Graphical Models 85 (2016) 46–55

Contents lists available at ScienceDirect

Graphical Models

journal homepage: www.elsevier.com/locate/gmod

Structure guided interior scene synthesis via graph matching

Shi-Sheng Huang[a,*], Hongbo Fu[b], Shi-Min Hu[a]

[a] Tsinghua University, Beijing, China
[b] City University of Hong Kong, Hong Kong, China

ARTICLE INFO

Article history:
Received 14 December 2015
Revised 4 March 2016
Accepted 24 March 2016
Available online 6 May 2016

Keywords:
Scene synthesis
Graph matching
Augmented graph

ABSTRACT

We present a method for reshuffle-based 3D interior scene synthesis guided by scene structures. Given several 3D scenes, we form each 3D scene as a structure graph associated with a relationship set. Considering both the object similarity and relation similarity, we then establish a furniture-object-based matching between scene pairs via graph matching. Such a matching allows us to merge the structure graphs into a unified structure, i.e., *Augmented Graph* (*AG*). Guided by the *AG*, we perform scene synthesis by reshuffling objects through three simple operations, i.e., *replacing*, *growing* and *transfer*. A synthesis compatibility measure considering the environment of the furniture objects is also introduced to filter out poor-quality results. We show that our method is able to generate high-quality scene variations and outperforms the state of the art.

1. Introduction

Recently, 3D interior scenes have received more and more attention due to the huge demand in the industries such as computer games and virtual reality. However, designing and creating 3D digital scenes are still time-consuming even for artists. Fisher et al. [1] provided an efficient solution for 3D scene synthesis from examples based on moderate-to-large scene datasets. However, such a learning based algorithm is still complicated due to the complexity for data collection. Besides, the learned probability model might not always achieve user-desired constraints, such as rigid grid layouts or exact alignment relationships. Such issues might be solved by utilizing the original examples rather than learning from the example dataset.

It is still a desirable way to synthesize scenes directly from a small set of 3D scene examples without learning algorithms. Starting from such a point, Xie et al. [2] introduced a non-learning-based scene synthesis method by grouping the furniture objects into different types of units and reshuffling the interchangeable objects from the same units. Although their method could generate some kind of diverse new scenes, it is still rather limited due to the limited grouping types. In addition, their local analysis ignored the scene's layout structure information, which is, however, a very important guidance cue for scene generation. We observed that there is a latent rule in the layout distribution of the scene furniture objects *locally* and *globally*. Locally, furniture objects often 'contact' with each other, following a certain kind of relation. For example, a *chair* often closely faces a *table*, and a *bedside cabinet* is always at one side of a *bed* with one side aligned. Globally, these furniture objects with the local relationships form a layout structure. Based on the above observations, we carefully analyze the layout structures of the exemplar scenes and synthesize new scenes utilizing the relations between the layout structures, which have not been explored by Xie et al. [2].

Given several 3D interior scenes as examples, our goal is to synthesize new scenes with variations using a geometric approach rather than a learning-based strategy. Although furniture objects vary a lot in geometry, they latently relate with each other according to the relations among objects. In this paper, we first define five kinds of relations between furniture objects (Fig. 4(a-e)), i.e., *support relation*, *vertical contact relation*, *facing relation*, *aligned relation* and *close relation*, which widely exist in the 3D interior scenes. Then we represent each 3D scene as a structure graph. Our structure graph is different from the previous ones [2], since we associate a relationship set rather than a single relationship with each edge in the structure graph. We establish a matching between the layout subgraphs (Fig. 4(f)) via graph matching, which provides a cue to relate two structure graphs. Based on the matching, we *merge* the scene structures into an *Augmented Graph* (*AG*), which encodes all the layout structure information among the examples. We then utilize the *AG* to guide scene synthesis by using several simple and efficient operations, i.e., *replacing*, *growing* and *transfer*. The *growing* operation is especially efficient for adding a new object. These operations provide a flexible and user-friendly way to synthesize diverse scenes. To evaluate scene quality and avoid

* Corresponding author.
E-mail address: shishenghuang0@gmail.com (S.-S. Huang).

http://dx.doi.org/10.1016/j.gmod.2016.03.004
1524-0703/

low-quality scenes during the synthesis, we introduce a synthesis compatibility value to measure each synthesis operation and the quality of a resulting scene.

Our main contribution lies in the following three points:

(1) We represent a 3D interior scene as a structure graph associated with a relationship set, and introduce a furniture object matching method between scene pairs via graph matching. Our scene matching is general and efficient, which can be used for other applications besides scene synthesis.
(2) We introduce a unified structure, *Augmented Graph*, to encode all the layout information from examples, augmented from the matched structure graphs. Guided by the *AG*, we provide three simple reshuffle-based synthesis operations, i.e., *replacing*, *growing* and *transfer*, to generate diverse new scenes.
(3) We also introduce a synthesis compatibility metric to measure scene quality during the synthesis, making it efficient to filter out poor quality synthesis results.

2. Related work

It has still been a challenging problem for rapidly designing and creating 3D contents, such as shapes and 3D scenes. In recent years, continuous progresses have been made for shape processing (see [3] for more details). Here we only focus on example-based manipulation and analysis for shapes and 3D scenes.

Part-based shape synthesis. Shape synthesis by reusing parts is an efficient way to generate new shapes [4]. Jain et al. [5] provided a method to synthesize man-made objects by blending a small set of segmented shapes via recombining their object parts. Functional structure plays an important role in shape understanding. Recently, a few part-based substructures have been introduced, such as sFARR-s structure [6], Support Substructure [7] and Replaceable Substructure [8]. Based on such part-based substructures, shape synthesis with plausible results can be efficiently achieved with functional computability maintained. Evangelos et al. [9] learned a probability model from a moderate-to-large shapeset to guide shape synthesis. The data-driven part-based algorithms also show us the efficiency in 3D modeling [10,11]. Xu et al. [12] introduced an approach to generate new shapes via set evolution. Recently, Alhashim et al. [13] provided a shape blending method to synthesize new shapes by topology varying. Our reshuffle-based algorithm is inspired by these part-based shape synthesis methods, which can be extended to scene synthesis.

Scene analysis. In general, interior scene understanding is a challenging problem due to the variation in object geometry and functional arrangement. The relationship between objects can be used as a useful cue to guide scene analysis, especially for scene matching and retrieval. Object retrieval can be enhanced using the context information [14]. Fisher et al. [15] introduced an efficient method to compare scene objects using Graph Kernel defined on a relation graph. Learning based algorithms have also been introduced to synthesize scenes. For example, Fisher et al. [1] proposed a learning based method to synthesize 3D scenes from a given scene set. Su et al. [16] proposed a probabilistic scene model using object frames. In contrast, our method is not learning-based and directly synthesizes scenes from examples. Xu et al. [17] provided a method to organize heterogeneous scene collection using focal points. Recently, Liu et al. [18] provided a method to infer consistent grouping information via parsing with a probabilistic grammar learned from examples. Existing works has paid more attention on the similarity between pairs of either single objects or scenes and few works have studied the matching of furniture objects in the layout structures.

Scene reconstruction. Our work is also related to scene reconstruction. Recently there has been a significant progress on scene reconstruction from LiDAR data [19,20] and RGB-D data [21]. Xu et al. [22] presented a novel approach to reconstruct 3D scenes from user-drawn rough sketches. Scenes reconstructed from sensor data always lack semantic labels or tags, which are time-consuming to manually label or tag. Our method thus aims to handle scenes without any category labels.

Our work is closely related to [1], both aiming at synthesizing scenes by example. However, our algorithm is non-learning based, and works well for a small number of examples, reducing the complexity on constructing a large dataset of scenes. Our work also bears close resemblance to [2]: both of them are reshuffle-based scene synthesis. However, our approach is more flexible on scene structures since structures and relations in [2] are very limited.

3. Overview

Inspired by the part-based methods for shape synthesis and scene analysis (see the discussions in the previous section), we provide a structure-guided method for synthesizing 3D interior scenes from a small set of examples. Our input is a small set of exemplar 3D interior scenes. Each interior scene has been segmented into single furniture objects (Fig. 1) and oriented uprightly [23]. As discussed previously our approach does not need the furniture objects to be semantically tagged or labeled. We also assume that the facing direction of each furniture object is available (see Fig. 2). In general, the shape's orientation detection is not an easy problem on its own. We use the prior knowledge of each 3D scene to determine the facing direction of each furniture object (see Section 4 for more details).

As shown in Fig. 1, our method involves three stages. In the first stage, we extract all the relations in each scene according to the five relations to be formally introduced later. Then we represent each scene as a structure graph $G = \{V, E, A\}$ by associating a relationship set $A_e \in A$ with each edge $e \in E$. In the second stage, we perform scene matching between scene pairs based on the layout subgraphs G_L (Fig. 4 (e)). Following the scene matching scheme, we introduce a greedy method to augment all the scene graphs into a unified *Augmented Graph* (*AG*). In the last stage, we perform scene synthesis according to the three scene synthesis operations defined on the *Augmented Graph* (*AG*). During the synthesis, we use the synthesis compatibility measurement to filter out poor synthesis operations.

Next we first introduce how we extract the relations and perform scene matching via graph matching in Section 4. In Section 5, we will discuss how to augment a set of structure graphs into a final *AG* and to perform scene synthesis guided by *AG*. We will present lots of diverse results synthesized by our algorithm in Section 6 and conclude the paper in Section 7.

4. Scene matching

In this section we first show how we determine the facing direction of each furniture object. First, we compute a symmetry plane (if any) (Fig. 3) for each furniture object. The facing direction is always parallel to the symmetry plane. Users can specify the facing direction manually if none or multiple symmetry planes exist. In general, furniture objects which are located in the boundary region of a 3D scene often have facing directions pointing to the scene's center. This motivated us to assign the facing directions of such boundary objects as the directions which are parallel to their symmetry planes and point to the scene's center. Then for the rest of the furniture objects, their facing directions are assigned as the directions parallel with the symmetry planes and pointing to the nearest objects.