Obstacle Detection for Intelligent Transportation Systems Using Deep Stacked Autoencoder and k-Nearest Neighbor Scheme

Abdelkader Dairi, Fouzi Harrou, *Member, IEEE*, Ying Sun, Mohamed Senouci

***Abstract*—Obstacle detection is an essential element for the development of intelligent transportation systems so that accidents can be avoided. In this study, we propose a stereovision-based method for detecting obstacles in urban environment. The proposed method uses a deep stacked auto-encoders (DSA) model that combines the greedy learning features with the dimensionality reduction capacity and employs an unsupervised k-nearest neighbors algorithm (KNN) to accurately and reliably detect the presence of obstacles. We consider obstacle detection as an anomaly detection problem. We evaluated the proposed method by using practical data from three publicly available datasets, the Malaga stereovision urban dataset (MSVUD), the Daimler urban segmentation dataset (DUSD), and Bahnhof dataset. Also, we compared the efficiency of DSA-KNN approach to the deep belief network (DBN)-based clustering schemes. Results show that the DSA-KNN is suitable to visually monitor urban scenes.**

***Index Terms*—Obstacle detection, autonomous vehicles, intelligent transportation systems, deep learning, clustering algorithms.**

I. Introduction

Management and monitoring of road traffic and congestion are becoming important factors for economic growth of countries. Therefore, the development of intelligent transportation systems (ITS) is more necessary than ever before. Obstacle detection is an essential element for the development of ITS so that accidents can be avoided [1]–[7]. In road environment, road obstacles detection system provides important information for driving safety and comfort. Moreover, obstacle detection has been involved in many practical applications, including smart wheelchairs, unmanned aerial vehicles and agricultural applications [8], [9].

Advances made in the areas of autonomous vehicles and intelligent transportation systems have made to ensure permanent monitoring of the road environment and detect obstacles. Detecting obstacles as soon as possible is therefore very important to avoid accidents and improve the driving safety and comfort. Several advanced technologies have been developed using sophisticated sensors, such as RADAR and LIDAR systems, and 3D cameras [5], [10]. In [11], an approach based on a special array of ultrasound sensors has been proposed to detect obstacles in extreme conditions (e.g., cloudy and foggy). The obstacles detection method in [12] is introduced for railway tracks environment based on vibration and accelerometer. In [13], Diego et al. proposed an obstacles detection system using signal processing for airborne transmission. In [14], an intelligent fiber grating (FG)-based 3D vision sensory system has been proposed for real-time obstacles detection, monitoring, and tracking by using laser technology combined with CCD camera for detection. In [15], using Microsoft Kinect, Javier et al. introduced an outdoors technique for detection of obstacle close to the ground. In [16], Yamaguchi et al. proposed a vision-based approach to detect moving obstacles on roads based on a vehicle-mounted monocular camera. In [17], Appiah and Bandaru presented a stereo vision-based methodology using 360 vertical cameras to detect obstacles around an autonomous vehicle.

Vision-based people detection is the subject of more recent works due to its large application such as in robotics and human tracking. In [18], Benenson et al. introduced an efficient algorithm based on boosting cascade classifiers with histograms of oriented gradient (HOG) like features to detect pedestrians in images and videos. This approach is implemented using a single CPU+GPU desktop machine. In [19], Pfeiffer and Franke proposed a compact way to encode the free space and obstacles based on Stixel World, in which 3D-situation are represented by a set of rectangular sticks named stixels. This approach uses the Semi-Global Matching (SGM) running on FPGA board and a GPU to compute dense optical ow for tracking purpose. Morales et al. also presented an innovative object tracking method based on the Stixel world [20]. These works use image processing for detection and focus only on pedestrians and ignore other obstacles. In [21], Wang et al. use the U-Disparity as key features for the on-road objects detection and particle filtering for multiple object tracking. However, the run-time of this algorithm is about 3.4s per frame for on-road detection processing and 0.05s per frame for multiple obstacle tracking, which make it a difficult task for real-time application. In these aforementioned approaches, the detection and recognition tasks are performed based on image processing. Once the regions of interest (ROI) are built based on different methods such as Stixel, then a classifier is applied to identify the potential object. This paper is concerned with the problem of stereovision-based obstacle detection for mobile robots in driving environments. The main objective of our approach is detecting obstacles (e.g.,

A. Dairi and M. Senouci are with 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

F. Harrou and Y. Sun are with 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

pedestrians, cars, and bikes) in the urban environment.

In obstacle detection, machine learning turn out to play an important role [22]–[24]. Deep convolutional neural networks (CNNs) [25] showed good performance in the field of object recognition via supervised classification. In [26], a deep CNN based approach was proposed to detect obstacles on a road. However, some tasks are not possible via CNN including dimensionality reduction, unsupervised learning, and require huge training data [27]. To bypass these limitations, Stacked-autoencoders, which are efficient and robust deep learning architectures, can be used [28]. These deep-learning based approaches are generally comprised of three main stages. First, images are fully scanned, then the potential ROIs are identified and surrounded, and finally, the recognition stage can be started. Such process with high computational cost and time-consuming is executed whenever the obstacles exist or not. In fact, this issue represents the main drawback of such an approach.

This study was motivated by the fact that the deep stacked auto-encoders (DSA) model combines the greedy learning features with the dimensionality reduction capacity. Therefore, the primary objective of this study was to exploit the deep DSA model to accurately and reliably detect the presence of obstacles. 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 [29], [30], is mostly stable with small variations due to measurement noise. The V-Disparity can significantly change in the presence of obstacles. In fact, we start with unsupervised greedy layer-wise training of the deep encoder using the normalize V-Disparity dataset. We address obstacle detection as an anomaly detection problem based on the KNN as a classifier, which requires only obstacle-free data in training. The choice of KNN scheme is motivated by its flexibility to do not make assumptions on the statistical distribution of data at all, and its capacity to deal with the non-linearity in the data [31]. The data encoded by the deep encoder model are the input to KNN for obstacle detection. The role of KNN scheme is to distinguish free scenes from busy scenes in the testing data. With three publicly available datasets, the Malaga stereovision urban dataset, the Daimler urban segmentation dataset, and Bahnhof dataset, we evaluated the proposed method. Results show the capacity of the proposed approach to reliably detect obstacles and superior performance compared to DSA-based KMeans (KM), Mean Shift (MS), Expectation maximization (EM), Birch, Spectral clustering (SC), Agglomerative (AG), and Affinity Propagation (AP). Furthermore,we provide comparisons of the proposed approach with the deep belief network (DBN)-based clustering schemes and showed that we achieve better results.

The remainder of this paper is organized as follows. Section II provides a brief overview of machine learning algorithms used in this study. Section III briefly presents stereovision. Section IV introduces the proposed method. Section V presents and discusses the results, and Section VI lists the conclusions of the study.

II. Research Method

This section primarily introduces a brief overview of data clustering algorithms used in this study (see Table I) and stacked autoencoders used to build deep learning models.

TABLE I: Clustering techniques.

Algorithm	Clustering technique
K-nearest neighbor (KNN)	Distance
Mean Shift (MS)	Density
Affinity Propagation (AP)	Similarity
Agglomerative Clustering (AG)	Agglomerative
Birch	Hierarchical
KMean (KM)	Partition
Spectral clustering (SC)	Graph
Expectation Maximization (EM)	Probabilistic

A. K-nearest neighbor

The k-nearest neighbor (KNN) approach is considered as non-parametric lazy learning approach, which means that no prior assumptions on the underlying data structure are required [32]. This property is very useful in many practical situations where the collected data are non-Gaussian distributed or cannot be linearly separable. Classification via KNN algorithm is performed by evaluating distances between training samples (e.g., the Euclidean, Manhattan, or Minkowski distance function). For a given a new data point x and a training dataset $\mathcal{D}$, the classification of x is achieved based on the k nearest neighbors, which have the smallest distances. KNN compute all distances d_i, defined by $d_i = distance(x, \mathcal{D}_i)$ of the training set $\mathcal{D}$ elements. KNN was applied successfully to large datasets like handwritten digits or satellite image scenes which prove its capacity to deal with high dimensionality problems. Here, we have used ball-tree algorithm, which is hierarchical data-structure that proved efficient in speed up the search of neighborhood points in high dimensionality cases [33].

B. K-means clustering

The K-means is a partitioning clustering technique based on minimization of the average squared distance between points in the same cluster [34]. It is an iterative algorithm that attempts to assign n observations to one cluster from the k clusters prefixed a priori by centroids. Specifically, for a given k the number of desired clusters, where each cluster has its own centroid, data point are assigned to the closest centroid. Each cluster update it centroid based on the new assignments. This step is repeated until the centroids remain unchanged. Given an integer k and a set of m data points in $Z \subset R^d$, the goal is to choose k centers C so that the total squared distance between each point and its closest center, $\varphi = \sum_{z \in Z}^{I} \min_{c \in C} \|z - c\|^2$, is minimum.

C. Mean shift

The mean shift clustering is a recursive technique that does not require a predefined number of clusters. It permits

performing nonparametric clustering based on kernel density estimation of the probability density function of the input data. It is comprised of three main steps 1) after defining a window around data points, 2) we compute the data point mean, and then 3) we shift the center of the window to the mean. These steps are repeated till convergence. In fact, Mean shift estimate window density based on kernel function. Given n data points z in R^d, the kernel density function gotten with kernel $\kappa(x)$ and window radius r is

$$f(z) = \frac{1}{nr^d} \sum_{i=1}^{n} \kappa(\frac{z - z_i}{h}) \quad (1)$$

In the case of radially symmetric kernels, the profile of kernel $k(x)$ can be defined as

$$\kappa(x) = c_{k,d} k(\|x\|^2) \quad (2)$$

where $c_{k,d}$ represents a normalization constant which guarantee $\kappa(x)$ integrates to 1. More details on Mean shift clustering refer to [35]. The window size should carefully be selected.

D. Expectation maximization

Expectation maximization (EM) algorithm was proposed by [36], to solve problems of maximizing likelihood estimation for data in which some variables can not be observed (Latent variables), In others words EM algorithm attempts to approximate the observed data distributions among *Gaussian Mixture* distributions. For given observed data X we need to determine the value of Φ (model parameters) that maximizes the log likelihood, $\mathcal{L}(\Phi) = log\ P(x|\Phi)$, by introducing latent variables Z we can write $\mathcal{L}(\Phi|X, Z) = log\ P(X|\Phi, Z)$. We assume that elements of the observed data X follow $\mathcal{C}$ Gaussian distributions, characterized by the parameters $\Phi = \{m, \sigma\}$,Here unobserved data or Latent variables are Gaussian-selector random variable. Expectation Maximization determine memberships to a cluster by computing probabilities of based on *Gaussian Mixture* distributions. Then the probability that maximize the observed data likelihood is used to choose the cluster, we can say that at the end each observation is member of all clusters under certain probability.

E. Birch

BIRCH (balanced iterative reducing and clustering using hierarchies) is a hierarchical clustering algorithm designed to deal with large datasets [37]. This algorithm, which is based on clustering feature tree (CF-Tree), incrementally changes the quality of sub-clusters. BIRCH clustering algorithm is easy to implement in few steps: (1) CF-Tree is built based on a full dataset scan. (2) A new compressed version of CF-Tree is created. (3) Perform a global clustering, and (4) refine clustering.

F. Spectral clustering

Spectral clustering algorithm addresses clustering as a graph partitioning problem [38]. In this procedure, data points are presented as the vertices of a graph. Vertices are connected by edges and each edge has a weight. Large weights indicate that the adjacent vertices are very similar and vice versa. Spectral clustering algorithm uses the spectrum of the similarity matrix to cluster data points [38]. For details, refer to the reference [38].

G. Affinity Propagation algorithm

Affinity Propagation is a machine learning algorithm that identifies a set of *exemplars* that represents the dataset, and used as input similarity/dissimilarity measures between pairs of data points [39]. In this algorithm, the number of clusters is not prefixed before running as the case of k-means algorithm. The main advantage if this algorithm is its general applicability, and ability to cluster a large number of clusters. However, using this algorithm, it is difficult to know the value of parameters to achieve optimal clustering solutions.

H. Agglomerative clustering

Hierarchical clustering algorithms provide a nested sequence of clusters organized as a hierarchical tree [40] Agglomerative hierarchical procedures begin with data points as separate clusters. In the Agglomerative hierarchical algorithm, the number of clusters k is not specified as an input. At each step of the algorithm, the most similar clusters are merged. The procedure is repeated until the distance between two closest clusters is above a certain threshold distance.

I. Autoencoders

An autoencoder is an artificial neural network [28] 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 1). Autoencoders are widely used for 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 the capacity to learn more features than linear principal component analysis [28]. The purpose of the decoder part is to reconstruct its own inputs via the decoder function, $\widehat{x} = Decoder(h)$. The learning process of an autoencoder is achieved by minimization of the negative log-likelihood (loss function) of the reconstruction, given the encoding $Encoder(x)$ [28]:

$$Reconstruction_{error} = -log(P(x|Encoder(x)), \quad (3)$$

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. Autoencoders can be stacked to build deep architectures getting the so called *stacked autoencoder*. Stacked autoencoder models (see Figure 2) have been widely applied in image denoising [41], [42], content-based image retrieval, speech enhancement, bilingual word representations [43]–[45], and medical object recognition [46], especially organ detection.

III. Stereovision

Stereovision is a process that provides depth information from two images of the same scene. It usually depends on

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2018.2831082, IEEE Sensors Journal

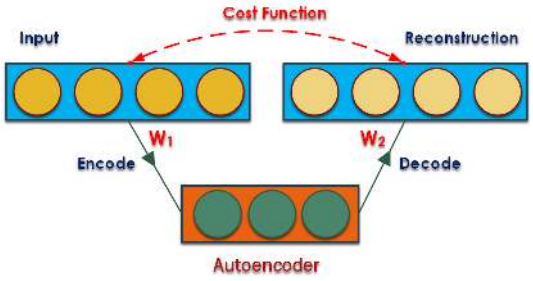

Fig. 1: Schematic representation of an Autoencoder.

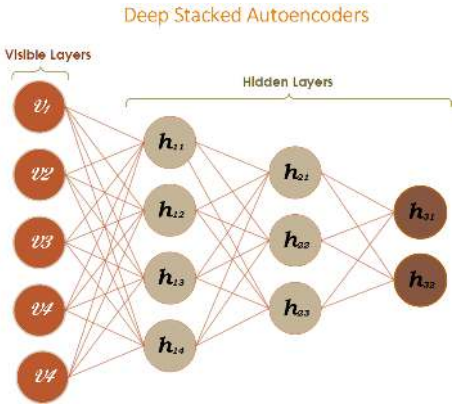

Fig. 2: A Stacked Autoencoder.

epipolar geometry for performing spatial perception and depth estimation using a disparity map of two rectified images (left and right) [1], [3]. A stereo images pair are captured by two cameras, separated by a distance called baseline, looking to the same scene. A rectification of the two collected images is required to project them onto the same plane, thus having the same y-axis. To do so, the object's position in the left image will be shifted in the right image by an amount (displacement) inversely proportional to the distance between objects and the stereo vision platform (device). We called this displacement a "Disparity". A Disparity map is built by computing the disparity for all pixel in the original image (usually in left image). In stereo, computing the disparity is treated as a problem of correspondence or block matching using cross-correlation approach [1], [3], [30], [47]. Several metrics have been developed to compute disparity mapsh [1], [3], [30], [47], $\mathcal{M}$, via different correlation metrics. The sum of absolute differences (SAD) is one of the most used metric to compute the disparity map (see Equation 4) because of its simplicity (fast computation) which meet the real-time requirement [1], [3], [30], [47].

$$\mathcal{M}(i,j,d) = \sum_{u=-w}^{w} \sum_{v=-w}^{w} |X_L(i+u,j+v) - X_R(i+u,j-d+v)| \quad (4)$$

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

Disparity maps, which are defined as the differences between all points in the rectified left and right, can be used as change indicators. The disparity decrease when the distance between the object and camera increases, and vice versa. V-disparity and U-disparity are built respectively using the numbers of pixels in rows and columns of the disparity map [1], [3]. V-Disparity map provides the depth estimation and serves to the estimation of a road's profile using the Hough transform. It gives information about obstacles height and positions with respect to the ground [1], [3]. On the other hand, a U-Disparity map provides depth estimation and obstacles width [1], [3], [48]. Based on U-Disparity and V-Disparity maps region of interest (ROI) surrounding obstacles can be determined as shown in Figure 3.

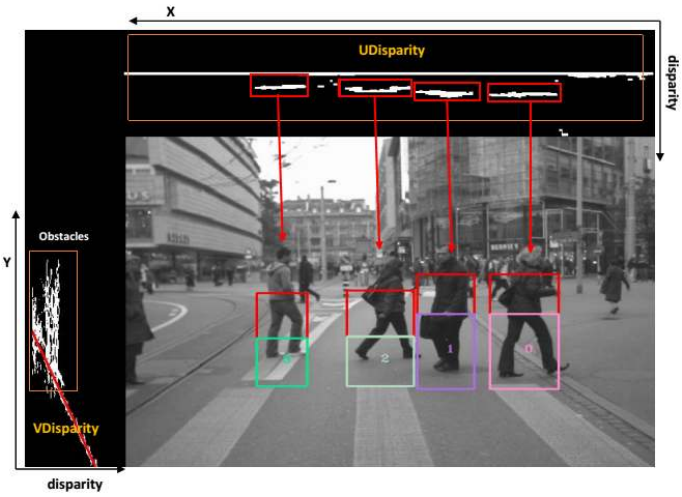

Fig. 3: An example of using V and U disparity maps to identify obstacles.

Figure 4(a-b) shows respectively a case of V-Disparity in the absence of obstacles and another case in the presence of an obstacle. From Figure 4(a), the V-Disparity of road profile is presented by the inclined line (accumulation of pixel intensities). The static environment is presented as vertical thick points located on the lower disparity because it is away from the vehicle. Generally, the road profile in the absence of obstacles is clearly visible without significant deformation (Figure 4(a)). Obstacles on a road are arising as a vertical cloud of dots with high intensities on the road profile; thick sus as bus, truck or close car and less thick in case of pedestrians (Figure 4(b)). When the obstacle moves away from the vehicle, the thickness will decrease. Figure 4(b) illustrate walking pedestrians vertical lines on the road profile indicate the presence of these obstacles (i.e., pedestrians).

IV. Proposed Obstacle Detection Procedure

The proposed deep stacked autoencoder-based (DSA) obstacle detection approach consists of four layers, where each layer extract features and encode its input to an output to be an input for the next layer. The normalized V-disparity map is the visible input of the DSA model. The input to the KNN classifier is the output of the last layer. The proposed deep learning architecture is trained in an unsupervised way without any labeling of the training data. The proposed procedure is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2018.2831082, IEEE Sensors Journal

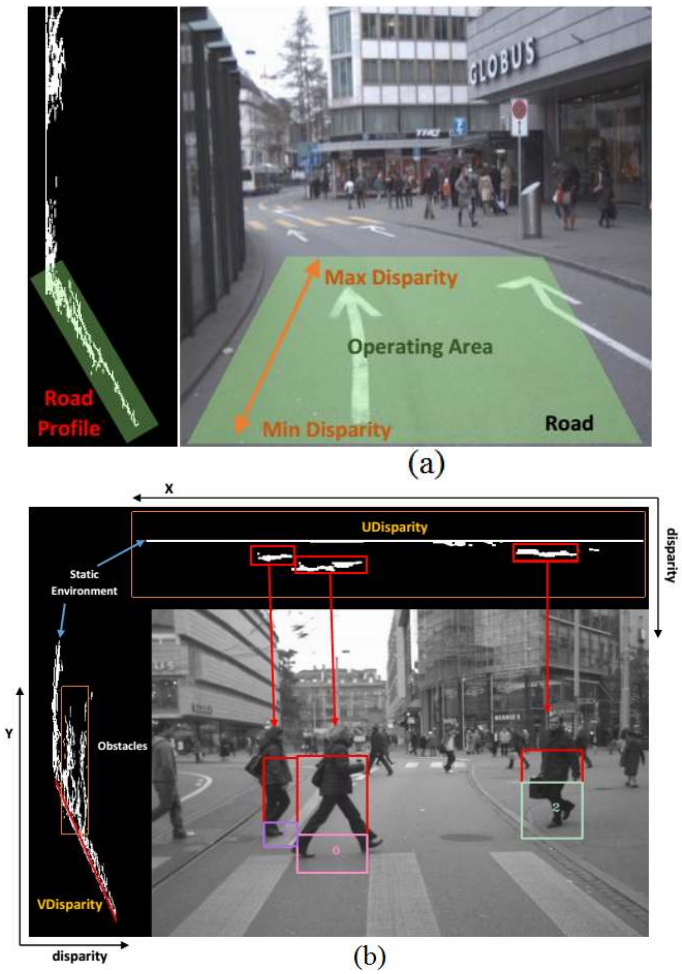

Fig. 4: Samples of V-Disparity maps in the case: (a) free-scene, and (b) busy scene.

implemented in two main steps. First, the system is based on an innovative framework for feature extraction and encoding. This is based on a stacked autoencoder model that play at the same time the role of feature extractor and encoder for dimensionality reduction. In fact, we start with unsupervised greedy layer-wise training of the deep 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. Second, we address obstacle detection as an anomaly detection problem based on the KNN as binary classifier. The data encoded by the deep encoder model is used as the input to KNN scheme for obstacle detection (Figure 5). The central role of the KNN is to separate inliers from outliers in the testing data by computing distances. The KNN rule classifies a new sample, x, based on its similarity with k nearest neighboring samples in the training set. Specifically, the KNN algorithm computes all distances d_i, defined as d_i = distance(x, Di) of the training set D elements. The distance between an abnormal sample (i.e., the presence of obstacle) and its k nearest neighbor normal samples (training) is larger than the distance between the normal sample (i.e., obstacle-free scene) and its k nearest neighbor normal samples (training). In other words, a normal sample obtained from free-scene is similar to those training samples obtained under the free-scene situation, while an abnormal sample (i.e., the presence of obstacles) significantly deviates from the normal training samples. Hence, obstacle detection can be done by checking the distance between a new sample and training observations. If the distance exceeds a decision threshold, T, then an obstacle is detected (i.e., abnormal sample). Otherwise, it is a free scene. To fix a threshold for obstacle detection based on KNN approach, we applied 3-Sigma rule to the distances obtained from the KNN algorithm [49].

$$T = \mu_D + 3\sigma_D, \tag{5}$$

where μ_D and σ_D are the mean and standard deviation of KNN distances under obstacle-free cases. We get a signal of an outlier (e.g., the presence of obstacle) at the i-th time point if the KNN distance, d_i, exceeds the decision threshold, i.e.,

$$d_i > T. \tag{6}$$

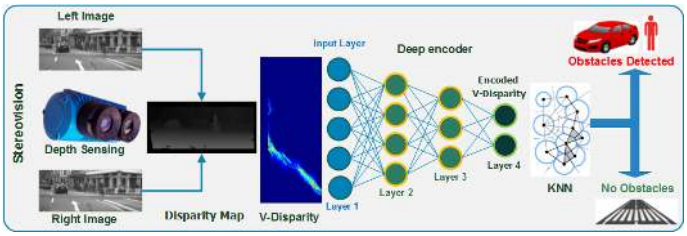

Fig. 5: A schematic diagram of the proposed DSA-based KNN obstacle detection system.

In the proposed approach, obstacles detected by the KNN algorithm are considered as potential obstacles only if they are within the operating area (see Figure 6). This area, which is the region in front of the vehicle must be kept free to avoid collisions, is defined as disparity range (min-max).

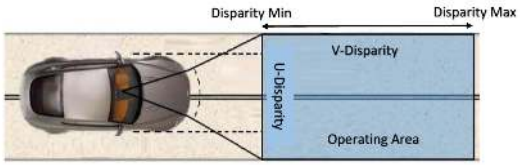

Fig. 6: Operating area.

A. Deep architecture training

In this section, we describe the main steps to train the proposed deep architecture. We build the deep encoder model via unsupervised training. The main steps of the training approach are summarized in Algorithm 1.

The dataset used to train the model contains rectified images (left, right) mostly free scenes with a few obstacles. V-disparity map computed from a pair of images (stereo-vision) is used as the input in the proposed approach. The deep encoder has the ability to learn a complex data distribution and yielding low dimensional outputs. Indeed, training phase

Algorithm 1: Training DSA-KNN approach

Input: Dataset of images (Left, Right): TrainingDataset
Output: Deep Stacked autoencoder Model: DSA_{Model}, and K-Nearest Neighbors threshold: $KNN_{Threshold}$

for *Each tuple (Left, Right) in TrainingDataset* **do**
  $DMap \leftarrow Build_{DisparityMap}(Left, Right)$;
  $VDisparity \leftarrow Build_{VDisparity}(DMap)$;
  $\mathcal{X} \leftarrow VDisparity$;
  for *Each Layer $\mathcal{L}$ in DSA Layers* **do**
    $output_{\mathcal{L}} \leftarrow LearnAndEncode(\mathcal{X})$;
    $\mathcal{X} \leftarrow output_{\mathcal{L}}$;
  Add $\mathcal{X}$ to $EncodedDataset$;
$DSA_{Model} \leftarrow BuildDSAModel()$;
$KNN_{Distances} \leftarrow ComputeKNN_{Distances}(EncodedDataset)$;
$KNN_{threshold} \leftarrow 3Sigma(KNN_{Distances})$;
return DSA_{Model}, $KNN_{Threshold}$

aims to minimize reconstruction error computed through cross-entropy, $\mathcal{L}$ (see equation 7). Cross-entropy metric, $\mathcal{L}$, is usually used to validate the constructed deep learning model. Cross-entropy, $\mathcal{L}$, measures the divergence between two probability distribution of input data and reconstructed data from the deep learning model. Its value quantifies the dissimilarity between the two distributions [42].

$$\mathcal{L}(X, \hat{X}) = -\sum_{i}^{n}\sum_{j}^{m}(\hat{x}_{ij}\log(x_{ij}) + (1-\hat{x}_{ij})\log(1-x_{ij})) \tag{7}$$

where X is the input $(n \times m)$ normalized VDisparity matrix and $\hat{X}$ is the reconstructed VDisparity matrix via the built model of size $(n \times m)$.

In this approach, an unsupervised KNN scheme is used to classify the encoded V-Disparity map generated from the DSA model. A free-scene training data set was used to determine the KNN decision threshold via the three-sigma rule, which is then applied to the new encoded data during the testing phase to discriminate between free scenes and busy scenes.

V. Results and discussion

A. Data description

This section reports on the effectiveness of the proposed deep encoder approach. Towards this end, we performed experiments on three practical datasets: the Malaga stereovision urban dataset (MSVUD) [50], the Daimler urban segmentation dataset (DUSD) [51], [52] and Bannhof dataset [53]. 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 [52]. In Bannhof dataset, the resolution of images is (800×600) pixels. After, we unified the image size of all images from the three datasets to the same resolution 800×600), which is common practice in vision-based deep learning approaches [54]. The use of a fixed resolution for all images permits avoiding the adjustment of the number of units in the visible layer of the Deep Stacked Autoencoder.

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 extracted 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 [50]. The first dataset termed FREE-DST contains images sequences of free roads. The second dataset called BUSY-DST contains images sequences 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.

Here, we assess the effectiveness of the proposed DSA-KNN detection approach by using real data. We also compare the detection quality of DSA-KNN to that of DSA-based KMeans (KM), Mean Shift (MS), Expectation maximization (EM), Birch, Spectral clustering (SC), Agglomerative (AG), and Affinity Propagation (AP) approaches. In this study, the clustering algorithms are used as binary classifiers trained in a similar way as KNN. The output of the last layer of DSA from the training samples (obstacle-free data) is used as input to train the clustering algorithms. Then, the DSA model and clustering algorithms are used together for testing new datasets. The experimental parameters of the machine learning approaches and clustering algorithms studied in this paper are presented in Table II. The used deep stacked auto-encoder consists of four layers.

- Layer 1: it is called visible layer or input later. This layer is fed directly from the input which is the VDisparity in our case. The number of neurons in this layer is (600×256) neurons.
- Layer 2: it is the first hidden layer, which is used as the first level of extracting features and reducing dimensionality of the input to quarter $((600 \times 256)/4)$.
- Layer 3: it is the second hidden layer, used as next level to extract features and reduce dimensionality of the output of previous hidden layer. This level contains $((600 \times 256)/64)$ neurons.
- Layer 4: it is the last hidden layer also called output layer, the final dimension 1024 neurons.

In the proposed approach, we fulfill the real-time requirements achieving 12 frames per second on ordinary desktop

TABLE II: Values of parameters used in the studied schemes.

Models	Parameter	Value
KNN	weights	uniform
	algorithm	BallTree
	metric	minkowski
	leaf_size	30
Autoencoder	Learning rate	0.01
	Training epochs	100
MS	bandwidth	0.44
AP	damping	0.5
	iteration	200
	convergence	15
AG	affinity	euclidean
	linkage	ward
Birch	threshold	0.5
	branch	50
KM	init	k-means++
	init	10
	iteration	300
SC	affinity	rbf
	gamma	1.0
EM	covarianceType	full
	covar	1e-06
	iteration	100
Operating area	Disparity Min	32 (pixels)
	Disparity Max	64 (pixels)

equipped with Intel i7 CPU and using Intel Streaming SIMD Extensions Technology (SSE4).

Here, we quantify the performance of detection procedures by the number of true positive (TP), false positive (FP), true negative (TN), false negative (FN), the recall or sensitivity, which is the true positive rate (TPR), false positive rate (FPR). Also, the Area Under Curve (AUC) metric is used to evaluate the performance of the studied algorithms [55].

B. Model trained with free scenes (FSM):

For obstacle detection, we construct a DSA model with V-Disparities of free scenes. Samples of free scenes and their corresponding V-Disparity maps are given in Figure 7. The road profile is presented in the V-disparity map by an inclined line (Figure 7). The static environment presented by a vertical line is located in the low V-disparity. From Figure 7, we conclude that there are no obstacles in these scenes and the static environment is far from the vehicle.

Now, we investigate how the performance of the proposed approach can be affected when varying the size of the training data. Table III summarize the accuracy of the proposed DSA-KNN algorithm obtained using the training dataset with different size of FREE-DST data. Table III shows that the DSA-KNN algorithm has got very satisfactory accuracy with 5000 training samples.

Here, the performance of the designed model will be evaluated using BUSY-DST. Samples of busy scenes are presented in Figure 9. Figure 9 shows that the road profile in the VDisparity map contains vertical cloud of points. This means that obstacles are present in these scenes. Tables IV provides results of the prediction performances of the proposed

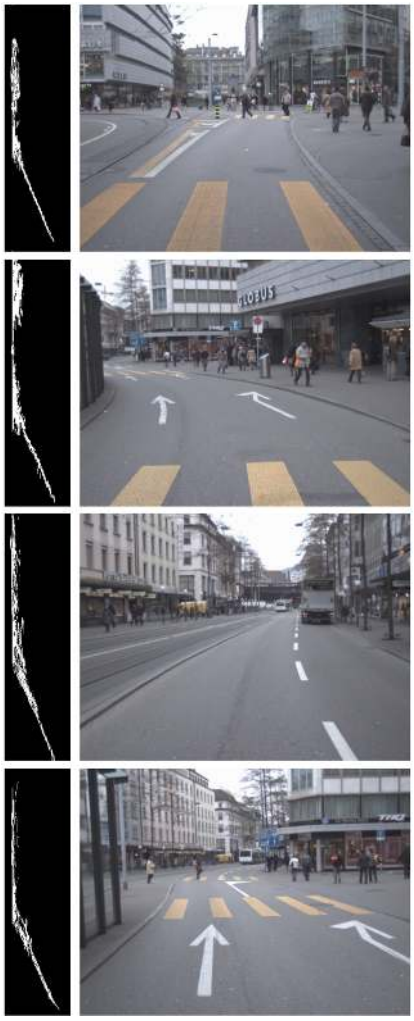

Fig. 7: Samples of free scenes (Right) original image and (Left) its corresponding V-Disparity map.

TABLE III: Performance of the proposed DSA-KNN using FREE-DST dataset.

$Dataset$ (Samples)	Inliers (TP)	Outliers (FP)
500	85.62	14.38
1000	94.25	5.75
2000	96.52	3.48
5000	97.26	2.73

approach for different sizes of BUSY-DST data. This result clearly demonstrates that the DSA-KNN approach performed well in detecting obstacles, in particular when the number of samples is large.

Figure 8 shows the convergence of the cross-entropy loss versus the number of epochs. An Epoch represents a single forward and backward pass of the entire training dataset through the deep neural network layers. From Figure 8, it can be seen that the cross-entropy values are quite close to zero in cases when the number of epochs is around 120. This means

TABLE IV: Performance of the proposed DSA-KNN based on the BUSY-DST dataset.

Dataset (Samples)	Inliers (TN)	Outliers (FN)
500	92.91	7.09
1000	98.89	1.11
2000	98.33	1.67
5000	98.61	1.39

that the reconstruction error decreases progressively to small values close to zero. It can be concluded that the designed deep learning model DSA has fit and learn the join distribution of the input (VDisparity) and able to reconstruct it with small error.

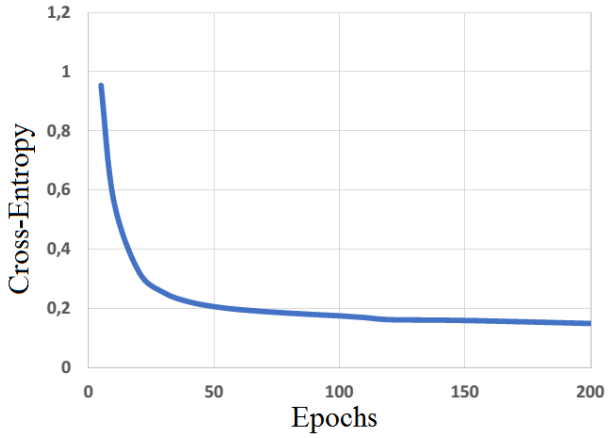

Fig. 8: The cross-entropy loss versus the number of epochs.

C. Obstacle detection-based one class classifiers:

In this subsection, we compare the proposed DSA-KNN obstacle detection approach with DSA-based KM, MS, EM, BIRCH, SC, AG, and AP schemes (see Table V). By comparing results shown in Tables V, we noted that the detection efficiency greatly enhanced by using the proposed DSA-KNN approach.

TABLE V: Detection performances of DSA-based clustering schemes when applied to MSVUD and DUSD datasets.

MODEL	TP	FP	TN	FN	TPR	FPR	AUC
KM	82.3	17.7	38.2	61.8	0.57	0.31	0.62
BIRCH	86.6	13.4	22.1	77.9	0.52	0.37	0.57
KNN	**97.2**	**2.8**	**84.4**	**15.6**	**0.86**	**0.03**	**0.91**
AG	63.6	36.4	87.9	12.1	0.84	0.29	0.77
SC	41.7	58.3	48.3	51.7	0.44	0.54	0.44
EM	76	24	57.5	42.5	0.64	0.29	0.67
AP	38.1	61.9	56.7	43.3	0.46	0.52	0.47
MS	22.3	77.7	58.7	41.3	0.35	0.56	0.39

This fact is due to the flexibility of DSA and sensitivity of KNN algorithm to small variations in the features. KNN is simple to use and implement. In addition KNN is non-parametric approach and do not has assumptions about data convexity (cluster shape) as in EM approach. Another advantage of KNN is that the number of clusters is not specified as input and it is not sensitive to the order of the data record like Birch algorithm. Also, KNN does not require any data type and do not need the notion of center (centroid). Furthermore, KNN can handle large dataset with high dimensional and is robust to noise which is not the case of some algorithms, such as Agglomerative and KMeans.

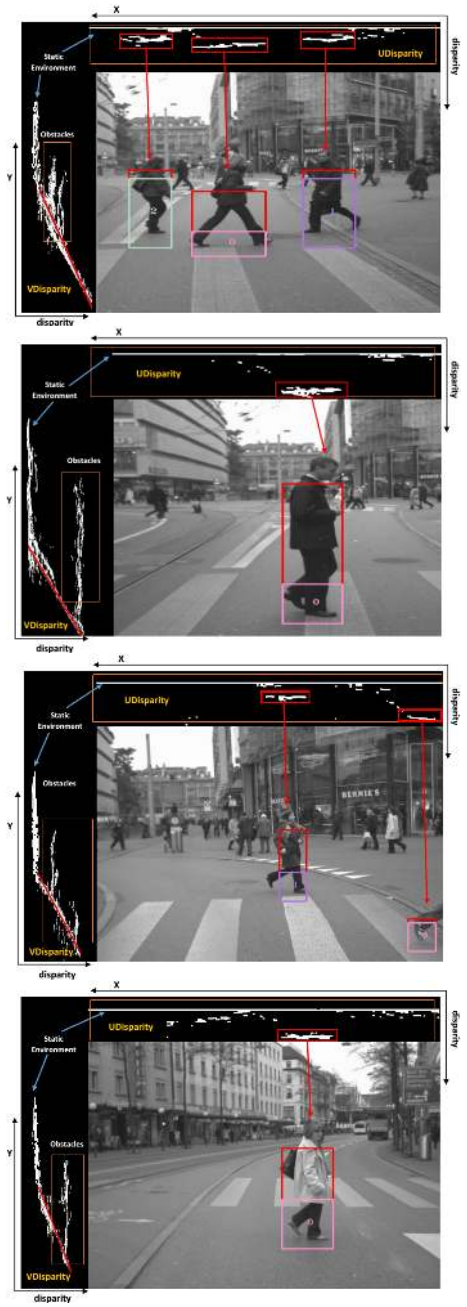

Fig. 9: Samples of busy scenes with their corresponding V-Disparity and U-Disparity maps.

Also, we have compared the performance of the proposed approach based on deep stacked autoencoder with deep belief

network (DBN)-based clustering schemes (Figure 10) while keeping the same parameters setup of the clustering schemes of the previous scenario (see Table VI). For more details about DBN refer to [27]. Results testify the powerful of the proposed DSA-KNN approach in detecting obstacles compared to DBN-based clusterin approaches, DBN-KM, DBN-MS, DBN-EM, DBN-BIRCH, DBN-SC, DBN-AG, and DBN-AP, (see Figure 11). The results show that DSA-KNN method performed better than the other models.

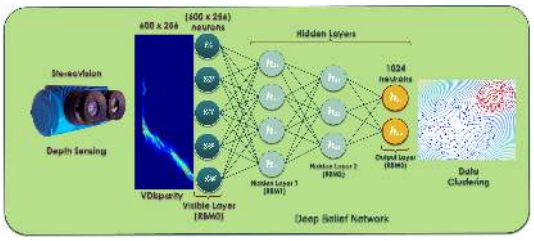

Fig. 10: DBN clustering obstacle detector

TABLE VI: Detection performances of DBN-based clustering schemes when applied to MSVUD and DUSD datasets.

Model	TP	FP	TN	FN	TPR	FPR	AUC
KNN	97.26	2.73	81.4 6	18.53	0.83	0.11	0.86
EM	60.64	39.35	84.94	15.05	0.80	0.61	0.59
Birch	83.68	16.31	76.39	23.60	0.77	0.50	0.63
AG	72.63	27.36	91.94	8.057	0.90	0.46	0.72
MS	74.65	25.34	80.01	19.98	0.78	0.56	0.61
AP	81.95	18.04	61.64	38.35	0.71	0.63	0.54
KMeans	81.60	18.39	77.51	22.48	0.78	0.52	0.63
SC	39.01	60.98	90.53	9.46	0.80	0.67	0.56

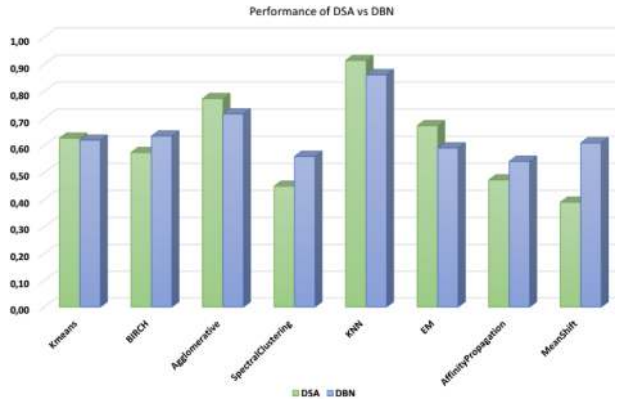

Fig. 11: AUC of the proposed DSA-KNN method compared to other methods.

We also performed experiment based on Bahnhof datasets [53], which is focused on pedestrians's tracking-by-detection in busy inner-city scenes. The Bahnhof consists of 800 stereo image pairs, we identify about 280 busy scenes and 520 free scenes. Tables VII and VIII present respectively a comparison between the DSA-KNN method with other studied classifiers, and DBN-based clustering approaches when they applied to the Bahnhof dataset. Furthermore, Figure 12 shows the AUC comparison between DSA and DBN-based clustering schemes. The combined DSA-KNN detection scheme also surpassed the other algorithms used in this study.

TABLE VII: Detection performances of DBN-based clustering schemes when applied to Bahnhof dataset.

Model	TP	FP	TN	FN	TPR	FPR	AUC
KNN	0.85	0.15	0.8	0.2	0.80	0.15	0.82
Kmeans	0.51	0.49	0.54	0.46	0.52	0.47	0.52
BIRCH	0.52	0.48	0.38	0.62	0.45	0.55	0.44
AG	0.81	0.19	0.57	0.43	0.65	0.25	0.70
SC	0.51	0.49	0.45	0.55	0.48	0.52	0.47
EM	0.74	0.26	0.51	0.49	0.60	0.33	0.63
AP	0.47	0.53	0.41	0.59	0.44	0.56	0.43
MS	0.45	0.55	0.39	0.61	0.42	0.58	0.41

TABLE VIII: Detection performances of DSA-based clustering schemes when applied to Bahnhof dataset.

Model	TP	FP	TN	FN	TPR	FPR	AUC
KNN	0.98	0.02	0.91	0.09	0.91	0.02	0.94
Kmeans	0.82	0.18	0.32	0.68	0.54	0.36	0.59
BIRCH	0.82	0.18	0.2	0.8	0.50	0.47	0.51
AG	0.81	0.19	0.59	0.41	0.66	0.24	0.71
SC	0.57	0.43	0.505	0.495	0.53	0.45	0.53
EM	0.73	0.27	0.51	0.49	0.59	0.34	0.62
AP	0.52	0.48	0.49	0.51	0.50	0.49	0.50
MS	0.52	0.48	0.47	0.53	0.49	0.50	0.49

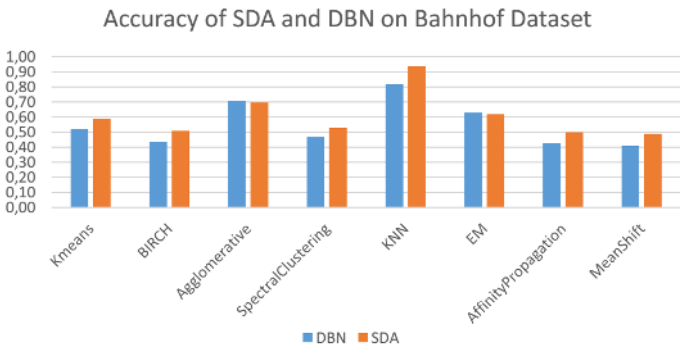

Fig. 12: Bahnhof dataset experiment results

VI. Conclusion

Obstacle detection is an essential element in the development of automatic driver assistance tool for enhancing safety on the roads. In this paper, we proposed a novel stereo vision method capable of detecting obstacles in a road environment. The proposed obstacle detection system merges the flexibility and accuracy of a new deep encoder and the extended capacity of KNN in anomaly detection. We evaluated the proposed approach using practical data from three databases, the Malaga stereovision urban dataset (MSVUD), the Daimler urban segmentation dataset (DUSD) and Bahnhof dataset. We provided comparisons of the proposed DSA-KNN method with DSA-based KMeans (KM), Mean Shift (MS), Expectation maximization (EM), Birch, Spectral clustering (SC), Agglomerative (AG), and Affinity Propagation (AP),

and showed that we achieve better results. Also, the proposed method showed superior performance compared to the deep belief network (DBN)-based clustering schemes.

ACKNOWLEDGMENT

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 (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. We are grateful to the five referees, the Associate Editor, and the Editor-in-Chief for their comments.

REFERENCES

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

[2] M. Fathollahi and R. Kasturi, "Autonomous driving challenge: To infer the property of a dynamic object based on its motion pattern," in *European Conference on Computer Vision*. Springer, 2016, pp. 40–46.

[3] N. Fakhfakh, D. Gruyer, and D. Aubert, "Weighted v-disparity approach for obstacles localization in highway environments," in *Intelligent Vehicles Symposium (IV), 2013 IEEE*. IEEE, 2013, pp. 1271–1278.

[4] H. Sun, H. Zou, S. Zhou, C. Wang, and N. El-Sheimy, "Surrounding moving obstacle detection for autonomous driving using stereo vision," *International Journal of Advanced Robotic Systems*, vol. 10, no. 6, p. 261, 2013.

[5] N. Appiah and N. Bandaru, "Obstacle detection using stereo vision for self-driving cars," in *Not Found*, 2015.

[6] L. Matthies, R. Brockers, Y. Kuwata, and S. Weiss, "Stereo vision-based obstacle avoidance for micro air vehicles using disparity space," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3242–3249.

[7] X. Zhang, Y. Song, Y. Yang, and H. Pan, "Stereo vision based autonomous robot calibration," *Robotics and Autonomous Systems*, vol. 93, pp. 43–51, 2017.

[8] C. Del, S. Skaar, A. Cardenas, and L. Fehr, "A sonar approach to obstacle detection for a vision-based autonomous wheelchair," *Robotics and Autonomous Systems*, vol. 54, no. 12, pp. 967–981, 2006.

[9] P. Fleischmann and K. Berns, "A stereo vision based obstacle detection system for agricultural applications," in *Field and Service Robotics*. Springer, 2016, pp. 217–231.

[10] A. Asvadi, C. Premebida, P. Peixoto, and U. Nunes, "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*, vol. 83, pp. 299–311, 2016.

[11] A. Lay-Ekuakille, P. Vergallo, D. Saracino, and A. Trotta, "Optimizing and post processing of a smart beamformer for obstacle retrieval," *IEEE Sensors Journal*, vol. 12, no. 5, pp. 1294–1299, 2012.

[12] D. Sinha and F. Feroz, "Obstacle detection on railway tracks using vibration sensors and signal filtering using bayesian analysis," *IEEE Sensors Journal*, vol. 16, no. 3, pp. 642–649, 2016.

[13] C. Diego, A. Jiménez, Á. Hernández, C. J. Martín-Arguedas, and C. G. Fernández, "Improved ultrasonic phased array based on encoded transmissions for obstacle detection," *IEEE Sensors Journal*, vol. 15, no. 2, pp. 827–835, 2015.

[14] M. K. Habib, "Fiber-grating-based vision system for real-time tracking, monitoring, and obstacle detection," *IEEE Sensors Journal*, vol. 7, no. 1, pp. 105–121, 2007.

[15] J. Hernandez-Aceituno, R. Arnay, J. Toledo, and L. Acosta, "Using kinect on an autonomous vehicle for outdoors obstacle detection," *IEEE Sensors Journal*, vol. 16, no. 10, pp. 3603–3610, 2016.

[16] K. Yamaguchi, T. Kato, and Y. Ninomiya, "Moving obstacle detection using monocular vision," in *Intelligent Vehicles Symposium, 2006 IEEE*. IEEE, 2006, pp. 288–293.

[17] N. Appiah and N. Bandaru, "Obstacle detection using stereo vision for self-driving cars," 2011.

[18] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, "Pedestrian detection at 100 frames per second," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2903–2910.

[19] D. Pfeiffer and U. Franke, "Efficient representation of traffic scenes by means of dynamic stixels," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 217–224.

[20] N. Morales, A. Morell, J. Toledo, and L. Acosta, "Fast object motion estimation based on dynamic stixels," *Sensors*, vol. 16, no. 8, p. 1182, 2016.

[21] B. Wang, S. A. R. Florez, and V. Frémont, "Multiple obstacle detection and tracking using stereo vision: application and analysis," in *Control Automation Robotics & Vision (ICARCV), 2014 13th International Conference on*. IEEE, 2014, pp. 1074–1079.

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

[23] M. Duguleana, F. G. Barbuceanu, A. Teirelbar, and G. Mogan, "Obstacle avoidance of redundant manipulators using neural networks based reinforcement learning," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 132–146, 2012.

[24] Y. Bengio, Y. LeCun *et al.*, "Scaling learning algorithms towards ai," *Large-scale kernel machines*, vol. 34, no. 5, pp. 1–41, 2007.

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

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

[27] G. E. Hinton, "Learning multiple layers of representation," *Trends in cognitive sciences*, vol. 11, no. 10, pp. 428–434, 2007.

[28] Y. Bengio *et al.*, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

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

[30] Z. Hu and K. Uchimura, "UV-disparity: an efficient algorithm for stereovision based scene analysis," in *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*. IEEE, 2005, pp. 48–54.

[31] Q. P. He and J. Wang, "Fault detection using the k-nearest neighbor rule for semiconductor manufacturing processes," *IEEE Transactions on Semiconductor Manufacturing*, vol. 20, no. 4, pp. 345–354, 2007.

[32] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip *et al.*, "Top 10 algorithms in data mining," *Knowledge and information systems*, vol. 14, no. 1, pp. 1–37, 2008.

[33] S. M. Omohundro, *Five balltree construction algorithms*. International Computer Science Institute Berkeley, 1989.

[34] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[35] D. Comaneci and P. M. M. Shift, "A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, 2002.

[36] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[37] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: an efficient data clustering method for very large databases," in *ACM Sigmod Record*, vol. 25, no. 2. ACM, 1996, pp. 103–114.

[38] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[39] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315, no. 5814, pp. 972–976, 2007.

[40] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

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

[42] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a

deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[43] A. Krizhevsky and G. E. Hinton, "Using very deep autoencoders for content-based image retrieval." in *ESANN*, 2011.

[44] X. Lu, Y. Tsao, S. Matsuda, and C. Hori, "Speech enhancement based on deep denoising autoencoder." in *Interspeech*, 2013, pp. 436–440.

[45] S. C. AP, S. Lauly, H. Larochelle, M. Khapra, B. Ravindran, V. C. Raykar, and A. Saha, "An autoencoder approach to learning bilingual word representations," in *Advances in Neural Information Processing Systems*, 2014, pp. 1853–1861.

[46] H.-C. Shin, M. R. Orton, D. J. Collins, S. J. Doran, and M. O. Leach, "Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1930–1943, 2013.

[47] C. Georgoulas, L. Kotoulas, G. C. Sirakoulis, I. Andreadis, and A. Gasteratos, "Real-time disparity map computation module," *Microprocessors and Microsystems*, vol. 32, no. 3, pp. 159–170, 2008.

[48] Z. Hu and K. Uchimura, "Uv-disparity: an efficient algorithm for stereovision based scene analysis," in *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*. IEEE, 2005, pp. 48–54.

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

[50] J.-L. Blanco, F.-A. Moreno, and J. Gonzlez-Jimnez, "The mlaga urban dataset: High-rate stereo and lidars in a realistic urban scenario," *International Journal of Robotics Research*, vol. 33, no. 2, pp. 207–214, 2014. [Online]. Available: http://www.mrpt.org/MalagaUrbanDataset

[51] T. Scharwächter, M. Enzweiler, U. Franke, and S. Roth, "Efficient multi-cue scene segmentation," in *German Conference on Pattern Recognition*. Springer, 2013, pp. 435–445.

[52] ——, "Stixmantics: A medium-level model for real-time semantic scene understanding," in *European Conference on Computer Vision*. Springer, 2014, pp. 533–548.

[53] A. Ess, B. Leibe, K. Schindler, and L. Van Gool, "Moving obstacle detection in highly dynamic scenes," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 56–63.

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[55] W. Wang, S. Chen, and G. Qu, "Incident detection algorithm based on partial least squares regression," *Transportation Research Part C: Emerging Technologies*, vol. 16, no. 1, pp. 54–70, 2008.