegetation detection [20]. Under illumination changes, Nguyen et al. [2] found that the hyperplane to classify vegetation and others could be in a logarithmic form instead of a linear one in the standard form of NDVI, and thus they proposed the Modification of NDVI (MNDVI). They also experimentally proved that the MNDVI performs more robust and stable vegetation detection than the NDVI under various illumination effects such as shadow, shining, under- and over-exposure. However, the softening red reflectance impact in MNDVI presents problems in an under-exposure or a dim lighting condition. In contrast, NDVI reveals good performance in those circumstances. Thus, a combination of them was adopted in [13] to achieve more robustness against illumination changes. Wurm et al. [21] measured the remission of laser to classify vegetation and non-vegetation regions in structured environments. Spectral reflectance of vegetation was introduced by Bradley et al. [20] for ground-based terrain classification, and this method was

Table 2.3 Summary of typical invisible approaches for vegetation segmentation

Ref.	Feature	Classifier	Object	Data	Result (%)
[20]	Density, surface normal, scatter matrix eigenvalue, RGB, NIR, and NDVI	Multi-class logistic regression	Veg, obstacle or ground	Two physical environments	95.1
[2]	MNDVI	Threshold	Veg versus non-veg	5000 I, 20 V	91
[13]	MNDVI, NDVI, background subtraction, dense optical flow	Fusion	Passable veg versus others	1000 I	98.4
[22]	Laser reflectivity, measured distance, incidence angle	SVM	Flat veg versus drivable surfaces	36,304 veg 28,883 street	99.9
[23]	Filter banks in Lab and infrared	Joint boost + CRF	Eight classes (road, sky, tree, car, etc.)	2 V	87.3
[9]	Ladar scatter features, intensity mean and std., scatter, surface, histogram	SVM	Veg versus non-veg	500 I	81.5

Veg vegetation, *non-veg* non vegetation, *I* image, *V* video

further improved with better robustness by adopting the laser. It has been shown that, by adding independent light and varying the exposure time, a vegetation detection system is able to perform more robustly against varied illumination conditions [2]. Table 2.3 lists typical invisible approaches for vegetation segmentation in existing studies.

However, one major drawback of invisible feature approaches is that they require specialized data capturing equipment, such as laser scanner, near-infrared camera, and sensor. Thus, a direct adoption of invisible features in various applications is still restricted by this requirement. How to define VIs capable of being adaptive robustly to any kind of natural conditions is still a question in this field?

2.3.1.3 Hybrid Feature Approach

Hybrid feature approaches combine invisible and visible features for more robust and accurate classification results, by utilizing both the capacity of visible features in representing visual appearance of objects and the robustness of invisible features to environmental effects.

Nguyen et al. [8] introduced an active method for a double-check of passable vegetation detection. To calculate statistics features, they used a sliding cube across 3D point clouds, and in each sliding cube, a positive definite covariance matrix was calculated. From the covariance matrix, eigenvalues and eigenvectors were

extracted to represent two types of 3D point cloud statistics, including scatter and surface, where scatter represents vegetation such as bushes, tall grasses, and tree canopy, while surface represents solid objects like rocks, ground surface, and tree trunks. However, the 3D features would have difficulty achieving robust vegetation segmentation as they did not consider color information. Therefore, Nguyen et al. [9] proposed a 2D and 3D fusion approach for outdoor automobile guidance with the consideration of color information to detect the location of vegetation areas in the viewed scene. They used six features for training an SVM classifier, including intensity, histograms of color features and 3D scatter features. The intensity features include mean and standard deviation of brightness and color in the HSV space, while the 3D scatter features reflect the spatial structure of vegetation in the local neighborhood of LADAR data. The limitations of this approach lie in the requirement of a long processing time as well as feature values being highly dependent on the environment, the type of sensors, the number of scanned points and the point density. In a similar fashion, Liu et al. [18] combined 2D and 3D features to discriminate between grass and non-grass areas, where the height and color information were obtained from a multi-layer radar and a color camera respectively. The color information is represented by the H and S components in the HSV space. Lu et al. [24] and Nguyen et al. [25] presented approaches combining multiple features e.g. color, texture and 3D distribution information, for vegetation segmentation. In [23], a feature vector from 20-D filter banks was extracted from both visible L, a, b and infrared channels for road object segmentation. A hierarchical bag-of-textons method was then introduced to capture the spatial neighborhood information by extracting multiscale texton features from larger neighboring regions. The textons are essentially centres of each pattern generated using a clustering algorithm. Around 87% global accuracy was achieved for classifying eight objects on the authors' own road scene video dataset. In [8], the opponent color space and Gabor features were combined to measure the similarity between a pixel and its neighbors for vegetation pixel spreading. The NDVI and MNDVI were fused to select chlorophyll-rich vegetation pixels as the initial seed pixels for spreading.

Hybrid feature approaches often produce better results than visible or invisible feature approaches, but they also inherit the drawbacks of both of them such as requiring specialized data capturing equipment. How to select a proper set of visible and invisible features for combination is still another question that should be further explored.

2.3.2 Generic Object Segmentation and Classification

Generic object segmentation and classification approaches aim to find the region where a specific object is present. They often share similar concepts with scene labelling, object categorization, semantic segmentation, etc. in computer vision tasks, and can be potentially applied into roadside vegetation segmentation.

The design of an object classification system has to address several processing tasks, including choosing suitable elementary regions (e.g. pixel, patch and superpixel—groups of neighboring pixels sharing similar appearance and perceptually meaningful atomic regions, as examples shown in Fig. 2.10), selecting discriminative visual features to characterize them (e.g. color, texture and geometry), building robust prediction models for obtaining class label confidence, extracting effective contextual features, and integrating prediction models with contextual information. According to the techniques or features used, existing approaches can be divided into different categories, such as parametric versus non-parametric, supervised versus unsupervised, and pixel based versus region based.

Early approaches to object segmentation obtain class labels for image pixels using a set of low-level visual features extracted at a pixel-level [15, 27] or patch-level [23]. However, because pixel-level features treat each pixel individually, they are unable to capture statistical characteristics of objects in local regions. While patch-level features are able to capture regional statistic features, they are prone to noise from background objects due to the difficulty in accurate segmentation of object boundaries. Recent studies [28–31] have focused more on the use of superpixel-level features as the basic unit for object segmentation, which showed

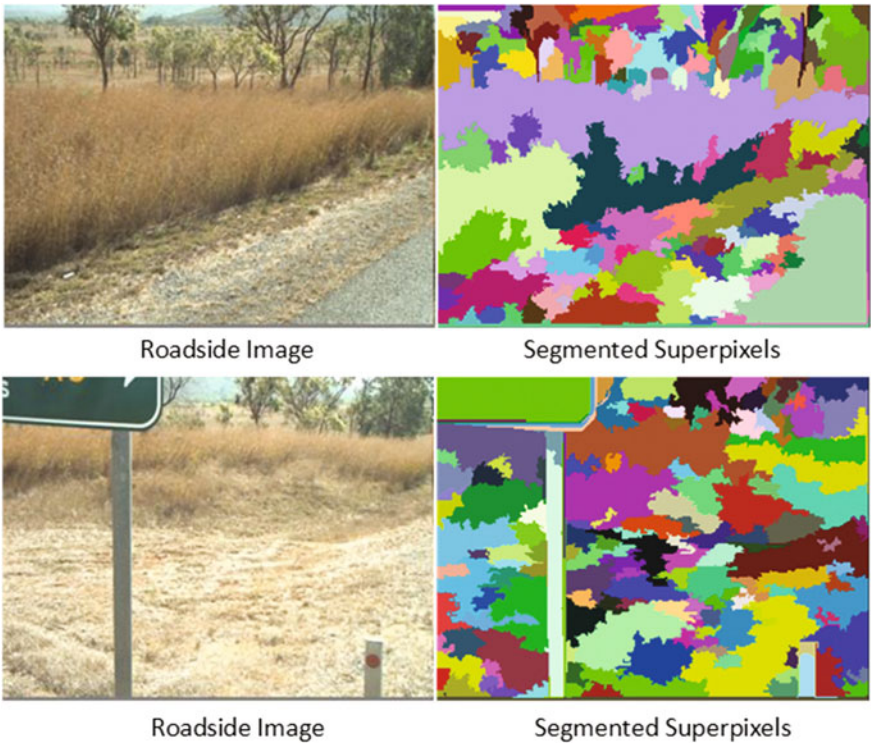

Fig. 2.10 Visual displays of segmented superpixels in roadside images using a graph-based segmentation algorithm [26]. Different colors indicate different superpixel regions

promising results of extracting discriminative features. Superpixel-level features have several advantages over traditional patch-level features, including being coherent support regions for a single labelling on a naturally adaptive domain rather than on a fixed window, supporting more consistent statistic feature extraction capturing contextual information by pooling over feature responses from multiple pixels, and requiring less computational time. The most commonly adopted superpixel-level features include color (e.g. RGB [32, 33] and CIELab [27, 28, 32, 34, 35]), texture (e.g. SIFT [33, 36], texton [27, 33], Gaussian filter [32], Gist [34] and Pyramid Histogram of Oriented Gradients—PHOGs [34]), appearance (e.g. color thumbnail [33]), location [37], and shape.

Despite the benefits, as visual feature based prediction treats each superpixel independently and does not consider semantic context about the scene, it often faces challenges for object segmentation in complex scenes. For segmentation algorithms, the vast majority of existing approaches focus on graphical models, such as CRF [31] and Markov Random Field (MRF) [33]. Work [31] investigated the use of superpixel neighborhoods for object identification by merging the histogram of a superpixel with neighbors in conjunction with CRF. But the approach still has a high dependence on initial seed superpixel selection. Tighe and Lazebnik [33] obtained superpixel labels using Naive Bayes and utilized a minimization of MRF energy over superpixel labels to enforce contextual constraints of objects. Recently, Balali and Golparvar-Fard [11] extracted a set of superpixel features for the recognition of roadway assets using MRF. Current superpixel based approaches pre-dominantly rely on graphical models (e.g. CRF and MRF), which enforce the spatial consistency of category labels between neighboring superpixels (or pixels) by jointly minimizing the total energy of two items—unary potentials which indicate the likelihood of each superpixel (or pixel) belonging to one of the semantic categories and pairwise potentials which account for the spatial consistency of category labels between neighboring superpixels (or pixels). However, flat graphical models have limited capacity of capturing higher order context.

To overcome this limitation, a wide variety of approaches have been proposed to incorporate contextual information to improve object segmentation accuracy, which is often conducted at two stages: feature extraction and label inference. Feature extraction incorporates the context by designing a rich set of semantic descriptions that represent intrinsic correlations between objects in different types of scenes. The commonly used contextual features include absolute location [27], which captures the dependence of class labels on the absolute location of pixels in the scene, relative location [32], which represents the relative location offsets between objects in a virtually enlarged image, directional spatial relationships [38, 39] which encode spatial arrangements of objects such as beside, below, above and enclosed, and object co-occurrence statistics [36], which reflect the likelihood of two objects co-existing in the same scene. The main drawback of the relative location, directional spatial relationships and object co-occurrence statistics is that they completely discard absolute spatial coordinates of objects in the scene and therefore they cannot capture spatial contextual information such as a high likelihood of sky appearing at the top part of a scene. By contrast, the absolute location excessively

retains all pixel coordinates of objects and thus it often demands a large training dataset to collect reliable prior statistics for each object and each image pixel.

Another popular approach of incorporating context is to use graphical models at the label inference stage, such as CRF [29, 40, 41], MRF [42] and energy function [15]. However, these graphical models have two shortcomings: (1) label purity of superpixels cannot be guaranteed due to the difficulty of prefect image segmentation; and (2) only contextual information in local neighborhoods is considered. To handle these shortcomings, one approach is to adopt hierarchical models, such as hierarchical CRF [43], stacked hierarchical learning [34] and pylon model [43], which generate a pyramid of image superpixels and perform classification optimization over multi-levels of images to alleviate the effect of inaccurate region boundaries and utilize higher order contextual information. Another approach is to extract feature descriptions from multiple regions, such as aggregated histograms [31] and weighted appearance features [32]. Although these approaches account for larger context, they still cannot fully capture long-range dependencies of objects in the entire scene and are unable to adapt to scene content.

Another drawback of graphical models is that their parameters are solely learnt from the training data, and thus their performance heavily depends on the availability of adequate training data and they have a generalization issue for new test data. For real-world applications, it is often very difficult or even impossible to collect a large number of training data to ensure adequate training, on top of the fact that it is also extremely time consuming and labor intensive. One solution to this problem, particularly for large datasets, is adopting non-parametric approaches [44], which retrieve the most similar training images to a query image and then perform class label transfer from K-nearest neighbours in the retrieval set to the query image. However, non-parametric approaches still depend on the reliability and accuracy of the retrieval strategies.

Recently, deep learning techniques have shown great advantages in extracting discriminative and compact features from raw image pixels rather than using hand-engineered features. The widely used CNNs utilize convolutional and pooling layers to progressively extract more abstract patterns and demonstrate state-of-the-art performance in many vision tasks [45] including object segmentation. The extracted CNN features can be combined with various classifiers (e.g. MRF, CRF and SVM) to predict class labels. A representative work is by Farabet et al. [29], which applied hierarchical CNN features into CRFs for class label inference in natural scenes. However, the CRF inference is completely independent from CNN training, and thus Zheng et al. [46] formulated the CRF inference as recurrent neural networks and integrated them in a unified framework. In [47], the recurrent CNN feeds back the outputs of CNNs to the input of another instance of the same network, but it works only on sequential data. Recent extensions to CNN models include AlexNet, VGG-19 net, GoogLeNet, and ResNet [48]. However, these models often require adequate image resolutions and may not be directly applicable for roadside vegetation segmentation on datasets such as the cropped roadside object dataset which has lower resolutions and substantial variations in the shape and size.

2.4 Matlab Code for Data Processing

This section introduces several commonly used algorithms for video data pre-processing, feature extraction, object segmentation and classification. The Matlab codes are provided to illustrate the processing steps.

1) Extract all frames from video data.

```
% this code takes as input video data, extracts all frames from the video, and
% saves all framesin a new folder.

% read video data into a variable ‘mov’. The path and name of the video are
% represented by the string ‘videoFilePathName’.
mov = VideoReader(videoFilePathName);

% set the output folder ‘outFrameFold’ for storing extracted frames.
frameFolder = outFrameFold;

% create the folder if it does not exist.
if ~exist(frameFolder, 'dir')
   mkdir(frameFolder);
end

% get the total number of frames in the video.
numFrame = mov.NumberOfFrames;

% extract and save each of all frames in a loop.
For iFrame = 1 : numFrame
    % read the ith frame.
    I = read(mov, iFrame);

    % set an index for each frame.
    frameIndex = sprintf('%4.4d', iFrame);

     % write frames with names like ‘Frame0001.jpg’ into the output folder.
    imwrite(I, [frameFolder 'Frame' frameNewName '.jpg'], 'jpg');

    % show the progress of data processing.
```

```
    progIndication = sprintf('frame %4d of %d.', iFrame, numFrame);
    disp(progIndication);
end
```

2) Convert RGB frames to grayscale frames.

```
% this code converts extracted video frames from a RGB color format to a
% gray scale.

% read frame data into a variable 'I'. The path and name of the frame are
% represented by the string 'imageFilePathName'.
I = imread(imageFilePathName);

% perform frame conversion.
I = rgb2gray(I);

% show converted frames in figures.
imshow(I);
```

3) Frame resizing.

```
% this code takes as input a frame, and resizes the frame into a desired
% width and height.

% perform frame resizing.
I = imresize(I,[numrows numcols]);

% show resized frames in figures.
imshow(I);
```

4) Apply a median filter to a frame.

```
% this code takes as input a frame, and applies a median filter to remove
% noise in the frame.

% convert RGB to grayscale frames.
I = rgb2gray(I);

% perform image filtering using a median filter.
K = medfilt2(I);

% show filtered frames in figures.
imshowpair(I,K,'montage');
```

5) Pixel-level R, G, B feature extraction.

```
% this code takes as input a frame, and extract R, G, and B values at each
% pixel.

% get the dimension of the input frame.
```

```
[numRows, numCols] = size(I);

% scan pixels across all rows and columns of the frame.
for iRow = 1 : numRows
   for iCol = 1 : numCols
          % get R, G and B values at each pixel of the frame.
          pixelRValue = I(iRow, iCol, 1);
          pixelGValue = I(iRow, iCol, 2);
          pixelBValue = I(iRow, iCol, 3);
   end
end
```

6) Patch-level Gaussian feature extraction.

```
% this code takes as input a frame, extracts Gaussian features from a local
% patch centred at each pixel, and stores the resulting features in a 3D matrix.

% set parameters of Gaussian filters.
fixV = 0.7;
size = [7 7];
sigma = 1;
% create Gaussian filters.
fgaus = fspecial('gaussian',size, sigma*fixV);

% get the dimension of the input frame.
[numRows, numCols] = size(I);

% get R, G, and B matrixes separately from the frame.
R = I(:, :, 1);
G = I(:, :, 2);
B = I(:, :, 3);

% apply Gaussian filters to R, G, and B matrixes separately.  The resulting
% Gaussian features are stored in a 3D matrix 'filterI'.
filterI (:, :, 1)=conv2(R,fgaus,'same');
filterI (:, :, 2)=conv2(G,fgaus,'same');
filterI (:, :, 3)=conv2(B,fgaus,'same');
```

7) Patch-level statistical feature extraction.

```
% this code takes as input a frame, extracts statistical features from a local
% patch centred at each pixel, and stores the resulting features in variables.

% get R, G, and B matrixes separately from the input frame.
R = I(:, :, 1);
G = I(:, :, 2);
B = I(:, :, 3);
```

```
% set the half size of patches.
nHalfBlock = 4;

% set parameters for handling border pixels.
adHeight = nHeight - nHalfBlock;
adWidth = nWidth - nHalfBlock;
adBegin = nHalfBlock +1;

% scan across all rows and columns of the frame.
for iRow = adBegin:adHeight
      rowBeg = iRow - nHalfBlock;
      rowEnd = iRow + nHalfBlock;
      for iCol = adBegin:adWidth
            colBeg = iCol - nHalfBlock;
            colEnd = iCol + nHalfBlock;

            % get a patch region from which statistical features are extracted.
            patchR = R(rowBeg: rowEnd, colBeg: colEnd);

            % calculate mean, standard deviation, and skewness of R values
            % in the patch region.
            meanPatchR = mean(patchR(:));
            stdPatchR = std(patchR(:));
            skewPatchR = skewness(patchR(:));
      end
end

8) Object segmentation and classification.
% this code demonstrates the segmentation/classification of objects in
% frames, which accepts a frame as the input and assigns each of all pixels into
% an object category.

% the dimension of the input frame.
[numRows, numCols] = size(I);

% create a variable for storing object categories of all pixels.
objCategory = zeros(numRows,numCols);

% scan across all rows and columns in the frame.
for iRow = 1 : numRows
   for iCol = 1 : numCols
         % get features at each pixel, i.e. RGB values in this example.
         pixelRGBValue = I(iRow, iCol, :);

         % apply a function named 'objSegAlgorithm' to obtain the object
         % category for each pixel based on its features.
```

```
            objCategory(iRow,iCol) = objSegAlgorithm(pixelRGBValue);
        end
    end
```

9) Train and test an ANNclassifier.

```
% this code explains the creation and application of a three-layer feedforward
% ANN to classifying new test data.

% create an ANN with parameters of hidden neuron, activation function, and
% learning algorithm.
net = newff(double(TrainX'),double(TrainY'),[15],{'tansig','tansig'},'trainrp');

% parameters controlling the train-test processes of the ANN.
% generate command-line output.
net.trainParam.showCommandLine = true;
% randomly divide training, validation, and test data subsets.
net.divideFcn = 'dividerand';
% ratios of training, validation, and test data subsets.
net.divideParam.trainRatio = 1;
net.divideParam.valRatio = 0;
net.divideParam.testRatio = 0;
% performance measurement.
net.performFcn='mse';
% performance goal.
net.trainParam.goal=0.0001;
% maximum number of training epochs.
net.trainParam.epochs=500;
% minimum performance gradient.
net.trainParam.min_grad = 1e-8;
% maximum validation failures.
net.trainParam.max_fail = 10;

% train the ANN with the training data and store it in the variable ‘net’.
[net,tr] = train(net,double(TrainX'),double(TrainY'));

% prediction accuracy on the training data.
outTrainTag = sim(net,TrainX');
% prediction accuracy on the test data.
outTestTag = sim(net,TestX');
```

References

1. W. Maddern, A. Stewart, C. McManus, B. Upcroft, W. Churchill et al., Illumination invariant imaging: applications in robust vision-based localisation, mapping and classification for autonomous vehicles, in *Proceedings of the Visual Place Recognition in Changing Environments Workshop, IEEE International Conference on Robotics and Automation (ICRA)*, 2014
2. D.V. Nguyen, L. Kuhnert, K.D. Kuhnert, Structure overview of vegetation detection. A novel approach for efficient vegetation detection using an active lighting system. Robot. Auton. Syst. **60**, 498–508 (2012)
3. M.P. Ponti, Segmentation of low-cost remote sensing images combining vegetation indices and mean shift. IEEE Geosci. Remote Sens. Lett. **10**, 67–70 (2013)
4. M.R. Blas, M. Agrawal, A. Sundaresan, K. Konolige, Fast color/texture segmentation for outdoor robots, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 2008, pp. 4078–4085
5. B. Zafarifar, P.H.N. de With, Grass field detection for TV picture quality enhancement, in *International Conference on Consumer Electronics (ICCE), Digest of Technical Papers*, 2008, pp. 1–2
6. I. Harbas, M. Subasic, Motion estimation aided detection of roadside vegetation, in *7th International Congress on Image and Signal Processing (CISP)*, 2014, pp. 420–425
7. I. Harbas, M. Subasic, Detection of roadside vegetation using features from the visible spectrum, in *37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2014, pp. 1204–1209
8. D.V. Nguyen, L. Kuhnert, K.D. Kuhnert, Spreading algorithm for efficient vegetation detection in cluttered outdoor environments. Robot. Auton. Syst. **60**, 1498–1507 (2012)
9. D.V. Nguyen, L. Kuhnert, T. Jiang, S. Thamke, K.D. Kuhnert, Vegetation detection for outdoor automobile guidance, in *IEEE International Conference on Industrial Technology (ICIT)*, 2011, pp. 358–364
10. A. Schepelmann, R.E. Hudson, F.L. Merat, R.D. Quinn, Visual segmentation of lawn grass for a mobile robotic lawnmower, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 734–739
11. V. Balali, M. Golparvar-Fard, Segmentation and recognition of roadway assets from car-mounted camera video streams using a scalable non-parametric image parsing method. Autom. Constr. **49**(Part A), 27–39 (2015)
12. N.W. Campbell, B.T. Thomas, T. Troscianko, Automatic segmentation and classification of outdoor images using neural networks. Int. J. Neural Syst. **8**, 137–144 (1997)
13. D.V. Nguyen, L. Kuhnert, S. Thamke, J. Schlemper, K.D. Kuhnert, A novel approach for a double-check of passable vegetation detection in autonomous ground vehicles, in *15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2012, pp. 230–236
14. S. Chowdhury, B. Verma, D. Stockwell, A novel texture feature based multiple classifier technique for roadside vegetation classification. Exp. Syst. Appl. **42**, 5047–5055 (2015)
15. A. Bosch, X. Muñoz, J. Freixenet, Segmentation and description of natural outdoor scenes. Image Vis. Comput. **25**, 727–740 (2007)
16. W. Guo, U.K. Rage, S. Ninomiya, Illumination invariant segmentation of vegetation for time series wheat images based on decision tree model. Comput. Electron. Agric. **96**, 58–66 (2013)
17. F. Ahmed, H.A. Al-Mamun, A.S.M.H. Bari, E. Hossain, P. Kwan, Classification of crops and weeds from digital images: a support vector machine approach. Crop Prot. **40**, 98–104 (2012)
18. D.-X. Liu, T. Wu, B. Dai, Fusing ladar and color image for detection grass off-road scenario, in *IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, 2007, pp. 1–4
19. L. Zhang, B. Verma, D. Stockwell, Spatial contextual superpixel model for natural roadside vegetation classification. Pattern Recogn. **60**, 444–457 (2016)

20. D.M. Bradley, R. Unnikrishnan, J. Bagnell, Vegetation detection for driving in complex environments, in *IEEE International Conference on Robotics and Automation*, 2007, pp. 503–508
21. K.M. Wurm, R. Kummerle, C. Stachniss, W. Burgard, Improving robot navigation in structured outdoor environments by identifying vegetation from laser data, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 1217–1222
22. K.M. Wurm, H. Kretzschmar, R. Kümmerle, C. Stachniss, W. Burgard, Identifying vegetation from laser data in structured outdoor environments. Robot. Auton. Syst. **62**, 675–684 (2014)
23. Y. Kang, K. Yamaguchi, T. Naito, Y. Ninomiya, Multiband image segmentation and object recognition for understanding road scenes. IEEE Trans. Intell. Transp. Syst. **12**, 1423–1433 (2011)
24. L. Lu, C. Ordonez, E.G. Collins Jr., E.M. DuPont, Terrain surface classification for autonomous ground vehicles using a 2D laser stripe-based structured light sensor, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 2174–2181
25. D.-V. Nguyen, L. Kuhnert, T. Jiang, S. Thamke, K.-D. Kuhnert, Vegetation detection for outdoor automobile guidance, in *IEEE International Conference on Industrial Technology (ICIT)*, 2011, pp. 358–364
26. P. Felzenszwalb, D. Huttenlocher, Efficient graph-based image segmentation. Int. J. Comput. Vis. **59**, 167–181 (2004)
27. J. Shotton, J. Winn, C. Rother, A. Criminisi, Textonboost for image understanding: multi-class object recognition and segmentation by jointly modeling texture, layout, and context. Int. J. Comput. Vis. **81**, 2–23 (2009)
28. S. Gould, R. Fulton, D. Koller, Decomposing a scene into geometric and semantically consistent regions, in *IEEE 12th International Conference on Computer Vision (ICCV)*, 2009, pp. 1–8
29. C. Farabet, C. Couprie, L. Najman, Y. LeCun, Learning hierarchical features for scene labeling. IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1915–1929 (2013)
30. A. Sharma, O. Tuzel, D.W. Jacobs, Deep hierarchical parsing for semantic segmentation, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 530–538
31. B. Fulkerson, A. Vedaldi, S. Soatto, Class segmentation and object localization with superpixel neighborhoods, in *IEEE 12th International Conference on Computer Vision (ICCV)*, 2009, pp. 670–677
32. S. Gould, J. Rodgers, D. Cohen, G. Elidan, D. Koller, Multi-class segmentation with relative location prior. Int. J. Comput. Vis. **80**, 300–316 (2008)
33. J. Tighe, S. Lazebnik, Superparsing: scalable nonparametric image parsing with superpixels, in *European Conference on Computer Vision (ECCV)*, 2010, pp. 352–365
34. D. Munoz, J.A. Bagnell, M. Hebert, Stacked hierarchical labeling, in *European Conference on Computer Vision (ECCV)*, 2010, pp. 57–70
35. R. Socher, C.C. Lin, C. Manning, A.Y. Ng, Parsing natural scenes and natural language with recursive neural networks, in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 129–136
36. B. Micusik, J. Kosecka, Semantic segmentation of street scenes by superpixel co-occurrence and 3D geometry, in *IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, 2009, pp. 625–632
37. L. Zhang, B. Verma, D. Stockwell, S. Chowdhury, Spatially constrained location prior for scene parsing, in *International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 1480–1486
38. Y. Jimei, B. Price, S. Cohen, Y. Ming-Hsuan, Context driven scene parsing with attention to rare classes, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 3294–3301
39. A. Singhal, L. Jiebo, Z. Weiyu, Probabilistic spatial context models for scene content understanding, in *IEEE Conference on Computer Vision and Pattern Recognition, (CVPR)*, 2003, pp. 235–241

40. D. Batra, R. Sukthankar, C. Tsuhan, Learning class-specific affinities for image labelling, in *IEEE Conference on Computer Vision and Pattern Recognition, (CVPR)*, 2008, pp. 1–8
41. Z. Lei, J. Qiang, Image segmentation with a unified graphical model. IEEE Trans. Pattern Anal. Mach. Intell. **32**, 1406–1425 (2010)
42. R. Xiaofeng, B. Liefeng, D. Fox, RGB-(D) scene labeling: features and algorithms, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 2759–2766
43. V. Lempitsky, A. Vedaldi, A. Zisserman, Pylon model for semantic segmentation, in *Advances in Neural Information Processing Systems*, 2011, pp. 1485–1493
44. F. Tung, J.J. Little, Scene parsing by nonparametric label transfer of content-adaptive windows. Comput. Vis. Image Underst. **143**, 191–200 (2016)
45. L. Zheng, Y. Zhao, S. Wang, J. Wang, Q. Tian, Good practice in CNN feature transfer. *arXiv preprint* arXiv:1604.00133 (2016)
46. S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su et al., Conditional random fields as recurrent neural networks. *arXiv preprint* arXiv:1502.03240 (2015)
47. P.H. Pinheiro, R. Collobert, Recurrent convolutional neural networks for scene parsing. *arXiv preprint* arXiv:1306.2795 (2013)
48. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition. *arXiv preprint* arXiv:1512.03385 (2015)

http://www.springer.com/978-981-10-4538-7

Roadside Video Data Analysis
Deep Learning
Verma, B.; Zhang, L.; Stockwell, D.
2017, XXV, 189 p. 79 illus., 68 illus. in color., Hardcover
ISBN: 978-981-10-4538-7