ISSN-0973-5151
Subscribers copy
Not for sale

Volume 13. No. 1
July - September 2018

i-manager's

Journal on Software Engineering

A brand new matrix of thoughts for the technology savvy of the modern era

i-manager®
Publications

ISSN-0973-5151
Volume 13. No. 4
April - June 2019

i-manager's

Journal on Software Engineering

About the Journal

i-manager's Journal on Software Engineering is an earnest effort to evolve the youth to achieve their full potentials. The Journal is solely devoted to the illumination of IT based younger generation and through this Journal we intend to serve as a platform for the researchers.

i-manager's Journal on Software Engineering is presently in its 13th Year. The first issue was launched in 2006.

i-manager's Journal on Software Engineering is published by i-manager Publications, one of India's leading Academic Journal Publisher, publishing 28 Academic Journals in diverse fields of Engineering, Education, Management and Science.

Why Publish with us

i-manager Publications currently publishes academic Journals in Education, Engineering, Scientific and Management streams. All of i-manager's Journals are supported by highly qualified Editorial Board members who help in presenting high quality content issue after issue. We follow stringent Double Blind Peer Review process to maintain the high quality of our Journals. Our Journals target both Indian as well as International researchers and serve as a medium for knowledge transfer between the developed and developing countries. The Journals have a good mix of International and Indian academic contributions, with the peer-review committee set up with International Educators.

Submission Procedure

Researchers and practitioners are invited to submit an abstract of maximum 200 words/full paper on or before the stipulated deadline, along with a one page proposal, including Title of the paper, author name, job title, organization/institution and biographical note.

Authors of accepted proposals will be notified about the status of their proposals before the stipulated deadline. All submitted articles in full text are expected to be submitted before the stipulated deadline, along with an acknowledgment stating that it is an original contribution.

Review Procedure

All submissions will undergo an abstract review and a double blind review on the full papers. The abstracts would be reviewed initially and the acceptance and rejection of the abstracts would be notified to the corresponding authors. Once the authors submit the full papers in accordance to the suggestions in the abstract review report, the papers would be forwarded for final review. The final selection of the papers would be based on the report of the review panel members.

Format for Citing Papers

Author surname, initials (s.) (2019). Title of paper. i-manager's Journal on Software Engineering, 13(4), xx-xx.

Copyright

Contact e-mails

editor_soft.engg@imanagerpublications.com
submissions@imanagerpublications.com

ISSN-0973-5151
Volume 13. No. 4
April - June 2019

i-manager's

Journal on Software Engineering

ISSN-0973-5151
Volume 13. No. 4
April - June 2019

i-manager's

Journal on Software Engineering

OUR OFFICES

Registered Office

3/343,Hill view,
Town Railway Nager,
Nagercoil, Kanyakumari District - 629001
Ph : 91-4652- 277675
E-mail : info@imanagerpublications.com

Editorial Office

13-B, Popular Building,
Mead Street, College Road,
Nagercoil, Kanyakumari District - 629001
Ph : (91-4652) 231675, 232675, 276675
E-mail : editor_soft.engg@imanagerpublications.com

Abstracting / Indexing

Join with us

https://www.facebook.com/imanJSE/

https://www.facebook.com/imanagerPublishing/

https://twitter.com/imanagerpub

CONTENTS

RESEARCH PAPERS

EDITORIAL

i-manager's Journal on Software Engineering provides some interesting topics in this issue onthe Effect of Attacker on three of Sampling Strategies in Active Learning, Time Estimation as Critical Factor of software Failure, Speech to 3D Scene Generation, Student Data Integration and Consolidation System and Multithreading in Game Development.

Ghofran M. Alqaralleh et al. have proposed a study about the effect of attacker on three of sampling strategies in active learning. This paper studies the effect of the attacker on the strategies that are used in the selection of most informative instances in active learning for network intrusion detection. For that, during the sampling stage, the representative or most informative instances were selected from the unlabeled data to the label based on the sampling strategy. The experiment was tested using a Matlab R2015b. The experimental results showed that the expected model change strategy is not significantly affected by the attack compared with other strategies.

Warda Shah has presented a study about the time estimation as a critical factor of software failure. The objective of this research is to find all those factors that can effect timely delivery of software and increase the actual time estimation. A step by step SLR protocol design and process is discussed with preliminary results in this study. Systematic Literature Review (SLR) was conducted to achieve the goals and SLR protocol was developed and executed in different digital libraries. The authors concluded that the result of the study will help to identify any research gap where more work is possible for successful and timely delivery of software projects.

Manthan Turakhia et al. have presented a study about Speech to 3d Scene Generation. This paper aimed to reduce the human efforts for the same by generating 3D scenes described by the user with precision. Speech to 3D Scene Generation is software developed to provide a digital/graphical output of the literal spokenwords of the user. This study intends to replace the existing methods of teaching and learning by using speech to 3D scene generation to depict exactly what the professor is trying to explain. To achieve that, the system was designed as a very simple and usable User Interface. The study concluded that the use of Panda 3D viewer and the way of maintenance by developers are to be improved to meet the expectations of multiple industries and domains.

Nishith Khandor et al. have conducted a study about the student data integration and consolidation system. The authors proposed an approach to manage and access the large amounts of distributed unstructured data of an organization in an efficient manner. Hence, a web-application has been designed to efficiently access and manage the data and to save precious time spent in accessing the distributed data. The authors also describe about the basic features provided by the web-application along with a set of special features that can help reduce effort and provide useful analysis and data visualization based on the data of the organization. The study concluded that, the proposed system is an ideal approach for satisfying the requirements for the organizations and educational institutions and the system can solve the data problems such as data availability and data accessibility.

Sainath Prasanna et al. have proposed a study about the use and importance of multithreading using the example of a basic game development in the Java environment. The concept of the paper is multithreading that allows programmers to write simpler, more easily understandable code when it comes to making games or any other application. The results concluded that, in terms of performance and ease of development, multithreaded programming is better since it has several advantages over the single threaded approach; but poorly implemented multithreaded programs can lead to several problems and can be very difficult to debug.

The current issue of i-manager's Journal on Software Engineering explores various concepts of the field of Software testing and we hope it serves the needs of the academic readers!

Warm Regards,

Ramya R.
Associate Editor
i-manager Publications

SPEECH TO 3D SCENE GENERATION

By

MANTHAN TURAKHIA * UMANG NANDU ** PRAYESH SHAH ***

SIDDHARTH SHARMA **** SAGAR KORDE *****

*-***** Department of Information Technology, K. J. Somaiya College of Engineering, Mumbai, Maharashtra, India.

Date Received: 15/04/2019 Date Revised: 02/06/2019 Date Accepted: 30/06/2019

ABSTRACT

3D scenes and graphics are widely used in the creative industry. However, the entire task of imagination and then depicting the same as 3D graphics is done manually today, which consumes a lot of time, not to mention the inability to depict the scene precisely as imagined. The authors aim to reduce human efforts for the same by generating 3D scenes described by the user with precision.

On the other hand, some industries currently lack the use of appropriate technology to make their tasks easier and more captivating such as the education industry. The existing methods of teaching and learning are intended to by replaced by using speech to 3D scene generation to depict exactly what the professor is trying to explain.

Keywords: Speech to Scene, Linguistic Analysis, Spatial Relationship, Natural Language Processing.

INTRODUCTION

We examine the task of speech to 3D scene generation. There is a myriad of applications for this technology, mainly for creative and educational industries. Designers can use this technology to interpret and display their thoughts and imaginations. Students can be taught with a graphical depiction of the topic. Commercial meetings and conference sessions can make the most out of this technology.

The algorithm uses the concept of semantic parsing (Monroe, 2008) for identifying objects and their nature, and spatial knowledge (Hossain & Salam, 2017) to map the identified objects and scenes according to the users' imaginations. The goal is to achieve a 3D generated scene, so as to practically match the on-going speech of the user, such that the scene on the screen will keep changing and manipulating to match the speaker's requirements and descriptions.

In future, the authors plan to introduce complex Artificial Intelligence algorithms to this research work so as to make it learn the usual demands and imaginations for each user, which would make the scene generation more reliable, efficient, and faster at the same time.

1. Objective

The objective of this research paper is to provide a platform for users to enable them to display exactly what they have said in the form of speech on their screens. As of now, 75% rendering accuracy is targeted and expected, with speech to text conversion (Coyne & Sproat, 2001) expected within 2 seconds and rendering in the next 2 seconds, making the complete process to be completed in the time frame of 3-4 seconds. The database for models and objects is going to be ever-growing and scaling of the models and objects should be dynamic in the future.

2. Problem Definition

There is a stale in multiple industries with respect to the presentation of knowledge. In simpler words, ways of presenting and imparting knowledge, and creating architectural masterpieces have become mundane and monotonous. Therefore, to bring a change, a way to paint a picture with words has been introduced.

3. Literature Survey

The task of speech to 3D scene generation has been examined. There is a myriad of applications for this technology, mainly for creative and educational industries. Designers can use this technology to interpret and display their thoughts and imaginations. Students can be taught with a near real-time (Seversky & Yin, 2006) graphical depiction of the topic. Commercial meetings and conference sessions can make the most of this technology.

The following observations were found in the literature survey:

- Text to scene (limited capabilities).
- Limited size databases (no dynamic generation or manipulation).
- Scenes generated are not intelligent and precise hence, cannot be used for real-world applications.
- Language used is unnatural.
- Carsim is the only text-to-scene conversion system that has been developed and tested using non-invented narratives (Johansson, Berglund, Danielsson, & Nugues, 2005).
- Identification of object categories using scoring model (Chang, Monroe, Savva, Potts, & Manning, 2015).
- Focus on translating the semantic intent of the user, as expressed in language, into a graphic representation (Coyne & Sproat, 2001).
- Semantic parsing model gives negative weights to the word skipping features (with the exception of verbs, wh pronouns, and punctuation) and the grammar rule application features with counts of zero (Monroe, 2008).

4. Speech to 3D Scene Generation

Speech to 3D Scene Generation is a software developed to provide a digital/graphical output to the literal spoken words of the user. It goes through various stages before providing the final output. First, the speech is converted to text, then the text is passed to the interpretation protocol, and finally the interpreted text is used to render the output.

The best part of "Speech to 3D Scene Generation" is that it is meant to be used by any person who can speak. It is targeted to be used at various industries like education, creative, etc. as well as corporates.

5. Methodology

5.1 Speech to Text

Google Speech to text API is used to serve the purpose of converting the given Speech audio to a text format.

5.2 Speech Recognition Library

This library is used to recognize the audio and remove the disturbance if any, and generate a clean speech audio. It is easy to tweak (one can set threshold to listen the audio). It is easy to handle the exception technically and is fast and reliable.

5.3 Parts-of-Speech Tagging

This part is important while processing the text. It identifies the text and tags the parts of speech. It stores each lexical element along with its POS tags in Json format. We convert this Json file to a parse tree. For example, Figure 1 presents the POS tagging for the speech input given as, "The ball is on the wooden table".

The JSON file along with Parse Tree will be used later in the form of 'Linguistic knowledge' and 'World knowledge' to interpret the same and render a scene accordingly.

5.4 Object Categorization

Spacy helps categorize the objects in the given input. It uses the rule-based models, in order to fetch the nouns as identifiers. The core identification is then passed to the scoring model $\{\varphi_i\}$ (Chang et al., 2015).

$$Score(c \setminus p) = \sum_{\phi_i \in \phi(p)} \theta_{(i,c)} \qquad (1)$$

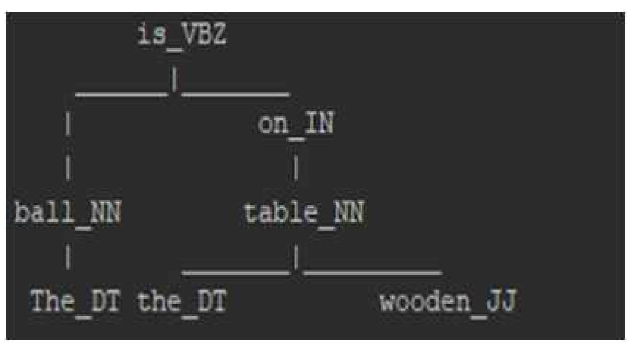

Figure 1. Example of Parse Tree

5.5 Panda3D (Speech to 3D Scene)

Panda3D, an open-source and free-to-use engine for real-time 3D visualizations, simulations, and experiments.

Here we use Panda3D to render, place, and generate a 3D model and eventually a scene.

The aim of using this platform is that it provides a structure to render and place the objects with respect to the given text input in a single 3D view. The implementation, packages, and classes used in this software will be seen further as we go through the paper.

6. Requirements

6.1 External Interface Requirements

6.1.1 User Interfaces

The user interface requirements for title are very general because it is a Desktop application. The PC at the user end should have only the basic screen layouts with no requirements for latest OS. However, it may not be compatible for very earlier versions of Windows OS. The user should be able to easily navigate to the part where it enables the speaker and the software should immediately start recording, converting, and rendering. It is essential that it is simply a one-step process for the user and then it should all be a completely automatic process.

6.1.2 Hardware Interfaces

There isn't much hardware interfaces required since it is a completely software oriented product. The only requirement for it is to work on any type of PC (Laptop, Computer), which match the basic OS and version requirements.

6.1.3 Software Interfaces

6.1.3.1 Software Requirement

- Google Speech-to-Text API (Integrated Library)
- SpaCy Version 2.0.13
- Panda3D

6.2 Software Product Features

"Speech to 3D Scene Generation" will provide following features:

6.2.1 Functional Requirements

6.2.1.1 Input Data Requirements

- Speech Input.

6.2.1.2 Operational requirements

- Conversion of speech to text.
- POS tagging.
- Parse tree generation.
- Information gathering and rendering.

6.2.2 Non-functional Requirements

- *Performance*: 75% conversion accuracy. Worst case 15s generation. Best case 3s.
- *Data Integrity*: Data and modules to be kept abstract.
- *Usability*: Smooth screen-to-screen movement.

6.3 Software System Attributes

6.3.1 Reliability

- Mean Time to Failure (MTTF) is Twenty Seconds.
- Expected optimal time for rendering and displaying is Seven Seconds.
- Speech-to-Text 75% accuracy.

6.3.2 Availability

- Failure at any point of the process will lead to complete termination and the user will have to start and perform the process all over again.

6.3.3 Security

- Since it is a Desktop application, the basic security measures taken by the user are sufficient with no additional requirements except for basic login credentials.
- Data/image/graph rendering is over the internet therefore simple internet security is more than enough.

6.3.4 Portability

- Entire software is mainly Python-oriented.
- No need of external compiler because of integrated environment.
- Most commonly used OS (Windows) is all that is required with no additional features.

6.3.5 Performance

- As mentioned, minimum 75% accuracy for Google speech-to-text API. Minimum latency for rendering.
- Users are expected to provide clear speech inputs, avoiding grammatical errors.

- Users are expected to be in a relatively quiet environment so as to ease the processing of the API.
- Data storage integrated using cloud therefore not much physical storage required.

6.4 Database Requirements

Dataset required is the .egg models, accessed from the storage.

7. Implementation Details

7.1 Overview of Implementation

Figure 2 includes all the modules and interfaces involved in the research work. The foremost step is to provide the speech input. This speech input will be received by the speech to text API, which will convert the provided speech to text. Next step is interpretation; this is where the converted text is being processed by SpaCy library used in Python environment. The term interpretation means that the text will be divided into multiple parts of speech identified by the library. As you can see, the library will make use of linguistic and word knowledge to precisely identify which word from the word knowledge (dictionary) fits into which category of the English language. One the text is vividly classified. Note that the hierarchy created by SpaCy is the most important part. The database will use the hierarchy to identify the order in which the scene is to be generated and more importantly, the relation between the objects of the scene, along with the attributes of each object and the scene as a whole. It will require extensive geometric knowledge (Chang, 2015; Cheng, Sun, Bi, Li, & Xi, 2017; Sneha, Dessai, & Dhanaraj, 2016) to place the objects exactly where required and also for mathematical purposes for forming a grid. Once all this is done, the scene will simply be rendered to the designed UI.

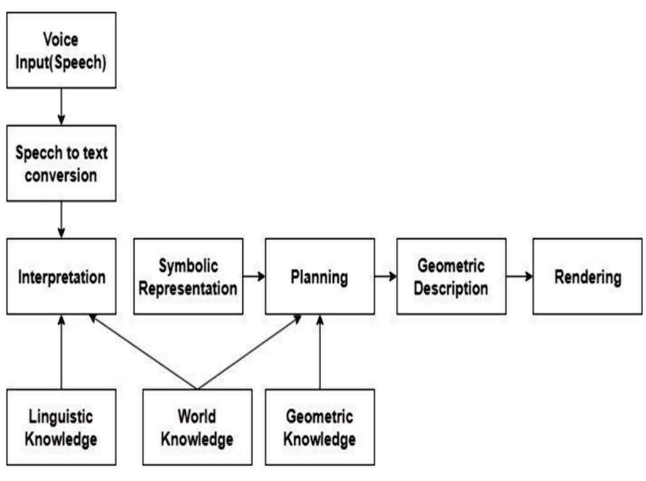

Figure 2. Implementation Architecture

7.2 System Interface Description

The software is a desktop application and thus, will work on Windows OS. The user will simply need to install the software and it will be ready to use. All the libraries and database files will run in the background, potentially using cloud, therefore the user will only need to download and install the installer and main file. The APIs and libraries used by the software are Speech to Text API, SpaCy library, and 3D dataset. All the components are back end products and therefore beyond users control and reach. The knowledge/information sets are also integrated with the APIs and libraries hence not concerning the user.

7.3 Implemented Speech to Scene

Panda3D is an open-source, completely free-to-use engine for real-time 3D games, visualizations, simulations, experiments (Panda3D).

7.3.1 Child and Parent Relationships

In Panda3D, a parent model is denoted by 'render'. It is the parent scene graph. The model first needs to be loaded by inserting the same to the scene graph.

For example:

loader. loadModel ("c:/ProgramFiles/MyScene/Models/Model.egg");

This loader is not enough to render the Model. It requires to be reparented. This is achieved by reparenting, which we will see in the later part of the paper. The child needs to be made a link with its parent.

7.3.2 Reparenting

Panda3D allows us to make a relationship between 2 objects or models by the process known as Reparenting. This is done by first Reparenting the base object/model to the child object, i.e. render in our case, and then reparenting other objects with available relevant objects.

For example:

base. reaparent To (render);

obj1.reparentTo(base);

obj2.reparentTo(obj3);

7.3.3 Rendering Method with Positioning

Show Base is a class inherited from Direct Object class. It has various attributes like base, render, render2d, camera, etc.

Figure 3 depicts the 3D Scene generated for positioning. Figure 4 depicts the 3D Scene generated for the input speech "A crt monitor is on the desk.", "A chair is in front of the couch.", "A couch is to the right of the desk." and "A printer is to the left on the desk."

Figure 5 depicts the 3D Scene Generated with prepositions for the input speech "A printer, crt monitor, and a lab chair is behind the big table".

7.3.4 Model Type

Panda3D renders a specific type of models. Models in the .egg format are supported for rendering on the Panda3D platform.

7.3.5 Pseudo Codes

7.3.5.1 Noun Exclusion

for token in doc:

Figure 3. 3D Scene Generated with Positioning

Figure 4. 3D Scene Generated with Multiple Inputs

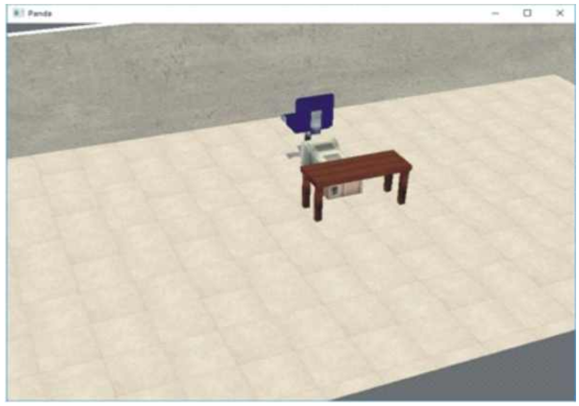

Figure 5. 3D Scene Generated with Prepositions

if (token.dep_ == "amod" or token.dep_ == "compound") and not (any(token.text in s for s in models)) and token.text not in exc:

models.append("_".join([token.lemma_, token.head.lemma_]))

elif token.pos_ == "NOUN" and not (token.text == "left" or token.text == "right" or token.text == "front") and not (any(token.text in s for s in models)) and token.text not in exc and token.text not in exc1:

models.append(token.lemma_)

7.3.5.2 Loading Model Objects into 'model_array' List

room = loader.loadModel(mydir + "/panda_models/model/model.egg") model_array = {"room": room}

7.3.5.3 Tokenization (subject, object, prepositions)

sub.append("_".join([token.lemma_, token.head.lemma_])) my_dict.update({str(key): (sub, prep, obj)})

7.3.5.4 Setting a West Position

model_pos.setPosWest(model_array.get(subj), room)

7.4 Design Principle

7.4.1 DFD

Figure 6 depicts the Flow of Data over the system.

7.4.2 Use Cases

Figure 7 depicts the use cases of the system.

8. Testing

8.1 Test Approach

Testing of the software application will require 10-15 days.

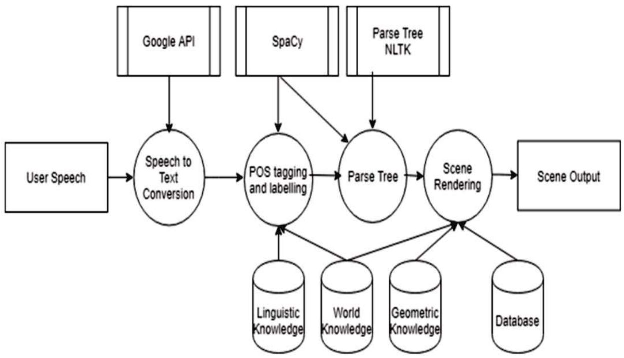

Figure 6. Data Flow Diagram

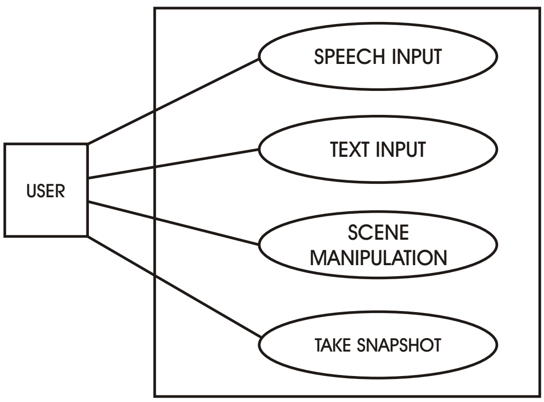

Figure 7. Use Case Diagram

Manual testing approaches is to be applied.

8.1.1 Testing of Speech Input to Text Output

The speech "A dog is sitting on the right of the table" needs to be tested for "Right"/"Write"/"White". The learning curve needs to be set here by the Speech Recognition Package with respect to the subject, object, and the predicate of the given test input.

8.2 Features to be Tested

- Speech to text Conversion
- Scene Generation
- Scene Manipulation

8.3 Features not to be Tested

- Parts of Speech Tagging.
- Label are not to be tested.

8.4 Test Cases

Table 1 briefly describes the test cases, their purposes, input, and their associated outputs.

Test Case	Purpose	Input	Expected
Speech to text conversion	Whether the input speech converted into the text is valid for further processing or not. To check how accurately speech is converted to text	Voice input	The text is valid if the text converted is same as the speech input given by the user. If it is then text is further processed else usercan rerecord the input.
Tagging and Labelling	To check whether the parts of speech tagging and labelling of the text is done meaningfully or not.	Text Converted using Speech to text Recognition.	JSON or XML file which will contain properparts of speech tagging and labellingof the text.
Rendered Models	To check whether therendered models from data ware house are perfectly suitable with the input provided in first stage.	No input from the user, the converted text is processed further	Actual model of specific objects specified by the user are correctly rendered else final output will be incorrect, Models should not overlap.
Positions of the object (models)	To check whether the models rendered and displayed on the output screen are at proper co-ordinates as user wants.	No specific input, text is processed	Objects are at proper positionas mentioned by the user into the speech input.

Table 1. Test Cases

9. Future Scope

The first and foremost improvisation that can be done in the future is to scale the database and/or find ways to convert other formats of data files (collada, jpeg, etc.) to the supporting format (.egg).

We also intend to configure the current algorithms to make the system learn and predict users' choices for generating the scenes for a better user experience, using deep learning. We can move the repository online, instead of a local database, to reduce security risks and increase the capacity simultaneously. Over time, we can also increase the accuracy and response time with further optimization. We also aim to train the system to get more accustomed to Natural Language Processing (NLP) to improve the response efficiency and provide better, more relevant results since this is the heart of the project.

10. Results

There have been many attempts made towards depicting human language in a digital form. Instead of using written or typed input from the users, which has been the case hitherto, we took it up a notch by directly processing the natural language spoken by the users.

We have used NLP to enhance user experience for various applications like education, corporate meetings, etc. With the right data set and deep learning planned for the future, we believe that this project can be further improved and used as the go-to technology and channel to explain, learn, and even communicate.

The 3D Scene to be generated is given a Speech input via a microphone. The input is then passed to the SpaCy package in order to identify the text and objects present in the same, which makes it a better move.

The Json file created as an output contains categorized objects like {'Google': "organization"}. The objects finds the storage to point to the associated model and render the same.

This rendering is done at a better speed than existing.

Conclusion

We have examined a strategy to convert a speech input to a 3D Scene in a stepwise procedure. A very simple and usable User Interface to give a Speech Input and a text input as exceptions to modify the corrections, if any. The Panda3D viewer helps view the generated 3D Scene in a 3D space with all the necessary parameters considered. This system will be maintained by the developers and be improved to meet the expectations of multiple industries and domains (Education, Interior Designing, Corporates, etc.).

In future we intend to include internationalization (Multiple language support). A better approach to the template making and a dynamic and real-time Scene manipulation of the same will be implemented in near future.

Acknowledgment

We are grateful to our project guide Prof. Sagar Korde, Assistant Professor, Information Technology Department, K. J. Somaiya College of Engineering, for his continuous support for the project, and constantly upgrading us with real world scenarios and making us thorough with new. Also for giving us a new experience on working under pressure and teaching us professionalism and team work. Also we would like to thank all the faculty members of K. J. Somaiya College of Engineering. We are thankful for their cooperation throughout the period of our assignment.

References

[1]. **Chang, A. X. (2015).** *Text to 3D Scene Generation* (Doctoral Dissertation, Stanford University, USA).

[2]. **Chang, A., Monroe, W., Savva, M., Potts, C., & Manning, C. D. (2015).** Text to 3D scene generation with rich lexical grounding. *arXiv preprint arXiv:1505.06289* (pp. 53-62).

[3]. **Cheng, Y., Sun, Z., Bi, S., Li, C., & Xi, N. (2017, December).** A supervisory hierarchical control approach for text to 2D scene generation. In 2017 *IEEE International Conference on Robotics and Biomimetics (ROBIO)* (pp. 2261-2266). IEEE.

[4]. **Coyne, B., & Sproat, R. (2001, August).** WordsEye: An automatic text-to-scene conversion system. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (pp. 487-496). ACM.

[5]. **Hossain, M. S., & Salam, A. (2017).** Text-to-3D Scene Generation using Semantic Parsing and Spatial Knowledge with Rule Based System. *International Journal of Computer Science Issues (IJCSI),* 14(5), 37-41.

[6]. **Johansson, R., Berglund, A., Danielsson, M., & Nugues, P. (2005, July).** Automatic text-to-scene conversion in the traffic accident domain. *In IJCAI*, 5, 1073-1078.

[7]. **Monroe, W. (2008).** *3D Scene Retrieval from Text with Semantic Parsing.* Stanford University Computer Science Department Stanford, USA.

[8]. **Panda3D, (n. d).** *The Open Source Framework for 3D Rendering and Games.* Retrieved from https://www.panda3d.org/

[9]. **Seversky, L. M., & Yin, L. (2006, October).** Real-time automatic 3D scene generation from natural language voice and text descriptions. In *Proceedings of the 14th ACM International Conference on Multimedia* (pp. 61-64). ACM.

[10]. **Sneha, N., Dessai, & Dhanaraj, R. (2016).** Text to 3D scene generation. In *International Journal of Latest Trends in Engineering and Technology (IJLTET)*, 6(3).

ABOUT THE AUTHORS

Manthan Turakhia has completed Engineering from K. J. Somaiya College of Engineering (KJSCE) in Information Technology (IT) and he also holds a Diploma Certificate in Information Technology from K. K. Wagh Polytechnic, Nashik. He has worked on a project called "A Detection Rover using Raspberry Pi" during his final year diploma.

Umang Nandu has completed his B.Tech in Information Technology (IT) from K. J. Somaiya College of Engineering (KJSCE), Mumbai, India, in 2019. He completed his Diploma in Computer Technology from Shah and Anchor Kutchhi Polytechnic (SAKP), Mumbai, in 2016. He has published papers in IJRITCC. His project experience includes his Diploma major project "E-Automobile Documentation" and his Degree project "Speech to 3D Scene Generation" and other minor free projects in Android and web development and IoT projects.

Prayesh Shah is currently working as a Developer at Mirraw. He has completed Engineering from K. J. Somaiya College of Engineering (KJSCE) in Information Technology (IT), in 2019.

Siddharth Sharma has completed Engineering from K. J. Somaiya College of Engineering (KJSCE) in Information Technology (IT), in 2019. He is pursuing further studies.

Sagar Korde is working as an Assistant Professor in the Department of Information Technology at K. J. Somaiya College of Engineering, Mumbai, India. He holds a Master of Engineering Degree with a specialization in Embedded System and Computing. He has published a handful of papers in the same areas. His areas of interest and work include Embedded System, Internet of Things, Computer Vision, Operating System, and Usability Engineering.

Newspapers of India registered number TNENG/04550/100306
RDO Nagercoil-Declaration No:AB/6305/2006

i-manager®
Publications

3/343, Hill view, Town Railway Nager, Nagercoil
Kanyakumari Dist. Pin-629 001.
Tel: +91-4652-276675, 277675

e-mail: info@imanagerpublications.com
contact@imanagerpublications.com
www.imanagerpublications.com

i-manager Printing Press