Creating an Artificial Bokeh Blur

Viswanandha Raju Thotakura
SCPD
Stanford University
viswanad@stanford.edu

Hitha Revalla
Masters,EE
Stanford University
hitha@stanford.edu

Abstract

This work focused on how bokeh can be created and applied to images artificially. It utilized advancements made in machine learning, depth estimation techniques, and work in computational photography. Here we explore two techniques namely, Random forest based depth estimation and Disparity map generation form Stereo images to get depth/disparity maps and apply a defocus algorithm resulting in an artificial bokeh blur.

1. Introduction

Artistic enhancement of photographs is a very popular field of research with several applications that are common place in the world of smartphone photography. Recently researchers have sought to produce an effect known as Bokeh in smartphone photos using digital image processing techniques. Bokeh is the shallow-depth of field effect which blurs the background of portrait photos (typically) to bring emphasis to the subject in the foreground.

Figure 1. Left :Image without Bokeh effect, Right :Image with Bokeh effect,

Figure 1 shows the same image with and without Bokeh effect. Bokeh's nature originates from the shape of camera lenses, aperture, distance to background objects, and their distinctive light and shadow patterns.Bokeh effect is usually achieved in high end SLR cameras using portrait lenses that are relatively large in size and have a shallow depth of field. It is extremely difficult to achieve the same effect (physically) in smart phones which have miniaturized camera lenses and sensors.

Many of today's amateur photographers need to create artistic photos on a budget. One of the convenient ways this can be achieved is through the use of post processing toolkits. These toolkits provide a variety of features such as blurring, bokeh, patching, and many more. The creation of these toolkits fall under computational photography, and are an ever-growing research field. In this project we choose to look into methods for creating an artificial bokeh blur by using the depth maps generated.

Currently, there are many different approaches to apply artificial bokeh in post processing. Two of the most popular methods utilize either a measured depth ground truth map, or stereo pairs to infer a depth ground truth map. We explored both these methods for the project.

2. Related Work

In recent years, there are a variety of methods were proposed to get the defocus mask.Elder and Zucker [1] used the first and second order derivatives of the input image to determine the size of the Gaussian blur kernel. Bae and Durand [2] extended their work by adding a bilateral filter to remove outliers and estimate the blur amount of the whole image. Using defocus map, they increased the blurriness of the blurry regions and kept the in focus region sharp. Levin et al. [3] converted a statistical model to recover defocus map with a modified camera. Tai and Brown [4] used local contrast prior to measurements into a defocus map and then apply MRF propagation to refine the estimated defocus map. The advantage of their method is that it does not rely on frequency decomposition of the input image. It only compares the local gradients with local contrast. Zhuo and Sim [5] generated a full defocus map with good accuracy by using matting interpolation.

One common approach in todays world, is to use machine learning to tackle difficult problems where there is a wealth of data and information. Luckily, computational photography problems often have a wealth of information to mine for patterns. As such, Saxena et al. have developed a depth estimator for monocular images. Their work was crucial to our work as it provided a solid foundation for generating approximate depth maps. They utilized indoor and outdoor scenes and generated various models on these scenes in order to cover a wide variety of applications and use cases. Apart from this we also used Stereo rectification in order to get disparity maps. The defocus mask is obtanied by segementing the depth/disparity map into foreground and background. Then, bokeh effect is generated by applying blur kernel to further blur the background while keep the foreground sharpen. The report shall give details on the Depth estimation and the Defocus algorithm used in the project.

3. Depth estimation techniques

For the project we wanted to explore the two most popular methods of depth estimation.

3.1. Machine Learning

This method was mostly guided by the Depth Estimation from Single Image using Structured Forest by Shuai Fang et.al.The core idea of this approach is to exploit the structure properties exhibit in local patches of depth map to learn the depth level for each pixel. Therefore, the problem of depth estimation can be framed as a structured learning framework based on random decision forests. The approach firstly maps structured labels into a discrete space at each branch in the tree while training the decision trees, and then use these labels to determine the splitting function. Each forest infers a patch of structured labels that are accumulated across the image to obtain the final depth map.

3.1.1 Structured random forest for depth estimation

Structured random forest is an extension of random forest with structured outputs. A decision tree $f_t(x)$ classifies a data sample $x \in X$ by recursively splitting left or right down the tree till a leaf node is reached. Specifically, each node j in the tree is associated with a binary split function:

$$h(x,_j) \in 0, 1$$

where $_j$ is the parameter. If $h(x,_j) = 0$, node j direct x left, otherwise right. The splitting process terminates until reaches at a leaf node. The output of decision tree on x is the inference stored at the reached leaf, which may be a label $y \in Y$ or a distribution over the labels Y .

Given an image patch, our approach takes the depth relevant features as the input x X, and the corresponding labels indicating the depth level as the output y Y . For example, for a 16x16 local patch, there are N^{256} (N means the number of depth levels) unique depth labels. Accordingly, we need to use an approximate splitting method to reduce the dimensionality of Y . Specifically, we map the output labels $y \in Y$ into a discrete set of labels $c \in C$, where C = 1, ..., k. The goal is clustering similar lables into same discrete label, see Figure 2.

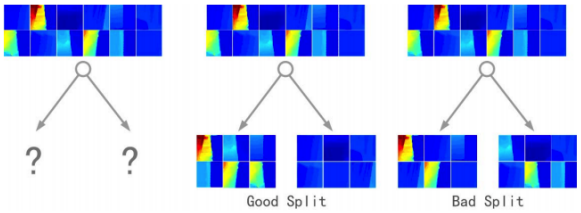

Figure 2. Illustration of the decision tree node splits: (a) Given a set of structured labels as depth labels, a splitting function must be determined. Intuitively a good split (b) groups similar depth labels, whereas a bad split (c) does not.

3.1.2 Input depth relevant features

It is important to note that, different image features that are related to the properties of scene structure such as texture variations, texture gradients, occlusion, haze and defocus. The paper mentioned about draws inspiration from Saxena et al.s work [11], two types of features are extracted from a given an image patch: absolute depth features and relative depth features, which are used to estimate the absolute and relative depth within the image patch, respectively. The features that the paper proposes are as follows:

a. SCN Features

In [11], Saxena et al. propose to use SCN features for absolute depth estimation. The SCN can be estimated by the output of 17 filters (9 Laws masks, 2 color channels in YCbCr space and 6 oriented edges) for each image patch. These filters capture the texture of a 33 patch and the edges at various orientations, which have high correlation to scene depth.

b. Dark Channel Features

Dark channel prior is based on the observation that most local patches in outdoor images contain some pixels with very low intensities in at least one color channel. Therefore, distant objects tend to reflect more atmospheric light. From this perspective, dark channel is indeed a depth cue. Dark Channel of an image J is defined as:

$$J_{dark} = min_{c \in r,g,b}(min_{y \in \Omega(x)}(J^c(y)))$$

where J^c is a color channel of J and $\Omega(x)$ is a local patch centered at position x.

3.1.3 Implementation

We implemented the structured forest training based on the Matlab toolbox and also inherited the features and toolboxes from Shuai Fang et. al implementation of the random forest with 4-tree forests. We implemented our own version of a 1-tree depth estimator version of the implementation.The results were not as good and the comparison of the two was reported in the poster. Therefore the first goal was to connect the make3d model to onto our matlab Bokeh script. This would provide the basis for the depth map creation needed for the defocusing algorithm. We were able to create the necessary matlab calls to generate the depth map from the 4-tree depth estimator.

3.1.4 Results

These are a few of the results from the depth estimator:

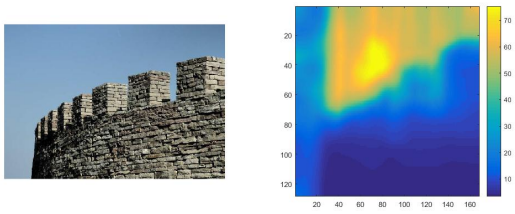

Figure 3. Depth example 1

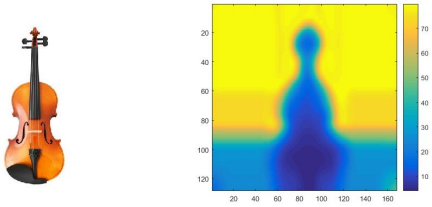

Figure 4. Depth example 2

3.2. Generating Disparity based on Stereo Images

Generating disparity from stereo images involves two main stages.
1.Rectification of stereo images
2.Disparity map generation from stereo correspondence.

Figure 5 explains multiple stages in generating disparity we are going to discuss in this paper.

3.2.1 Rectification of stereo images:

The important stage in generating the stereo images are capturing images in sync. An iOS application is written to capture from tele and wide camera images in synchronously.

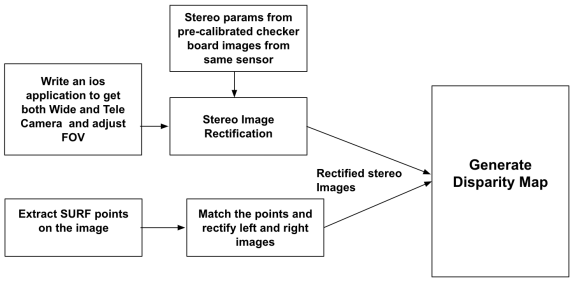

Figure 5. Stages in generating disparity map using stereo rectification

Once captured, Tele and Wide images are stored in the photos app. These captured frames are fed to the stereo image rectification process. Stereo Image rectification is mainly based on the camera intrinsic, extrinsic parameters parameters of a camera. This approach projects images onto a common image plane in such a way that the corresponding points have the same row coordinates. This process is useful for stereo vision, because the 2-D stereo correspondence problem is reduced to a 1-D problem. Stereo image rectification is pre processing step for computing disparity. The camera intrinisic matrix is parameterized as Intrinsic

$$K = \begin{pmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

Figure 6. Intrinsic Matrix of Camera

parameters describe geometry property of camera. f_x and f_y is focal length of camera in x and y directions. The values x_0 and y_0 are the offsets of the principal point from the top-left corner of the image frame. s is axis skew. Extrinsic matrix explains about the camera's position in the world coordinates. This matrix has two components, rotation matrix and translation vector. Extrinsic matrix corresponds to how to transform world coordinates to camera coordinates. The extrinsic matrix is of the form 3x3 rotation matrix and 3x1 translation column vector. $e = [R|t]$;

$$[R \mid \boldsymbol{t}] = \left[\begin{array}{ccc|c} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \end{array}\right]$$

Figure 7. Extrinsic Matrix of Camera

Here in this paper we employed two methods of stereo camera calibration.

A) Stereo Camera Calibration: Stereo camera calibration estimates the parameters of a lens and image sensor of both the images. These parameters are used to lens distortion, measure size of an object in world units or determine the location of camera in world scene. This method uses checkerboard pattern images at different areas in the camera field of view to determine camera intrinsic and extrinsic parameters.

Image Rectification: The image rectification process removes the lens distortions that are measured from the calibration step and transforms the images such that they are coplanar. This applies the epipolar constraint which says that a point must lie on epipolar line determined by its corresponding point to the stereo pair which enables a faster and more accurate stereo correspondence. Figure 8 shows some of the images we used for checkerboard calibration. Each row corresponds to image capture from same camera. Consecutive rows corresponds images from different cameras. Figure 9 shows mean projection errors and Figure 10

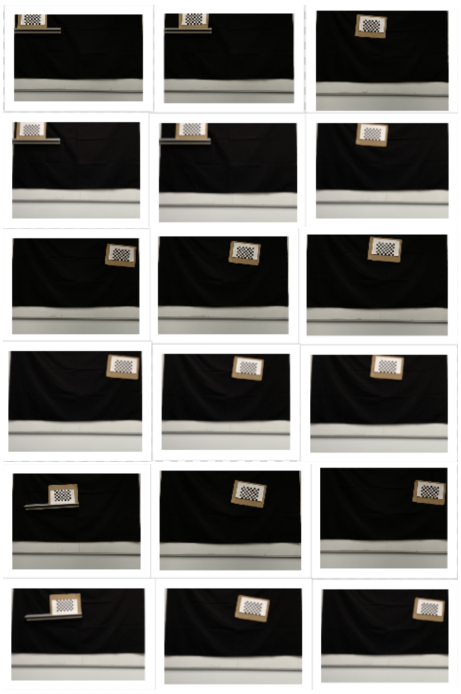

Figure 8. Some of the checkerboard calibration images

shows the extrinsic parameter visualization.

B) Uncalibrated stereo rectification: This method estimates fundamental matrices to compute rectifications of two uncalibrated images when camera intrinsics are not known. The rectification process requires to identify a set of point correspondences between the two images. To generate these correspondences, points of interest are collected between the images and and then potential matches are chosen. The correctly matched points must satisfy epipolar

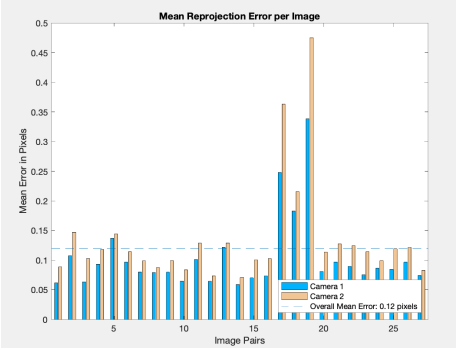

Figure 9. Mean Projection Errors

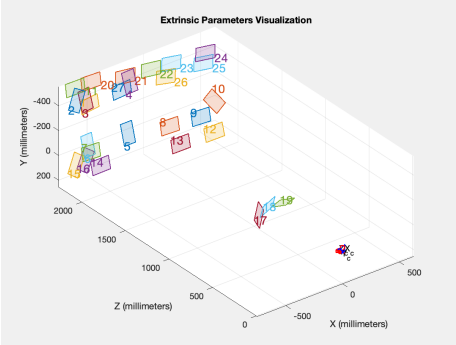

Figure 10. Extrinsic parameter Visualization

constraints.

3.2.2 Stereo Camera Disparity Map:

Stereo camera disparity map is computed based on the Semi Global Block Matching(SGBM) Algorithm. . It uses a pixel-wise, Mutual Information based matching cost for compensating radiometric differences of input images. Pixel wise matching is supported by a smoothness constraint that is usually expressed as a global cost function. SGM performs a fast approximation by path-wise optimizations from all directions. Semi-global matching uses information from neighboring pixels in multiple directions to calculate the disparity of a pixel. Analysis in multiple directions results in a lot of computation. Instead of using the whole image, the disparity of a pixel can be calculated by considering a smaller block of pixels for ease of computation. Thus, the Semi-Global Block Matching (SGBM) algorithm uses block-based cost matching that is smoothed by path-wise information from multiple directions. Figure 11 [6] explains the block diagram of the SGBM algorithm.

The SGBM algorithm implementation has three major modules: Matching Cost Calculation, Directional Cost Calculation and Post-processing. In this paper, CSCT transform is used in matching cost computation. Matching

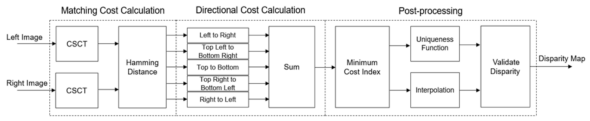

Figure 11. Block Diagram of Semi Global Block Matching Algorithm

cost computation block is divided into two steps: Center-Symmetric Census Transform (CSCT) of left and right images and Hamming Distance computation. First, the model computes the CSCT on each of the left and right images using a sliding window. For a given pixel, a 9-by-7 pixel window is considered around it. CSCT for the center pixel in that window is estimated by comparing the value of each pixel with its corresponding center-symmetric counterpart in the window. If the pixel value is larger than its corresponding center-symmetric pixel, the result is 1, otherwise the result is 0. In the Hamming Distance module, the CSCT outputs of the left and right images are pixel-wise XOR'd and set bits are counted to generate the matching cost for each disparity level. To generate D disparity levels, D pixel-wise Hamming distance computation blocks are used. The matching cost for D disparity levels at a given pixel position, p, in the left image is computed by computing the Hamming distance with (p to D+p) pixel positions in the right image.

The second module of SGBM algorithm is directional cost estimation. In general, due to noise, the matching cost result is ambiguous and some wrong matches could have lower cost than correct ones. Therefore additional constraints are required to increase smoothness by penalizing changes of neighboring disparities. This constraint is realized by aggregating 1-D minimum cost paths from multiple directions. It is represented by aggregated cost from r directions at each pixel position, S(p,d), as given by

$$S(p,d) = Sum(L_r(p,d))$$

The 1-D minimum cost path for a given direction, Lr(p,d), is computed as shown in the equation.

$$\begin{aligned} L_r(p,d) = C(p,d) + min(&L_r(p-r,d),\\ &L_r(p-r,d-1)+P1,\\ &L_r(p-r,d+1)+P1,\\ &min_i L_r(p-r,i)+P2-min_k L_r(p-r,k)) \end{aligned} \quad (1)$$

$L_r(p,d)$ =current cost of pixel p and disparity d in dir r
$C(p,d)$ = matching cost at pixel p and disparity d
$L_r(p-r,d-1)$ = previous cost of pixel in direction r and disparity d - 1
$L_r(p-r,d+1)$ = previous cost of pixel in direction r at disparity d +1
$min_i L_r(p-r,i)$ = minimum cost of pixel in r for previous computation
$P1, P2$ = Penalty for discontinuity

In the post-processing subsystem, the index of the minimum cost is calculated at each pixel position from user needed disparity levels by using Min block. The min index value obtained is the disparity of each pixel. Along with minimum cost index computation, the minimum cost value at the computed index, and the cost values at index-1 and index+1 are also computed.A uniqueness function is applied to the index calculated by min blocks, to ensure reliable disparity results. As a last step, invalid disparities are identified and replaced with -1.

4. Defocus Algorithm

To achieve Bokeh effect after the full depth map estimation, the image can be segmented into two segments i.e.,foreground and background.The constraint is to make this process as automated as possible. The algorithm chosen requires some hand tuning. We chose to use the K-Means Algorithm on the depth/disparity information. We hard coded a cluster count of 2, which for an image such as the one in figure 12, with a single object of interest without nearby objects is quite suitable.This basically helps us make a binary mask for applying the blur.

For dealing with more cluttered environments a larger cluster count would ensure a more granular region identification.This would be done by deciding an appropriate threshold and considering all the points with their depth/defocus blur below the threshold as a part of the foreground and the remaining as background. Thus, the bit masks of these segmentations would be obtained. The bit mask of the foreground would be applied on the original image and the bit mask of the background would be applied on the blurred image which is obtained by applying a Gaussian kernel to the original image. Finally, both images are combined to produce the final image with the Bokeh effect. We tried out the algorithm on a pair of images (original and depth maps) we found online. Figure 12 shows the results.

5. Results

Figure13 - Figure17 are set of results that are generated with Machine Learning and Stereo Camera Rectification methods.

6. Discussion

Overall the results from both the machine learning and stereo image rectification are satisfactory. With Random forest based approach we were able to extract approxi-

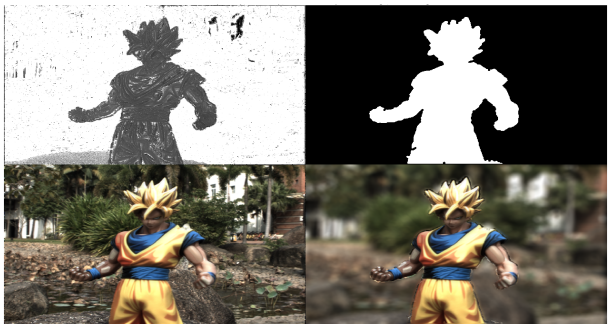

Figure 12. Results from Defocus Algorithm, Top-left : Depth map, Top-right : Binary mask, Bottom-left : Original Image, Bottom-right : Bokeh Image

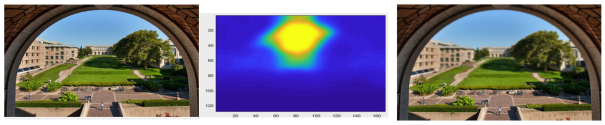

Figure 13. Results from Machine Learning Method,LtoR Original Picture, Depth map, Bokeh effect picture

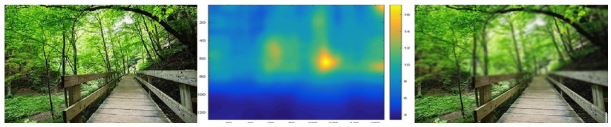

Figure 14. Results from Machine Learning Method, LtoR Original Picture, Depth map, Bokeh effect picture

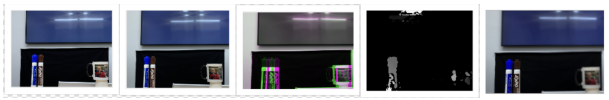

Figure 15. Results from Calibration Method , LtoR: Tele Camera Image, Wide Camera Image(after crop and re-scale), stereo anaglyph Image, disparityMap, Image with bokeh effect.

Figure 16. Results from UnCalibration Method , LtoR: Tele Camera Image, Wide Camera Image(after crop and re-scale), stereo anaglyph Image, disparityMap, Image with bokeh effect.

mate depth Map and while employing the stereo image rectification model we could achieve good disparity maps from both the calibrated and uncalibrated stereo camera approaches. The depth maps from the Random forest didnt have very good definition and this could be because of the selection of the right depth relevant input to train the trees. Also a bigger forest could have resulted in better depth

Figure 17. Results from UnCalibration Method LtoR: Tele Camera Image, Wide Camera Image(after adjusting), mask map and bokeh image

differentiation. The Bokeh effect based on the maps that were pulled from the original implementation did do good.

On the other hand, the bokeh effect from the stereo camera results are not as good as that of iPhone. We believe there are few reasons where this project lacked compared to the current portrait mode standard in mobile industry. 1. AutoFocus and OIS shift. We believe that the setup what we were using is having in-built Image stabilization and we couldn't find the API to make is center-lock(in the case of OIS) or disable it(in case of Software IS/EIS). We could disable AutoFocus and did experiments and capture the frames. In real time scenario we might need to account the lens position for each snapshot frame. We tried to get the camera calibration parameters from the iPhone through our custom camera app which provides current Lens position through intrinsic parameters and read other camera calibration parameters like extrinsic parameters, pixel size on per frame basis. When we feed these real-time parameters to the Matlab camera stereo rectification api the results are not convincing. We can also extend this project in that direction. 2. Irregularities in depth map, the depth map generated is not having perfect edges. This might be because of irregular disparity generated at the some parts of image is not reliable then we assign them max value after checking the uniqueness, to avoid this instead of applying smoothening or blur filter kernel we could do edge aware blur after reducing the imperfections in the image.

7. Future Work

As a concluding section, we would like to think about the future possibilities with this project.

• Work in this space could involve placing the corrected median around a normal distribution, to basically create focus at a particular depth range. This may produce more interesting results as a particular depth level is now isolated completely from the rest of the scene.

• Another interesting future work would be to incorporate multiple points of focus in the image. This would allow pictures to have bokeh effects perhaps around two subjects or two parts of a scene, with an artistic effect around them both.

References

[1] J. Elder, S. Zucker, Local scale control for edge detection and blur estimation, IEEE Trans. Pattern Anal. Mach. Intell. 20 (7) (1998) 699-716.

[2] S. Bae, F. Durand, Defocus magnification, Proc. Eurographics (2007) 571-579. [3] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. ACM Trans. Graphics, 2007.

[4] Y.-W. Tai, M. S. Brown, Single image defocus map estimation using local contrast prior, in: Proc. ICIP, 2009.

[5] Zhuo, Shaojie, and Terence Sim. "Defocus map estimation from a single image." Pattern Recognition 44.9 (2011): 1852-1858. [6] https://www.mathworks.com/help/vision/examples/uncalibrated-stereo-image-rectification.html

[7] "FPGA Implementation of Stereo Disparity using Semi-Global Block Matching" https://www.mathworks.com/help/visionhdl/examples/stereoscopic-disparity.html

[8] "Stereo Processing by Semi-Global Matching and Mutual Information Heiko Hirschmuller " https://core.ac.uk/download/pdf/11134866.pdf

[9] Hirschmuller H., Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information, International Conference on Computer Vision and Pattern Recognition, 2005.

[10] Spangenberg R., Langner T., and Rojas R., Weighted Semi-Global matching and Center-Symmetric Census Transform for Robust Driver Assistance, Computer Analysis of Images and Patterns, 2013.

[11]Saxena Ashutosh, Sun Min, and Andrew Y Ng, "Make3d: learning 3d scene structure from a single still image.," IEEE Transactions on Pattern Analysis Machine Intelligence, vol. 31, no. 5, pp. 824–840, 2009.