3D Scene Generation by Learning from Examples

Mesfin A. Dema and Hamed Sari-Sarraf
Department of Electrical and Computer Engineering
Texas Tech University
Lubbock, Texas, USA
e-mail: {mesfin.dema, hamed.sari-sarraf}@ttu.edu

***Abstract*—Due to overwhelming use of 3D models in video games and virtual environments, there is a growing interest in 3D scene generation, scene understanding and 3D model retrieval. In this paper, we introduce a data-driven 3D scene generation approach from a Maximum Entropy (MaxEnt) model selection perspective. Using this model selection criterion, new scenes can be sampled by matching a set of contextual constraints that are extracted from training and synthesized scenes. Starting from a set of random synthesized configurations of objects in 3D, the MaxEnt distribution is iteratively sampled (using Metropolis sampling) and updated until the constraints between training and synthesized scenes match, indicating the generation of plausible synthesized 3D scenes. To illustrate the proposed methodology, we use 3D training desk scenes that are all composed of seven predefined objects with different position, scale and orientation arrangements. After applying the MaxEnt framework, the synthesized scenes show that the proposed strategy can generate reasonably similar scenes to the training examples without any human supervision during sampling. We would like to mention, however, that such an approach is not limited to desk scene generation as described here and can be extended to any 3D scene generation problem.**

Keywords-3D scene generation; Maximum Entropy ; Markov Random Field

I. INTRODUCTION

The ever growing interest in video games and virtual environments (such as Second Life [1]) is causing a surge in the availability of 3D models. As a result, access to these 3D models is easier than ever before, especially after the creation of databases such as Google SketchUp 3D warehouse [2]. The abundance of these resources is making it easier for researchers to participate in study areas that focus on 3D scene understanding, scene generation and retrieval. 3D scene generation, through object composition, has been one of the active research areas in the computer graphics community. Over the years, different approaches have been proposed to automate the tedious process of scene generation and among these approaches some of the relevant ones are described briefly in the next section.

II. RELATED WORKS

In [3], Coyne and Sproat showed that a text description can be changed to a corresponding 3D scene. In their work, the scene generation process involves language parsing, semantic information extraction using sets of predefined rules, and 3D scene generation by respecting the semantics. Due to the difficulties of parsing complex sentences, their approach is limited to the generation of simple scenes.

In [4], Xu et al. proposed a constraint-based 3D scene generation approach that is capable of generating complex scenes. In their Constraint-based Automatic Placement System (CAPS), they used pseudo-physics constraints (for enforcing stability of object arrangements) and a set of predefined semantic constraints as a guide for composing complex scenes from objects. Similarly, Arkazawa et al. [5] and Tutenel et al. [6] demonstrated the strength of semantic constraints for 3D scene generation. Taking a different approach is the recent work of Merrell et al. [7] that proposed an interactive furniture arrangement framework for encoding a set of predefined constraints by their fuzzy formulations into a Boltzman-like function, which is then sampled using Metropolis sampling.

Even though the contributions of [3-7] are significant, these methods rely on a predefined set of constraints to represent a given scene. Fisher and Hanrahan [8] and Fisher et al. [9], on the other hand, have shown that contextual constraints can be extracted from a set of training scenes to aid ranking of 3D models for content-based retrieval applications. Similarly, Yu et al. [10] proposed an automatic, data-driven furniture arrangement approach that embraces the work of [7-9]. In their work, which is very similar to [7], furniture arrangement is formulated into a Boltzman-like function that is also sampled using Metropolis sampling.

In this paper, we introduce an automatic, data-driven furniture arrangement approach that can sample plausible scenes from a Gibbs distribution. Unlike [10], which uses very few training data that exhibit small variations of object attributes and arrangements, our approach is designed mainly to accommodate many training scenes that display larger, more realistic object variations. To incorporate these capabilities, we introduce a different constraint representation and also synthesize multiple scenes in a single run of the algorithm as compared with [10]. The advantages of these two main components in scene synthesis are explained next.

A. Constraint Representation

Constraints extracted from training scenes can be represented in a variety of ways depending on the application at hand. In [10], because only a few training scenes were considered, representing the constraints with the first moment was found to be reasonable, adequate and perhaps most importantly, computationally efficient. On the other hand, our main goal here is to synthesize scenes by learning from many training examples that adequately capture the

978-0-7695-4875-3/12 $26.00 © 2012 IEEE
DOI 10.1109/ISM.2012.19

expected object attribute and arrangement variations. As the size of the training examples increases, the corresponding constraints may give rise to multimodal distributions that clearly cannot be represented by the first moment alone. Therefore, we have chosen to represent these constraints with their histograms that can adequately capture the multimodal nature of the constraints when dealing with a large and diverse training data.

B. Number of synthesized scenes

The other main difference between the approach in [10] and our work is the number of synthesized scenes that are produced from a single run of the algorithm. In [10], a single run produces a single synthesized scene, requiring one to run the algorithm multiple times if additional synthesized scenes are desired. A potential problem with this approach is that since each synthesized scene is optimized independently using the same mean-based constraints, the range of variations between the synthesized instances will be small. On the other hand, our approach produces multiple synthesized scenes in a single run of the algorithm and these scenes are optimized *dependently under the same histogram-based constraint*s. This means that among the multiple scenes that are synthesized in a single run of the our algorithm, each scene can sample any of the bins in the histogram and the number of synthesized scenes sampling from a given bin in any histogram will be proportional to the number of training scenes observed to fall in that particular bin (i.e. high probability bins from training constraints will be sampled more frequently than those with low probability). As a result, the variations observed from the training examples are well captured and encoded in the synthesized scenes using our approach.

In addition, it is noteworthy that, unlike the method in [10], other non-contextual variations observed from the training examples (such as object color, texture and type) can also be easily incorporated in our approach while synthesizing scenes.

The proposed approach uses the MaxEnt model selection criterion to formulate the furniture arrangement problem with the Gibbs distribution that is sampled through Metropolis sampling. It should be mentioned that MaxEnt model has been used extensively in the past for different applications such as natural language processing [11], texture synthesis [12], and image parsing [13-15]. To the best of our knowledge, this is the first time that MaxEnt model has been used for 3D scene generation. The scene synthesis approach can be useful in applications such as 3D database amplification, as described in [16], as well as synthetic image generation by introducing a back-end image rendering framework as we proposed in [17].

This paper is organized as follows. Section III first describes the necessary mathematical formulations required to encode furniture arrangement as a Markov Random Field (MRF) problem followed by model selection using MaxEnt criterion. Then after, Section IV and V present the implementation details and the obtained results using the proposed approach, respectively. Finally the last section presents the discussions of the paper.

III. Mathematical Formulations

In this section, we present the mathematical groundwork that is necessary to formulate 3D scene generation as an MRF problem under the MaxEnt model selection criterion.

A. Context Representation

In many computer vision problems, MRF is regarded as an elegant framework to represent contextual constraints with undirected graphs. These undirected graphical models are built from nodes, which define sites of interest, and edges connecting the nodes, which embrace the strength of the local neighborhood structure.

In order to cast 3D scene generation as an MRF problem, we define a scene graph $\mathcal{G}$ as the tuple:

$$\mathcal{G} = \langle \mathcal{V}, \mathcal{R}, \mathcal{P} \rangle \tag{1}$$

where $\mathcal{V}$ represents the nodes defined in the scene, $\mathcal{R}$ represents a set of relationships defined between nodes respecting the Markovian property and $\mathcal{P}$ represents a probabilistic distribution defined on the scene of interest.

In our model, the nodes, $\mathcal{V}$, are defined as high-level objects that can be extracted from COLLADA scene graphs of 3D scenes [18] and the relationships, $\mathcal{R}$, are defined to extract the necessary contextual information, such as relative position, scale and orientations between nodes. Furthermore, we approach the selection of a probability distribution, $\mathcal{P}$, using the MaxEnt criterion in order to identify the most unprejudiced distribution [19].

B. MaxEnt Modeling

With the availability of limited information to identify a generative probability distribution of interest $\mathcal{F}(\mathcal{S})$, one can employ varieties of model selection strategies from which maximum entropy criterion is proven to be the most consistent and the least biased approach [19]. This model selection criterion is briefly described below.

Given an unobserved true distribution $\mathcal{F}(\mathcal{S})$ that generates a particular scene $(\mathcal{S})$, an unbiased distribution $\mathcal{P}(\mathcal{S})$ that approximates $\mathcal{F}(\mathcal{S})$ is the one with maximum entropy, satisfying the constraints simultaneously [19]. Using a set of contextual relationships that can be extracted from the training 3D scenes as observed constraints of $\mathcal{F}(\mathcal{S})$, an unbiased probability distribution is selected using the MaxEnt criterion as follows.

$$\begin{aligned} \tilde{\mathcal{P}}(\mathcal{S}) = argmax\ & \left\{ -\sum \mathcal{P}(\mathcal{S}) \log\big(\mathcal{P}(\mathcal{S})\big) \right\} \\ subject\ to\ & E^i_{\mathcal{F}}[\mathcal{R}(\mathcal{S})] = E^i_{\mathcal{P}}[\mathcal{R}(\mathcal{S})] \\ & i = 1, \dots, N \end{aligned} \tag{2}$$

In (2), E is the expectation defined on $\mathcal{R}$ and N represents the number of constraints (relationships) defined.

Solving the above constrained objective function using Lagrangian method results in the following Gibbs distribution [11-15, 19]

$$\tilde{\mathcal{P}}(\mathcal{S}) = \frac{1}{Z} exp\left\{-\sum_{i=1}^{N} \langle \lambda_i, H_{\mathcal{P}}^{i}[\mathcal{R}(\mathcal{S})] \rangle\right\}. \quad (3)$$

Here, λ_i represents the Lagrange multipliers for the constrained optimization problem, $H_{\mathcal{P}}^{i}[\mathcal{R}(\mathcal{S})]$ represents the histogram of the constraints (used as an approximation to $E_{\mathcal{P}}^{i}[\mathcal{R}(\mathcal{S})]$), Z represents a normalizing constant (partition function) and $\langle \cdot \rangle$ represents the inner product.

The Lagrangian multipliers can be computed by applying Maximum Likelihood Estimate (MLE) over $\tilde{\mathcal{P}}(\mathcal{S})$ and are updated through gradient ascent iteratively as follows [12-15, 20]

$$\lambda_i^{t+1} = \lambda_i^t + \eta\left(H_{\mathcal{P}}^{i}[\mathcal{R}(\mathcal{S})] - H_{\mathcal{F}}^{i}[\mathcal{R}(\mathcal{S})]\right) \quad (4)$$

where η represents the specified learning rate and is set to 1 in our implementation.

Given the Gibbs distribution, one can sample synthetic scenes by applying Metropolis or other Markov Chain Monte Carlo (MCMC) simulation techniques [21] while simultaneously updating the Lagrange multipliers using (4).

Due to the high dimensionality of the solution space for $\tilde{\mathcal{P}}(\mathcal{S})$, optimizing the parameters for all constraints simultaneously is computationally expensive. Instead, a greedy parameter estimation approach is followed that iteratively updates a single parameter while fixing the remaining parameters with feature pursuit strategy as suggested in [12-15].

IV. Implementation

To demonstrate the applicability of the proposed approach, we considered 3D desk scenes. These scenes are represented with 7 nodes (objects) namely: table, chair, laptop, lamp, paper, book and cellphone. To allow intraclass variations of object models, we have downloaded 2 tables, 4 chairs, 4 laptops, 3 lamps, 4 books, 2 cell phones and a single paper from Google SketchUp 3D warehouse.

Manually applying different translations, rotations and scales on individual object models, 19 training scenes are composed (as shown in Fig. 1(a), (b) & (c)) and a dataset $\mathcal{D}$ is created. Thereafter, these 3D scenes are exported from Google SketchUp to COLLADA file format that can be readily imported into MATLAB. Using MATLAB software, by reading the COLLADA files of each scene, 3D points are first extracted and then converted to individual objects with the aid of their annotations, and the corresponding minimal vertical bounding boxes are extracted as shown in Fig. 1(d), (e) & (f).

Once every node in a given training 3D scene is represented as boxes, each node is formulated with an 8-dimensional vector as:

$$v = \left[l, x, y, z, \sigma_M, \sigma_m, \sigma_z, \theta_{xy}\right] \quad , v \in \mathcal{V} \quad (5)$$

where l represents node label; x, y, z represent centroid of the box in 3D; σ_M and σ_m represent major and minor scale, respectively; σ_z represents scale along the Z-axis and θ_{xy} represents orientation along XY-plane, measured between the positive X-axis and the major axis.

After representing every node with an 8D vector, sets of self-explanatory constraints (as defined in Table I) are extracted from the dataset $\mathcal{D}$ while restricting the local neighborhood over a single scene (under the Markovian local neighborhood property).

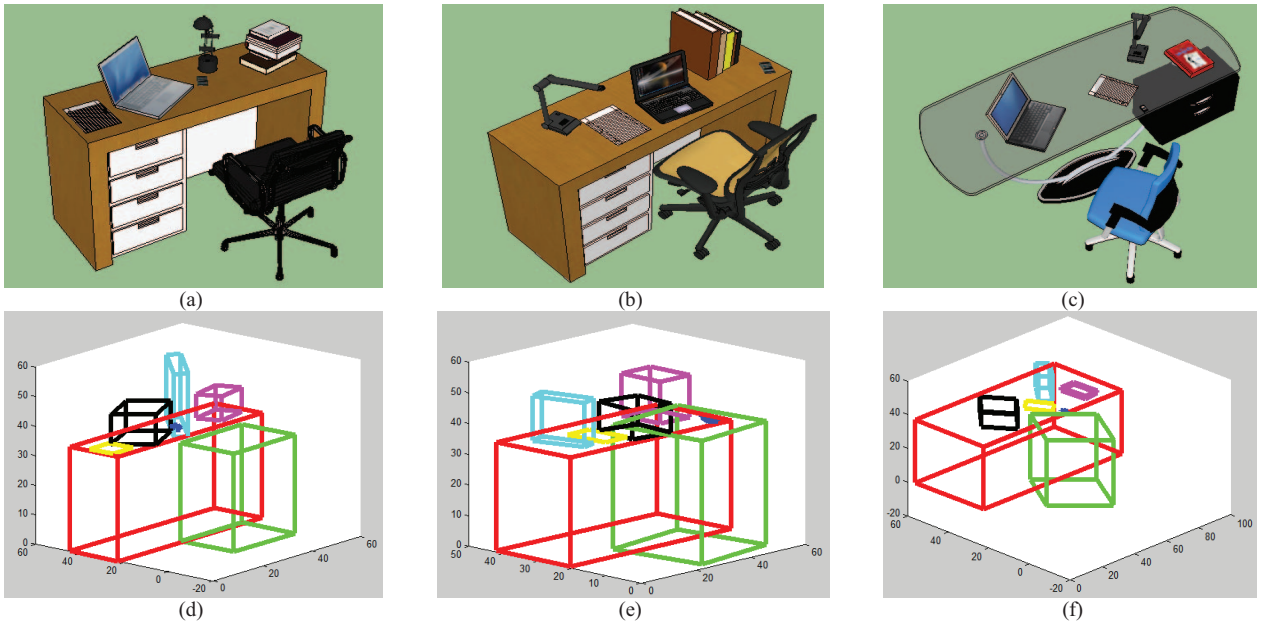

Figure 1. (a)(b)(c) Training scenes. (d)(e)(f) Minimum vertical bounding box for objects with color labels: red=table, green=chair, blue=cellphone, black=laptop, cyan=lamp, magenta=book and yellow=paper.

TABLE I. DEFINED RELATIONSHIPS

Relationships	Formula
Aspect ratio 1	σ_i^m/σ_i^M
Aspect ratio 2	σ_i^z/σ_i^M
Aspect ratio 3	σ_i^z/σ_i^m
Relative position in X axis	$(x_i - x_j)/\sigma_j^M$
Relative position in Y axis	$(y_i - y_j)/\sigma_j^m$
Relative position in Z axis	$(z_i - z_j)/\sigma_j^z$
Relative scale in Major axis	σ_i^M/σ_j^M
Relative scale in Minor axis	σ_i^m/σ_j^m
Relative scale in Z axis	σ_i^z/σ_j^z
Relative Orientation	$\theta_i^{xy} - \theta_j^{xy}$

Using these, a total of 147 pairwise relationships between nodes (objects) and 21 singleton relationships are extracted and converted to normalized histograms (with 30 predefined bin sizes) to serve as observed constraints, $H_{\mathcal{F}}[\mathcal{R}(\mathcal{S})]$. Thereafter, a set of 3D synthetic scenes (we considered 50 scenes in our implementation) are randomly initialized as an initial state of the Markov Chain configuration and corresponding histograms are extracted to serve as approximated constraints , $H_{\mathcal{P}}[\mathcal{R}(\mathcal{S})]$.

Then after, all except the node label in the 8D vectors of the synthetic 3D scenes are updated by sampling from the Gibbs distribution based on the outcome of the transition probabilities defined in the Metropolis sampling. The sampling and parameter learning are continually applied until the two constraints ($H_{\mathcal{P}}[\mathcal{R}(\mathcal{S})]$ and $H_{\mathcal{F}}[\mathcal{R}(\mathcal{S})]$) converge within a pre-specified value as measured by

$$d(H_{\mathcal{F}}, H_{\mathcal{P}}) = \sum_{i=1}^{N} \left(H_{\mathcal{P}}^i[\mathcal{R}(\mathcal{S})] - H_{\mathcal{F}}^i[\mathcal{R}(\mathcal{S})]\right)^2. \qquad (6)$$

Upon convergence of the algorithm, the learned scenes are converted into COLLADA files that eventually are exported into Google SketchUp to produce the 3D scenes. The general framework of the implementation presented in this section is shown in the flow diagram of Fig. 2.

V. RESULTS

In this section, we present realizations of the desk environment that are synthesized using the proposed approach. Using the proposed framework, 50 synthetic desk scenes are initialized randomly by sampling an 8D vector for every node, from which three sample realizations in terms of vertical box representations are shown in Fig. 3(a), (b) & (c). Observing these initial scene configurations, one can clearly identify that the initial states do not capture any recognizable pattern that depict desk scenes in terms of object position, scale and orientation. However, after applying the MaxEnt learning strategy, the starting configurations are updated into reasonable desk scene box representations, as shown in Fig. 3(d), (e) & (f).

After convergence of the process, the bounding box representations are changed into an actual 3D scene. To perform this task, one needs to compose a 3D scene, in Google SketchUp, by first downloading individual 3D object models that are supported in the scene definition (in a very similar way as the training data is prepared except no manual manipulation of the 3D models in translation, rotation and scale is needed). Thereafter, the 3D scene is exported as a COLLADA file and the learned 8D vectors are applied to the corresponding objects to modify their position, orientation and scale.

Finally, the COLLADA file is imported back to Google SketchUp to generate the actual synthesized 3D scene. To demonstrate this idea, we selected the 7 object models that we define to produce desk scene and configure them as shown in Fig. 4. In this figure, we made our best effort to manually scale all the objects with the same proportion in XYZ axes, orient them in the same direction and place them horizontally to make visual comparison easy.

In order to generate the actual scene, all 7 objects are initially translated to the origin and the learned parameters that produced the outputs in Fig. 3(d), (e) & (f) are fed into the corresponding COLLADA files. Thereafter, these COLLADA files are imported into Google SketchUp to produce the synthesized scenes as shown in Fig. 5. Visual inspection of the results shown in Fig. 5 clearly illustrate that the proposed approach has the capability to learn from few training scenes and capture this pattern to synthesize plausible synthetic scenes as output.

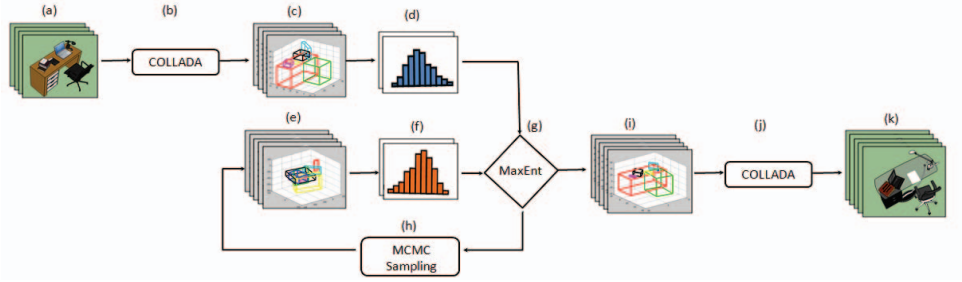

Figure 2: Flow diagram of the proposed approach. (a) Input 3D training scenes, (b) COLLADA files from (a), (c) minimal bounding box representation for (b), (d) histogram representation of relationships from (c), (e) minimal bounding box representation of synthetic scenes, (f) histogram representation of relationships from (e), (g) MaxEnt learning until convergence, (h) MCMC sampling of synthetic scenes, (i) output minimal bounding box representation of synthetic scenes, (j) generate COLLADA files from (i), (k) synthesized 3D scenes.

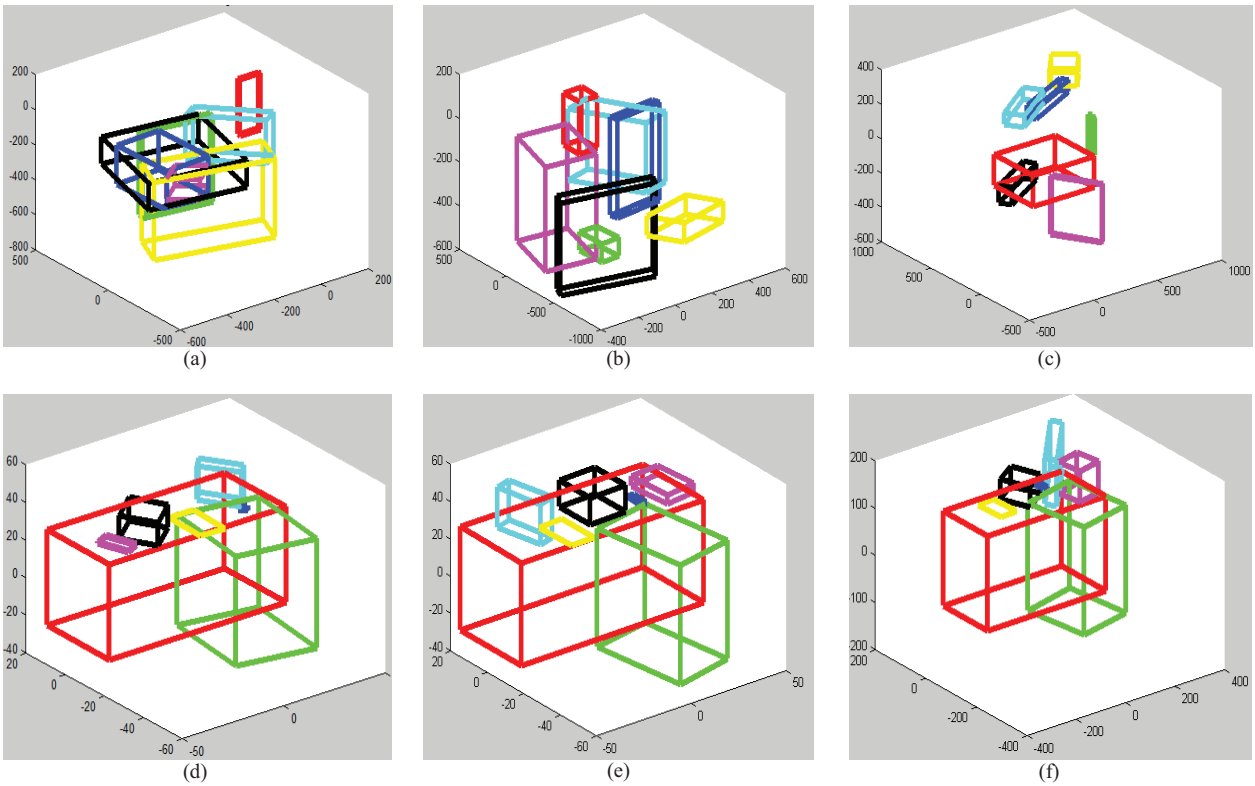

Figure 3. Synthesized desk scenes. (a)(b)(c) Initial random scenes, (d)(e)(f) After learning through the proposed approach with color label: red=table, green=chair, blue=cellphone, black=laptop, cyan=lamp, magenta=book and yellow=paper.

Figure 4. 3D object models [from left to right: chair, lamp, table (with cabinet), book, paper (stack), cellphone and laptop].

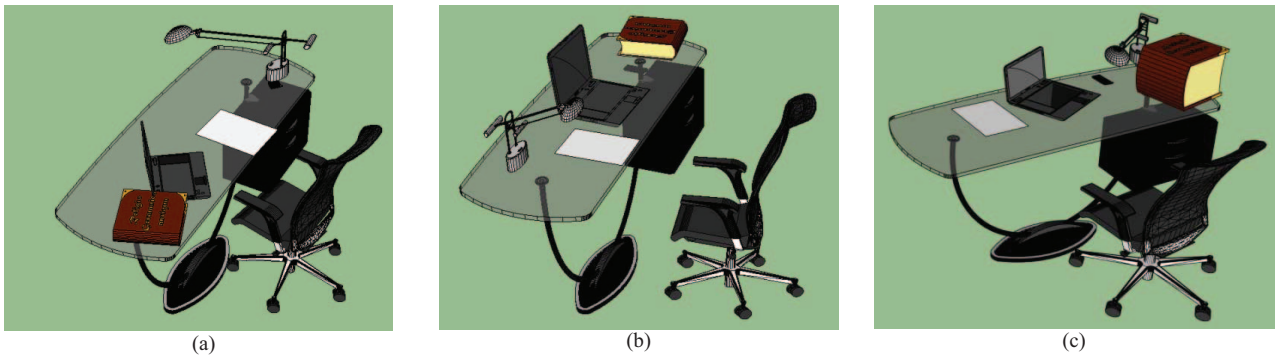

Figure 5. Synthesized desk scenes in Google SketchUp

A. Analysis

In order to assess the performance of the proposed framework, we collected the responses of three independent observers about the quality of the synthesized scenes. In this study, the observers were first presented with the minimal bounding box representation of the training examples as shown in Fig. 1(d), (e) & (f). Thereafter, we requested them to grade the 50 synthesized scenes on a scale of one to ten; ten being the highest quality. We later converted their response into three qualitative output categories: "bad" ranging from scale 1 to 3.3, "need improvement" ranging from scale 3.4 to 6.7 and "good" ranging from scale 6.8 to 10.

After averaging their responses, we found that 17% of the scenes were categorized as "bad", 27% of the scenes were categorized as "need improvement" and the rest, 56%, of the scenes were categorized as "good". This means that using only 19 training scenes, we sampled 28 new scenes that are plausible interpretation of the training data in a single run of the algorithm. Therefore, it indicates that the proposed approach has demonstrated its learning capability of 3D scenes from few training examples.

Nonetheless, it also points that further work is required to improve the performance of the approach. Investigating some of the outputs categorized under "need improvement" and "bad", we discovered that in most of these cases, objects are levitating in the air (outside the range of the table), as shown in Fig. 6. Adding constraints will improve performance (i.e. higher percent of plausible outcomes) but at the cost of increased computational complexity. Therefore, one has to trade-off between performance and computation when synthesizing scenes.

VI. Discussions

In this paper, we have introduced a new approach to 3D scene generation using the MaxEnt framework. Using a set of training 3D desk scenes, which are composed of 7 predefined objects, we have extracted more than 150 constraints that define the scene. These constraints are then utilized in a generative framework to sample a set of synthetic 3D scenes whose constraints capture the training data.

The proposed approach embraces a constraint extraction strategy from a set of training examples and a learning framework to sample plausible 3D scenes from the Gibbs distribution. As a result, the proposed framework is very extensible to most of 3D scene generation problems without requiring much algorithm tuning.

Even though the importance of the proposed approach for 3D scene generation is unquestionable, there are issues that need to be addressed. Firstly, the proposed framework is computationally expensive (requiring orders of hours to synthesize scenes) and further work is needed to reduce the incurred computational cost. The other issue is that the model produces 3D scenes with defect in object position and scale that do not satisfy the observed statistics from training samples. Therefore, we are currently studying approaches that can mitigate these issues for better usage of the framework for 3D scene generation.

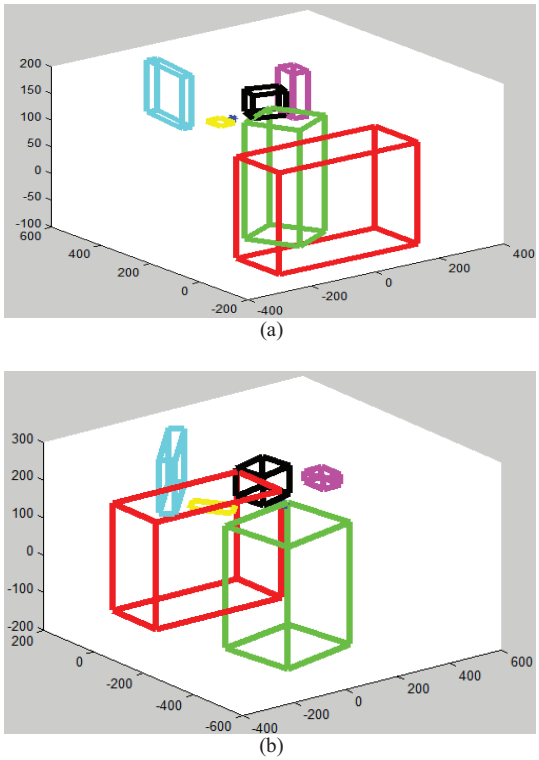

Figure 6. (a) "bad" output, (b) "need improvement" output

Acknowledgment

We would like to thank Dr. Jake Porway for his timely and valuable comments and advice on the subject matter.

References

[1] http://secondlife.com/ [Accessed in 06, 2012]

[2] http://sketchup.google.com/ [Accessed in 06, 2012]

[3] B. Coyne, and R. Sproat, "Wordseye: An automatic text-to-scene conversion system", In SIGGRAPH, 2001

[4] K. Xu, J. Stewart,and E. Fiume, "Constraint-based automatic placement for scene composition", In Graphics Interface, 25–34, 2002

[5] Y. Akazawa, Y. Okada, and K. Niijima, "Automatic 3D scene generation based on contact constraints", In 3IA International Conference on Computer Graphics and Artificial Intelligence, 2005

[6] T. Tutenel, R.M. Smelik, R. Bidarra, K. J. D. Kraker, "Using Semantics to Improve the Design of Game Worlds", In Artificial Intelligence and Interactive Digital Entertainment, 2009

[7] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun, "Interactive Furniture Layout Using Interior Design Guidelines", In SIGGRAPH, 2011

[8] M. Fisher, and P. Hanrahan, "Context-based search for 3D models", In SIGGRAPH Asia, 2010

[9] M. Fisher, M. Savva, and P. Hanrahan, “Characterizing Structural Relationships in Scenes Using Graph Kernels”, In SIGGRAPH, 2011

[10] L. Yu, S. Yeung, C. Tang, D. Terzopoulos, T. F. Chan, and S.J. Osher,” Make it home: Automatic optimization of furniture arrangement”, In SIGGRAPH, 2011

[11] A. Berger, S. D. Pietra, and V. D. Pietra,”A maximum entropy approach to natural language processing”, Computational Linguistics, 1996

[12] S. C. Zhu, Y. Wu, and D. Mumford, “Filters, Random Fields and Maximum Entropy (FRAME): Towards a unified theory for texture modeling”, International Journal of Computer Vision, 1998

[13] C. Liu, S.C. Zhu, and H.Y. Shum, “Learning inhomogeneous Gibbs model of faces by Minimax Entropy”, In International Conference of Computer Vision, 2001

[14] J. Porway, B. Yao, and S. C. Zhu , “Learning compositional models for object categories from small sample sets”, In Object Categorization: Computer and Human Vision Perspectives, B. S. Sven Dickson, Ales Leonardis and M. J.Tarr, Eds. Cambridge Press, 2009

[15] J. Porway, Q. Wang, and S. C. Zhu, “A hierarchichal and contextual model for aerial image parsing”, International Journal of Computer Vision, 2010

[16] E. Kalogerakis, S. Chaudhuri, D. Koller, and V. Koltun, “A probabilistic model for component-based shape synthesis”, In SIGGRAPH, 2012

[17] S. Gleason, M. Dema, H. Sari-Sarraf, A. Cheriyadat, R. Vatsavai, and R. Ferrell. “Verification & validation of a semantic image tagging framework via generation of geospatial imagery ground truth”, In IGARSS, 2011

[18] https://collada.org [Accessed in 06, 2012]

[19] E. Jaynes, “Discrete prior probabilities: The entropy principle”, In Probability Theory: The Logic of Science, G. L. Bretthorst, Ed. Cambridge Press, 343–371, 2003

[20] R. Malouf, “Maximum entropy models”, In Handbook of Computational Linguistics and Natural Language Processing, C. F. Alex Clark and S. Lappin, Eds.Wiley Blackwell, 133–155. 2010

[21] B. Walsh,” Markov Chain Monte Carlo and Gibbs Sampling”, Lecture Notes, 2004