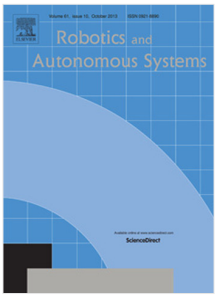
Unsupervised obstacle detection in driving environments using deep-learning-based stereovision

Accepted Manuscript



Unsupervised obstacle detection in driving environments using deep-learning-based stereovision

Abdelkader Dairi, Fouzi Harrou, Mohamed Senouci, Ying Sun

PII: S0921-8890(17)30473-6
DOI: https://doi.org/10.1016/j.robot.2017.11.014
Reference: ROBOT 2957

To appear in: *Robotics and Autonomous Systems*

Received date : 11 July 2017
Revised date : 13 October 2017
Accepted date : 26 November 2017

Please cite this article as: A. Dairi, F. Harrou, M. Senouci, Y. Sun, Unsupervised obstacle detection in driving environments using deep-learning-based stereovision, *Robotics and Autonomous Systems* (2017), https://doi.org/10.1016/j.robot.2017.11.014

Unsupervised obstacle detection in driving environments using deep-learning-based stereovision

Abdelkader Dairi[a], Fouzi Harrou[b], Mohamed Senouci[a], Ying Sun[b]

[a] *Computer Science Department, University of Oran 1 Ahmed Ben Bella , Algeria Street El senia el mnouer bp 31000 Oran, Algeria. E-mail: dairi.aek@gmail.com*
[b] *King Abdullah University of Science and Technology (KAUST) Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division, Thuwal 23955-6900, Saudi Arabia, E-mail: fouzi.harrou@kaust.edu.sa*

Abstract

A vision-based obstacle detection system is a key enabler for the development of autonomous robots and vehicles and intelligent transportation systems. This paper addresses the problem of urban scene monitoring and tracking of obstacles based on unsupervised, deep-learning approaches. Here, we design an innovative hybrid encoder that integrates deep Boltzmann machines (DBM) and auto-encoders (AE). This hybrid auto-encode (HAE) model combines the greedy learning features of DBM with the dimensionality reduction capacity of AE to accurately and reliably detect the presence of obstacles. We combine the proposed hybrid model with the one-class support vector machines (OCSVM) to visually monitor an urban scene. We also propose an efficient approach to estimating obstacles location and track their positions via scene densities. Specifically, we address obstacle detection as an anomaly detection problem. If an obstacle is detected by the OCSVM algorithm, then localization and tracking algorithm is executed. We validated the effectiveness of our approach by using experimental data from two publicly available dataset, the Malaga stereovision urban dataset (MSVUD) and the Daimler urban segmentation dataset (DUSD). Results show the capacity of the proposed approach to reliably detect obstacles.

Keywords: Deep learning, DBM, Autoencoder, OCSVM, Monitoring, Stereovision

1. Introduction

1.1. Background

Over the past two decades, intelligent transport systems, driver assistance systems and autonomous vehicles have received increasing research attention (Labayrade et al., 2002; Fakhfakh et al., 2013; Sun et al., 2013; Appiah and Bandaru, 2015; Nalpantidis et al., 2016; Zhang et al., 2017). Localization and obstacle detection systems are key enablers in the developmentof practical autonomous robots and vehicles and for intelligent transportation systems so that accidents can be avoid. Indeed, the main objective of a detection and localization of obstacles system is to improve safety and comfort, while reducing the risk of collisions by alerting the driver or providing useful information for rapid decision making. Moreover,

obstacle detection is useful in other applications, such as smart wheelchairs, unmanned aerial vehicles and agricultural applications (Woo and Kim, 2016; Del et al., 2006; Fleischmann and Berns, 2016).

To guarantee reliable obstacle detection, researchers and engineers have developed autonomous vehicles and robots that are fully equipped with sophisticated sensors, such as ultrasound sensors, RADAR and LIDAR systems, 3D and 360-degree cameras (Appiah and Bandaru, 2015; Asvadi et al., 2016). However, these sensors are costly, and require continuous maintenance and complex synchronization in the fusion of different sources of data. To remedy these limitations, low-cost, vision-based obstacle detection and localization systems have been developed (Labayrade et al., 2002; Fakhfakh et al., 2013; Yoo et al., 2016). Such systems are mainly based on multiple collection of views using visual sensors that can estimate depth and perceive three-dimensional (3D) components in a scene. For example, binocular stereovision is based on two rectified images (left and right) that are used to compute a disparity map (i.e., displacement of an object between two rectified images) such that the epipolar geometry constraints are fulfilled (Labayrade et al., 2002; Fakhfakh et al., 2013; Hu and Uchimura, 2005a; Nalpantidis et al., 2016).

In the literature, there has been much discussion on obstacle detection techniques. For instance, some approaches are based on images descriptors such as scale invariant feature transform (SIFT), local binary pattern (LBP), regions of interest (ROI) based on sliding windows, and histograms of oriented gradient (HOG) (Dalal and Triggs, 2005). Indeed, these techniques usually utilize manually designated features, such as vehicle motion, color and texture. Nadav and Katz (2016); Broggi et al. (2005); K Yamaguchi (2006) proposed obstacle detection using a monocular camera in the off-road environment. Häne et al. (2015) proposed an obstacle detection approach in the on-road environment using monocular cameras. Labayrade et al. (2002); Fakhfakh et al. (2013); Hu and Uchimura (2005a) proposed a binocular stereo vision system based on depth estimation via disparity maps for highways. Sun et al. (2013) proposed a system for detection and tracking of moving obstacles in urban driving scenarios. Appiah and Bandaru (2015) proposed an approach using stacked stereo 360 vertical cameras to perceive obstacles around an autonomous vehicle. Nalpantidis et al. (2016) introduced a new representation of 3D scene structure named theta-disparity. The key idea of theta-disparity is to get a radial representation of the significant objects in a set with respect to a point of interest based on a disparity map (Appiah and Bandaru, 2015). Woo and Kim (2016) proposed vision-based obstacle detection and collision risk estimation of an unmanned surface vehicle. Based on the work of Labayrade et al. (2002); Fakhfakh et al. (2013); Nalpantidis et al. (2016), Burlacu et al. (2016) presented an obstacle detection approach in stereo sequences using multiple representations of the disparity map. However, this approach is based on heavy scanning of images to look for obstacles without any certainty about the existence and kind of obstacles. This method requires intensive computation and is difficult to adapt in real-time applications. In addition, this method cannot distinguish obstacles from other objects.

In obstacle detection and localization, machine learning turn out to play an important role (Petković et al., 2016; Duguleana et al., 2012; Bengio et al., 2007). Many methods have been developed for improving obstacle detection and for handling new applications (Dollar et al., 2012; Dalal and Triggs, 2005; Bengio et al., 2009; Hinton, 2007; Bengio et al., 2007). In learning-based obstacle detection methods, two classes can be distinguished: approaches based on shallow learning approaches and those based on deep learning approaches. Various shallow learning-based approaches have been investigated, such as training different classifiers by support vector machines (SVM), AdaBoost, and neural networks in supervised learning with one or two layers (Dollar et al., 2012). Robust approaches have been proposed by merging HOG with SVM for human detection based on single views (Dalal and Triggs, 2005). However, shallow learning approaches are not suitable for representing dependencies between multiple variables, and they are inefficient in dealing with problems with high-dimensionality data, leading to unsuitable generalized models (Bengio et al., 2009, 2007).

On the other hand, deep learning-based approaches have been developed to overcome these limitations. Indeed, deep convolutional neural networks are powerful tools in image classification. They have proved to be efficient for Google's ImageNet, which contains more then 1.3 million high-resolution images. Deep convolutional neural networks (CNNs) were first proposed by Nguyen et al. (2016) for obstacle detection and recognition, but their efficiency was limited to 2D images. Ramos et al. (2016) proposed an approach based on deep CNNs to detect unexpected obstacles. Despite the promising results obtained using the deep CNN approach for obstacle detection and recognition based on 2D images, some tasks, such as learning more about data distribution, encoding data, reducing dimensionality, generating new data with a given joint distribution, and unsupervised learning are not possible (Hinton, 2007). Restricted Boltzmann machines (RBM) and autoencoders are powerful deep architectures that overcome most of these limitations (Bengio et al., 2009). These deep-learning based approaches are usually implemented in three main steps. First, a heavy scanning of images. The next step is to locate the surrounding ROI. The last step is to start a recognition process. This complex process is automatically executed in both the presence and absence of obstacles, which is the main drawback of such an approach.

1.2. Motivation and contribution

To improve obstacle detection and classification, we start by checking the presence of obstacles before starting any heavy scanning of input images. In other words, our objective is to optimize the obstacle detection process by answering the question, *are there any obstacles?* Then, the localization, estimation and recognition processes can be executed only if a potential obstacle exists.

Here, we treat the problem of obstacle detection as an anomaly detection problem based on the V-Disparity data distribution. In urban settings or on highways a V-Disparity data distribution, which is the

vertical coordinate in the (u, v) disparity map coordinate system (Labayrade and Aubert, 2003; Hu and Uchimura, 2005b), is mostly stable with small variations due to measurement noise. The V-Disparity can significantly change in the presence of obstacles. Our proposed system has four main stages as shown in Figure 1.

- First, the system employs an innovative hybrid framework for feature extraction and encoding. This is based on a hybrid encoder model that combines multiple layers of deep Boltzmann machine (DBM) as the feature extractor and autoencoder (AE) for dimensionality reduction (V-Disparity $\Rightarrow$ Code). In fact, we start with unsupervised greedy layer-wise training of the hybrid encoder using the V-Disparity dataset. Two tasks are accomplished at the end of each layer: 1) discover and extract new features; 2) generate a new encoded output that will be used as input for the next layer. This proposed hybrid encoder architecture is built on four layers of DBM and AE.

- Second,we address obstacle detection as an anomaly detection problem based on the one-class support vector machine (OCSVM) classifier, which requires only obstacle-free data in training. The training of OCSVM is unsupervised from data encoded by the hybrid encoder model. The central role of the OCSVM classifier is to separate inliers from outliers in the testing data by building a hyper-plan (Erfani et al., 2016). Third, the presence of obstacles can predicted. Towards this end, for a given V-Disparity, a code is generated using the hybrid encoder model and the OCSVM classifier predict if it is an inlier or an outlier. Here, two models are built, the first model identifies free scenes and the second model to identifies busy scenes. The main reason to use two models is to improve decision making and reduce false alarms.

- Finally, the location of obstacles can be estimated based on density maps computed for both V-Disparity and U-Disparity by checking changes in residuals, which represent the difference between the current values of the density maps and the previous values. Here, the three-sigma rule is used to detect changes in residuals.

The effectiveness of the developed hybrid approach is validated using experimental data from two publicly available datasets, the Malaga stereovision urban dataset and the Daimler urban segmentation dataset. Results show that the proposed approach is able to reliably detect obstacles.

The remainder of this paper is organized as follows. Section 2 gives a brief overview of machine learning generative models and the OCSVM algorithm. In Section 3, stereovision is briefly presented. In Section 4, we present the proposed hybrid deep-learning-based obstacle detection approach. In Section 5, we assess the performance of the developed approach using publically available experimental data. Finally, Section 6 concludes with a discussion and suggestions for future research directions.

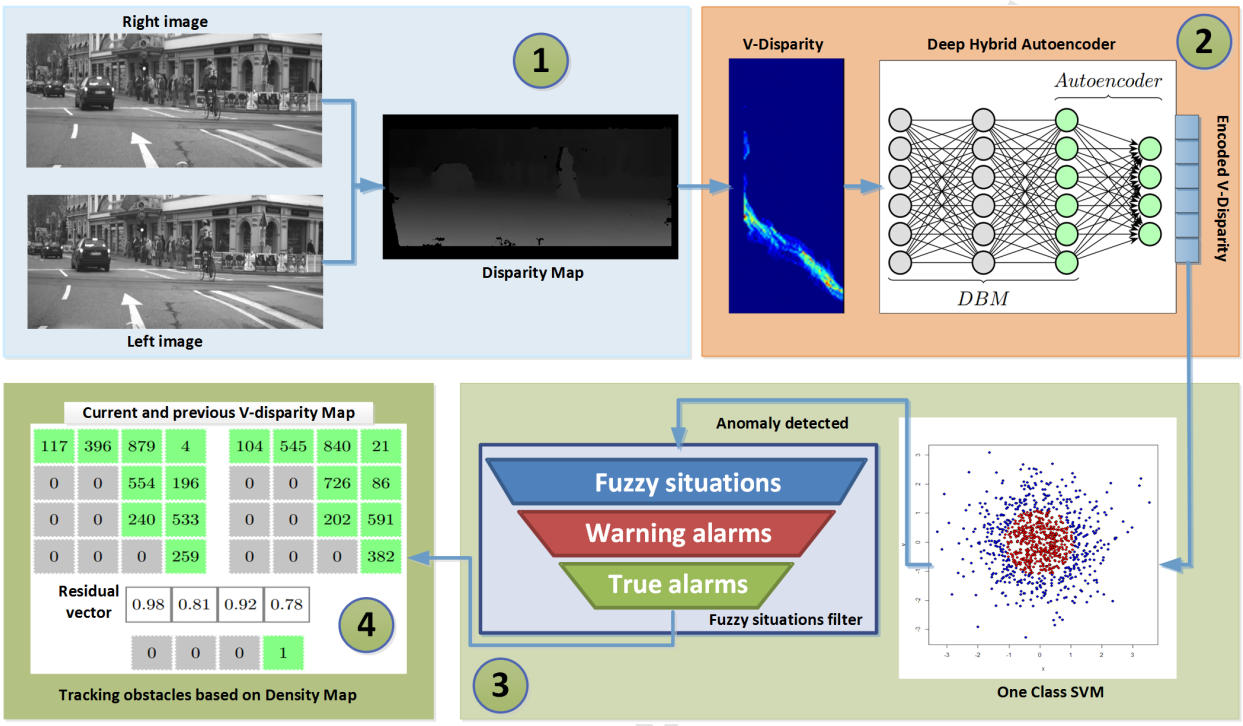


Figure 1: Flowchart of the proposed vision-based obstacle detection and localization system.

2. Preliminary materials

In this section, we briefly present an overview of machine learning generative models used to build deep learning architectures, such as deep autoencoders, Boltzmann machine and restricted Boltzmann machine. More details about these generative models can be found in (Hinton, 2007; Xu et al., 2015).

2.1. Autoencoders

An autoencoder is an artificial neural network (Bengio et al., 2009) used for unsupervised learning that is trained to reconstruct its own inputs (i.e., predicting the value of output $\hat{x}$ given input x via hidden layer h, see Figure 2). Autoencoders are widely used in dimensionality reduction and feature learning. Autoencoders comprise two parts: the encoder and the decoder. The encoder can be defined with encoder function $h = Encoder(x)$, which can be defined by a linear or nonlinear function. If the encoder function is nonlinear, the autoencoder will have capacity to learn more features than linear principal component analysis (Bengio et al., 2009). The purpose of the decoder part is to reconstruct its own inputs via the decoder function, $\hat{x} = Decoder(h)$. The learning process of an autoencoder is achieved by minimization of the negative log-likelihood (loss function) of the reconstruction, given the encoding $Encoder(x)$ (Bengio et al., 2009):

$$Reconstruction_{error} = -log(P(x|Encoder(x)), \quad (1)$$

where P is the probability assigned to the input vector x by the model. Indeed, incorporating latent variable models has caused autoencoders to behave like generative models. Stacked autoencoder models have been widely applied in image denoising (Vincent et al., 2008, 2010) and content-based image retrieval (Krizhevsky and Hinton, 2011).

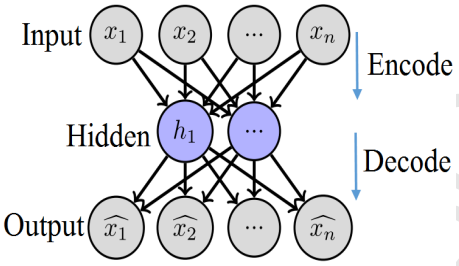


Figure 2: Autoencoders

2.2. Restricted Boltzmann Machine

Restricted Boltzmann machines (RBMs) can be viewed as stochastic neural networks (Smolensky, 1986) (see Figure 3). RBMs consist of m visible units, $v \in \{0,1\}^m$ and n hidden units, $h \in \{0,1\}^n$. There are no visible-to-visible and hidden-to-hidden connections, although v and h are fully connected (see Figure 3). The learning procedure comprises many steps of Gibbs sampling (propagate: sample hidden given visibles; reconstruct: sample visible given hidden; repeat) and selecting the weights with minimum reconstruction error. Different learning algorithms for RBMs have been proposed mostly based on Markov chain Monte Carlo (MCMC) sampling using Gibbs sampling to obtain an estimator of the log-likelihood gradient (Bengio et al., 2009; Hinton et al., 2006). Moreover, RBMs are used to construct deeper models, such as Deep Belief Networks (DBN) and the hierarchical probabilistic model deep Boltzmann machine (DBM) (Salakhutdinov and Hinton, 2009).

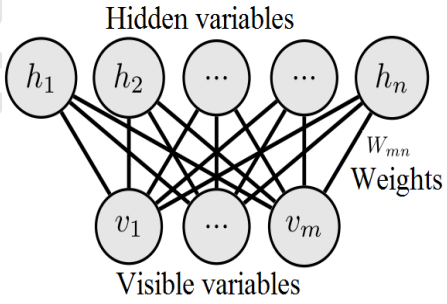


Figure 3: Schematic presentation of a Restricted Boltzmann Machine.

RBMs are particularly energy-based models and have been used as generative models for several types of data (Bengio et al., 2009) such as text, speech and images. The energy function of the RBM configuration

is defined as (Mohamed et al., 2012):

$$Energy(v,h) = -\sum_{i=1}^{m}\sum_{j=1}^{n} W_{ij} v_i h_j - \sum_{i=1}^{m} b_i v_i - \sum_{j=1}^{n} c_j h_j, \tag{2}$$

where W_{ij} is the weight matrix between visible variable v_i and hidden variable h_j and b and c are model parameters. The joint distribution of the configuration is given as:

$$P(v,h) = \frac{1}{Z}\exp(-Energy(v,h)) = \frac{1}{Z}\prod_{ij} e^{W_{ij} v_i h_j} \prod_{i} e^{b_i v_i} \prod_{j} e^{a_j h_j}, \tag{3}$$

where

$$Z = \sum_{v}\sum_{h} \exp(-Energy(v,h)) \tag{4}$$

is the partition function. Since only v is observed, the hidden variables h are marginalized.

$$P(v) = \sum_{h} \frac{e^{-Energy(v,h)}}{Z}, \tag{5}$$

where $P(v)$ is the probability assigned by the mode to a given visible vector v. In terms of probability since the hidden nodes are conditionally independent from the visible units, we can derive from equation 3:

$$P(v|h) = \prod_{i} p(v_i|h), \tag{6}$$

$$P(h|v) = \prod_{j} p(h_j|v). \tag{7}$$

For binary visible unit $v \in \{0,1\}^m$ and hidden units $h \in \{0,1\}^n$, the marginal probability of the RBM is expressed by:

$$P(v_i = 1|h) = \sigma(\sum_{j} W_{ij} h_j + c_i), \tag{8}$$

$$P(h_j = 1|v) = \sigma(\sum_{i} W_{ij} v_i + b_j), \tag{9}$$

where $\sigma(.)$ is the logistic function and $\sigma(x) = (1+\exp(-x))^{-1}$. Hinton et al. (2006) developed an extension of RBMs, Gaussian Bernoulli RBMs, to deal with different data types like real-valued vectors (e.g., pixel intensities of an image), in which $v \in R^m$ and hidden units $h \in \{0,1\}^n$. For the Gaussian Bernoulli RBMs, the joint energy is:

$$Energy(v,h) = \sum_{i=1}^{I} \frac{(v_i, c_i)^2}{2\sigma_i^2} - \sum_{i=1}^{I}\sum_{j=1}^{J} W_{ij} h_j - \frac{v_i}{\sigma_i}\sum_{j=1}^{J} b_j h_j. \tag{10}$$

The aim of training RBMs is to adjust the model's parameters (weights matrix w) (see equation 11). This task is achieved by maximizing the probability of the training data under the model. In other words, it is doneby maximizing the log-likelihood of the parameters given the training data, where the derivative

of the log-likelihood with respect to W takes the following form (Hinton et al., 2006):

$$\Delta w_{ij} = \alpha(E(v_i, h_j) - \hat{E}(v_i, h_j)), \tag{11}$$

where α is the learning rate and $\hat{E}(v_i, h_j)$ is the energy expected from the distribution learned by the model. which is intractable (Hinton et al., 2006) Gibbs Sampling is used instead. RBMs have been successfully applied in various applications such as in blocks of deep learning architectures, classification, feature extraction and dimensionality reduction.

2.3. Deep Belief Networks

Deep belief networks (DBNs) are probabilistic generative models that are based on stacked RBMs (see Figure 4). DBNs have been used in many challenging learning problems, such as in real-time classification (O'Connor et al., 2013), audio classification (Lee et al., 2009), speech synthesis (Kang et al., 2013), and facial expression recognition (Liu et al., 2014). They exhibited high efficiency in discovering layer-by-layer complex nonlinearity. Furthermore, DBNs have been used successfully in dimensionality reduction (Hinton et al., 2006; Salakhutdinov and Hinton, 2007). Hinton et al. (2006) introduced a fast unsupervised learning algorithm for DBN in which the joint distribution between observed vector x and ℓ hidden layers h^k is expressed as follows:

$$P(x, h^1, \ldots, h^l) = (\prod_{k=0}^{\ell-2} P(h^k|h^{k+1}))P(h^{\ell-1}, h^\ell), \tag{12}$$

where $x = h^0$ and $P(h^k|h^{k+1})$ is a visible given hidden conditional distribution in an RBM associated with level k of the DBN, and $P(h^{\ell-1}, h^\ell)$ is the joint distribution in the top-level RBM.

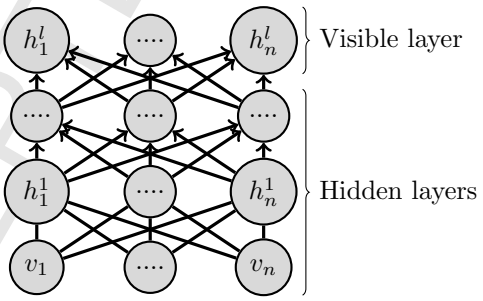


Figure 4: The structure of Deep Belief Networks.

Indeed, adding more layers of the DBN allows an increase in the probability of the training data. Specifically, accuracy of the energy expression is improved by adding more layers in the network. The training time will be reduced because only one step is required to learn the maximum likelihood.

2.4. Deep Boltzmann Machines

Salakhutdinov and Hinton (2009) proposed a new learning algorithm for a hierarchical probabilistic model called deep Boltzmann machine (DBM). DBM is a generative model with many layers of hidden variables in which connections between layers are undirected (see Figure 5). Whereas RBMs are a kind of Markov random field, DBMs learn increasingly from complex representations of given data and incorporate uncertainty about ambiguous and missing or noisy inputs. DBMs are able to extract complex statistical structures and are applicable to various applications, such as object recognition (Leng et al., 2015), and computer vision (Gan et al., 2015). Salakhutdinov and Larochelle (2010) optimized all layers of DBM parameters jointly by following the approximate gradient of a variational lower-bound on the likelihood function. Salakhutdinov and Hinton (2009) proposed greedy, layer-by-layer pre-training by learning a stack of RBMs with a small change to initialize the model parameters of a DBM. The DBM energy function of the state {v, h} is defined as:

$$E(v, h^1, h^2; \theta) = -v^T W^1 h^1 - h^1 W^2 h^2, \tag{13}$$

where $\theta = \{W^1, W^2\}$ are the model parameters, the vector of visible units $v \in \{0,1\}^D$ and the vectors of hidden units $h^1, h^2 \in \{0,1\}^P$. The probability that the model assigns to a visible vector v is:

$$p(v;\theta) = \frac{1}{Z(\theta)} \sum_{h^1, h^2} exp(-E(v, h^1, h^2; \theta)). \tag{14}$$

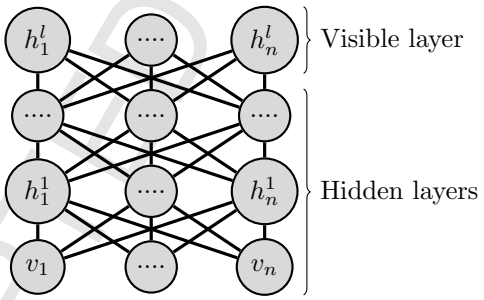


Figure 5: A Deep Boltzmann Machine

2.5. The one-class support vector machine (OCSVM):

The one-class support vector machine (OCSVM) (Schölkopf et al., 2001) is an efficient, unsupervised learning algorithm that learns decision functions for anomaly detection. OCSVM returns a function $f(x)$ with **+1** or **-1** to indicate whether the data is an "inlier" or "outlier" respectively. Its decision function $f(x)$ is defined as:

$$f(x) = \begin{cases} +1, & \text{if region capturing most of the data points} \\ -1, & \text{otherwise.} \end{cases} \tag{15}$$

OCSVM, which is based on kernels (see equation 16) such as the radial basis function (RBF) (see equation 17), maps input data into a high-dimensional feature space $\mathcal{F}$, the hyperplane that maximizes the margin that best separates the training data from the origin.

$$\mathcal{K}(x, y) = (\Psi(x) \ . \ \Psi(y)), \tag{16}$$

where x and y are the input vectors, Ψ is a feature map $\mathcal{X} \rightarrow \mathcal{F}$ and $\mathcal{X}$ is set of observed x. The RBF kernel is also known as a Gaussian kernel:

$$\mathcal{K}_{RBF}(x, y) = \exp(-\frac{\|x - y\|^2}{2\sigma^2}). \tag{17}$$

The selection of the hyperplane separating the training dataset from the origin is achieved by solving the following quadratic optimization problem:

$$\min_{w \in \mathcal{F}, \xi \in R^l, \rho \in R} \ \frac{1}{2}\|w\|^2 \frac{1}{\nu l} \sum_i^l \xi_i - \rho, \tag{18}$$

$$\text{subject to } (w \ . \ \Psi(x)) \geq \rho - \xi_i \ , \ \xi_i \geq 0$$

where $\nu \in [0, 1]$ is a parameter that characterizes the solution, w is a weight vector and ρ is an offset.

The decision function $f(x)$ can be estimated by equation 19 since nonzero slack variables ξ_i are penalized in the objective function:

$$f(x) = sgn((w \ . \ \Psi(x)) - \rho). \tag{19}$$

An hyperplane is constructed based on two parameters w and ρ, the distance of all the data points in $\mathcal{F}$ from the hyperplane to the origin.

Tax and Duin (2004) introduced an other one-class classifier called support vector data description (SVDD). In the SVDD algorithm, boundaries used to detect novel data as inliers or outliers are spherically shaped to contain the training samples. However, the spherical boundary of SVDD suffers from the non-spherical shapes of some training datasets, resulting in empty space in the hyper sphere. In this paper, we use the SVDD algorithm as a benchmark for obstacle detection using our hybrid deep learning approach.

3. Stereovision

Stereovision is the process of extracting 3-D information from multiple 2-D views of a scene. Stereovision techniques usually depend on epipolar geometry to perform spatial perception and depth estimation based on a disparity map of two rectified images (left and right) (Labayrade et al., 2002; Fakhfakh et al., 2013). Disparity maps indicate the difference (*"disparity"*) in position of an object in two corresponding rectified images. The disparity becomes smaller as the distance between the object and camera decreases, and vice versa. Several algorithms have been proposed to compute disparity maps (Labayrade et al., 2002; Fakhfakh

et al., 2013; Georgoulas et al., 2008), $\mathcal{D}$, using different matching correlation measures, such as the sum of absolute differences (SAD) (see Equation 20).

$$\mathcal{D}_{SAD}(i,j,d) = \sum_{u=-\omega}^{\omega} \sum_{u=-\omega}^{\omega} |I_{left}(i+u, j+v) - I_{right}(i+u, j-d+v)| \tag{20}$$

where I_{left} and I_{right} respectively denote the left and right image pixel intensities, d is the disparity range $[d_{min}, d_{max}]$, d_{min} and d_{min} are respectively the minimum and maximum disparity values, ω is the window size and i, j are the coordinates (rows, columns respectively) of the center pixel of the SAD or any correlation measures.

V-disparity map, which gives a good estimation of a road's profile based on the Hough transform and depth estimation, provides information about the height of obstacles and their positions with respect to the ground (Labayrade et al., 2002; Fakhfakh et al., 2013). The main steps used to compute the V-disparity are given in Algorithm 1.

Algorithm 1: V-disparity computation steps

Input: Disparity map DispMap(rows, cols)

Input: D_{max}: Max disparity value.

Output: V-disparity $DispMap_v$ (rows, D_{max})

```
for Each row r^th in DispMap do
    for Each column c^th in DispMap do
        currentDisparity ← DispMap(r, c)
        if currentDisparity > 0 then
            DispMap_v(r, c) ← (currentDisparity + 1)
```

On the other hand, a U-disparity map provides information about the width of obstacles and depth estimation (Labayrade et al., 2002; Fakhfakh et al., 2013; Hu and Uchimura, 2005a). Algorithm 2 describe the main steps to compute U-disparity.

A density map is a compact representation of V-disparity without losing of essential information. To compute the density map, the V-disparity is segmented into many small cells (see Figure 6), and the density map for each cell is derived as follows:

$$Density_{Cell} = (\sum^{Cell} I(i,j))/(w * h),$$

where $I(i,j)$ is the intensity of the pixel in row i and column j, w and h are respectively the width and height of the cell.

Algorithm 2: U-disparity computation steps.

Input: Disparity map DispMap(rows, cols)

Input: D_{max}: Max Disparity value.

Output: UDisparity $DispMap_u$ (D_{max}, cols)

```
for Each row r^th in DispMap do
    for Each column c^th in DispMap do
        currentDisparity ← DispMap(r, c)
        if currentDisparity > 0 then
            DispMap_u(r, c) ← (currentDisparity + 1)
```

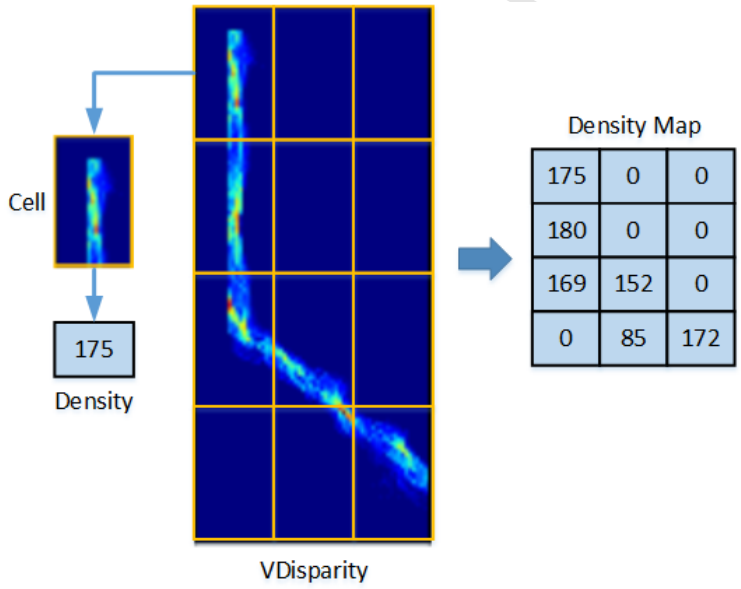


Figure 6: Example of density map.

The region of interest (ROI) of obstacles can be determined using both U-disparity and V-disparity maps. Figure 7 illustrates how V-disparity and U-disparity maps are used to find a region of interest containing potential obstacles. Indeed, each detected obstacle belongs to an interval of disparity, which allows the estimation of its distance from the vehicle.

Indeed, V-disparity and U-disparity are respectively computed based on the numbers of pixels with the same disparity level at rows and columns wise in the disparity map. After the V-disparity and U-disparity maps are computed, the road profile is extracted from V-disparity using the Hough transform. Figure 8(a-b) shows an example of V-disparity with free-scene (i.e., the absence of obstacles) and in the presence of an obstacle, respectively. The road surface determines an inclined straight line in V-disparity

Figure 7: Example of using V-disparity and U-disparity to locate obstacles.

space (see Figure 8(a)). Figure 8(a) shows that the V-disparity concept simplifies the process of separating obstacles in an image. The vertical cloud of points on the lower disparity represents a static environment (see Figure 8(a)), its thickness depends on its texture richness (e.g., buildings, and trees). Obstacles on a road will are presented by vertical lines with high intensities (see Figure 8(a). If the obstacle is closer to the right side of the V-disparity map, the distance between the obstacle and the vehicle is smaller. The thickness of the detected obstacle decrease when the obstacle moves away further from the mobile robot. The vertical length of the vertical line represents the height, h, of the actual obstacle in the image. The greater the thickness of the obstacle in the V-disparity map, the bigger is the obstacle in the image (e.g., bus, cars, and pedestrians). Figure 8(b) shows pedestrians walking on the road. From the V-disparity, it can be seen that vertical lines to the road profile indicate the presence of these obstacles (i.e., pedestrians). In U-disparity, obstacles appear as a fragment of horizontal lines (see Figure 8(b)). The length of a fragment is the width of the detected obstacle, and the starting x-coordinate of each fragment represents the x-coordinate of the obstacle. By using V-disparity and U-disparity, the width, high, x and y coordinates of the detected obstacle can be extracted. The algorithm 3 describes the steps to surround obstacle on ROI.

4. Proposed hybrid deep autoencoder-based obstacle detection approach

The proposed hybrid deep autoencoder (HAE) consists of four layers. Each layer is the combination of a DBM and an autoencoder. In each layer, useful features are extracted and encoded in an output code.

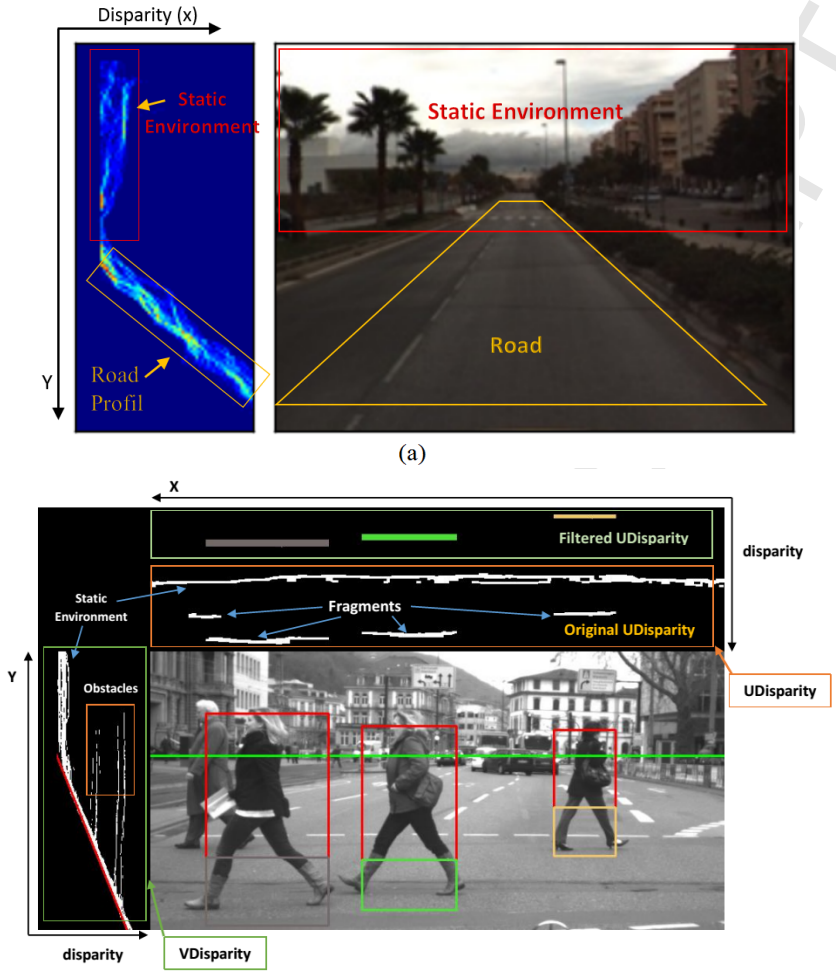


Figure 8: Example of V-disparity in the situation of free-scene (a). Example of V-disparity and U-disparity in the presence of obstacles (b).

Then, the generated code is used for the next layer. The output of the last layer will be used as the input to the one-class classifier. Specifically, the one-class classifier builds boundaries to separate normal (without obstacles) and abnormal (presence of obstacles) cases. In this approach, two models are constructed to enhance accuracy and reduce false alarms. The first is built with unsupervised learning of images with obstacles and the second is built with the unsupervised learning based on images without obstacles. False alarms can be reduced by comparing the outputs of two models. Figure 9 schematically summarizes the proposed system that is based on a deep learning architecture trained entirely in an unsupervised way. The main steps of the proposed approach are summarized in Algorithm 4.

Algorithm 3: Obstacle localization steps.

Input: Disparity map: $DMap$

Output: Vector of Region of Interest: $\mathcal{R}_{ROI}$

$\mathcal{V} \leftarrow Build_{VDisparity}(DMap)$;
$\mathcal{U} \leftarrow Build_{UDisparity}(DMap)$;
$\mathcal{D}$: is the disparity range of the obstacle;
(x, y): coordinate of the obstacle in the original image;
(h, w): height and width of the obstacle;
Extract Road Profile RP from $\mathcal{V}$;
$OBS \leftarrow FindStandingObstacle(RP)$;
for *Each obstacle* $\mathcal{O}$ *in* OBS **do**
    ➥ Determine $\mathcal{D}$ and y from $\mathcal{V}$;
    ➥ Determine h obstacle height located in $\mathcal{V}$;
    ➥ Determine w and x using $\mathcal{D}$ from $\mathcal{U}$;
    ➥ Append (x, y, h, w) to $\mathcal{R}_{ROI}$;
return $\mathcal{R}_{ROI}$

Algorithm 4: Hybrid deep encoder approach.

Input: images DataSet of (Left, right): TrainingDataset

Output: DataSet of Encoded V-disparity: EncodedDataset

for *Each tuple (Left, Right) in training dataset* **do**
    $DisparityMap \leftarrow ComputeDisparityMap(Left, Right)$
    $V\text{-}Disparity \leftarrow ComputeVDisparity(DisparityMap)$
    $X \leftarrow V\text{-}Disparity$
    for *Each layer* λ *in HAE layers* **do**
        $output_{DBM} \leftarrow LearnFeatures_{DBM}(X)$
        $output_{\lambda} \leftarrow Encode_{AE}(output_{DBM})$
        $X \leftarrow output_{\lambda}$
    $EncodedDataset \Longleftarrow add(X)$
    /*Add X to EncodedDataset*/
$OCSVM_{Model} \leftarrow train(EncodedDataset)$

Definition 1 (Operating area). *Let us define an operating area as the region in front of a vehicle (see Figure 10). The dimensions of this region are expressed as a range of disparities, where δ is the disparity*

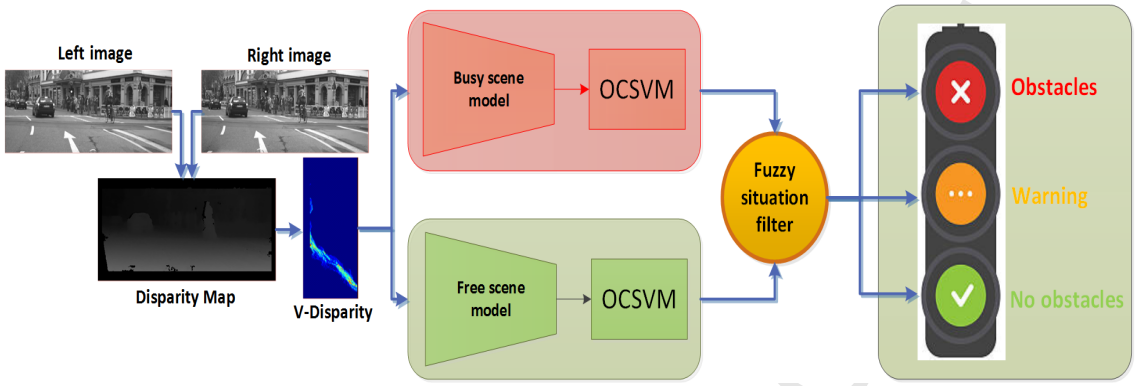


Figure 9: Block diagram of the deep encoders architecture with two OCSVM classifiers.

range, δ_{min} and δ_{max} are he minimum and maximum disparity values, respectively.

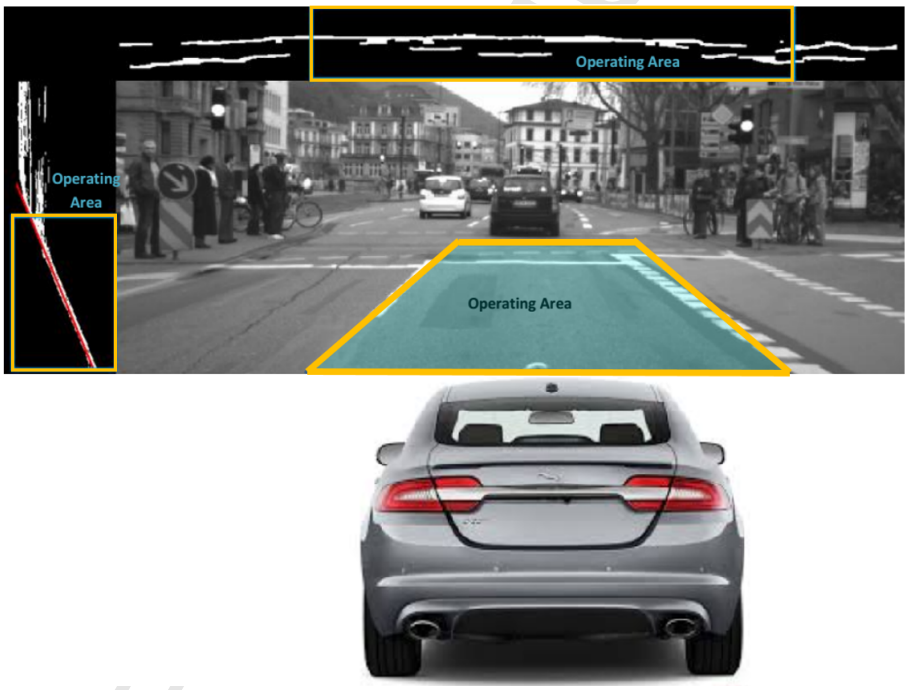


Figure 10: Vehicle operating area.

The proposed procedure is implemented in several steps as summarized in Table 1.

Step	Action
❶	**Stereoimage acquisition** from the stereovision device. ➥ Input: Left and right images. ➥ Output: Rectified left and right images.
❷	**Compute disparity map**: ➥ Input: Rectified left and right images. ➥ Output: Disparity map.
❸	**Compute V-disparity map** (see algorithm 1) ➥ Input: Disparity map. ➥ Output: V-disparity map.
❹	**Check existence of obstacles (Detection)**: Apply the hybrid deep encoder-based OCSVM for obstacle detection. ➥ Input: Encoded V-disparity map. ➥ Output: Prediction, $P \in \langle Yes, No \rangle$.
❺	**Compute scene density**: Compute Density map using V-disparity density ➥ Input: Encoded V-disparity map. ➥ Output: Density estimation.
❻	**Track obstacles localization (tracking)**: Based on the previous density map, predict the new obstacles localization by tracking density changes. ➥ Input: Density map. ➥ Output: Estimation of the obstacles localization.
❼	**Compute U-disparity map**: Compute U-disparity map based the boundaries of the vehicle operating area (see algorithm 2). ➥ Input: Disparity map. ➥ Output: Obstacles region of interest (ROI) (see Figure 12).

Table 1: Main steps of the proposed system.

4.1. Hybrid deep architecture training

In this section, we describe the approach used to train the proposed deep architecture, starting with building the hybrid deep encoder based on unsupervised training. Then, the one-class classifier is trained to learn how to classify the encoded data obtained from the hybrid deep encoder.

Deep Hybrid Encoder training:. The proposed system is based on two models, which are implemented in parallel (see Figure 9). Each model merges a deep DBM with an autoencoder to enhance the quality of the

generated encoded datasets (see Figure 11). These models are trained with an input dataset that contain rectified left and right images. Specifically, we train the first model with image sequences that contain mostly free scenes with a few obstacles. At the same time, we train the second model with data containing mostly scenes with obstacles. This hybrid deep encoder allows the system to learn a complex data distribution and encode the input images. It is also able to reconstruct the input with reduced errors.

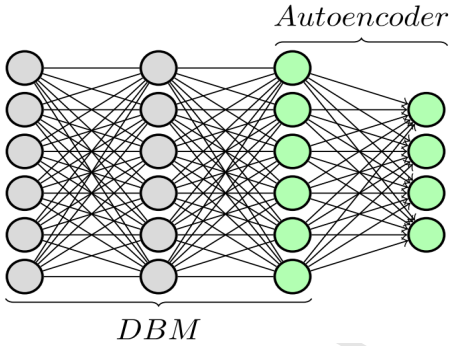


Figure 11: Deep Boltzmann Machines with Autoencoders.

Training the one-class classifier:. In the proposed approach, the OCSVM classifier, which is an unsupervised classifier, is trained with the encoded V-disparity map generated from the two constructed models of the hybrid deep encoder. As described above, we implement two OCSVMs, the first aims to detect outliers from the encoded V-disparity map of the model trained with obstacles; the second is used to detect outliers from the encoded V-disparity map of the model trained without obstacles (see Figure 9).

Obstacle localization and tracking. After detecting an obstacle using our HAE-OCSVM approach, it is important to locate its position. The proposed approach for obstacle localization and tracking is schematically presented in Figure 12. This approach is based on V-disparity and U-disparity maps, which are useful for obstacle localization. In fact, each row in the density map of the V-disparity map represents an area that potentially contains obstacles (following the Y-coordinate axis). The density map of the V-disparity is useful for detecting and tracking obstacles moving vertically. On other hand, columns of the density map obtained from the U-disparity represent an area that potentially contains obstacles (following the X-coordinate axis). Thus, the density map of the V-disparity can be used to detect and track obstacles moving horizontally. In this approach, the V-disparity and U-disparity maps are computed based on the disparity map of the two input images. Of course, the density map can be used as an indicator to determine the position of the detected obstacle. Towards this end, we analyze the trend of previous density map to track changes. Specifically, we applied the three-sigma rule (i.e., Shewhart monitoring chart) (Montgomery, 2009) on the density map column-wise to detect change.

The proposed approach is implemented in several steps: firstly, the V-disparity and U-disparity maps

are computed based on the disparity map of the two input images. Then, the road profile is extracted using the Hough transform, which helps to determine a line representing the road. Obstacles on the road are represented by vertical lines on the V-disparity map. Their height and depth can be estimated via distances in the V-Disparity. Their width can be determined based on processing the U-disparity map. Thus, we can surround the obstacles in the ROI. Specifically, by crossing the U-disparity and V-disparity maps, we can surround the obstacles and estimate their positions and distances.

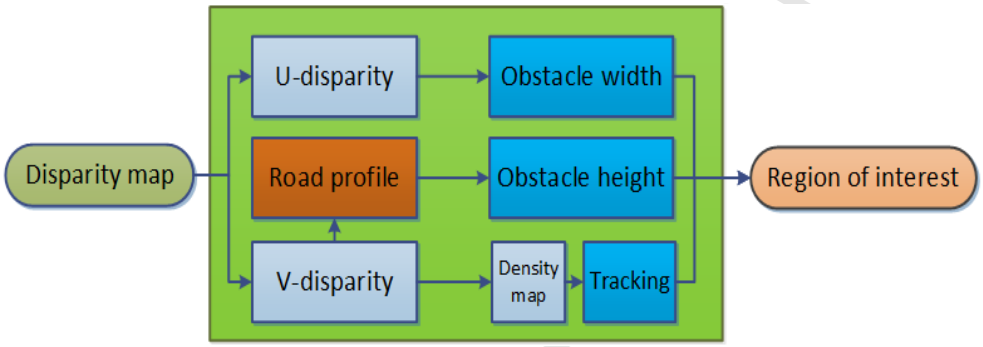


Figure 12: Block diagram of the obstacle localization process.

5. Experimental Results and Discussion

5.1. Data description

This section reports on the effectiveness of the proposed hydrid encoder approach. Towards this end, we performed experiments on two practical datasets: the Malaga stereovision urban dataset (MSVUD) (Blanco et al., 2014) and the Daimler urban segmentation dataset (DUSD) (Scharwächter et al., 2013, 2014). The MSVUD comprises 15 sub-datasets (extracts) of rich urban scenarios of more than 20 km in length with a resolution of 800×600 pixels recorded under different situations (with and without traffic), such as a straight path, turns, roundabouts, avenue traffic and highway. The DUSD contains images sequences recorded in urban traffic. It consists of rectified stereo image pairs with a resolution of 1024×440 pixels (Scharwächter et al., 2014).

Two sub-datasets of MSVUD are used in the training phase. The first dataset, which is extract number 5 (avenue loop closure 1.7 km), consists of 5000 pairs of images and the second dataset is extract number 8 (long loop closure, 4.5 km), which consists of 10,000 pairs of images. These two extracts (5,8) are composed mainly of free scenes. In the testing phase, we used two sub-datasets of MSVUD, extract number 10 (multiple loop closures) which consists of 9000 pairs of images, and extract number 12 (a long avenue of 3.7km with traffic), which consists of 11,000 pairs of images. In addition, the DUSD dataset is used for obstacle detection with 500 pairs of images.

To do so, we used two MSVUD datasets for testing purposes (Blanco et al., 2014). The first dataset termed FREE-DST contains 20% of fuzzy situations and 80% of free roads. The second dataset called BUSY-DST contains a 90% of fuzzy situations and 10% of true obstacles (vehicles, motorbikes and pedestrians). This distribution is motivated by the fact that in normal urban driving scenarios, the car is moving most of the time unless. The vehicle can be stuck in traffic. Both datasets, FREE-DST (3563 pairs of images) and BUSY-DST (1437 pairs of images), were generated randomly from extracts 10 and 12 of MSVUD.

In this study, the effectiveness of three obstacle detection approaches consisting of two layers: deep encoders and one-class encoders, is assessed and compared. Indeed, we used three different deep encoders: i) the proposed Hybrid Autoencode (HAE), (ii) Deep Belief Network (DBN), and (iii) Stacked Autoencoders (SDA). Also, we used two one-class classifiers OCSVM and SVDD. The experimental parameters of the machine learning approaches studied in this paper are presented in Table 2.

Table 2: Parameter settings of the studied approaches.

Models	Parameter	Value
DBM	learning rate	0.01
	Gibbs sampling (k)	15
	Training epochs	100
Autoencoder	Learning rate	0.01
	Training epochs	100
OCSVM	Kernel	RBF
RBF	γ	0.1
RBF	ν	0.1
Operating area	$\delta_{\min}$	32 (pixels)
	$\delta_{\max}$	64 (pixels)

5.2. Model trained with free scenes (FSM):

To build an efficient and accurate model able to predict free scenes and reject the scenes with obstacles, we trained the one-class classifier with V-disparities of free scenes. Sometimes there were confusing (fuzzy) situations in which obstacles were in the field of view of the vehicle but they were not in the operating area. This classifier is constructed to fit with the free scenes and fuzzy situations and to reject busy scenes. Examples of free scenes and their corresponding V-disparity map are shown in Figure 13. From the V-disparity shown in Figure 13 (a-d), it can be seen that the road profile is clearly apparent with visible inclined line of cloud points without accumulation of high intensities pixels. So, from Figure 13 (a-d), it seems that there is no obstacle in the road. It can also be seen that the static environment (vertical line) is

in the low V-disparity area, which means it is away from the vehicle.

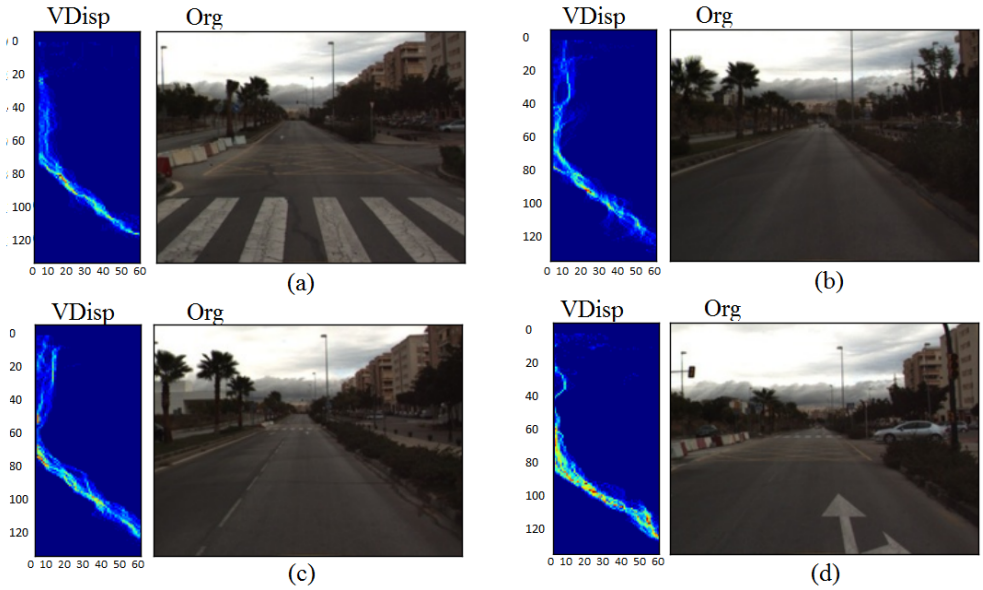


Figure 13: Examples of free scenes (Right) Original input image and (Left) its corresponding V-disparity map.

We evaluated the effect of the number of samples in the training dataset on the accuracy of the proposed hybrid model. To do so, we varied the number of samples in the training dataset from 500, 1000, 2000 to 5000 and evaluated the accuracy of the proposed HAE-OCSVM algorithm compared to both SDA and DBN-based OCSVM algorithms (see Table 3). In each experiment, we measure the inliers called true positives (TP); accepted by OCSVM and the outliers, called false positives (FP), rejected by OCSVM. Table 3 shows that when 500 samples were used for training, the accuracy in percentage % (TP, FP) of the proposed HAE-OCSVM method was 99.51 and 0.49 respectively, and that of DBN-OCSVM and SDA-OCSVM was 89.78 and 10.22 and 89.39 and 10.61, respectively. It can be seen that the accuracy of the proposed method increases with the number of samples in the training data.

With 5000 training samples, the HAE-OCSVM method, the DBN-OCSVM, and the SDA-OCSVM method respectively yielded 99.73 and 0.27, 89.89 and 10.11 and 90.57 and 9.43 percent accuracy. Results show that the proposed method outperformed DBN-OCSVM and SDA-OCSVM, and exhibited the highest accuracy. This is mainly due to its strong ability to learn complex structures from training data.

We also assessed, the performance of previously constructed models trained with free scenes using BUSY-DST. Figure 14 shows examples of busy situations. From Figure 14 (a,c and d), it can be seen that an area with visible pixel intensities is present in the road profile, and the static environment (vertical line) is located in the middle of the V-disparity. Thus, the scene contains an obstacle and its static environment is close to the vehicle. The static environment in Figure 14 (b) has an unusually thick due to the sky fragment with low in texture. Table 4 shows the high prediction accuracy of the proposed method compared to the DBN-

Table 3: Performance comparison between HAE-OCSVM, DBN-OCSVM, and SDA-OCSVM based on FREE-DST.

Dataset (Samples)	Approach	Inliers (TP)	Outliers (FP)
500	DBN-OCSVM	89.78	10.22
	HAE-OCSVM	**99.51**	**0.49**
	SDA-OCSVM	89.39	10.61
1000	DBN-OCSVM	89.45	10.55
	HAE-OCSVM	**99.95**	**0.05**
	SDA-OCSVM	90.3	9.70
2000	DBN-OCSVM	90.25	9.75
	HAE-OCSVM	**99.92**	**0.08**
	SDA-OCSVM	90.08	9.92
5000	DBN-OCSVM	89.89	10.11
	HAE-OCSVM	**99.73**	**0.27**
	SDA-OCSVM	90.57	9.43

OCSVM and SDA-OCSVM methods. This fact is due to integrating the DBM, which is able to learn and extract complex data, with encoder-based dimensionality reduction, thus improving the feature extraction. These results indicate that the proposed method learns complex structures of input data.

Figure 15 presents area-under-curve (AUC) values corresponding to the proposed HAE-OCSVM method, and the DBN-OCSVM and SDA-OCSVM methods for different training data sizes. We note that the HAE-OCSVM method performed better than the other models due to the combination of two powerful deep learning architecture DBMs as feature extractors, the autoencoder for dimensionality reduction and the extended capacity of OCSVM algorithm to detect outliers.

5.3. Model trained with busy scenes (BSM):

To build a model that rejects free scenes and describes busy scenes, we trained three deep encoders (HAE,SDA,DBN) with a dateset containing sequences of busy roads (with traffic) as described above. To validate the proposed model we generated a new dataset from BUSY-DST composed of 400 true obstacles named OBS-DST. Table 5 presents the testing results of the HAE, SDA and DBN-based OCSVM methods applied to the OBS-DST dataset. The proposed method achieved a high prediction accuracy of 99.79% compared to 91.12% and 95.20% accuracy using DBN-OCSVM and SDA-OCSVM, respectively (see Table 5). Again, the overall performance of the proposed HAE-OCSVM is better that of DBN-OCSVM and SDA-

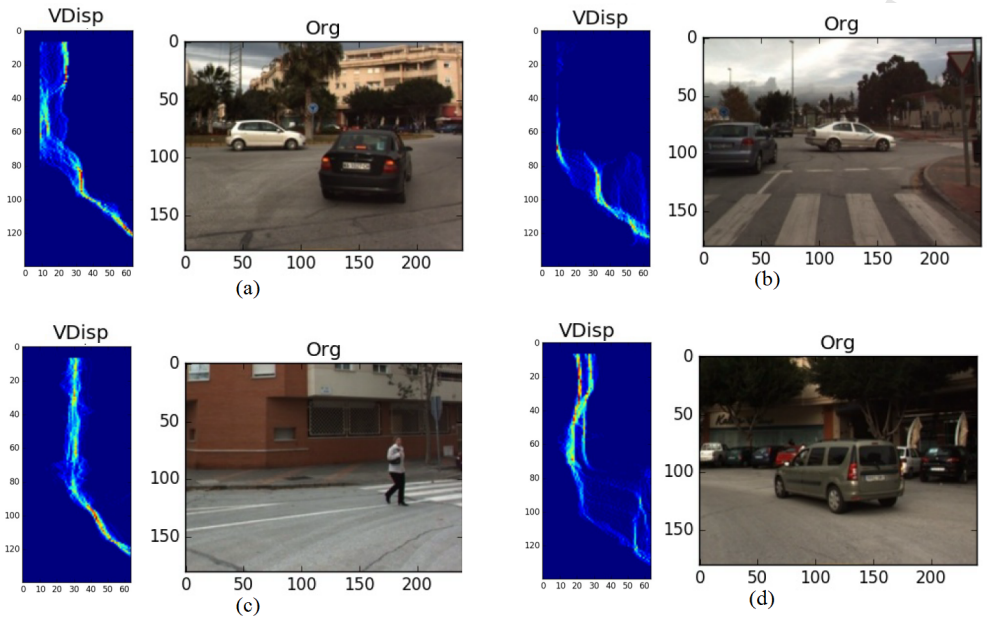


Figure 14: Examples of busy scenes (Right) Original input image and (Left) its corresponding V-disparity map.

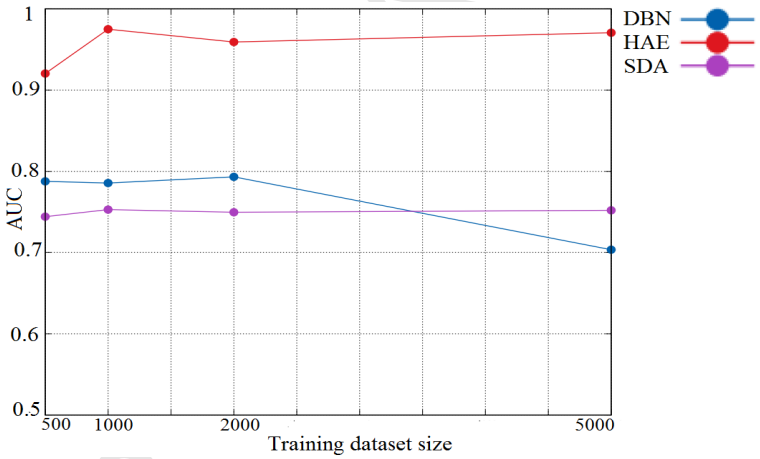


Figure 15: AUC of the proposed HAE-OCSVM method compared to DBN-OCSVM and SDA-OCSVM methods for different training-sample size.

OCSVM due to fact that DBMs are robust feature detectors that capture data correlations. In addition, complex data-dependent statistics can be discovered for learning through multiple layers.

5.4. Identification of confusing (fuzzy) situations:

Now, we focus on the identification of confusing situations. In such situations, the output response could be free scene, busy scene or fuzzy or confusing situation. These confusing situations can increase the number of false alarms. For this reason, we have to deal with fuzzy situations. Figure 16 shows few examples of fuzzy situations in which it is not easy to determining wheter or not the scene is free. From Figure 16(a-d),

Table 4: Performance comparison between HAE-OCSVM, DBN-OCSVM, and SDA-OCSVM methods applied to BUSY-DST dataset.

Dataset (Samples)	approach	Inliers (TP)	Outliers (FN)
500	DBN-OCSVM	63.89	36.11
	HAE-OCSVM	**81.98**	**18.02**
	SDA-OCSVM	52.96	47.04
1000	DBN-OCSVM	63.96	36.04
	HAE-OCSVM	**94.79**	**5.21**
	SDA-OCSVM	53.38	46.62
2000	DBN-OCSVM	64.51	35.49
	HAE-OCSVM	**91.24**	**8.76**
	SDA-OCSVM	52.96	47.04
5000	DBN-OCSVM	41.13	58.87
	HAE-OCSVM	**86.44**	**13.56**
	SDA-OCSVM	52.55	47.45

it can be seen that the environment static is close to the vehicle, which is not the case of a free scene. Also, here the vehicle is coming close to a bend. These are confusing situations.

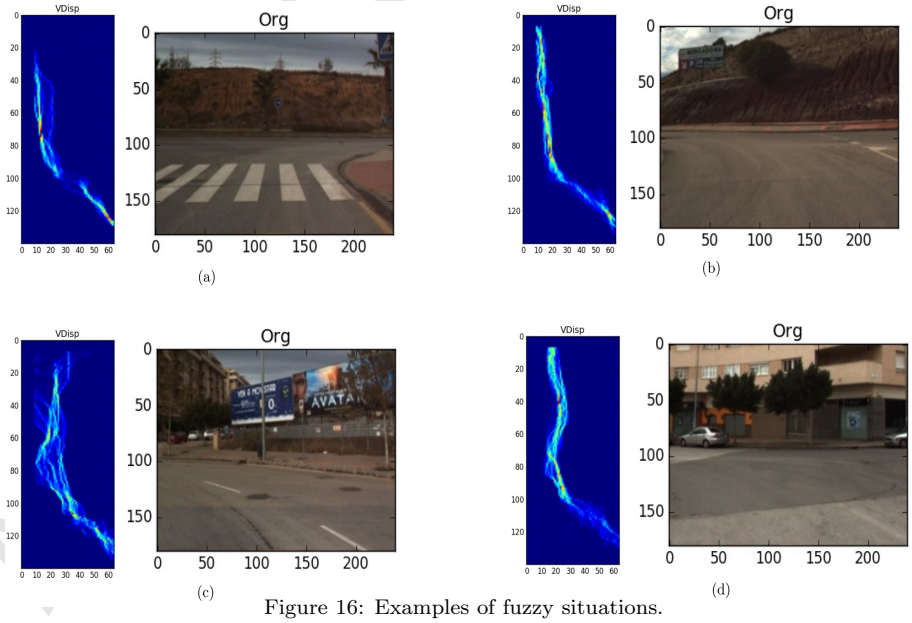


Figure 16: Examples of fuzzy situations.

Table 5: Performance of HAE-OCSVM, DBN-OCSVM and SDA-OCSVM methods trained with busy scenes and tested on OBS-DST.

Encoders	Inliers (TN)	Outliers (FN)
DBN-OCSVM	91.12	8.88
HAE-OCSVM	**99.79**	**0.21**
SDA-OCSVM	95.20	4.80

Here, we propose an approach to identify fuzzy situations as different from busy and free scenes. Towards this end, we compare responses of the FSM and BSM models to identify fuzzy situations. If both models are flagged, the tested case is considered as a fuzzy situation. By this, we can identify and filter fuzzy situations from busy and free situations.

After identifying fuzzy scenes, two cases can be distinguished: true alarm and warning alarm. A true alarm occurs if there is an obstacle in the operating area of the vehicle (see Figure 10). Figure 17 shows two examples of true alarms (i.e., the presence of obstacles in the operating area). On other hand, a warning alarm is declared if there is an obstacle in the field of view but outside the operating area of the vehicle (see Figure 18). To distinguish between true alarms and warning alarms, we used U-V disparity on the operating area of the vehicle to estimate the obstacle locations. If the obstacle is inside the operating area, then it considered as true alarm; otherwise, it considered as a warning alarm.

(a) (b)

Figure 17: Obstacle detection for true alarms examples. In each image, the colored boxes represent the predicted ROI area of the detected obstacles.

(a) (b)

Figure 18: False alarm examples. In each image, the colored boxes represent the predicted ROI area of the detected obstacles.

Here, we investigate the capabilty of this approach to distinguish between warning and true alarms. To do so, we test both BSM and FSM models with the BUSY-DST dataset, which comprises 1437 examples of confirmed obstacles (417 scenes) and 1020 fuzzy situations. After applying the identification approach, we find 59% warning alarms and 41% true alarms. This distribution (see Figure 19) is obtained according to the chosen dimensions of operating area. We can make this area stricter or more flexible by extending or reducing the disparity range.

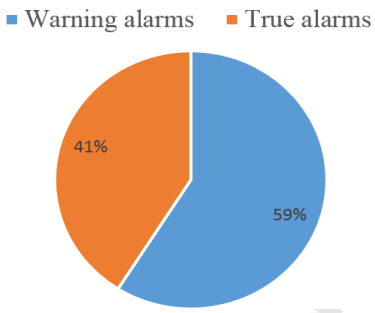


Figure 19: Filtering fuzzy situations.

5.5. Obstacle detection-based one class classifiers:

After constructing the two hybrid models trained with BUSY-DST and FREE-DST, respectively, we assess the performance of the proposed HAE-based OCSVM obstacle detection approach and we compare our results with results from five algorithms: DBN-OCSVM, SDA-OCSVM, HAE-SVDD, DBN-SVDD and SDA-SVDD. A benefit of SVMs is their ability to map problems into higher spatial dimensions using kernels, allowing a non-linear relationship to appear to be fairly linear. Here, we aim to exploit the advantages of the HAE model and those of the OCSVM with RBF kernel functions to improve the detection of obstacles. Table 6 presents a comparison between the HAE-OCSVM method with other studied classifiers. Results show that the combined HAE-OCSVM detection scheme outperforms the other algorithms used in this study. OCSVM-based detection also surpassed SVDD-based detection algorithms. This is related to the phenomenon of empty spaces inside the hyper sphere suffered by the SVDD.

Table 6: Accuracy of the OCSVM vs SVDD.

	TN	FN	TP	FP	TPR	FPR	AUC
HAE-OCSVM	86,43	13.57	99,73	0.27	0.95	0.007	0.97
DBN-OCSVM	41.12	58.88	89,88	10.12	0.78	0.38	0.70
SDA-OCSVM	56,29	43.71	90,56	9.44	0.84	0.3	0.77
HAE-SVDD	81,21	19.79	98,76	1.24	0.93	0.03	0.94
DBN-SVDD	34.65	64.35	86,82	13.18	0.77	0.49	0.64
SDA-SVDD	51,56	48.44	87,37	12.63	0.82	0.38	0.72

Implementation of these methods consist of two phases. Off-line training or learning, in which models are constructed. The models are then used to detect obstacles in future data (i.e., testing). On-line detection, in which the online measurement data are processed and the constructed models are used to to detect obstacles. For each obstacle detection method, a processing time is computed (see Table 7). In Table 7, 'Encoding time' is the time needed to encode V-disparity map, 'Total time' is the total time required to perform all steps of obstacle detection, and 'FPS number' is the number of images processed per second. Processing time in testing phase is a signicant indicator to measure the complexity of models. Meanwhile, the testing time for all the methods are within 100 ms, that means the testing size is very small. We implemented these approaches using a fast algorithm based Intel SSE (CPU i7 come with version 4) technology to accelerate computation (10 FPS) to meet real time application requirements. The processing time could be decreased further by using this approach implementing on GPU (up to 15 FSP).

Table 7: Testing processing times (ms) for each detection approach

Model	Encoding time (ms)	Total time (ms)	FPS
DBN	70	98	10.20
HAE	72	100	10
SDA	64	92	10.86

5.6. Estimation of obstacle locations:

In this subsection, we describe a statistical approach to estimate obstacle locations in a current scene. This approach is based on tracking changes in density map columns. Indeed, the motion of obstacles will generate changes in the V-disparity map, which are reflected in the density map as well. Let the Density

map matrix, $\theta_{VDisparity}$, which provides a compact representation of V-disparity, be defined as follows:

$$\theta_{VDisparity} = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & d_{1n} \\ d_{21} & d_{22} & d_{23} & \dots & d_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{m1} & d_{m2} & d_{m3} & \dots & d_{mn} \end{bmatrix} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_n],$$

where $m \in [1, M], n \in [1, N]$, with :
$M = \frac{Rows_{VDisparity}}{4}$ and $N = \frac{Columns_{VDisparity}}{4}$.

$$d_{mn} = \frac{\sum_{r=R,c=C}^{R+4,C+4} V - disparity(r,c)}{M.N}$$

where $R = (m-1) * M$ and $C = (n-1) * N$.

We check if there is any change in the columns of the density map by using density map information from previous scenes. In other words, we use residuals, $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_n]$, which represent the difference between columns of the current density map and the previous density map, as change indicators. Without obstacles, residuals are close to zero due to measurement noise, and they deviate significantly from zero in the presence of obstacles. First, we remove the density map data mean and then we apply the three-sigma rule on the residuals to detect potential changes. Upper and lower control limits, which are denoted respectively by UCL and LCL, for the residuals are defined as

$$UCL, LCL = \mu_0 \pm 3\sigma_0,$$

where μ_0 and σ_0 are respectively the mean and standard deviation of the obstacle-free residuals. The width of the control limits is usually chosen to be 3 in practice, by using this control width, the Shewhart chart would have a 0.27% probability to give a false alarm the absence of obstacles. This implies that 99.73% of the observations should be contained within the control limits in the absence of obstacles. Such a choice is motivated by the detection ability of Shewhart chart and its low-computational cost making it easy to implement in real time. Figure 20 shows an example of tracking obstacle locations by applying the three-sigma rule to the columns of the density map. Whenever the most recently measured point or a consecutive sequence of points is outside the control limits, a change is encountered. The detection of a change means the presence of the obstacle in column i of the density map. This procedure to track changes in density map columns is summarized in Algorithm 5.

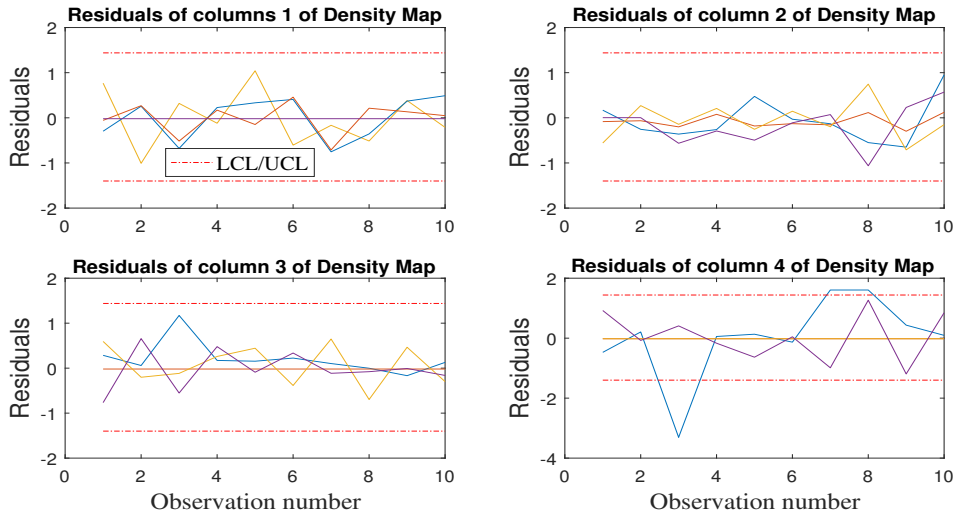


Figure 20: Tracking obstacle locations based on a density map. In each plot, the solid colored lines represent the residuals of density map. The dashed horizontal lines labeled UCL and LCL denote the upper control limit and the lower control limit of Shewhart chart.

Algorithm 5: Obstacle tracking based on density map change detection. respectively.

Input: Density Map : $\{\theta_{Previews}, \theta_{Current}\}$

Output: vector of response $r \in \{0,1\}^4$

$a_i \leftarrow$ columns of $\theta_{Previews}$

$b_i \leftarrow$ columns of $\theta_{Current}$

for *i in 1..4* **do**

  $Residue_i \leftarrow computeResidue(a_i, b_i)$

  if $Residue_i < threshold_{UCL}$ *and* $Residue_i > threshold_{LCL}$ **then**

    $r_i \leftarrow 0$

  else

    $r_i \leftarrow 1$

return r

6. Conclusion

Accurate detection, localization and tracking of obstacles in urban scenes is a key enabler to improving traffic efficiency and safety by avoiding accidents during driving. In this paper, a novel obstacle detection method based on stereovision is proposed. Specifically, we presented an obstacle detection system by combining the flexibility and accuracy of a new hybrid encoder and the extended capacity of OCSVM in anomaly detection. Indeed, the developed hybrid model merges the greedy features of deep Boltzmann Machines

(DBM) with the dimensionality reduction capacity of an auto-encoders (AE). We evaluated the proposed approach using practical data from two databases, the Malaga stereovision urban dataset (MSVUD) and the Daimler urban segmentation dataset (DUSD). We provided comparisons of the proposed model with state-of-the-art models based on the deep belief network (DBN) and stacked auto-encoders (SDA) and showed that we achieve better results. Also, we compared the detection quality of OCSVM to that of support vector data descriptor (SVDD) and found better performance. To reduce the number of false alarms and fuzzy situations, we constructed two models and used them for detection, one trained with free scenes and the other with busy scenes. We showed that by using both models together, a higher accuracy is achieved compared to DBN, SDA, and the use of one model alone. Furthermore, we developed a fast approach to estimating obstacle locations by tracking changes on a density map using the three-sigma rule.

The presence of highly noisy images makes obstacle detection more difficult as the presence of noise degrades the quality of anomaly detection. In fact, wavelet-based multiscale representation of data has been shown to provide effective noise-feature separation in the data and to approximately decorrelate auto-correlated data. As future work, we plan to exploit the advantages of multiscale denoising and those of the proposed obstacle detection approach to further enhance the performance of this technique, especially when the observed data are very noisy. To further improve the performance of this technique, the proposed method can be parallelized by implementing it on GPU to enable its real-time use in vision-based driver assistance systems. Also, other methods can be integrated with HE model for online obstacle detection should be explored. One potential approach could be to use statistical hypothesis testing approaches, such as generalized likelihood ratio test, which does not require any learning step to take place. Finally, other available databases can be used for obstacles tracking, such as the Karlsuhe Institute of Technology and Toyota Technological Institute (KITTI) car dataset.

Acknowledgement

The authors (Abdelkader Dairi and Mohamed Senouci) would like to thank the Computer Science Department, University of Oran 1 Ahmed Ben Bella for the continued support during the research. This publication is based upon work supported by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under Award No: OSR-2015-CRG4-2582. The authors would like to thank two anonymous referees whose comments and suggestions have improved the content and presentation of this work.

References

Appiah, N., Bandaru, N., 2015. Obstacle detection using stereo vision for self-driving cars.

Asvadi, A., Premebida, C., Peixoto, P., Nunes, U., 2016. 3D Lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes. Robotics and Autonomous Systems 83, 299–311.

Bengio, Y., LeCun, Y., et al., 2007. Scaling learning algorithms towards ai. Large-scale kernel machines 34 (5), 1–41.

Bengio, Y., et al., 2009. Learning deep architectures for ai. Foundations and trends® in Machine Learning 2 (1), 1–127.

Blanco, J.-L., Moreno, F.-A., Gonzlez-Jimnez, J., 2014. The mlaga urban dataset: High-rate stereo and lidars in a realistic urban scenario. International Journal of Robotics Research 33 (2), 207–214.
URL http://www.mrpt.org/MalagaUrbanDataset

Broggi, A., Caraffi, C., Fedriga, R. I., Grisleri, P., 2005. Obstacle detection with stereo vision for off-road vehicle navigation. In: Computer Vision and Pattern Recognition-Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on. IEEE, pp. 65–65.

Burlacu, A., Bostaca, S., Hector, I., Herghelegiu, P., Ivanica, G., Moldoveanul, A., Caraiman, S., 2016. Obstacle detection in stereo sequences using multiple representations of the disparity map. In: System Theory, Control and Computing (ICSTCC), 2016 20th International Conference on. IEEE, pp. 854–859.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE, pp. 886–893.

Del, C., Skaar, S., Cardenas, A., Fehr, L., 2006. A sonar approach to obstacle detection for a vision-based autonomous wheelchair. Robotics and Autonomous Systems 54 (12), 967–981.

Dollar, P., Wojek, C., Schiele, B., Perona, P., 2012. Pedestrian detection: An evaluation of the state of the art. IEEE transactions on pattern analysis and machine intelligence 34 (4), 743–761.

Duguleana, M., Barbuceanu, F. G., Teirelbar, A., Mogan, G., 2012. Obstacle avoidance of redundant manipulators using neural networks based reinforcement learning. Robotics and Computer-Integrated Manufacturing 28 (2), 132–146.

Erfani, S. M., Rajasegarar, S., Karunasekera, S., Leckie, C., 2016. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. Pattern Recognition 58, 121–134.

Fakhfakh, N., Gruyer, D., Aubert, D., 2013. Weighted v-disparity approach for obstacles localization in highway environments. In: Intelligent Vehicles Symposium (IV), 2013 IEEE. IEEE, pp. 1271–1278.

Fleischmann, P., Berns, K., 2016. A stereo vision based obstacle detection system for agricultural applications. In: Field and Service Robotics. Springer, pp. 217–231.

Gan, Q., Wu, C., Wang, S., Ji, Q., 2015. Posed and spontaneous facial expression differentiation using deep Boltzmann machines. In: Affective Computing and Intelligent Interaction (ACII), 2015 International Conference on. IEEE, pp. 643–648.

Georgoulas, C., Kotoulas, L., Sirakoulis, G. C., Andreadis, I., Gasteratos, A., 2008. Real-time disparity map computation module. Microprocessors and Microsystems 32 (3), 159–170.

Häne, C., Sattler, T., Pollefeys, M., 2015. Obstacle detection for self-driving cars using only monocular cameras and wheel odometry. In: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on. IEEE, pp. 5101–5108.

Hinton, G. E., 2007. Learning multiple layers of representation. Trends in cognitive sciences 11 (10), 428–434.

Hinton, G. E., Osindero, S., Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural computation 18 (7), 1527–1554.

Hu, Z., Uchimura, K., 2005a. Uv-disparity: an efficient algorithm for stereovision based scene analysis. In: Intelligent Vehicles Symposium, 2005. Proceedings. IEEE. IEEE, pp. 48–54.

Hu, Z., Uchimura, K., 2005b. Uv-disparity: an efficient algorithm for stereovision based scene analysis. In: Intelligent Vehicles Symposium, 2005. Proceedings. IEEE. IEEE, pp. 48–54.

K Yamaguchi, T Kato, Y. N., 2006. Moving obstacle detection using monocular vision. In: Intelligent Vehicles Symposium. IEEE, pp. 288–293.

Kang, S., Qian, X., Meng, H., 2013. Multi-distribution deep belief network for speech synthesis. In: Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, pp. 8012–8016.

Krizhevsky, A., Hinton, G. E., 2011. Using very deep autoencoders for content-based image retrieval. In: ESANN.

Labayrade, R., Aubert, D., 2003. In-vehicle obstacles detection and characterization by stereovision. In: Proceedings of the 1st International Workshop on In-Vehicle Cognitive Computer Vision Systems, Graz, Austria.

Labayrade, R., Aubert, D., Tarel, J.-P., 2002. Real time obstacle detection in stereovision on non flat road geometry through" v-disparity" representation. In: Intelligent Vehicle Symposium, 2002. IEEE. Vol. 2. IEEE, pp. 646–651.

Lee, H., Pham, P., Largman, Y., Ng, A. Y., 2009. Unsupervised feature learning for audio classification using convolutional deep belief networks. In: Bengio, Y., Schuurmans, D., Lafferty, J. D., Williams, C. K. I., Culotta, A. (Eds.), Advances in Neural Information Processing Systems 22. Curran Associates, Inc., pp. 1096–1104.
URL http://papers.nips.cc/paper/3674-unsupervised-feature-learning-for-audio-classification-using-convolutional-deep-belief-networks.pdf

Leng, B., Zhang, X., Yao, M., Xiong, Z., 2015. A 3d model recognition mechanism based on deep boltzmann machines. Neurocomputing 151, 593–602.

Liu, P., Han, S., Meng, Z., Tong, Y., 2014. Facial expression recognition via a boosted deep belief network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1805–1812.

Mohamed, A.-r., Dahl, G. E., Hinton, G., 2012. Acoustic modeling using deep belief networks. IEEE Transactions on Audio, Speech, and Language Processing 20 (1), 14–22.

Montgomery, D. C., 2009. Introduction to statistical quality control. John Wiley & Sons (New York).

Nadav, I., Katz, E., 2016. Off-road path and obstacle detection using monocular camera. In: Science of Electrical Engineering (ICSEE), IEEE International Conference on the. IEEE, pp. 1–5.

Nalpantidis, L., Kragic, D., Kostavelis, I., Gasteratos, A., 2016. Theta-disparity: An efficient representation of the 3d scene structure. In: Intelligent Autonomous Systems 13. Springer, pp. 795–806.

Nguyen, V. D., Van Nguyen, H., Tran, D. T., Lee, S. J., Jeon, J. W., 2016. Learning framework for robust obstacle detection, recognition, and tracking. IEEE Transactions on Intelligent Transportation Systems.

O'Connor, P., Neil, D., Liu, S.-C., Delbruck, T., Pfeiffer, M., 2013. Real-time classification and sensor fusion with a spiking deep belief network. Frontiers in neuroscience 7.

Petković, D., Danesh, A. S., Dadkhah, M., Misaghian, N., Shamshirband, S., Zalnezhad, E., Pavlović, N. D., 2016. Adaptive control algorithm of flexible robotic gripper by extreme learning machine. Robotics and Computer-Integrated Manufacturing 37, 170–178.

Ramos, S., Gehrig, S., Pinggera, P., Franke, U., Rother, C., 2016. Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling. arXiv preprint arXiv:1612.06573.

Salakhutdinov, R., Hinton, G., 2009. Deep Boltzmann machines. In: Artificial Intelligence and Statistics. pp. 448–455.

Salakhutdinov, R., Hinton, G. E., 2007. Learning a nonlinear embedding by preserving class neighbourhood structure. In: AISTATS. Vol. 11.

Salakhutdinov, R., Larochelle, H., 2010. Efficient learning of Deep Boltzmann machines. In: AISTATs. Vol. 9. pp. 693–700.

Scharwächter, T., Enzweiler, M., Franke, U., Roth, S., 2013. Efficient multi-cue scene segmentation. In: German Conference on Pattern Recognition. Springer, pp. 435–445.

Scharwächter, T., Enzweiler, M., Franke, U., Roth, S., 2014. Stixmantics: A medium-level model for real-time semantic scene understanding. In: European Conference on Computer Vision. Springer, pp. 533–548.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., Williamson, R. C., 2001. Estimating the support of a high-dimensional distribution. Neural computation 13 (7), 1443–1471.

Smolensky, P., 1986. Information processing in dynamical systems: Foundations of harmony theory; cu-cs-321-86.

Sun, H., Zou, H., Zhou, S., Wang, C., El-Sheimy, N., 2013. Surrounding moving obstacle detection for autonomous driving using stereo vision. International Journal of Advanced Robotic Systems 10 (6), 261.

Tax, D. M., Duin, R. P., 2004. Support vector data description. Machine learning 54 (1), 45–66.

Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.-A., 2008. Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on Machine learning. ACM, pp. 1096–1103.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research 11 (Dec), 3371–3408.

Woo, J., Kim, N., 2016. Vision based obstacle detection and collision risk estimation of an unmanned surface vehicle. In: Ubiquitous Robots and Ambient Intelligence (URAI), 2016 13th International Conference on. IEEE, pp. 461–465.

Xu, J., Li, H., Zhou, S., 2015. An overview of deep generative models. IETE Technical Review 32 (2), 131–139.

Yoo, H., Son, J., Ham, B., Sohn, K., 2016. Real-time rear obstacle detection using reliable disparity for driver assistance. Expert Systems with Applications 56, 186–196.

Zhang, X., Song, Y., Yang, Y., Pan, H., 2017. Stereo vision based autonomous robot calibration. Robotics and Autonomous Systems 93, 43–51.

Dairi Abdelkader holds in 2003 an Engineer degree in computer science from the University of Oran 1 Ahmed Ben Bella, Algeria. He got a Magister degree in 2006 from the National Polytechnic School of Oran, Algeria. He is currently preparing his Ph.D. degree in computer sciences at Ben Bella Oran1 University under the supervision of Pr. Senouci Mohamed. His current research interests include machine learning, computer vision, image processing and mobile robotics.

Fouzi Harrou received the Dipl.-Ing in Telecommunications from Abou Bekr Belkaid University, Algeria, in 2004 and the M.Sc. degree in Telecommunications and Networking in 2006 from the University of Paris VI, France. In 2010, he received the Ph.D. degree in Systems Optimization and Security from the University of Technology of Troyes (UTT), France, and was an Assistant Professor at the UTT, from 2009 to 2010. In 2010, he was an Assistant Professor at the Institute of Automotive and Transport Engineering at Nevers, France. From 2011 to 2012, he was Postdoctoral Research Associate at the Systems Modelling and Dependability Laboratory, UTT. From 2012 to 2014, he was an Assistant Research Scientist, in Chemical Engineering Department at the Texas A&M University at Qatar, Doha, Qatar. Since 2015, he is Postdoctoral Fellow in the Division of Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) at King Abdullah University of Science and Technology (KAUST). His current research interests include statistical decision theory and its applications, fault detection and signal processing, and Spatio-temporal statistics with environmental applications. He is a Member of the IEEE Computational Intelligence Society.

Senouci Mohamed received the Engineer degree and Magister degrees in computer science from the University of Oran 1 Ben Bella, Algeria in 1979 and 1994, respectively, and the Ph.D. degree in computer sciences, from the Ben Bella Oran1 University Algeria in 2007, where he is currently a Professor. His research interests include embedded systems, machine learning, and artificial intelligence.

Ying Sun is an Assistant Professor of Statistics in the division of Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) at King Abdullah University of Science and Technology (KAUST) in Saudi Arabia. She joined KAUST in June 2014 after one-year service as an assistant professor in the Department of Statistics at the Ohio State University, USA. At KAUST, she leads a multidisciplinary research group on environmental statistics, dedicated to developing statistical models and methods for space-time data to solve important environmental problems. Prof. Sun received her Ph.D. degree in Statistics from Texas A&M University in 2011, and was a postdoctorate researcher in the research network of Statistics in the Atmospheric and Oceanic Sciences (STATMOS), affiliated with the University of Chicago and the Statistical and Applied Mathematical Sciences Institute (SAMSI). She demonstrated excellence in research and teaching, published research papers in top statistical journals as well as subject matter journals, won multiple best paper awards from the American Statistical Association and the Transportation Research Board National Academies. Her research interests include spatio-temporal statistics with environmental applications, computational methods for large datasets, uncertainty quantification and visualization, functional data analysis, robust statistics, statistics of extremes.

Dairi Abdelkader

Fouzi Harrou

Senouci Mohamed

Ying Sun

Unsupervised obstacle detection in driving environments using deep-learning-based stereovision

Highlights

- A stereovision-based hybrid deep autoencoder (HAE) approach to urban scene monitoring is developed.
- This system combines the advantages of deep Boltzmann Machines (DBM) and autoencoders.
- An unsupervised HAE-based one-class SVM is developed for obstacle detection in driving environments.
- A fast obstacle tracking approach based on density maps is developed.
- Two publically available datasets, Malaga and Daimler, are used for validation.
- The detection results show the superior performance of the new combined HAE-OCSVM strategy.