Automatic Learning based 2D-to-3D Image Conversion

Victoria M Baretto
Dept. of Computer Science and Engineering
Alva's Institute of Engineering and Technology (AIET)
Moodbidri, India
meetvics.online@gmail.com

Abstract— **in the last few years, the availability of 3D content is still less than 2D counterpart. Hence many 2D-to-3D image conversion methods have been proposed. Methods involving human operators have been most successful but also time-consuming and costly. Automatic methods, that make use of a deterministic 3D scene model, have not yet achieved the same level of quality for they rely on assumptions that are often violated in practice. Here two types of methods are developed. The first is based on learning a point mapping from local image/ attributes, such as color, spatial position. The second method is based on globally estimating the entire depth map of a query image directly from a repository of 3D images (image + depth pairs or stereo pairs) using a nearest-neighbour regression type idea. It demonstrates the ability and the computational efficiency of the methods on numerous 2D images and discusses their drawbacks and benefits.**

Keywords-Stereoscopic images, Image conversion, nearest neighbour Classification, Cross-bilateral filtering, 3D images

I. INTRODUCTION

The convenience of 3D-capable hardware today, such as TVs, Blu-Ray players, gaming consoles, and smart phones, is not yet matched by 3D content production. Today there exists an urgent need to convert the existing 2D content to 3D. A typical 2D-to-3D conversion process consists of two steps: depth estimation for a given 2D image and depth based rendering of a new image in order to form a stereo pair. While the rendering step is well understood, the challenge is in estimating depth from a single image. Therefore, throughout the focus is on depth recovery.

There are two basic approaches, semi-automatic and automatic methods. In the former case a skilled operator assigns depth to various parts of an image. Based on this sparse depth assignment, a computer algorithm estimates dense depth over the entire image or sequence. The involvement of a human operator may vary from just a few scribbles to assign depth to various locations in an image to a precise delimitation of objects and subsequent depth assignment to the delineated regions. In the case of automatic methods, no operator involvement is needed and a computer algorithm automatically estimates the depth for a single image. Recently, machine-learning-inspired methods have been proposed to automatically estimate the depth map of a single monocular image by applying image parsing.

The proposed methods carry the "big data" philosophy of machine learning. They apply to arbitrary scenes and require no manual explanation. Two types of methods are proposed. The first one is based on learning a point mapping from *local* image/ attributes, such as color, spatial position, and motion at each pixel, to scene-depth at that pixel using a regression type idea. The second one is based on *globally* estimating the entire depth map of a query image directly from a repository of 3D images (image + depth pairs or stereopairs) using a nearest-neighbor regression type idea. It introduces local method and evaluates the qualitative performance and the computational efficiency of both the local and global methods. The improved quality of the depth maps produced by the global method relative to state-of-the-art methods together with up to 4 orders of magnitude reduction in computational effort and weakness of the methods are also demonstrated.

II. CONVERSION METHODS

There are two types of 2D-to-3D image conversion methods: semi-automatic methods and automatic methods.

A. Semi-automatic methods

Semi-automatic methods are more effective. This method has been effectively used commercially by such companies as Imax Corp., Digital Domain Productions Inc. etc. In order to shorten operator involvement in the process and lower the cost while speeding up the conversion, research has recently focused on the most labor-intensive steps of the manual involvement, namely spatial depth assignment. Liao *et al.* [10] further simplify operator involvement by first computing optical flow, then applying structure-from-motion estimation and finally extracting moving object boundaries. The role of an operator is to correct errors in the automatically computed depth of moving objects and assign depth in undefined areas.

B. Automatic methods

The difficult of depth estimation from a single 2D image is the main step in 2D-to-3D image conversion. Methods called multiview stereo, attempt to improve depth by estimating scene geometry from multiple images not taken

instantaneously. Such methods are similar in spirit to the methods proposed here, the main difference is that while these methods use images known to show the same scene as the query image, all images accessible in a large repository and automatically select suitable ones for depth retrieval. Real-time methods have been implemented in Blu-Ray 3D players by LG, Samsung, Sony and others. DDD offers its TriDef 3D software for PCs, TVs and mobile devices. Recently, machine-learning-inspired techniques employing image parsing have been used to estimate the depth map of a single monocular image [14], [11]. Such methods have the potential to automatically generate depth maps, but work only on few types of images (mostly architectural scenes). Metric based on histogram of gradients was used for selecting most similar depth fields from a database. It has been observed that there is no significant quality degradation but a significant reduction of the computational complexity [9]. Karsch *et al.* [7] have proposed a depth extraction method based on SIFT warping that essentially follows the initial, unnecessarily complex, approach to depth extraction [8].

III. 2D-TO-3D CONVERSION BY LEARNING A LOCAL POINT TRANSFORMATION

This conversion method is presented on the basis of learning a point transformation that relates local low-level image or video attributes at a pixel to scene-depth at that pixel. Once the point transformation is learned, it is applied to a monocular image, i.e., depth is assigned to a pixel based on its attributes.

The point transformation is used to compute depth from image attributes. This transformation can be estimated either by training on a ground-truth dataset. Let I = $\{(I^1, d^1), (I^2, d^2),...,(I^K, d^K)\}$ denote a training dataset composed of K pairs (I^k, d^k), where I^k is a color image (usually in *YUV* format) and d^k is the corresponding depth field. Such a dataset can be constructed in various ways. One example is the Make3D dataset [21], [13], [14], NYU Kinect dataset [22], [15]. Examples of low-level video attributes that can be leveraged to compute relative depth of a pixel include color, spatial location, and local motion. Due to the dependency of color, Bluish color is often associated with a distant sky, the bottom of a picture usually depicts ground close to the camera and a moving object stays in front of the background.

Given a training set I consisting of K image-depth pairs, a general regression function can be learned that maps a tuple of local features such as (color, location, motion) to a depth value, i.e.,

$$f : (\text{color, location, motion}) \rightarrow \text{depth}$$

However, to ensure low run-time memory and processing costs, it learns a more restricted form of transformation:

$$f[\text{color}, \mathbf{x}, \text{motion}] = w_c f_c[\text{color}] + w_l f_l[\mathbf{x}] + w_m f_m[\text{motion}].$$

It also discusses how the individual color-depth, location depth, and motion-depth transformations as well as the weights are learned.

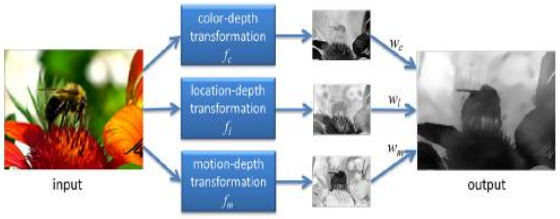

Figure1. Example of depth estimation from color, spatial location and motion.

Figure.1 shows a sample video frame with depth maps estimated from color, location and motion signs separately, with the final combined depth map. In order to obtain a color-depth transformation *fc*, it first transforms the *YUV* space to the *HSV* color space. It is found out that the saturation component (*S*) provides little depth discrimination capacity and therefore it limits the transformation attributes to hue (*H*) and value (*V*). Let $[H^k[\mathbf{x}], S^k[\mathbf{x}], V^k[\mathbf{x}]]^T$ be the *HSV* components of a pixel at spatial location **x** quantized to L levels. The depth mapping, $f_c[h, v]$, $h, v = 1, ..., L$ is computed as the average of depths at all pixels in I with hue h and value v:

$$f_c[h,v] = \frac{\sum_{k=1}^{K} \sum_{\mathbf{x}} 1(H^k[\mathbf{x}] = h, V^k[\mathbf{x}] = v) d^k[\mathbf{x}]}{\sum_{k=1}^{K} \sum_{\mathbf{x}} 1(H^k[\mathbf{x}] = h, V^k[\mathbf{x}] = v)} \qquad (1)$$

Where (1) is the indicator function which equals one if A is true and equals zero otherwise. In the final step, the local transformation outputs are linearly combined to produce the final depth field. While the location-depth weight w_l may be kept constant, the motion-depth weight w_m can be adjusted in proportion to the number of pixels deemed moving in the image being converted. Therefore, the color-depth weight w_c equals $1-w_l-w_m$. Assuming that the image to be converted to 3D is the left image of a fictitious stereopair, the right image is rendered from the left image and the inferred depth field.

IV. 2D-TO-3D CONVERSION BASED ON GLOBAL NEAREST–NEIGHBOR DEPTH LEARNING

Learning a local point transformation has the undisputed advantage of computational efficiency – the point transformation can be learned off-line and applied basically in real time because it is based on purely *local* image/video attributes, such as color, spatial position, and motion at each pixel. To address this limitation, a second method is developed that estimates the *global* depth map of a query image or video frame directly from a repository of 3D images (image+depth pairs or stereopairs) using a nearest-neighbor regression type idea.

This approach is built upon a key observation and an assumption. The key observation is that among millions of 3D images available on-line, there likely exist many whose 3D content matches that of a 2D input (query). An assumption is made that two images that are photometrically similar also have similar 3D structure (depth). Given a

monocular query image Q, assumed to be the left image of a stereopair that is to be computed, relies on the above observation and assumption to "learn" the entire depth from a repository of 3D images I and render a stereopair in the following steps:

- **Search for representative depth fields**: find k 3D images in the repository I that have most similar depth to the query image, for example by performing a k nearest-neighbor (kNN) search using a metric based on photometric properties.
- **Depth fusion**: combine the k representative depth fields, for example, by means of median filtering filtering across depth field.
- **Depth smoothing**: process the fused depth field to remove spurious variations, while preserving depth discontinuities, for example, by means of cross-bilateral filtering.
- **Stereo rendering**: generate the right image of a fictitious stereopair using the monocular query image and the smoothed depth field followed by suitable processing of occlusions and newly-exposed areas.

The above steps apply directly to 3D images represented as an image+depth pair. However, in the case of stereopairs a disparity field needs to be computed first for each left/right image pair. Then, each disparity field can be converted to a depth map. Alternatively, the fusion and smoothing can take place in the space of disparities (without converting to depth), and the final disparity used for right-image rendering.

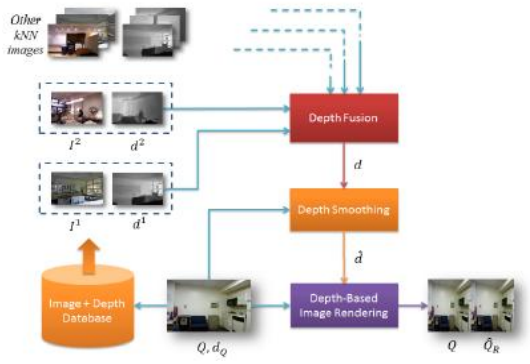

Figure 2: Block diagram of overall algorithm

Fig. 3 shows the block diagram of the approach. Q_R is the right image which is being sought for each query image Q, while d_Q is the query depth (ground truth) needed to numerically evaluate the performance of a depth computation. Again, it is assumed that a 3D dataset I is available by means of laser range finding, Kinect-based capture or disparity computation. The goal is to find a depth estimate d and then a right-image estimate Q_R given a 2D query image Q and the 3D dataset I.

A. *kNN Search*

There exist two types of images in a large 3D image repository: relevant and irrelevant images. Images that are not photometrically similar to the 2D query need to be rejected because they are not useful for estimating depth. One method for selecting a useful subset of depth relevant images from a large repository is to select only the k images that are closest to the query where closeness is measured by distance function capturing global image properties such as color, texture, edges, etc. The distance function used here is Euclidean norm of the difference between histograms of oriented gradients (HOGs) [3] computed from two images. It also performs a search for top matches to the monocular query Q among all images inverse of I^k, k = 1,..., K in the 3D database I. The search returns an ordered list of image + depth pairs from the most to the least photometrically similar *via* the query and discard all but the top k matches (kNNs) from this list.

The average photometric similarity between a query and its k-th nearest neighbor usually decays with the increasing k. While for large databases, larger values of k may be appropriate, since there are many good matches, for smaller databases this may not be true. Therefore, a judicious selection of k is important. K denotes the set of indices i of image + depth pairs that are the top k photometrically-nearest neighbors of the query Q.

B. *Depth Fusion*

The depth field is computed by applying the median operator across the kNN depths at each spatial location $\mathbf{x}$ as follows:

$$d[\mathbf{x}] = \text{median}\{d^i[\mathbf{x}], \forall i \in \mathcal{K}\}. \qquad (3)$$

Although these depths are overly smooth, they provide a globally-correct, although coarse, assignment of distances to various areas of the scene.

C. *Cross-Bilateral Filtering of Depth*

While the median-based fusion helps make depth more consistent globally, the fused depth is overly smooth and locally inconsistent with the query image due to edge misalignment between the depth fields of the kNNs and the query image. This results in the lack of edges in the fused depth where sharp object boundaries should occur and/or the lack of fused-depth smoothness where smooth depth is expected. In order to correct this, similarly to Agnot *et al.* [1], it applies cross-bilateral filtering (CBF). CBF is a variant of bilateral filtering, an edge-preserving image smoothing method that applies anisotropic diffusion controlled by the local content of the image itself [4]. In CBF, however, the diffusion is not controlled by the local content of the image under smoothing but by an external input. It applies CBF to the fused depth d using the query image Q to control diffusion. This allows us to achieve two goals simultaneously: alignment of the depth edges with those of the luminance Y in the query image Q and local

noise/granularity suppression in the fused depth *d*. This is implemented as follows:

$$\hat{d}[\mathbf{x}] = \frac{1}{\gamma[\mathbf{x}]} \sum_{\mathbf{y}} d[\mathbf{y}] h_{\sigma_s}(\mathbf{x}-\mathbf{y}) h_{\sigma_e}(Y[\mathbf{x}]-Y[\mathbf{y}]),$$
$$\gamma[\mathbf{x}] = \sum_{\mathbf{y}} h_{\sigma_s}(\mathbf{x}-\mathbf{y}) h_{\sigma_e}(Y[\mathbf{x}]-Y[\mathbf{y}]), \quad (4)$$

Where inverse of *d* is the filtered depth field and $h_\sigma(\mathbf{x}) = \exp(-\|\mathbf{x}\|^2/2\sigma^2)/2\pi\sigma^2$ is a Gaussian weighting function. Note that the directional smoothing of *d* is controlled by the query image *via* the weight $h_{\sigma e}(Y[\mathbf{x}] - Y[\mathbf{y}])$. For large luminance discontinuities, the weight $h_{\sigma e}(Y[\mathbf{x}]-Y[\mathbf{y}])$ is small and thus the contribution of $d[\mathbf{y}]$ to the output is small. However, when $Y[\mathbf{y}]$ is similar to $Y[\mathbf{x}]$ then $h_{\sigma e}(Y[\mathbf{x}] - Y[\mathbf{y}])$ is relatively large and the contribution of $d[\mathbf{y}]$ to the output is larger. In essence, depth filtering (smoothing) is happening along (and not across) query edges. The depth field is overall smooth (slowly varying) while depth edges, if any, are aligned with features in the query image. The filtered depth preserves the global properties captured by the unfiltered depth field *d*, and is smooth within objects and in the background. At the same time it keeps edges sharp and aligned with the query image structure.

D. Stereo Rendering

In order to generate an estimate of the right image Q_R from the monocular query *Q*, a disparity δ from the estimated depth *d* needs to be computed. Assuming that the fictitious image pair (Q,Q_R) was captured by parallel cameras with baseline *B* and focal length *f* , the disparity is simply $\delta[x, y] = Bf/d[\mathbf{x}]$, where $\mathbf{x} = [x, y]T$. It forwards projects the 2D query *Q* to produce the right image:

$$Q_R[x+\delta[x,y],y]=Q[x.y] \quad (5)$$

While rounding the location coordinates (x + ä[x, y], y) to the nearest sampling grid point. It handles occlusions by depth ordering: if $(x_i + ä[x_i, y_i], y_i) = (x_j + ä[x_j, y_i], y_i)$ for some i, j, it assigns to the location $(x_i + ä[x_i, y_i], y_i)$ in QR an RGB value from that location (x_i, y_i) in Q whose disparity $ä[x_i, y_i]$ is the largest. In newly-exposed areas, i.e., for x_j such that no x_i satisfies $(x_j, y_i) = (x_i + ä[x_i, y_i], y_i)$, Applying a more advanced depth-based rendering method would only improve this step of the proposed 2D-to-3D conversion.

V. EXPERIMENTAL RESULTS

The approach has been tested on two datasets: the Make3D dataset #1 [21], [13], [14] composed of 534 outdoor images with depth fields captured by a laser range finder and the NYU Kinect dataset [22], [15] composed of 1449 pairs of RGB images and corresponding depth fields. Note that the Make3D images are of 240×320 resolution but the corresponding depth fields are only of 55×305 spatial resolution and relatively coarse quantization. On the other hand, the Kinect dataset consists of both images and depth fields at 640×480 resolution and the depth precision is relatively high (11 bits).

In order to evaluate the performance of the proposed algorithms quantitatively, leave-one-out cross-validation (LOOCV) was applied as follows. One image+depth pair from a database was selected as the 2D query (Q, d_q) treating the remaining pairs as the 3D image repository I based on which a depth estimate the inverse of d and a right-image estimate Q_R are computed. As the quality metric, it is used normalized cross-covariance between the estimated depth d and the ground-truth depth d_q as follows:

$$C = \frac{1}{N\sigma_{\hat{d}}\sigma_{d_Q}} \sum_{\mathbf{x}} (\hat{d}[\mathbf{x}] - \mu_{\hat{d}})(d_Q[\mathbf{x}] - \mu_{d_Q}) \quad (6)$$

Where N is the number of pixels in d and d_Q, μd and μd_Q are the empirical means of d and d_Q, respectively, while σ_d and σ_{dQ} are the corresponding empirical standard deviations. The normalized cross-covariance C takes values between - 1 and +1 (for values close to +1 the depths are very similar and for values close to -1 they are complementary).

Table 1: Average and median normalized cross covariance C computed across all images in the Make3D Dataset.

	Local	Global		Make3D	Karsch et. al.
		Median	Median+CBF		
Average C	0.59	0.78	0.80	0.78	0.73
Median C	0.61	0.85	0.86	0.78	0.79

Table I shows experimental results obtained from 534 LOOCV tests on the Make3D dataset #1 using various algorithms. The performance of each algorithm has been captured by the average and median of cross-covariance C (6) across all LOOCV tests. The local method has been trained on the Make3D and Kinect datasets, respectively, i.e., the transformations fc and f_l (transformation f_m is not used since both datasets contain only still images), have been learned by analysing depth color and depth-location relationships in all image-depth pairs of either dataset. It used weights $w_c = 0.3$ and $w_l = 0.7$ in the experiments. The global method and the method by Karsch et al. [7] have no training phase but learn the depth from k best examples found for each query image. As it has already mentioned, the Make3D algorithm [14] has been trained on the Make3D dataset and there is no option available to re-train it on the Kinect dataset. Clearly, for both datasets the global method with cross bilateral filtering of the fused depths outperforms all other algorithms, although the same algorithm without the filtering performs very similarly. The numerical gain from filtering the fused depth is rather small since its greatest impact is at depth edges (re-alignment with edges in the query image). Consequently, it affects the normalized

cross covariance at just a few pixels. The Make3D algorithm performs almost as well as the global method on the Make3D dataset. However, this result is biased towards high values of C since the test images in the LOOCV test include images from the database on which the Make3D algorithm was trained (400 training images). Not surprisingly, the same algorithm applied to the Kinect dataset fairs rather poorly but, it has been already mentioned, re-training was not possible. The Karsch et al. method does not perform as well as the global algorithm on either dataset. Finally, the local method achieves a consistent but low performance which is not surprising given its simplicity.

In addition to LOOCV tests on the Make3D dataset #1, where the test image may belong to the set of original 400 training images, it also applied the test used by Saxena *et al.* [14]. Namely, the 534 images of this dataset were divided into 134 test images and 400 training images (on which the Make3D algorithm was trained). It selected the test image from the test set and used the training set to find the *k* nearest neighbors.

This is to be expected since LOOCV uses more training images. The performance of the local method appears to be only marginally affected. This can be attributed to the use of a fixed point mapping. Both Make3D and the global method with CBF experience a significant performance drop but Make3D continues to trail behind the method. The method of Karsch *et al.* appears to be more robust, even improving slightly in terms of the average *C* value, but it takes about 2 hours to execute while processing 12 images in parallel. In contrast, Make3D takes about 30mins and the global method with CBF takes about 1 second to process.

VI. Conclusion

A new class of methods is proposed to aim at 2D-to-3D image conversion that is based on the radically different approach of learning from examples. One method that is proposed is based on learning a point mapping from local image attributes to scene-depth. The other method is based on globally estimating the entire depth field of a query directly from a repository of image+depth pairs using nearest-neighbor-based regression. It objectively validates the algorithms' performance against state-of-the-art algorithms. While the local method was outperformed by other algorithms, it is extremely fast as it is, basically, based on table look-up. However, the global method performed better than the state-of-the-art algorithms in terms of cumulative performance across two datasets and two testing methods, and has done so at a fraction of CPU time. Anaglyph images produced by the algorithms result in a comfortable 3D experience but are not completely void of distortions. Clearly, there is room for improvement in the future. With the continuously increasing amount of 3D data on-line and with the rapidly growing computing power in the cloud, the proposed framework seems a promising alternative to operator-assisted 2D-to-3D image and video conversion.

VII. Acknowledgements

The author would like to acknowledge Mr. Geoffrey for the implementation of the very first, SIFT-based, and variant of the 2D-to-3D image conversion method reported here.

References

[1] L. Agnot, W.-J. Huang and K.-C. Liu. A 2D to 3D video and image conversion technique based on a bilateral filter. In *Proc. SPIE Three-Dimensional Image Processing and Applications*, volume 7526, Feb.2010.

[2] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *Proc. European Conf. Computer Vision*, pages 25–36, 2004.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 886–893, 2005.

[4] F. Durand and J. Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. *ACM Trans. Graph.*, 21:257–266, July 2002.

[5] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust L1 optimal camera paths. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, 2011.

[6] M. Guttmann, L. Wolf, and D. Cohen-Or. Semi-automatic stereo extraction from video footage. In *Proc. IEEE Int. Conf. Computer Vision*, pages 136–142, Oct. 2009.

[7] K. Karsch, C. Liu, and S. B. Kang. Depth extraction from video using non-parametric sampling. In *Proc. European Conf. Computer Vision*, 2012.

[8] 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. In *Proc. SPIE Stereoscopic Displays and Applications*, volume 8288, Jan. 2012.

[9] J. Konrad, M. Wang, and P. Ishwar. 2D-to-3D image conversion by learning depth from examples. In *3D Cinematography Workshop (3DCINE'12) at CVPR'12*, pages 16–22, June 2012.

[10] M. Liao, J. Gao, R. Yang, and M. Gong. Video stereolization: Combining motion analysis with user interaction. *IEEE Trans. Visualization and Computer Graphics*, 18(7):1079–1088, July 2012.

[11] B. Liu, S. Gould, and D. Koller. Single image depth estimation from predicted semantic labels. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 1253–1260, June 2010.

[12] 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 *Proc. IEEE Int. Conf. Image Processing*, Sept. 2011.

[13] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *NIPS*, 2005.

[14] A. Saxena, M. Sun, and A. Ng. Make3D: Learning 3D scene structure from a single still image. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(5):824 –840, May 2009.

[15] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *Proc. Int. Conf. on Computer Vision - Workshop on 3D Represent. and Recogn.*, 2011.

[16] M. Subbarao and G. Surya. Depth from defocus: A spatial domain approach. *Intern. J. Comput. Vis.*, 13:271–294, 1994.

[17] R. Szeliski and P. H. S. Torr. Geometrically constrained structure from motion: Points on planes. In *European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, page 171–186, 1998.

[18] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Trans. Pattern Anal. Machine Intell.*, 30(11):1958–1970, Nov. 2008.

[19] M. Wang, J. Konrad, P. Ishwar, K. Jing, and H. Rowley. Image saliency: From intrinsic to extrinsic context. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 417–424, June 2011.

[20] R. Zhang, P. S. Tsai, J. Cryer, and M. Shah. Shape-from-shading: A survey. *IEEE Trans. Pattern Anal. Machine Intell.*, 21(8):690–706, Aug. 1999.

[21]http://make3d.cs.cornell.edu/data.html.
[22]http://cs.nyu.edu/silberman/datasets/nyudepth v1.html.