Controllable Scene Generation from Natural Language

Cheng, Yu; Sun, Zhiyong; Shi, Yan; Dong, Lixin

Published in:
Procedia Computer Science

Published: 01/01/2022

Document Version:
Final Published version, also known as Publisher's PDF, Publisher's Final version or Version of Record

Published version (DOI):
10.1016/j.procs.2022.10.106

Publication details:
Cheng, Y., Sun, Z., Shi, Y., & Dong, L. (2022). Controllable Scene Generation from Natural Language. *Procedia Computer Science*, *209*, 122-131. https://doi.org/10.1016/j.procs.2022.10.106

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 209 (2022) 122–131

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the 2022 International Symposium on Biomimetic Intelligence and Robotics (ISBIR)

Controllable Scene Generation from Natural Language

Yu Cheng[a,*], Zhiyong Sun[a], Yan Shi[a], Lixin Dong[b]

[a]*Department of Electrical and Computer Engineering, Michigan State University, East Lansing, 48824, USA*
[b]*Department of Biomedical Engineering, City University of Hong Kong, Hong Kong, 999077, China*

Abstract

We propose a novel framework to generate recognizable scenes conditioned on natural language (NL) descriptions. The proposed modular approach decomposes the scene synthesis process into several manageable steps, in which it first infers a spatial layout of the desired scene from input descriptions by a spatial layout generator and generates the scene with a scene generator. Specifically, the proposed approach allows interactive tuning of the synthesized scene via NL, which helps to generate more complex and meaningful scenes, and to correct training errors or bias. We demonstrate the capability of the proposed approach on the challenging MS-COCO dataset and show that our approach can improve the quality of generated scenes, interpretability of the drawn scenes and semantic alignment to the input language descriptions.

Keywords: ; scene generation; natural language; discrete event system

1. Introduction

NL Control utilizes the rich expressibility of NL to deliver task requirements, complementary knowledge and information from a human instructor, which can benefit both the machines and human users. Existing approaches of NL based control focus on modeling the mapping relationship between language and primitive actions [14] [9]. Symbolic representations are generated to represent the task goal and specifications, and then are parameterized for reasoning of action selection. The success of existing approaches relies on well-trained mapping and correct language grounding using environmental information from sensors and knowledge from hand-engineered knowledge base. However, if the tasks are not dependent on environmental information, or there doesn't exist a well-built knowledge base for the machine to use, it is difficult to perform the task. As illustrated in Figure 1, a machine is tasked to synthesize a scene with NL description "*A lovely dog sits in front of a sofa in a room. A chair is at the left side of the*

☆ The first two authors contributed equally.

* Corresponding author.
E-mail address: chengyu9@msu.edu

1877-0509
10.1016/j.procs.2022.10.106

dog". It has to not only infer the context (*room*), object type (*sofa*, *dog*, *chair*), object property (*lovely*), the spatial layout among objects (*in*, *left*, *front*), but also generate motion plans to draw the scene on paper. There are a huge number of choices for object types and object properties (e.g., color, shape, texture, etc.), and it is nontrivial to build or train such a knowledge base.

Scene generation conditioned on NL has many applications. One important application is literacy development. For children who are learning to read and for second language learners, seeing scenes together with language can enhance learning [13]. Another application is as a reading helper for people with learning disabilities or brain damage. The machine can convert textual menus, signs, and operating instructions into graphical representations. In addition, scene generation conditioned on NL can extend to industrial application scenarios, such as robotic painting and polishing.

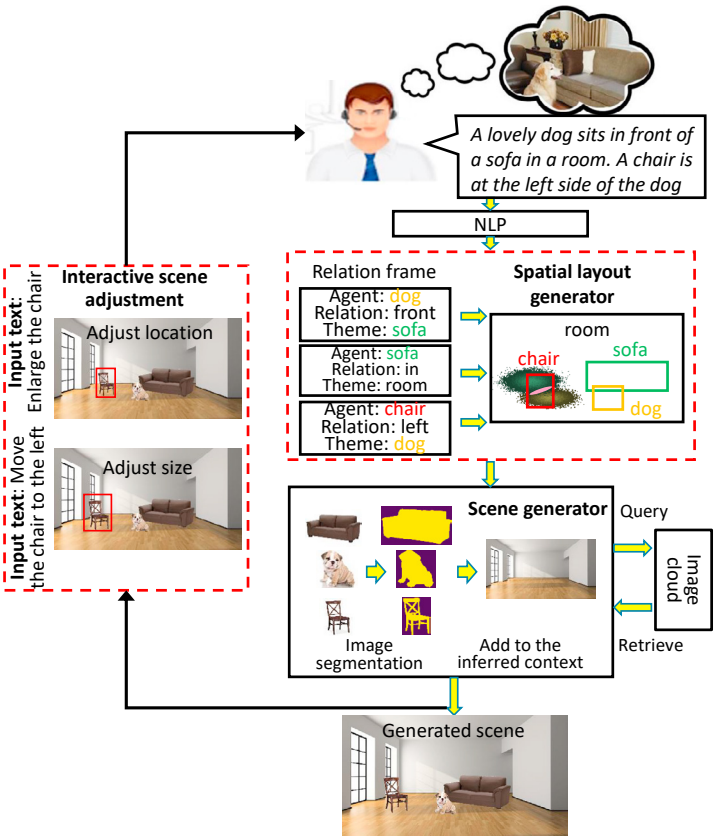


Fig. 1. Overall pipeline of the proposed approach. Given an language description, the proposed approach parses text into relation frames that captures semantic specification of the desired scene, and uses relation frame information to infer spatial layout of objects (circled in red dashed rectangle. The relations frames shown in this figure ignore information of *Verb* and *Property* for simplicity purposes. The dark yellow cloud shaped area represents relative left locations of the dog, and the dark green cloud shaped area denotes relative left locations of the sofa. The centroid position of the chair is randomly selected from the intersection area (marked in pink)). The scene generator retrieves images using relation frame information and segment objects from images. The extracted object instances are placed together to meet the spatial configuration. The instructor can adjust the synthesized scene by replacing objects, change size and location of objects, etc.

Some related research have been conducted on scene generation conditioned on text along two directions: object level and pixel level. Object level scene generation arranges objects to generate scenes in accordance with inferred spatial layout. While the later maps text to most likely object pixel distributions trained from data.

For object level scene synthesis, [25] use nouns appeared in the text as keywords to search for images of object instances and then combine them together to represent the meaning of the input text. Only object type information is used in scene generation, while visual attributes of objects and spatial layout information are ignored. [26] train a conditional random field model to generate clipart scenes using clipart characters. [3] employ Bayesian probability to

generate 3D office scenes using objects from a 3D model database. These two works utilize spatial relations among objects. The visual attributes of objects are not considered in scene generation.

More recent works focus on pixel level scene generation. Generative models are trained for text-to-pixel mapping. [23] use variational auto-encoder to generate scenes conditioned on visual attributes. [17] train generative adversarial networks (GANs) for scene generation. These approaches are limited to single objects, such as birds [21] and flowers [15]. [12] synthesize scenes on more complicated scene descriptions (MS-COCO) using a variational recurrent autoencoder with attention. [22] propose stacked GANs structure to generate and refine scenes for better recognizability than single GANs. [5] use cascaded convolutional neural network and recurrent neural network with GAN to augment the visual features of object in scene. The major focus of pixel level scene generation is on visual features, while spatial mappings are ignored.

In this paper, we propose a hierarchical approach to generate scenes in a compositional manner. This approach has two components: a spatial layout generator and a scene generator. The spatial layout generator infers spatial configuration among objects. The scene generator firstly retrieves images through query with object types and object properties from a ”dirty” image cloud (the open Internet dataset collected from distributed sources are often ”dirty” with erroneous or corrupted data [7]). Then the object instances are segmented from images and assembled together in accordance with the estimated spatial layout. In addition, the scene generator is modeled using supervisory control, which helps to deal with unexpected events in image retrieval and scene generation and allows interactive modification of generated scenes via NL.

The main contributions of the paper are as follows:

- We propose a novel approach for scene generation conditioned on NL descriptions. Compared with existing approaches, the proposed method employs both information of object property and spatial configuration for scene generation, which helps to generate more complex and realistic scenes.
- The proposed approach offers an interactive framework and interface to control scene generation process: users are allowed to modify the synthesized scene by replacing object instances, changing size and location of objects through NL. This helps to correct possible training errors or bias using human experience. As a result, the generated scenes can be more recognizable and well-aligned with the descriptions.
- Instead of interacting with a well-built knowledge base aimed for robots to use, the scene generator interacts with a ”dirty” image cloud and tackles unexpected events during scene generation.
- Quantitative and qualitative evaluations have been conducted on MS-COCO dataset, and demonstrate improvement on quality of the scene generation over baseline works.

The rest of the paper is organized as follows. Section II illustrates the proposed approach in detail. Section III discusses the experimental results. Finally, Section IV concludes the paper.

2. Scene Generation Approach

The overall pipeline of the proposed framework is illustrated in Figure 1. Given a language description of a scene, the robotic painter draws the generated scene on paper through the sequence of following modules.

2.1. Natural Language Processing

To convert the users instructions into a formal specification, the system must identify the underlying linguistic structure of the description and convert it into a logical representation. This section describes the process of this conversion and its implementation. The scene descriptions are processed through a pipeline of natural language components which identify the syntactic structure of the sentences, extract semantic information from them, and create formal representation to be used in action sequence synthesis. Different from many previous natural language based control systems that have relied on per-scenario grammars that combine semantic and syntactic information, this work uses a combination of robust, general-purpose components for parsing and tagging the input. An advantage of this approach compared to per-scenario grammars is that the core language models need not be modified across scenarios.

This reduces the role of the fragile process of grammar engineering and minimizes the cost of adapting the system to handle commands in new domains.

The NLP module uses a pipeline of domain-general natural language processing components. The input is parsed using AllenNLP semantic role labeling (SRL) parser to identify verbs and arguments [6]. Each argument is classified into one type of PropBank modifiers which indicates the argument's semantic function in the sentence [1]. Then each argument is tagged using Stanford Log-linear POS Tagger to identify the subject and corresponding properties as well as possible relations with neighboring items appeared in the sentence [8]. The identified information is matched to a relation frame (*Agent, Agent property, Verb, Relation, Theme, Theme property*). The expected relation (*Relation*) between the two items (*Agent* and *Theme*), specification of the items (*Agent property* and *Theme property*), and pairwise position dependency (*Agent* depends on *Theme*) are identified using their tags and syntactic positions.

2.2. Spatial Layout Generator

Spatial layout of objects is required in order to generate scenes using object instances extracted from real images. Pairwise spatial relations acquired through the first step of language processing contain partial information of the layout. The subfigure circled out with red dashed rectangle in Figure 1 shows an example of the inferred spatial layout corresponding to the input description *A lovely dog sits in front of a sofa in a room. A chair is at the left side of the dog*. Position of each object instance is determined by its surrounding objects on which it is dependent as described in the language. For example, the chair's position is directly dependent on the dog, and the dog's position depends on the sofa. As a result, only after the placement of the sofa and dog, the chair's position can be figured out using position and size information of the dog and the sofa.

In this work we use the dependency relation between objects to help the spatial configuration inference [4]. An agent is considered to be dependent on its theme. The more objects an object is dependent on, the higher dependency value it has, the latter it will be added into the scene. An ordered object manipulation sequence can be figured out based on the dependency values in an ascending manner. Usually, the first object in the sequence is the context because all the other objects are dependent on it. If the context is not explicitly specified in the language, it will be inferred using the objects appeared in the scene description:

$$P(context|object_{1:m}) = \prod_{i=1}^{m} P(context|object_i) \tag{1}$$

the context type that maximize the probability will be selected. In this work we consider five context types: indoor, road, city plaza, rural field, and sea shore (if a context is explicitly specified in the description, the scene generator can retrieve candidate context images from cloud resources; otherwise, the context will be inferred from the five categories). The priors for object occurrence in different context types are trained using 1708 scenes from MS-COCO training dataset (the object detector we used in this work can recognize 20 category of objects, so we filtered the dataset to get scenes comprised of objects lay in the category set):

$$P(context|object_i) = \frac{count(object_i \text{ in } context)}{count(context)} \tag{2}$$

After determining the context, each time when adding an object into the scene, its position is calculated based on other objects on which it is dependent. The spatial knowledge is trained using relative location data from [26], which consists of 10020 scenes created using 58 clipart objects. The centroid position of an object is randomly chosen from the intersection area of its relative locations in each spatial relation with a dependent object (theme). The dimensional ratio between each two object types is set as a priori.

2.3. Scene Generator

Scene generator retrieves images containing specified objects from an image cloud (in this work, the images are retrieved using Google image search engine) and synthesizes a scene using segmented objects processed from retrieved images. Object instances are from a "dirty" image cloud rather than manually labeled datasets [26] [3]. Unexpected events may happen and cause failures of scene generation, such as detection failure of objects in the retrieved images. In addition, it helps to generate semantically meaningful scenes if a human instructor can tune the synthesized scene through NL. To this end, a dynamic discrete event model is developed to tackle with unexpected events and interactive scene modification. Firstly, some basics of supervisory control theory (SCT) that we use to build the system model is illustrated [16]. Then the modeling process using SCT is illustrated in detail. Finally, property analysis of developed model is conducted to refine the model to guarantee successful scene generation.

2.3.1. Preliminaries

Different from the continuous variable dynamic systems (CVDS), where the system behaviors are governed by physical laws and modeled by differential equations, we model the system controlled by natural language as a discrete event dynamic system (DEDS), where its behaviors evolve in accordance with rules of operation or algorithms. Supervisory control is one of the modeling theory of DEDS, which focuses on maintaining the system behavior described by formal language. In the context of supervisory control, a DEDS is modeled using a five-tuple automaton: $G = \{Q, \Sigma, \delta, q_0, Q_m\}$. G denotes the plant to be controlled.

Q is a finite nonempty set of states abstracted from the system, denoting the status of the plant. Σ represents the set of events, in which the elements drive the system to evolve from one state to another (in this paper, event and action are used interchangeably). The events in Σ are classified into two categories: controllable event set Σ_c in which events can either be enabled or disabled, and uncontrollable event set Σ_{uc} where events are set to be always enabled by default. The language generated by an automaton is comprised of elements from the event set Σ, it is also called an alphabet. $\delta : \Sigma \times Q \rightarrow Q$ is the transition function that captures conditional state changes. q_0 denotes the initial state from where a language or a system starts. $Q_m \subseteq Q$: a subset of the state set Q, called the set of marker states. Usually, Q_m marks the termination of successfully implemented tasks.

Let Σ^* represents the set of all the finite strings s comprised of elements from Σ, including the empty string ε. The language $L(G)$ is the set of all event trajectories that are physically possible for the plant $L(G) = \{s : s \in \Sigma^*,\ \delta(q_0, s)!,\ q_0 \in Q_0\}$, where "!" means "is defined". The marker language $L_m(G)$, describes all the event sequences that can lead to the marker states $L_m(G) = \{s : s \in \Sigma^*,\ \delta(q_0, s) \in Q_m,\ q_0 \in Q_0\}$.

Supervisory control scheme separates the open loop dynamics (plant) from the feedback control. This separation is reminiscent of the feedback control scheme adopted in CVDS. The controller is modeled as a pair $S = (\mathfrak{R}, \psi)$ where $\mathfrak{R} = \{X, \Sigma_s, \xi, x_0, X_m\}$ is a deterministic automaton with state set X, event set Σ_s (Σ_s and Σ share the same event space), transition function ξ, initial state x_0, and marker state $X_m \subseteq X$. The ψ is a total function that maps supervisor states x into control patterns, which is defined as:

$$\phi(x)(\sigma) = \begin{cases} 1, & \text{for each } \sigma \in \Sigma_{uc} \text{ or } \sigma \text{ is allowed at x} \\ 0, & \text{for } \sigma \text{ not allowed at x} \end{cases}$$

The supervisor is synthesized by regulating the plant behaviors with physical constrains and task specifications. The sets of its behaviors, represented by formal language, are denoted as $L(S/G)$ and $L_m(S/G)$, respectively.

2.3.2. Modeling of Scene Generator

Receiving matched frames and inferred spatial layout, the scene generator retrieves images that probably contain desired objects from image cloud, segment the objects from their original images, and assembles them together in accordance with the required spatial layout. In interactive scene tuning phase, the scene generator replaces objects, modifies position and size of objects following the instructor's commands. Table 1 presents the event set and control-

lability of each event. The plant model is a shuffle product of the plant models:

$$G = \|_{i=1}^{8} G_i \tag{3}$$

where || represent parallel composition operation. The shuffled plant model has 896 states and 4096 transitions in total.

Table 1. List of basic actions for scene synthesis.

Primitive Actions	Controllability
retrieve_img (obj)	controllable
resize (obj)	controllable
detect (obj)	controllable
compare (obj1, obj2)	controllable
segment (obj)	controllable
merge (obj, context)	controllable
affirmative	uncontrollable
negative	uncontrollable
change_img (obj)	controllable
zoom_in (obj)	controllable
zoom_out (obj)	controllable
move (obj)	controllable
save (obj, img)	controllable
stop	controllable

When the system receives language descriptions of a scene, it retrieves images that ranked high in the query results using keyword combinations of (*Agent* + *Agent Property*) or (*Theme* + *Theme Property*). Then the system uses object detector to detect desired objects from retrieved images. If an image contains the object, then the object is extracted and resized to fit into the context. Otherwise, the system switches to the next image in the rank and repeat the process until all the required objects have been arranged to specified spatial layout.

The plant model captures all the physically possible behaviors, including desired behaviors (legal behaviors, i.e., behaviors lead to the marker states) and behaviors that should be forbidden (illegal behaviors, i.e., behaviors that cause system failure to reach the marker states). To guarantee success of scene generation, a supervisor is required to regulate the plant's behaviors. In addition to following the scene synthesis procedure, following properties should also be hold by the supervisor:

Controllability: Controllability characterizes the capability of a system to accomplish assigned task (reach target state) under abrupt and unexpected occurrence of uncontrollable events.

Definition of Controllability [16]: Let K and $L = \overline{L}$ be two arbitrary languages over event set Σ. K is said to be controllable with respect to L and Σ_{uc} if $\overline{K}\Sigma_{uc} \cap L \subseteq \overline{K}$.

If a supervisor is controllable, it is able to drive the system to its target state even encountering uncontrollable events during the implementation. To ensure successful scene generation, illegal behaviors should be removed from the supervisor (i.e. scene generator). Controllability guarantees the marker states are achievable. However, sometimes the robot may be trapped by a state or a subset of state such that it takes longer time to accomplish the task. To avoid this issue, the second property for behavior analysis is introduced next.

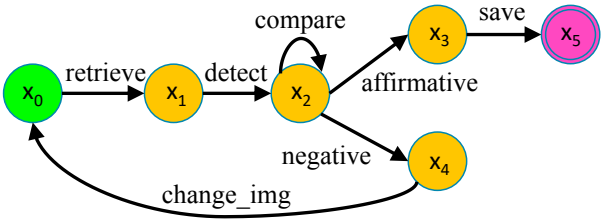
Nonblocking: Nonblocking characterizes the property of a system to avoid being stuck at a state or trapped in a subset of states. The following gives the definition and criteria of nonblocking property.

Definition of Nonblocking [16]: Let $L(G)$ and $L_m(G)$ represent the physically allowed behaviors and the marked behaviors (in this context, marked behavior represent successful scene generation) of the supervisor, respectively. A set of formal language satisfying $\overline{L_m(G)} = L(G)$ is said to be nonblocking.

An overline of a formal language represent all the prefixes of the formal language, including the language itself. The supervisor refers to the image generator in the proposed framework.

The system retrieves images from noisy and nonpreprocessed image cloud and may encounter events that hamper the scene generation. For example, the retrieved images that ranked high in the results may not contain the desired object. Then the system has to keep changing images until the target object has been detected. This is time-consuming and may trap the system in a livelock. Perform nonblocking check on the synthesized supervisor helps to figure out these blocking situations and ensure the accessibility of the marker states.

To avoid the blocking, the retrieved image evaluation procedure and image generation procedure are separated into two independent processes, as shown in Figure 2 (a) and (b), respectively. The retrieved images will be evaluated first to detect desired objects and only the qualified images will be saved to a local image library (Figure 2 (a)). Then the robot randomly retrieves an image from the local image library and extracts the object instance for image synthesis (Figure 2 (b)). The behaviors generated by these two refined models are nonblocking and controllable. They serve as partial supervisors for scene generation. In addition, the plant models G_6 to G_8 are both controllable and nonblocking. The supervisors for interactive scene tuning are designed with the same event and state sets, respectively. The operations on automata are implemented using DESUMA [18].



(a) S_1: retrieved image evaluation

(b) S_2: scene synthesis

Fig. 2. Supervisors for (a) image retrieve and (b) scene synthesis.

3. EXPERIMENTS

3.1. Experimental Setup

Object detection. The object detection performs twofold roles: detection and segmentation. The object detector developed in [24] is used in this work, which is able to detect 20 categories of objects. Object detection is implemented as pixel-level labelling tasks. The output is a 2D matrix that has the same dimension as the input image. Each value of the matrix denotes the object category of the pixel at the same location of the input image.

Dataset. We use the MS-COCO evaluation dataset [11] to evaluate our approach. Since the employed object detector can recognize 20 categories of objects, we filtered the MS-COCO dateset to find text descriptions that only contain objects lay in the range of processable object categories. In total we test our approach on 128 descriptions for scene generation and drawing.

Evaluation metrics. We evaluated the proposed approach using caption generation and human evaluation.

Baseline works. We compare our approach with the following baseline works. **Ground Truth**: The ground truth uses original scenes from the MS-COCO evaluation dataset paired with the scene description. **Random**: A random scene is selected from the ground truth. **Reed et al.** [17]: A GAN developed by Reed et al. to generate images according to the input text. **AttnGAN** [22]: Attentional GAN with attention-driven and multi-stage refinement for text to scene generation. **Ours**: the scenes are generated using the proposed method without further tuning by a human instructor. **Ours-Human**: the generated scenes have been tuned by human instructors via NL.

3.2. Scene Generation

Caption generation:Text-conditional scene generation performance are evaluated using caption generation. We generate sentences from the synthesized scenes and measure the similarity between input text and predicted descriptions. The underlying intuition is that if the generated scene is relevant to input text and its contents are recognizable, one should be able to guess the original text from the synthesized scene. An image caption generator [20] trained on MS-COCO is used to generate sentences, where one sentence is generated per scene. We report four standard language similarity on METEOR [2], CIDEr [19], and Rouge-L [10]. Table 2 summarized the quantitative evaluation results based on caption generation performance. The Random baseline shows the worst performance. This demonstrates that random scenes in the ground truth rarely convey the same semantic meaning. It can be seen that the proposed methods significantly outperforms the baseline approaches. Tuning from human instructors also helps to increase the semantics of generated scene dramatically. Caption generation performance shows that captions generated from our synthesized scenes are more correlated with the text than the baselines, which means that the scenes synthesized by our method are better aligned with the input descriptions and are easier to be recognized.

Table 2. Quantitative evaluation and human study results

Method	CIDEr	ROUGH_L	METEOR
Random	0.121	0.256	0.072
[17]	0.201	0.269	0.092
[22]	0.220	0.296	0.097
Ours	**0.416**	**0.335**	**0.134**
Ours-Human	**0.654**	**0.362**	**0.151**
Ground Truth	1.298	0.476	0.221

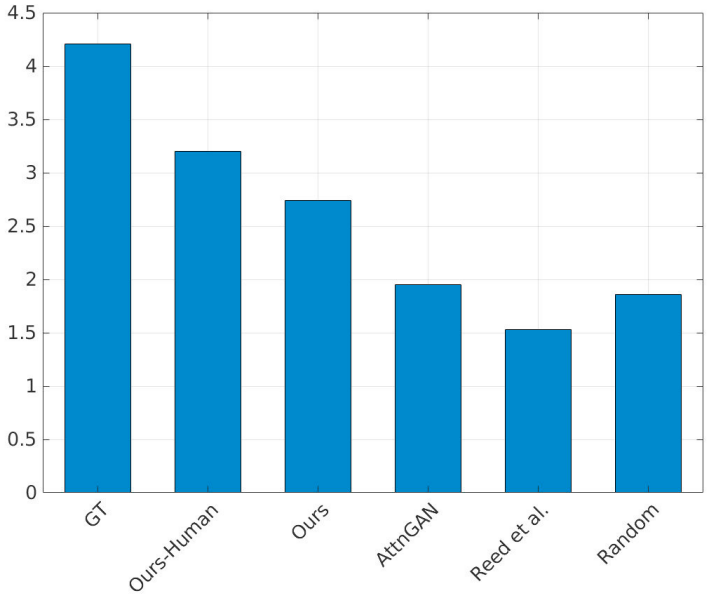


Fig. 3. Turkers are asked to score how well the scenes match the description on a scale of 1 (very poorly) to 5 (very well). "GT" denotes "Ground Truth". Blue bars represent results of human evaluation on generated scenes

Human evaluation: Using caption generator for evaluation is beneficial for large scale data. However, it may introduce unintended bias of the caption generator. To validate the caption generation evaluation, we have also conducted

human evaluation on Amazon Mechanical Turk. For each text description from the evaluation dataset, six scenes generated by different approaches are presented to the Turkers. They are asked to score how well the scenes match the description on a scale of 1 (very poorly) to 5 (very well). The results shown in Figure 3 and 4 demonstrates consistency with the caption evaluation performance. Figure 5 shows some randomly chosen qualitative results.

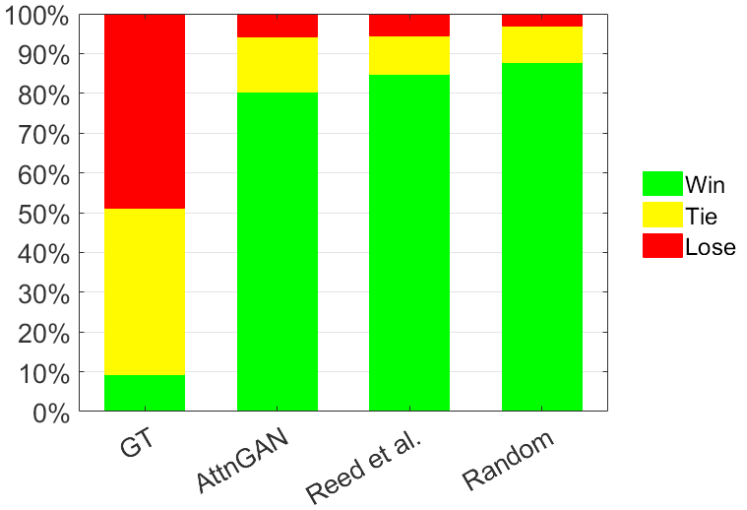


Fig. 4. The results of which scenes better depicted the input descriptions. The subjects find our scenes better represent the input sentences than other baseline approaches. In fact, our approach wins over or ties with the ground truth scenes frequently.

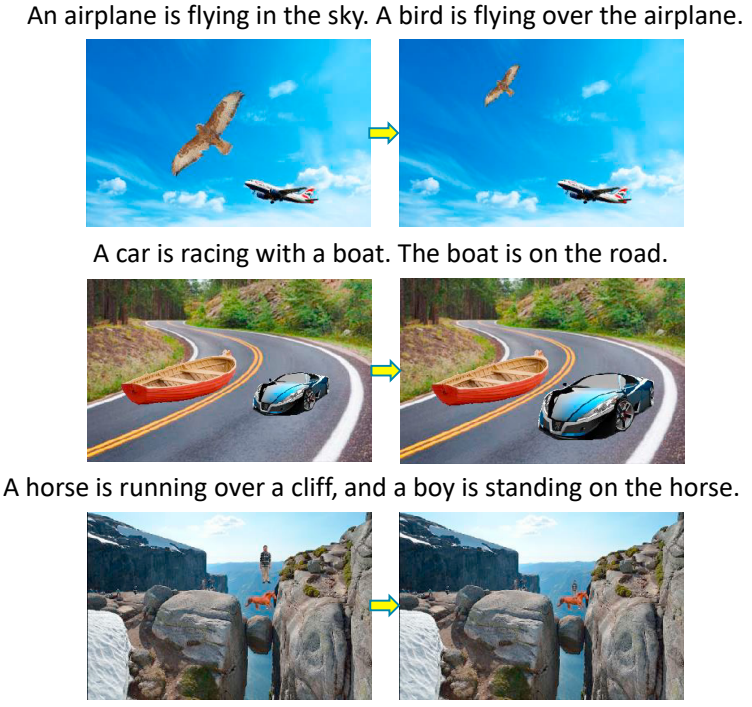


Fig. 5. Qualitative Examples.

4. CONCLUSION

We proposed a framework for scene generation conditioned on NL descriptions. The proposed approach decomposes the process into several manageable steps. Instead of learning a direct text-to-pixel mapping or specifically building a knowledge base, the proposed approach obtains and utilizes knowledge retrieved from Internet. Handling unexpected events makes the scene generation more stable and reliable. The proposed approach allows interactive modification of synthesized scenes. In addition, the object size and location data after interactive tuning can be recorded for later training augmentation.

References

[1] Babko-Malaya, O., 2005. Propbank annotation guidelines. URL: http://verbs. colorado. edu .

[2] Banerjee, S., Lavie, A., 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments, in: In Proc. of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, pp. 65–72.

[3] Chang, A.X., Savva, M., Manning, C.D., 2014. Learning spatial knowledge for text to 3d scene generation., in: EMNLP, pp. 2028–2038.

[4] Cheng, Y., Bao, J., Jia, Y., Deng, Z., Sun, Z., Bi, S., Li, C., Xi, N., 2017. Modeling robotic operations controlled by natural language. Control Theory and Technology 15, 258–266.

[5] Dong, H., Zhang, J., McIlwraith, D., Guo, Y., 2017. I2t2i: Learning text to image synthesis with textual data augmentation, in: Proc. of IEEE International Conference on Image Processing, IEEE. pp. 2015–2019.

[6] He, L., Lee, K., Lewis, M., Zettlemoyer, L., 2017. Deep semantic role labeling: What works and whatŁ next, in: Proc. of Annual Meeting of the Association for Computational Linguistics, pp. 473–483.

[7] Kehoe, B., Patil, S., Abbeel, P., Goldberg, K., 2015. A survey of research on cloud robotics and automation. IEEE Trans. Automation Science and Engineering 12, 398–409.

[8] Klein, D., Manning, C.D., 2003. Accurate unlexicalized parsing, in: Proc. of Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics. pp. 423–430.

[9] Lignos, C., Raman, V., Finucane, C., Marcus, M., Kress-Gazit, H., 2015. Provably correct reactive control from natural language. Autonomous Robots 38, 89–105.

[10] Lin, C.Y., 2004. Rouge: A package for automatic evaluation of summaries. Text Summarization Branches Out .

[11] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context, in: European conference on computer vision, Springer. pp. 740–755.

[12] Mansimov, E., Parisotto, E., Ba, J.L., Salakhutdinov, R., 2015. Generating images from captions with attention. arXiv preprint arXiv:1511.02793 .

[13] Mayer, R.E., 2002. Multimedia learning, in: Psychology of learning and motivation. Elsevier. volume 41, pp. 85–139.

[14] Misra, D.K., Sung, J., Lee, K., Saxena, A., 2015. Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. The International Journal of Robotics Research 35, 281–300.

[15] Nilsback, M.E., Zisserman, A., 2008. Automated flower classification over a large number of classes, in: Proc. Indian Conference on Computer Vision, Graphics and Image Processing, pp. 722–729.

[16] Ramadge, P.J., Wonham, W.M., 1989. The control of discrete event systems. Proceedings of the IEEE 77, 81–98.

[17] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H., 2016. Generative adversarial text to image synthesis. arXiv preprint arXiv:1605.05396 .

[18] Ricker, L., Lafortune, S., Genc, S., 2006. Desuma: A tool integrating giddes and umdes, in: International Workshop on Discrete Event Systems, pp. 392–393.

[19] Vedantam, R., Lawrence Zitnick, C., Parikh, D., 2015. Cider: Consensus-based image description evaluation, in: In Proc. of IEEE Conference on Computer Vision and Pattern Recognition, pp. 4566–4575.

[20] Vinyals, O., Toshev, A., Bengio, S., Erhan, D., 2015. Show and tell: A neural image caption generator, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164.

[21] Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S., 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001. California Institute of Technology.

[22] Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., He, X., 2017. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. arXiv preprint .

[23] Yan, X., Yang, J., Sohn, K., Lee, H., 2016. Attribute2image: Conditional image generation from visual attributes, in: European Conference on Computer Vision, Springer. pp. 776–791.

[24] Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P.H., 2015. Conditional random fields as recurrent neural networks, in: Proc. of IEEE International Conference on Computer Vision, pp. 1529–1537.

[25] Zhu, X., Goldberg, A.B., Eldawy, M., Dyer, C.R., Strock, B., 2007. A text-to-picture synthesis system for augmenting communication, in: AAAI, pp. 1590–1595.

[26] Zitnick, C.L., Parikh, D., Vanderwende, L., 2013. Learning the visual interpretation of sentences, in: Proc. of IEEE International Conference on Computer Vision (ICCV), pp. 1681–1688.