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/TIP.2018.2813093, IEEE Transactions on Image Processing

Automatic Depth Extraction from 2D Images using a Cluster-based Learning Framework

José L. Herrera, Carlos R. del Blanco, and Narciso García,

***Abstract*—There has been a significant increase in the availability of 3D players and displays in the last years. Nonetheless, the amount of 3D content has not experimented an increment of such magnitude. To alleviate this problem, many algorithms for converting images and video from 2D to 3D have been proposed. Here, we present an automatic learning-based 2D-3D image conversion approach, based on the key hypothesis that color images with similar structure likely present a similar depth structure. The presented algorithm estimates the depth of a color query image using the prior knowledge provided by a repository of color + depth images. The algorithm clusters this database attending to their structural similarity, and then creates a representative of each color-depth image cluster that will be used as prior depth map. The selection of the appropriate prior depth map corresponding to one given color query image is accomplished by comparing the structural similarity in the color domain between the query image and the database. The comparison is based on a K-Nearest Neighbor framework that uses a learning procedure to build an adaptive combination of image feature descriptors. The best correspondences determine the cluster, and in turn the associated prior depth map. Finally, this prior estimation is enhanced through a segmentation-guided filtering that obtains the final depth map estimation. This approach has been tested using two publicly available databases, and compared with several state of the art algorithms in order to prove its efficiency.**

***Index Terms*—Index Terms— depth extraction, 2D-to-3D conversion, machine learning, depth maps, clustering.**

I. Introduction

THE availability of displays and players with 3D capabilities, such as TVs, DVD/Blu-Ray players, smartphones, video game consoles, cinemas, or projectors, has significantly risen in the last years. However, the amount of 3D content, such as images, movies, or TV broadcastings, is still scarce. Although this gap between the number of 3D players/displays and the volume of 3D content could be compensated in the future, nowadays the existing solution is to convert the current 2D content into 3D using manual and automatic methods.

This 2D-to-3D conversion process is typically performed in two main steps [1]. The first one is the depth estimation from a given monocular image, and the second one is the Depth-Image-Based Rendering (DIBR) of a new image or images to form a stereo pair, or a multi-view set of images. For this rendering step, there exists many algorithm that generate good quality results, while the depth estimation from single images is still a more challenging process. For this reason, this paper focuses on inferring the depth information from 2D images.

There are two main approaches to 2D-to-3D image and video conversion depending on whether a human operator is needed or not. In the denominated semi-automatic methods, the intervention of a human operator in the process is needed to assign depth to different parts of the scene, creating a sparse depth map. Then, the sparse map is processed to create a dense depth map over the whole image or video. Alternatively, the human operator can assign a global variation to the whole scene to define a depth prior, which is then refined to make the different objects of the scene appear in the final depth map. The importance of the human actuation may vary from a small sketch that assigns depths to different regions of the scene up to an accurate delimitation of the objects in the scene in order to assign depth values to them [2].

In the case of automatic approaches, no human operation is needed to estimate the depth of the images or video sequences. An algorithm extracts the depth structure of the scene in an automatic way, representing an important saving in time and cost. For this purpose, many advanced depth extraction algorithms have been proposed that estimate the depth structure from defocus, motion, or shading. The main problem of these methods is that they cannot be universally applied to all the images that compose a movie (or generic video content) [1]. For example, structure from motion algorithms need that the camera is in motion. On the other hand, there are simple approaches that rely on different heuristic assumptions to provide a real-time 2D-to-3D conversion, which are even already embedded in many 3D commercial devices, such as TVs, video game consoles, or DVD/Blu-Ray players. However, they only provides a good conversion in some simple and restricted scenarios. Recently, a new family of methods based on a machine-learning approach have appeared, which automatically estimate the depth of an image or video sequence by processing a repository of color + depth images (RGBD) that is used as prior knowledge. They are based on the key hypothesis that color images with similar structure will likely present a similar depth structure [2]. These algorithms can be successfully applied to situations where others failed, such as scenes with static cameras.

In this sense, the proposed automatic 2D-3D image conversion algorithm belongs to this recent family of algorithms that adopts a machine learning philosophy, and consequently does not require a human intervention. The algorithm computes the depth map of a color image by means of a learning framework that uses a color + depth image database. First, an off-line stage processes the color + depth database to infer a set of representative depth scenarios, which will be used as prior

This work has been partially supported by the Ministerio de Economía, Industria y Competitividad (AEI/FEDER) of the Spanish Government under projects TEC2013-48453 (MR-UHDTV) and TEC2016-75981 (IVME).

J.L. Herrera, C.R. del-Blanco, and N. García are with the Grupo de Tratamiento de Imágenes, Universidad Politécnica de Madrid, Madrid, 28040, Spain (emails:{jhc,cda,narciso}@gti.ssr.upm.es).

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

depth maps. This is accomplished by dividing the database into clusters using a structural similarity criterion in the color domain. The depth images relative to the color ones in each obtained cluster are combined to create a cluster representative in the depth domain (a prior depth map). Then, an on-line stage estimates the depth map of a query color image using the computed prior depth maps in the off-line stage. The best prior depth map for every color query image is selected by finding the cluster whose color images are more similar to the query image using a K-Nearest Neighbor framework. The similarity criterion is based on distances between features vectors that encode the color structure of the query and database color images. For this purpose, a learning procedure is adopted to adaptively combine a pool of image feature descriptors. Finally, this prior estimation is refined through a segmentation-guided filtering that enhances the depth value transitions among different objects in the scene, assisted by a high level segmentation in the color image. As a result, an enhanced depth map of the color query image is obtained. This algorithm has been tested in two public and widely-used RGBD databases, along with other state-of-the-art algorithms to prove its performance.

The paper is organized according to the following structure. In Section II, a review of the state of the art in the field of 2D-to-3D image conversion is exposed. Then, in Section III, we describe the general aspects of the proposed conversion algorithm. Next, in Section IV and V, the details of the offline and online processing stages are, respectively, explained. In Section VI, we show and discuss the obtained results, and finally conclusions are drawn in Section VII.

II. State of the art

There are two different strategies for 2D-to-3D image conversion algorithms. Semi-automatic methods require a human operator intervention in the conversion process, while automatic methods do not need such human interaction.

Semi-automatic methods need a significant human interaction for the conversion process to segment the different objects in the scene, to provide them with an appropriate depth. Nowadays, these methods represent the most successful strategy for 2D-to-3D conversion, and they are used by the industry to convert many 2D films to 3D [2]. Nevertheless, the fact that a human operator must interact during the conversion process makes these methods highly time consuming and costly. For example, a semi-automatic system was proposed by Guttmann et al. [3] to recover a dense depth map from sparse depth values assigned by the human operator via diffusion. Angot [4] proposed an approach where cross-bilateral filtering is applied to an initial depth map that is selected by the human operator from a library of initial depth maps. A similar approach was presented in [5], where the cross-bilateral filtering is replaced by an edge-based filtering. Also, based on [3], Phan et al. [6] presented a simplified and more efficient conversion algorithm by using scale-space random walks and a graph cut strategy. Liao et al. [7] reduced the human burden by computing first the optical flow, and then applying a structure from motion stage, saving the role of the human operator to correct errors and manually assign depth to undefined areas in the generated depth maps.

On the contrary, automatic methods do not need any human intervention to perform the 2D-to-3D conversion. There are many classic approaches that have faced this problem attending to different cues, such as shape from shading [8] and visual saliency [9]. A combination of different cues (perspective geometry, defocus, and visual saliency) have been also used in [10]. All these methods rely on different assumptions, and therefore they only achieve satisfactory depth maps for specific situations. Alternatively, some approaches try to recover the depth of the scene from different non-simultaneous views of the same scene using structure from motion algorithms [11]. Depth-from defocus algorithm achieves the same goal by modifying the focal length of the camera [12].

In the last years, several machine learning-based algorithms have appeared as an alternative to the classical 2D-to-3D conversion methods. The main hypothesis behind this family of methods is that images with a high photometric similarity will likely have a similar 3D structure. Therefore, using a color+depth image database, the depth map of a query color image can be inferred by first finding the color images in the database most similar to the query image, and finally by combining the depth images associated to the previous selection of color images. In this sense, Saxena et al. [13] [14] developed a learning approach for estimating the depth of a scene from a single image using a parsing technique and Markov Random Fields, which estimate both 3D locations and orientations. In [15] and [16], the inclusion of semantic labels and more elaborated models were considered to achieve better depth maps. Konrad et al. [17] presented a similar strategy, but transferring depth data instead of labels. Following this methodology, Karsch et al. [18] incorporated a depth optimization processing to assure the smoothness and consistency of a candidate depth map. Also, they extended the approach to work with videos. More recently, Konrad et al. [19][2] developed a computationally-efficient approach by avoiding the SIFT-flow based image alignment. Instead, the Histogram of Oriented Gradients (HOG) feature descriptor were used to directly find photometrically-similar images. Moreover, they applied a Cross-Bilateral Filtering to enhance the final depth map estimation. This way, they achieved to reduce the processing time in several orders of magnitude due to the discarding of the SIFT-flow alignment, which had a minimal impact on the final quality. Herrera et al. [20] employed Local Binary Patterns (LBP) features, and an adaptive strategy to select a variable number of similar images for the conversion process. All these methods perform an exhaustive search in the database to find the most similar images, which implies that the computational cost of such methods is proportional to the size of the used database. For that reason, this strategy could be impractical for huge databases. To alleviate this problem, a clustering-based hierarchical search strategy was introduced by Herrera et al. [21].

One common point of all the machine learning-based methods is the use of feature descriptors to have a compact representation of the images. However, each descriptor usually achieves the best performance for some specific types of

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

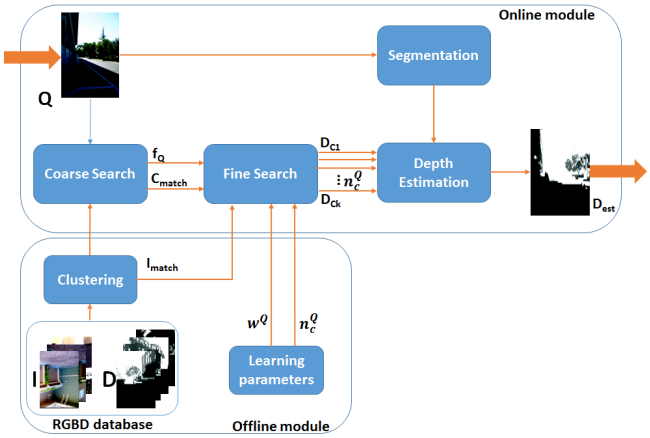


Fig. 1: Overall block diagram of the proposed 2D-to-3D conversion method.

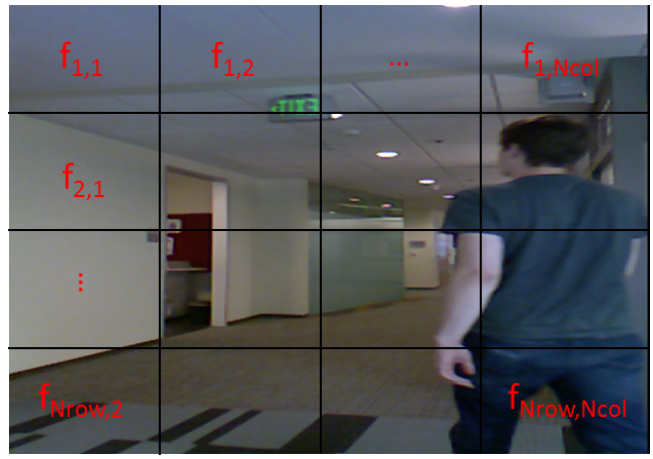


Fig. 2: The feature descriptors are computed in each image tile, and then concatenated to form a super-descriptor.

images, not existing a universal feature descriptor that is the best one for all images. In this sense, we propose in this paper to use a learned adaptive combination of previously used descriptors (HOG, LBP, GIST and SURF) to easily find the most similar images in the repository independently of the type of images. In addition, the optimal number of selected similar images for the conversion process is also learned by the algorithm in a training stage. Those innovations, along with the clustering of the database, improve the process of finding the images that are structurally similar, avoiding those that could distort our estimation. The clustering of the database also allows to work with large databases, while keeping the computational cost at a feasible level.

III. Algorithm Overview

The proposed automatic 2D-3D image conversion algorithm can be formulated as follows. Given a query color image Q, and a RGBD database DB, composed by color images and their corresponding depth maps, the goal is to estimate the depth map D_{est} of Q. The algorithm can be divided into two main modules: an offline processing, which adapts the DB and computes intermediate results for the posterior 2D-3D conversion process, and an online processing, which estimates the depth of each incoming query image Q. Fig. 1 shows the overall block diagram of the proposed algorithm, including the two main modules, and other sub-modules that are described in the following sections.

IV. Offline Processing

The offline processing module performs two tasks. First, the database DB is clustered to group the images that represent similar scenarios. And second, some critical parameters of the system are learnt from DB to achieve the best performance in the conversion process.

A. Database clustering

The division of the color images of the database DB into clusters is done using the k-means algorithm and the correlation function. For this purpose, the clustering is performed on a feature-based representation of the color images that captures the structural content. This feature-based representation is based on a combination of four state-of-the-art image descriptors: Histogram of Oriented Gradients (HOG) [22], Local Binary Patterns (LBP) [23], the GIST descriptor [24], and the Speeded-Up Robust Features (SURF). The exact procedure to obtain the feature descriptor that describes the color image structure is as follows. Every color image is divided into 4 x 4 tiles (as can be shown in Fig 2). Then, the previous four image descriptors are independently calculated for each tile, and stacked together by concatenating all the tile descriptors per type

$$\begin{aligned} f_{hog} &= [\text{hog}_{1,1}\ \text{hog}_{1,2} \ldots \text{hog}_{1,N_{col}} \ldots \text{hog}_{N_{row},N_{col}}], \\ f_{lbp} &= [\text{lbp}_{1,1}\ \text{lbp}_{1,2} \ldots \text{lbp}_{1,N_{col}} \ldots \text{lbp}_{N_{row},N_{col}}], \\ f_{gist} &= [\text{gist}_{1,1}\ \text{gist}_{1,2} \ldots \text{gist}_{1,N_{col}} \ldots \text{gist}_{N_{row},N_{col}}], \\ f_{surf} &= [\text{surf}_{1,1}\ \text{surf}_{1,2} \ldots \text{surf}_{1,N_{col}} \ldots \text{surf}_{N_{row},N_{col}}], \end{aligned} \tag{1}$$

where $N_{row} = 4$ and $N_{col} = 4$ are the number of rows and column, respectively, and, for example, $\text{hog}_{n,m}$ represents the HOG descriptor of the tile located in the n-th row and the m-th column. Finally, a super-descriptor is obtained by concatenating the previous resulting vectors as

$$f_{mix} = [f_{hog} \quad f_{lbp} \quad f_{gist} \quad f_{surf}]. \tag{2}$$

Additionally, a set of weights w are considered for each one of the four descriptors with the purpose of assigning different relevance to each one according to how well they capture the structure of the scene, since one descriptor can characterize better one image than the others, and therefore it should have more relevance in the feature-based representation.

Since every feature descriptor (HOG, LBP, GIST, and SURF) has different lengths, the super-descriptor f_{mix} is normalized to avoid that longer descriptors have more importance in the clustering process. The normalization process is performed using the mean $\mu_{f_{mix}}$ and the standard deviation

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

$\sigma_{f_{mix}}$ of the set of f_{mix} obtained from the color images in DB as

$$f_{mix}^{norm} = \frac{f_{mix} - \mu_{f_{mix}}}{\sigma_{f_{mix}}}. \quad (3)$$

Once the clustering is performed, the cluster centroids are stored as feature vector representatives of the set of color images that belong to every cluster. These will be used in the on-line processing module to speed up the K-Nearest Neighbor strategy used to find the most similar color image in DB to the query image.

The number of clusters $N_{cluster}$ is a critical parameter in the algorithm for two reasons. First, it determines the number of different types of considered scenarios and has implications on the quality and computational cost. A low value of $N_{cluster}$ can lead to clusters that have images from very different scenarios (i.e. their structural similarity is low), and therefore useless for a proper depth estimation. On the other hand, a large number of $N_{cluster}$ potentially generates clusters with images that cover a wider range of different scenarios, but at the expense of increasing the computational cost of the posterior on-line processing module.

B. Parameter learning

Regarding the second task of the off-line processing module, a learning strategy is carried out to find the optimal values of the weights w related to the different descriptors, and the number of candidate images n_c needed to generate a depth prior estimation D_{prior}. Fig. 3 shows an outline of the procedure. It starts by computing the feature descriptor f_{mix} for all the N_{Im} color images in DB. Then, the best parameters w and n_c are estimated from the descriptors and the depth images in DB by means of a Leave One Out learning strategy. As a result, a feature descriptor f_{mix}^{test}, a depth map D^{test}, the descriptors of the rest of images f_{mix}^{train}, and the rest of depth images D^{train} are obtained.

To find the optimum values for the weights of the descriptors w_{opt}, we compute the similarity between f_{mix}^{test} and each descriptor from the set f_{mix}^{train}. This similarity is calculated as a weighted correlation between them as

$$S_f = \sum_{i=1}^{4} w(i)\text{corr}(f_{mix}^{test}, f_{mix}^{train}). \quad (4)$$

For each combination of weights, all similarity values are sorted in descend order creating a vector of similarities S_{im}^{sorted}. The depth maps in D^{train} are accordingly sorted as D^{sorted}, and then they are fused to create $N_{Im}-1$ depth maps D^{fusion}, where the $i-th$ D^{fusion} is a weighted combination of the $i-th$ first depth maps in D^{sorted}. The weights are computed as the similarities of the feature descriptors S as

$$D^{fusion}(n) = \frac{\sum_{i=1}^{n} (S_{im}^{sorted}(i) D^{sorted}(i))}{\sum_{i=1}^{n} S_{im}^{sorted}(i)}, \quad (5)$$

where $D^{fusion}(n)$ is the $n-th$ built depth map, $S_{im}^{sorted}(i)$ is the $i-th$ similarity value (computed as the correlation), and $D^{sorted}(i)$ is the $i-th$ most similar depth map to D^{test}. Once

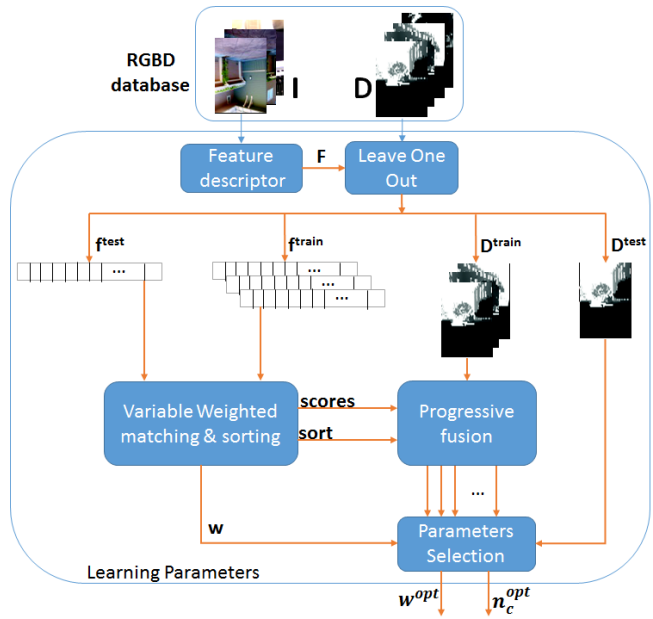


Fig. 3: Block diagram of the module: Learning Parameters.

all D^{fusion} have been built, the correlation between them, and D^{test} is computed as

$$S_d(n) = \text{corr}(D^{test}, D^{sorted}(n)). \quad (6)$$

This process is repeated for every combination of values of w. Then, the highest score for S_d is searched and the particular values of w and n_c that produced this score are taken as the optimum values (w^{opt} and n_c^{opt} respectively) for the test image represented by f_{mix}^{test}. The whole procedure is repeated until for each image in DB, a set of values for w^{opt} and n_c^{opt} is assigned.

V. Online module

The online processing of the algorithm can be divided into three main steps. The two first steps performs a hierarchical search to find similar images in DB to Q by doing a coarse search and then a fine search. The coarse search is performed to find the pre-computed cluster C_{match} in DB that matches Q the best, using the correlation as a similarity metric. In the second step, a fine search is performed by computing the similarity between Q and all images inside C_{match}. Then, the n_c most similar images are selected. For this calculation, a weighted correlation is considered, using as a weighting value the one estimated in the training step. The value n_c of the selected images is also determined by the results of the training step.

The final part of the algorithm is the fusion of the depth maps and the filtering of the result to get the final depth map estimation. With this purpose, a depth prior D_{prior} is first computed by fusing the depth maps associated to the selected images D_c. Then, a segmentation-based filtering is applied to D_{prior} to enhance the edges of the final depth estimation D_{est}.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

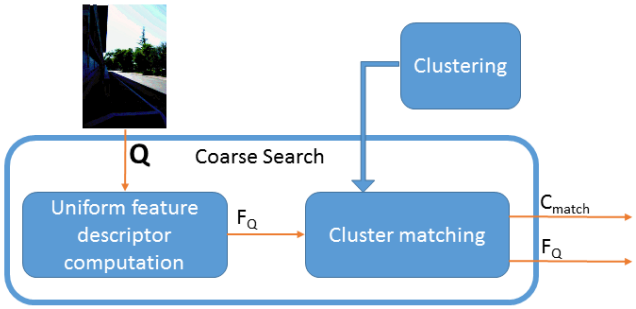


Fig. 4: Block diagram of the coarse search part of the algorithm.

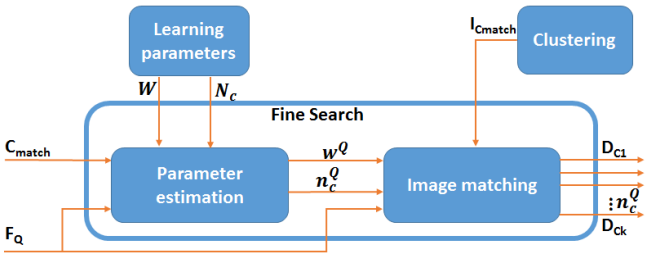


Fig. 5: Block diagram of the fine search part of the algorithm.

A. Coarse Search

The first step of the online part of the algorithm is a coarse search to find the clusters of images (in which the database was divided in the training stage) that best match the query image Q. The diagram for this part of the algorithm can be seen in Fig 4.

For this purpose, the combination of feature descriptors of Q, f_{mix_Q}, is computed (see Section IV), and compared with every cluster centroid f_{mix_c}, using the correlation metric as

$$S_{clu} = \text{corr}(f_{mix_Q}, f_{mix_c}). \tag{7}$$

As a result, the m nearest clusters C_{match} are selected. This value of the parameter m is a tradeoff between computational cost and accuracy in the search. While high values of m guarantee that the closest descriptors in the database are found, it also increases the computational cost. Low values of m speed-up the search, but does not ensure that the closest components are found.

B. Fine Search

The block diagram of the fine search part of the algorithm is shown in Fig 5. The first step is the estimation of the best parameters w^Q and k^Q for the query image Q, by computing the correlation of the query descriptor f_{mix_Q} and the set of descriptors $f_{mix_{im}}$ related to the images in C_{match} as

$$S_{im} = \text{corr}(f_{mix_Q}, f_{mix_{im}}). \tag{8}$$

Then, using the similarity scores S_{im} and the parameters w^{opt} and n_c^{opt} learned in the offline module, the values w^Q and n_c^Q are computed as as weighted average

$$\begin{aligned} w^Q &= \frac{\sum_i (S_{im}(i) w^{opt}(i))}{\sum_i S_{im}(i)}, \\ n_c^Q &= \frac{\sum_i (S_{im}(i) n_c^{opt}(i))}{\sum_i S_{im}(i)}. \end{aligned} \tag{9}$$

Next, an exhaustive search is performed over the images in C_{match} to find those n_c^Q images that best match Q. This search is applied using correlation as the similarity metric and weighting the descriptors by w^Q

$$\begin{aligned} S_{descr} &= \text{corr}(descr_Q, descr_{im}) \\ WS &= \sum_{descr} w_{descr} S_{descr}, \end{aligned} \tag{10}$$

where $descr$ is f_{hog}, f_{lbp}, f_{gist} or f_{surf} correspondingly, and S_{descr} is the similarity value between the feature descriptor of the query image f_{mix} and the descriptors of the images in C_{match}. Finally, WS is the weighted similarity using the set of weights w^Q. Fig 6 and 7 show some examples of the images selected by this search. As a result, the corresponding depth maps D_c to the selected n_c^Q images are obtained, which will be combined in the last part of the algorithm to estimate the final depth map.

C. Depth map estimation

The depth maps of the n_c^Q color images with highest similarity score are fused to get a depth prior D_{prior}. This fusion is performed by computing the weighted average of the selected depth maps as

$$D_{prior} = \frac{\sum_{i=1}^{n_c^Q} (WS(i) D_c(i))}{\sum_{i=1}^{k^Q} WS(i)}, \tag{11}$$

where D_c is the $i-th$ depth map selected to be fused, and $WS(i)$ is the weighted similarity score computed in Eq 10.

The obtained depth estimation is refined using a new smooth filtering technique that reduces noise and artifacts. The smooth filtering is guided by a hierarchical segmentation applied to every query color image. The main idea behind this filtering is to use the image structure of the query image to refine the depth estimation, since there exists a significant correspondence between many of the edges in the color image and in their associated depth images. Fig.9 shows an example of a depth image, the corresponding color image, and the resulting segmentation of the color image for a better understanding. As can be visually observed, a significant amount of the depth transitions are common to the color ones. It can be also noted that both the color image and the color segmentation include borders that do not correspond to depth transitions, because of the different reflectance patterns that an object can have at similar depth values. Two measures can provide an insight of this mismatch. On the one hand, the correlation coefficient between the thresholded edge maps of the depth and color images gives a value of 0.4964. On the other hand, the percentage of depth edge pixels that are also color edge pixels gives a value of 68.82%. Although a significant amount of edges are in common, neither one measure, nor the other,

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

Fig. 6: From left to right: query image Q and the three most similar images to Q sorted by similarity for two examples of NYU dataset.

Fig. 7: From left to right: query image Q and the three most similar images to Q sorted by similarity for two examples of Make3D dataset.

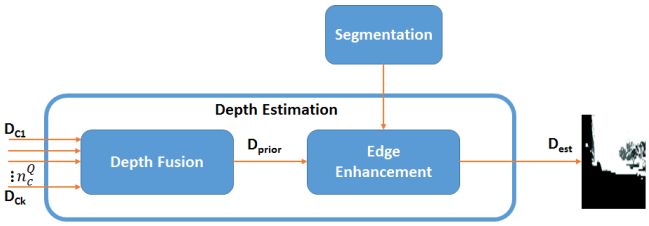


Fig. 8: Block diagram of the depth estimation part of the algorithm.

give a maximum value because of the additional edges found in the color image. However, the proposed smooth filtering algorithm handles this situation seamlessly.

Let's assume that the color image is segmented. Theoretically, the depth value of the regions corresponding to each region of the segmentation/partition would be coarsely the same. Then, an adaptive average filtering is applied over the depth estimation, using a uniform kernel with a custom spatial geometry defined according to the shape of every color segmented region. In other words, for every segmented region in the color image, an average filter is applied to only the pixels that belong to the region that is being filtered. The active pixels for each region are defined as:

$$s(x,y) = \begin{cases} 1 & \text{if } (x,y) \in R_s \\ 0 & \text{if } (x,y) \notin R_s \end{cases}, \tag{12}$$

where R_s is the 2D segmented region that contains the spatial coordinates of the depth pixel to be filtered.

On the other hand, an accurate segmentation of a color image is quite challenging. However, the proposed smooth strategy can perfectly work with over-segmented images (with more partitions that a human user could determine). Consider an object that is located at the same depth but it is split into

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

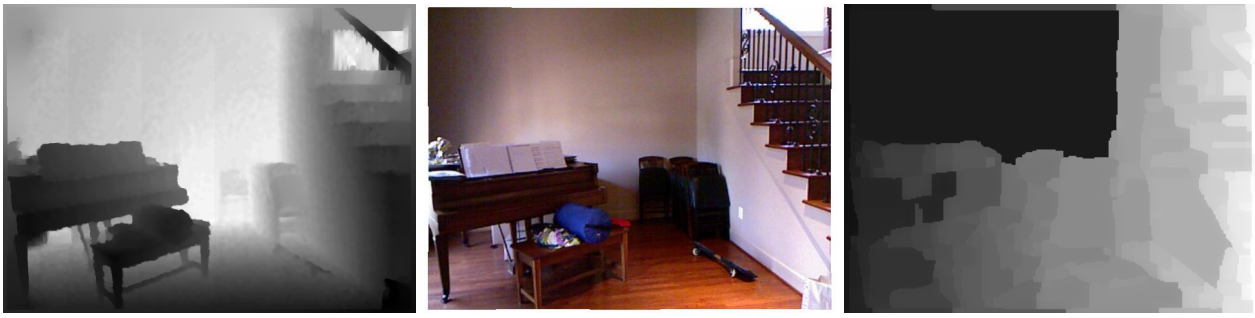


Fig. 9: From left to right: Query depth, Query color image, and segmentation of the query image. A significant correspondence between many of the edges in the color image and in its associated depth image can be observed.

two or more regions in the color segmented process. Then, the smoothing effect of those adaptive filters derived from those segmented regions will be very similar, since all of them will average similar depth values. On the contrary, an under-segmentation would have potentially very negative effects, since the adaptive filters would average very different depth values belonging to different objects. For the purpose of obtaining an over-segmentation, the algorithm presented in [25] has been used, which produces a hierarchical image segmentation that allows to control the degree of over-segmentation. This algorithm computes first the image contours by combining multiple local cues using a spectral clustering framework. And then, it performs the image segmentation by converting the computed contours into a hierarchical region tree.

VI. Results

The presented approach has been tested in two different databases: the Make3D database [14] and the NYU dataset [26]. The Make3D database is composed by 534 outdoor images with their corresponding depths maps, which have been acquired by a laser range finder. This database is divided into a test subset of 134 images and a train subset of 400 images. The resolution of the color and depth images are 1704 x 2272, and 55 x 305 pixels, respectively. On the other hand, the NYU database is formed by 1449 color images and their associated depth maps, captured by the Kinect sensor in indoor scenarios. The resolution of both color and depth images is 640 x 480 pixels. All color and depth images from both databases have been resized to 320 x 240 pixels for a straightforward comparison with the previous works. Additionally, both databases have been joined to create a third database that is used to test the behavior of our algorithm in hybrid databases containing indoor and outdoor situations.

As discussed in Sect. IV-B, the number of clusters $N_{cluster}$ is an important parameter that requires a tradeoff between accuracy and computational cost. In Fig 10, we can see the variation of the search time as a function of the parameter $N_{cluster}$. Each line represent a determined database configuration: Make3D composed by 400 images, NYU composed by 1449 images, and the database resulting from the union of the previous ones (1983 images). Every line shows the total search time (coarse and fine search) as a function of the number of clusters $N_{cluster}$ in which the database is divided. The values have been normalized by the search time corresponding to one cluster. This is virtually the same condition as not considering any cluster strategy (the search is accomplished by exhaustively checking all images). As can be observed, the search time decreases fast, and remains stable between the 10% and 20% of the search time corresponding to one cluster. Those time reduction percentages are obtained for values of $N_{cluster}$ between 25 and 100 clusters. Consequently, the clustering strategy would allow to use large databases that would be infeasible with an exhaustive search approach. The figure also shows that the time search for the Make3D dataset starts to rise slowly when the number of cluster is higher and higher. This can be explained by the fact that a hierarchical search performed over only one cluster (there is an exhaustive search inside the cluster, which contains all the images for only one cluster) is equivalent to a hierarchical search performed over a number of clusters equal to the total number of images in the database (the search is accomplished for all the clusters, whose number is the same as the total number of images). This increment of time for large numbers of clusters is not appreciated in the figure corresponding to the NYU dataset and the combined dataset, since they contain a larger number of images, and therefore the increment of time is produced for higher numbers of clusters than those represented in the figure. The figure also suggests that the benefit provided by the clustering and the hierarchical search would be also more significant for larger databases (in comparison with the exhaustive search strategy).

To evaluate the quality of the presented approach, we have trained our algorithm with each of the databases using a leave-one-out methodology. This selects one image+depth pair from the database, using the color image as the query image Q and the depth image as the ground truth D_Q, and leaving the other pairs for training. The quality of the generated depth map D_{est} is measured using the correlation coefficient (C), the Peak Signal to Noise Ratio ($PSNR$), the Structural Similarity ($SSIM$) [27], the depth error $|\log(D_Q) - \log(D_{est})|$ ($\log_{10}$), and the average relative depth error $\frac{|D_Q - D_{est}|}{D_Q}$ (REL). The correlation coefficient C is defined as:

$$C = \frac{\sum_i (D_{est}[i] - \mu_{D_{est}})(D_Q[i] - \mu_{D_Q})}{N_p \sigma_{D_{est}} \sigma_{D_Q}}, \qquad (13)$$

where N_p is the number of pixels in D_{est} and D_Q; $\mu_{D_{est}}$

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

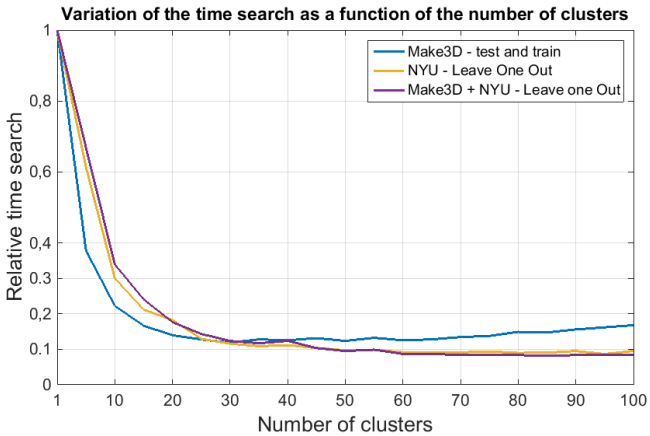


Fig. 10: Search time as a function of the number of clusters $N_{cluster}$ in which the database is divided. Time values are normalized to the search time corresponding to $N_{cluster} = 1$.

and μ_{DQ} are the empirical mean values of D_{est} and D_Q, respectively; and $\sigma_{D_{est}}$ and σ_{d_Q} are the corresponding empirical standard deviations. The correlation coefficient C takes values from -1 to +1 (values close to +1 indicate that the depth maps are very similar, and values close to -1 suggest they are complementary). The PSNR is mathematically expressed by

$$RMSE = \sqrt{\sum_i \left(D_Q(i) - D_{est}(i)\right)^2 / N_p}, \qquad PSNR = 20 \log_{10} \frac{R(D_Q)}{RMSE}, \tag{14}$$

where $R(x)$ is the maximum dynamic range of the image type. On the other hand, SSIM is an objective metrics that takes into account some perceptual assessment. In all situations, the higher the C, PSNR, and SSIM values are, the higher quality has the estimation.

First, examples of the depth map estimations using our algorithm for the considered datasets are shown in Figs. 11 to 14. From left to right, the images included in every figure are: query image Q, depth prior D_{prior}, final depth estimation D_{est}, and query depth ground truth D_Q. Figs. 11 and 12 show some good and bad examples, respectively, of conversion for the Make3D database, and Figs. 13 and 14 for the NYU database. As can be observed, the depth prior represents a global estimation that coarsely follows the general behaviour of the ground truth depth map. Then, the segmentation-based filtering enhances the edges/shapes of main objects in the scene, obtaining a final depth estimation that is consistent with the query image, and perceptually correlated with its depth. Potential reasons for not achieving a good quality conversion are due to problems with the segmentation. For example, an over-segmentation is shown in Fig. 14. Other reasons are related to problems with lights and shadows.

The results of our approach have been compared with several works of the state of the art. Tab. I shows the averaged computational time for the 2D-to-3D conversion for different algorithms in the state-of-the-art for the databases: Make3D, NYU, and the combination of this two previous datasets.

TABLE I: Evaluation of the computational cost for state-of-the-art for the Make3D, NYU, and the combination of this two previous datasets

Make3D [14] and NYU [26]			
Time(s)/image	Make3D	NYU	Make3D-NYU
Depth Transfer [18]	92.67	98.65	101.39
HOG learning based [2]	0.35	0.56	0.69
Adaptive LBP based [20]	0.38	0.58	0.73
(Ours)	0.58	0.62	0.67

TABLE II: Evaluation of the state-of-the-art algorithms using different metrics in the Make3D database with a test subset of 134 images and a train subset of 400 images. The results are the average over the 134 test images. NR indicates Not Reported

Make3D [14] - 134 Test, 400 Train					
Algorithm	C	PSNR	SSIM	$\log_{10}$	REL
Depth Transfer [18]	0.69	14.56	0.84	0.148	0.362
HOG learning based [2]	0.61	13.44	0.79	0.187	0.432
Adaptive LBP based [20]	0.66	14.06	0.77	0.156	0.384
kNN Learning based [28]	0.48	NR	NR	NR	NR
SCN [29]	NR	NR	NR	0.198	0.530
HEH [30]	NR	NR	NR	0.320	1.423
Pointwise MRF [31]	NR	NR	NR	0.149	0.458
PP-MRF [31]	NR	NR	NR	0.187	0.370
Superpixel MRF [32]	NR	NR	NR	0.148	0.379
(Ours no filt.)	0.66	14.10	0.75	0.161	0.388
(Ours)	0.67	14.15	0.81	0.153	0.375

Time measurements have been obtained running a Matlab implementation of the algorithms in a Intel. Core $i7$-$4510U$ CPU @2.00GHz 2.6GHz. Regarding our implementation, a number of clusters equal to 25 has been used for the proposed algorithm. A noticeable reduction in time is achieved by the proposed algorithm in comparison with the other approaches. This reduction in time is more significant as the size of the database is larger and larger. Presumably, this fact will make possible to use huge datasets that theoretically will increment the quality conversion. Table II, III and IV show the quantitative results (averaged over all the test images) for the Make3D [14] database (in a train and test configuration of 400 training images and 134 testing images), the NYU database (in a Leave-One-Out configuration), and the database combination of both Make3D and NYU databases. As can be seen, the quality results are quite similar for all the approaches, although our proposal is more computationally efficient and scalable than the others. Even more, our algorithm is slightly better for some cases. Another interesting fact is that the segmentation-based post-filtering seems to be marginal for some metrics (C, PSNR, and SSIM metrics), although some improvement is observed for the other metrics. Visually, it appears to increase the smoothness of the depth map, as it was observed in Figs. 11 to 14, but more experiments using subjective metrics should be carried out for a proper assessment. However, the design of these experiments are quite challenging and costly, justifying by themselves a new research process.

Lastly, some qualitative results about the rendering of stereoscopic views using a query color image plus the estimated depth image are presented. For this purpose, the DIBR algorithm presented in [33] has been used. Additionally,

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

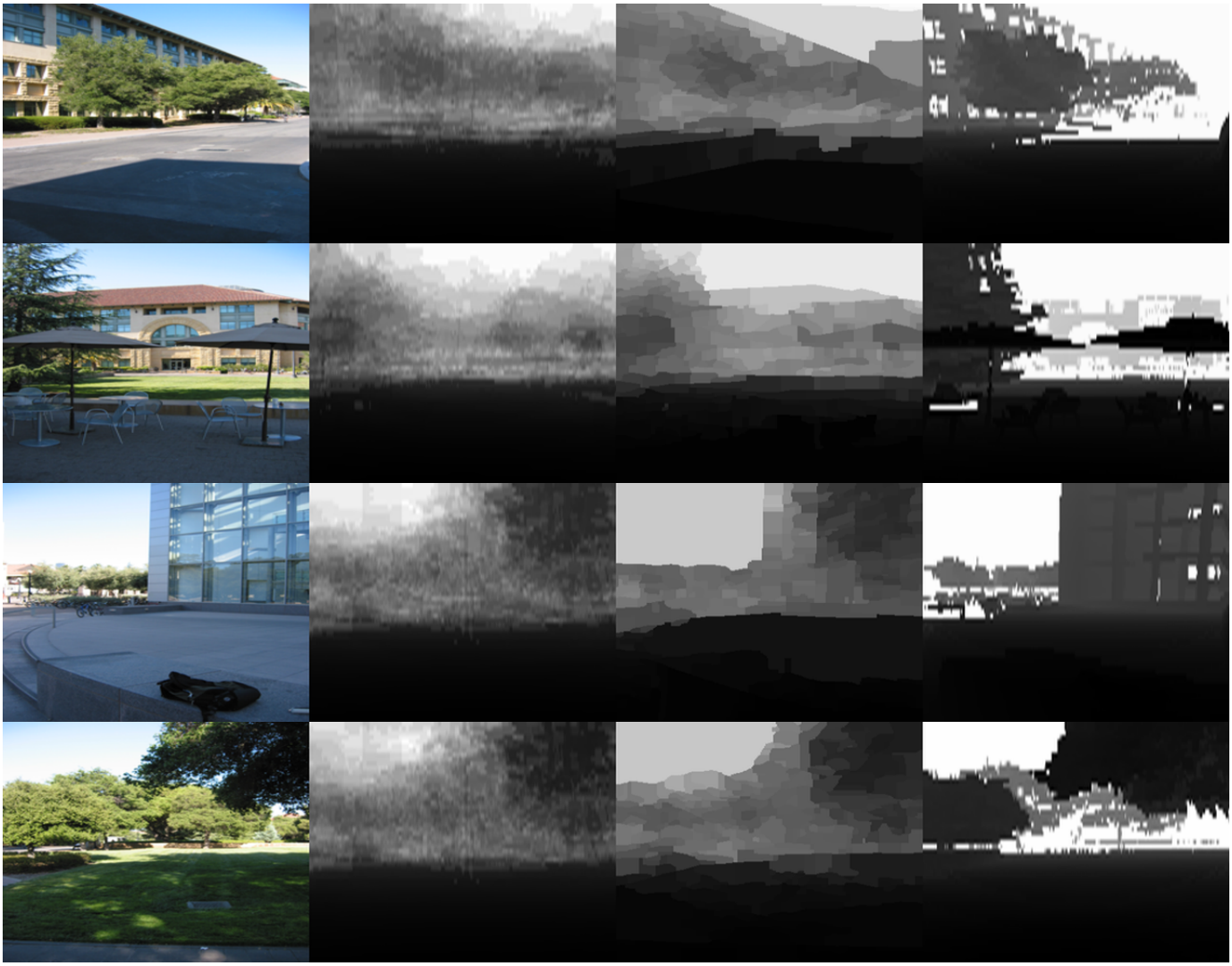


Fig. 11: From left to right: query image Q, depth prior D_{prior}, final depth estimation D_{est}, and depth ground truth D_Q for four examples of Make3D dataset.

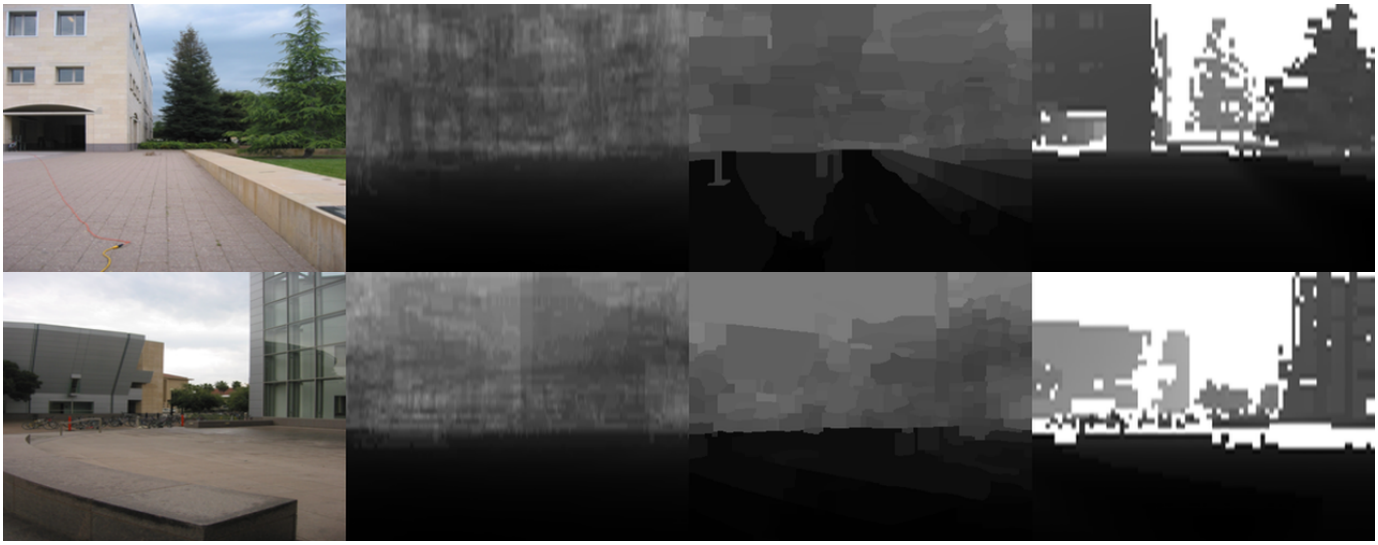


Fig. 12: From left to right: query image Q, depth prior D_{prior}, final depth estimation D_{est}, and depth ground truth D_Q for two bad examples of Make3D dataset.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

Fig. 13: From left to right: query image Q, depth prior D_{prior}, final depth estimation D_{est}, and depth ground truth D_Q for four examples of NYU dataset.

Fig. 14: From left to right: query image Q, depth prior D_{prior}, final depth estimation D_{est}, and depth ground truth D_Q for two bad examples of NYU dataset.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

TABLE III: Evaluation of state-of-the-art algorithms using different metrics in the NYU database in Leave One Out configuration. The results are the average over the 1449 images.

NYU [26] - Leave One Out					
Algorithm	C	PSNR	SSIM	$\log_{10}$	REL
Depth Transfer [18]	0.59	13.57	0.80	0.134	0.374
HOG learning based [2]	0.61	12.90	0.79	0.183	0.539
Adaptive LBP based [20]	0.63	13.74	0.81	0.153	0.422
(Ours no filt.)	0.63	13.90	0.78	0.140	0.407
(Ours)	0.63	13.86	0.80	0.137	0.352

TABLE IV: Evaluation of state-of-the-art algorithms using different metrics in the combination of Make3D and NYU databases in Leave One Out configuration. The results are the average over the 1983 images.

Make3D [14] and NYU [26]					
Algorithm	C	PSNR	SSIM	$\log_{10}$	REL
Depth Transfer [18]	0.60	12.48	0.76	0.196	0.559
HOG learning based [2]	0.60	13.58	0.79	0.245	1.01
Adaptive LBP based [20]	0.62	13.81	0.81	0.192	0.568
(Ours no filt.)	0.63	14.05	0.75	0.182	0.541
(Ours)	0.63	14.00	0.78	0.164	0.413

stereoscopic views using the depth ground truth are also shown to check more realistically the quality of the stereoscopic views from the estimated depth map, since there are several artifacts that are produced by the own limitations of the DIBR algorithm, and therefore they cannot be attributed to the estimated depth map. Fig. 15 shows four examples of rendered stereoscopic views in different rows. Every row contains (a) the left view rendered from the ground truth depth, (b) the left view rendered from the estimated depth, (c) the query color image, (d) the right view rendered from the estimated depth, and (e) the right view rendered from the ground truth depth, respectively.

VII. Conclusion

We have proposed a new 2D-to-3D image conversion algorithm based on machine learning techniques. The presented approach learns how to combine different feature descriptors, as well as the number of candidate images to use from the training dataset. The algorithm also clusters the database according to their similarity to find similar images in a more efficient way. The clustering and the learning phases are implemented offline. Then, when a query image arrive for the conversion process, a hierarchical search (using the estimated clusters) is performed to find those images in the training database that are the most similar to the query image. The depth prior is built by fusing the depth maps of the structurally similar images applying a weighted average, and this prior is refined with a segmentation-based filtering to improve the contour of the objects in the scene. The approach achieves similar or higher results to the best algorithms in the state of the art, outperforming them for the most challenging cases, such as the indoor scenes and for the combination of indoor and outdoor scenes. Also, the database clustering makes the algorithm feasible when large databases are used.

References

[1] S. Patil and P. Charles, "Review on 2D-to-3D image and video conversion methods," in *International Conference on Computing Communication Control and Automation*, Feb 2015, pp. 728–732.

[2] J. Konrad, M. Wang, P. Ishwar, C. Wu, and D. Mukherjee, "Learning-based, automatic 2D-to-3D image and video conversion," *IEEE Transaction on Image Processing*, vol. 22, no. 9, pp. 3485–3496, 2013.

[3] M. Guttmann, L. Wolf, and D. Cohen-Or, "Semi-automatic stereo extraction from video footage," in *International Conference on Computer Vision*, Sept 2009, pp. 136–142.

[4] L. J. Angot, W.-J. Huang, and K.-C. Liu, "A 2D to 3D video and image conversion technique based on a bilateral filter," in *Proc. SPIE*, vol. 7526, 2010, pp. 75 260D–75 260D–10.

[5] C.-C. Cheng, C.-T. Li, and L.-G. Chen, "A novel 2D-to-3D conversion system using edge information," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1739–1745, 2010.

[6] R. Phan, R. Rzeszutek, and D. Androutsos, "Semi-automatic 2D to 3D image conversion using scale-space random walks and a graph cuts based depth prior," in *IEEE International Conference on Image Processing*, Sept 2011, pp. 865–868.

[7] M. Liao, J. Gao, R. Yang, and M. Gong, "Video stereolization: Combining motion analysis with user interaction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 7, pp. 1079–1088, 2012.

[8] R. Zhang, P.-S. Tsai, J. Cryer, and M. Shah, "Shape-from-shading: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 8, pp. 690–706, 1999.

[9] C. Huang, Q. Liu, and S. Yu, "Regions of interest extraction from color image based on visual saliency," *The Journal of Supercomputing*, vol. 58, no. 1, pp. 20–33, 2011.

[10] P. Ji, L. Wang, D.-X. Li, and M. Zhang, "An automatic 2D to 3D conversion algorithm using multi-depth cues," in *International Conference on Audio, Language and Image Processing*, July 2012, pp. 546–550.

[11] R. Szeliski and P. Torr, "Geometrically constrained structure from motion: Points on planes," in *3D Structure from Multiple Images of Large-Scale Environments*, 1998, vol. 1506, pp. 171–186.

[12] M. Subbarao and G. Surya, "Depth from defocus: A spatial domain approach," *International Journal of Computer Vision*, vol. 13, no. 3, pp. 271–294, 1994.

[13] A. Saxena, H. C. Sung, and Y. N. Andrew, "Learning depth from single monocular images," in *Conference on Neural Information Processing Systems*, 2005, pp. 1–8.

[14] A. Saxena, M. Sun, and A. Ng, "Make3D: Learning 3D scene structure from a single still image," *IEEE Transactions on Pattern Recognition and Machine Intelligent*, vol. 31, no. 5, pp. 824–840, 2009.

[15] C. Liu, J. Yuen, and A. Torralba, "Nonparametric scene parsing: Label transfer via dense scene alignment," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 1972–1979.

[16] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 1253–1260.

[17] J. Konrad, G. Brown, M. Wang, P. Ishwar, C. Wu, and D. Mukherjee, "Automatic 2D-to-3D image conversion using 3D examples from the internet," *Proc. SPIE*, vol. 8288, pp. 82 880F–82 880F–12, 2012.

[18] K. Karsch, C. Liu, and S. Kang, "Depth extraction from video using non-parametric sampling," in *European Conference on Computer Vision*, vol. 7576, 2012, pp. 775–788.

[19] J. Konrad, M. Wang, and P. Ishwar, "2D-to-3D image conversion by learning depth from examples," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 16–22.

[20] J. Herrera, C. del Blanco, and N. Garcia, "Learning 3D structure from 2D images using LBP features," in *IEEE International Conference on Image Processing*, October 2014, pp. 2022–2025.

[21] ——, "Fast 2D to 3D conversion using a clustering-based hierarchical search in a machine learning framework," in *IEEE 3DTV-Conference*, July 2014, pp. 1–4.

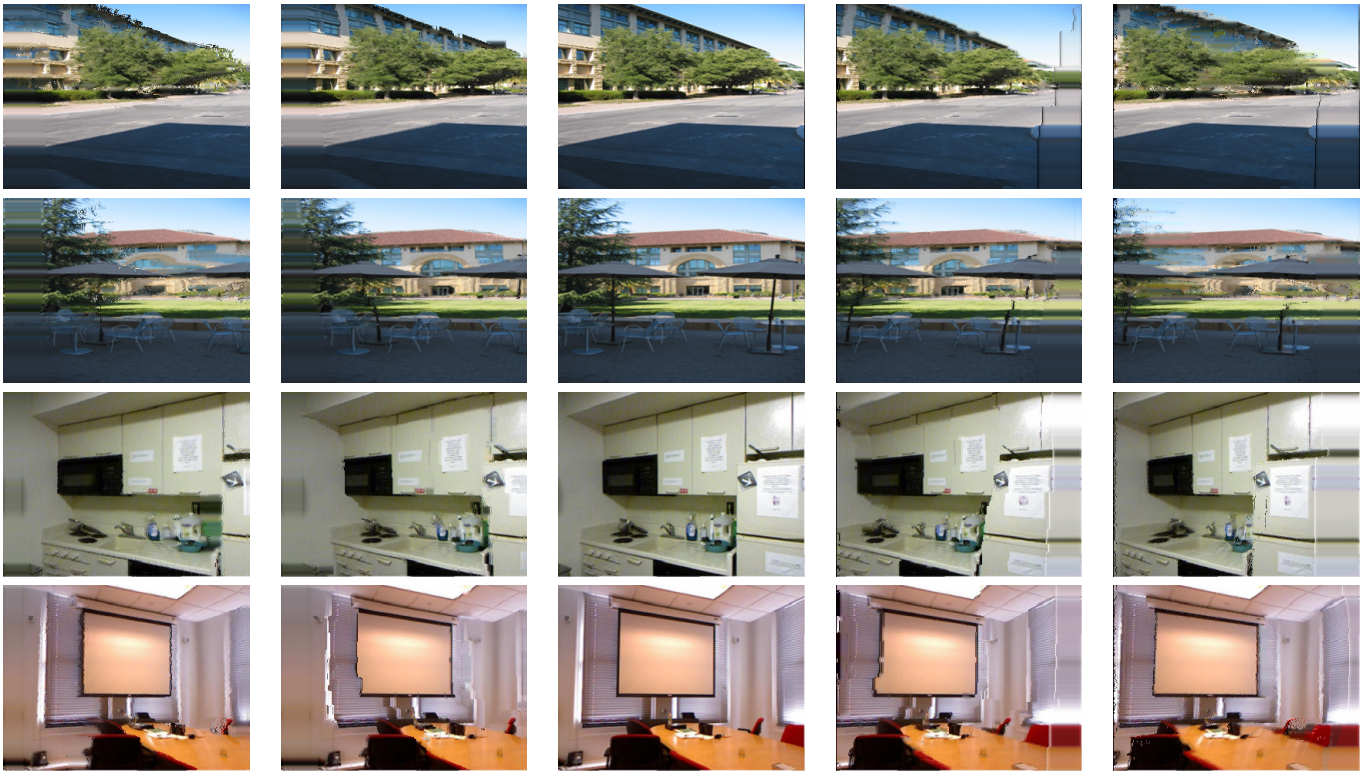
[22] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2005, pp. 886–893.

[23] K. Meena and A. Suruliandi, "Local binary patterns and its variants for face recognition," in *International Conference on Recent Trends in Information Technology*, June 2011, pp. 782–786.

[24] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing



(a) (b) (c) (d) (e)

Fig. 15: Four examples of rendered stereoscopic views. Every row contains the left rendered views from the ground truth depth (a) and the estimated depth (b), the query color image (c), and the right rendered views from the estimated depth (d) and the ground truth depth (e), respectively.

[25] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Recognition and Machine Intelligent*, vol. 33, no. 5, pp. 898–916, 2011.

[26] N. Silberman and R. Fergus, "Indoor scene segmentation using a structured light sensor," in *IEEE International Conference on Computer Vision Workshops*, Nov 2011, pp. 601–608.

[27] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transaction on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[28] J. Konrad, M. Wang, and P. Ishwar, "2D-to-3D image conversion by learning depth from examples," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 16–22.

[29] A. Saxena, S. H. Chung, and A. Y. Ng, "3-d depth reconstruction from a single still image," *International Journal of Computer Vision*, vol. 76, no. 1, pp. 53–69, 2008.

[30] D. Hoiem, A. A. Efros, and M. Hebert, "Geometric context from a single image," in *IEEE International Conference on Computer Vision*, 2005, pp. 654–661.

[31] A. Saxena, M. Sun, and A. Y. Ng, "Make3D: Learning 3D scene structure from a single still image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 824–840, 2009.

[32] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 1253–1260.

[33] D. V. S. X. D. Silva, W. A. C. Fernando, and H. K. Arachchi, "A new mode selection technique for coding depth maps of 3D video," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, March 2010, pp. 686–689.

José L. Herrera received the Telecommunication Engineering degree (integrated BSc-MSc accredited by ABET) from the Universidad Politécnica de Valencia (UPV), Spain in 2009, and the Technologies and Systems of Communications Master degree from the Universidad Politécnica de Madrid (UPM), Spain in 2013. He is currently a PhD student at the same University. Since 2011 he has been a member of the Image Processing Group of the UPM. His research interests are in the area of 3D image and video.

Carlos R. del Blanco received the Telecommunication Engineering degree and the Ph.D. degree in Telecommunication, both from the Universidad Politécnica de Madrid (UPM), in 2005 and 2011, respectively. Since 2005 he is a member of the Image Processing Group of the UPM. In addition, since 2011 he is a member of the faculty of the E.T.S. Ingenieros de Telecomunicacin as Assistant Professor of Signal Theory and Communications at the Department of Signals, Systems, and Communications. His professional interests include signal and image processing, computer vision, pattern recognition, and machine learning, and stochastic dynamic models.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

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/TIP.2018.2813093, IEEE Transactions on Image Processing

Narciso García received the Telecommunication Engineering degree and the Ph.D. degree in Telecommunication, both from the Universidad Politécnica de Madrid (UPM), in 1976 (Spanish National Graduation Award) and 1983 (Doctoral Graduation Award), respectively. Since 1977 he is a member of the faculty of the UPM, where is currently Professor of Signal Theory and Communications. He leads the Image Processing Group of the UPM. He was Area Coordinator of the Spanish Evaluation Agency from 1990 to 1992 and the General Coordinator of the Spanish Commission for the Evaluation of the Research Activity (CNEAI) from 2011 to 2014. His professional and research interests are in the areas of digital image and video compression and of computer vision.

1057-7149 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.