J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

A 3D Cartoon Extraction in Cartoon Sentences with GPT API☆

Yejin Jin[1] Janghwan Kim[2] Hyunseung Son[3] R. Young Chul Kim[4,*]

ABSTRACT

With rapid growth in the 3D Content market, there is an increasing demand for 3D technology to produce 3D content. Recently, in the cartoon industry, cartoonists are attempting to utilize AI image tools to generate background elements or props for cartoons using natural language sentences. However, Generative AI faces technical difficulties in maintaining the visual consistency of main characters across all scenes. Therefore, a methodology is required that ensures the consistent reuse of main characters throughout the cartoon story. To address this, we propose an automated mechanism for representing a 3D object's emotional expression by analyzing natural language-based cartoon sentences. This enables the emotional state of the main character in the cartoon to be reflected in the 3D character entity. The existing Text-to-3D tools generated various models for each natural language query, which cannot guarantee the accuracy of the results. To overcome these limitations, natural language sentences are analyzed using linguistic approaches via the GPT API. Subsequently, cartoon sentences are visualized through UML diagrams, integrating software engineering principles into cartoon engineering. By ensuring reusability and consistency for 3D characters, the proposed method is expected to enhance the productivity and convenience of 3D cartoon development.

☞ keyword : GPT, Natural Language Processing, State Diagram, 3D Modeling, Cartoon Engineering

1. Introduction

Recently, the 3D modeling market has been experiencing gradual growth [1]. The development of 3D modeling is utilized in various industries, including architecture, construction, and gaming. In particular, the trend in the comics industry is to create webtoons with a 3D modeling technique. Realistic expressions characterize the 3D webtoons compared to existing 2D methods, offering higher quality and faster work speed [2]. In general, 3D models utilizing AI are mainly used for background design rather than character design. This is because in comics, the same background or prop assets can be downloaded and used. However, it is not easy to apply AI to characters. A 3D model must accurately reflect all changes in the character's facial expressions or behaviors, depending on the scene. This process must accurately reflect all the complex characteristics of the character; it is both costly and time-consuming, requiring expert-level skills. Therefore, we need to generate a 3D character entity from natural language systematically. Although the current AI tools create 3D models from natural language, they have limitations in meeting detailed requirements. In addition, most AI technology faces challenges with low maintenance and reusability due to the unconfirmable nature of its internal mechanisms. To address this issue, the existing mechanism is enhanced through a procedural approach. We use the GPT API to analyze natural language, which identifies key objects, their actions, and emotions. This information is then visualized using a sequence diagram to illustrate the step-by-step actions among objects. The state transitions of an object are then used to generate a state diagram. The mechanism is further enhanced by extracting these actions from the sequence diagram and incorporating them into a state diagram to illustrate the state changes of a primary object. As a result, these detailed UML diagrams are utilized to generate a 3D model that incorporates

1,2,4 Dept. of Software and Communications Engineering, Hongik University, Sejong, 30016, Republic of Korea
3 Dept. of Computer Engineering, Mokpo National University, Mokpo, 58645, Republic of Korea
* Corresponding author: bob@hongik.ac.kr
[Received 24 August 2025, Reviewed 29 August 2025(R2 07 November 2025, R3 15 December 2025), Accepted 30 January 2026]
☆ This research was conducted with the support of the Korea Research Foundation's four, Brain Korea 21 (Project Name: Ultra-Distributed Autonomous Computing Service Technology Research Team, Project Number:202003520006).
☆ A preliminary version of this paper was presented at ICONI 2024.

all the defined properties and behaviors, thereby adapting cartoon engineering to software engineering. This method facilitates tracking between natural language sentences and UML models, and also generates a clear cartoon through a procedural approach. The paper is organized as follows: Section 2 introduces the 3D modeling process and the existing proposed mechanism. Section 3 describes this research with 3D cartoon extraction. Section 4 concludes the study.

2. Related works

2.1 A 3D Character Generation in the Game Industry

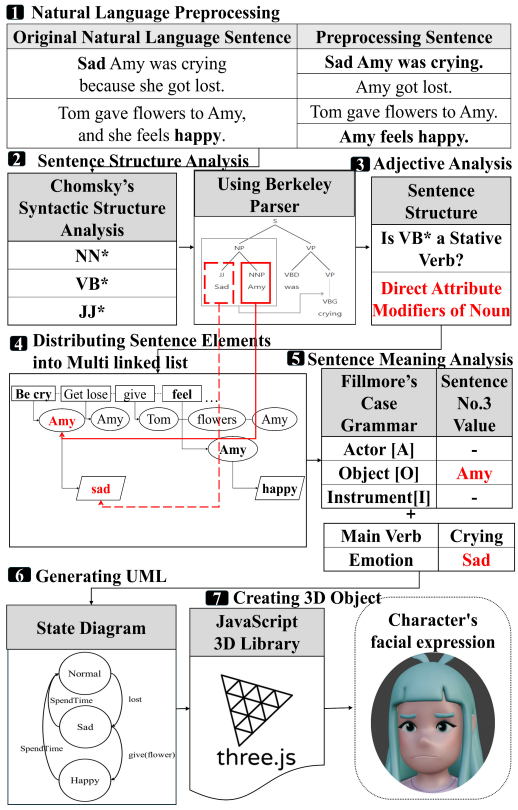
3D character models are primarily used in the game industry. They are now being applied to the cartoon industry. In the current production process, character modeling involves a series of detailed steps and can be broadly divided into six stages [3].

1) Conceptual Design: Determines the character's basic form and style and establishes modeling plans by considering the character's personality and distinct features. 2) Modeling: Builds the 3D form based on 2D designs, visualizes the character's structure, and expresses detailed features. 3) Texturing and Rendering: visualizes the 3D model with textures and lighting, and adds color and texture to the character through texturing. 4) Rigging: adds skeleton and joints to the 3D model and configures the character for natural movement. 5) Animating: applies expressions and motions to the 3D model and conveys the character's personality and emotions. 6) Exporting: prepares the completed character for export in formats compatible with the target environment and enables integration with game engines or other 3D software.

In this study, steps 5 (Animating) and 6 (Exporting) are performed using a 3D model already completed through steps 1 (Conceptual Design) to 4 (Rigging). This process focuses on adding emotional expressions and motions to an existing model and exporting it in a file format suitable for use in a web environment.

2.2 Text-to-3D Approach

A mechanism to generate 3D models from natural language was proposed in our previous study[4]. Figure 1 shows the process of the last study.



(Figure 1) Text to 3D Object Generation Process

Natural language sentences were systematically analyzed and visualized as UML diagrams. The resulting information was then used to generate 3D models, ensuring traceability and consistency between the natural language input, the UML, and the 3D model. A comparison between the proposed mechanism and commercial AI Text-to-3D (TT3D) tools was also presented. The AI tools were unable to generate identical characters or accurately express facial emotions. To solve this problem, we proposed analyzing the emotional state of a main character from natural language using a state diagram in UML models. This approach

enhances the reusability and consistency of cartoon engineering. However, in previous studies, humans had to process this process manually. Therefore, we need to improve this and develop an automated tool through the LLM API.

3. Text to 3D Object via Software Engineering Approach

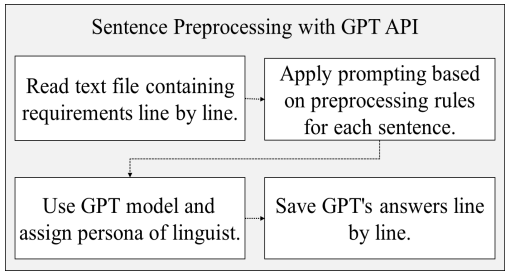
Our research improves the previously proposed mechanism [5, 6]. Also, the study focuses on a 3D authoring tool that creates 3D objects using natural language prompts. Through this approach, we understand the psychological states of objects in cartoon scenes, as inferred from natural language sentences, providing structured data that can be directly applied to 3D modeling. The research process is primarily divided into three parts: 1) Natural language analysis, 2) UML diagram generationand Emotion Extraction with Sequence Diagram, State Diagram, and 3) 3D Character Entity generation. We apply LLM to the natural language analysis process to automate this process. The research environment is presented in Table 1.

(Table 1) Development Environment and Tools

Component	Version	Description
Node.js	20.14.0	JavaScript Runtime
Three.js	0.167.1	3D Library
Express	4.19.2	Web Framework
NLTK	3.5	NLP Library
OpenAI	1.15.2	GPT API

3.1 Natural Language Analysis

When a natural language sentence describing a cartoon scene is input, it is processed in three ways. First, the natural language sentence is preprocessed. Sentences consisting of complex and compound sentences are generally long, and the results of analyzing them are also extended. Therefore, complex sentences are converted into simple sentences. Simple sentences refer to sentences consisting of one subject and one verb. This process is designed as shown in Figure 2, and utilizes the GPT API for sentence preprocessing.



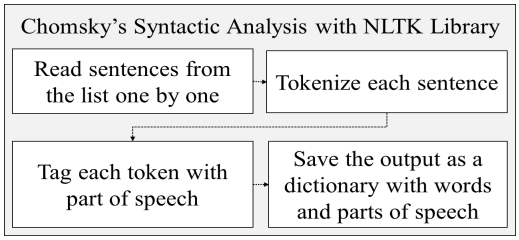
(Figure 2) Sentence Preprocessing Process

First, each line of the text file is input into GPT. Second, rules are defined to transform complex and compound sentences into simple sentences. Third, GPT is assigned the persona of a linguist, and reponses are stored in a single file. Simple sentences are structurally analyzed using Chomsky's syntactic theory of analysis [7]. Since it is essential to distinguish key morphemes in a sentence, the NLTK library is used to analyze the morphemes and part-of-speech tagging of the sentence. Furthermore, the structural type of each sentence is examined according to five basic patterns, as presented in Table 2 [8]. In Table 2, 'S' represents 'Subject,' 'V' for 'Verb,' 'O' for 'Object,' 'I.O' for 'Indirect Object,' 'D.O' for 'Direct Object,' 'C' for 'Complement,' and 'O.C' for 'Objective Complement'.

(Table 2) Types of Sentence Structure

Sentence Type	Sentence Structure
Type 1	S+V
Type 2	S+V+C
Type 3	S+V+O
Type 4	S+V+I.O+D.O
Type 5	S+V+O+O.C

After completing the structural analysis of all sentences, a JSON file is generated. This file contains the analyzed sentence, its order number, sentence structure type, morphemes, and parts of speech. POS tagging is conducted without using GPT. Instead, each preprocessed sentence is taken as input, and the NLTK library is used to segment it into tokens. Figure 3 illustrates the POS analysis procedure.



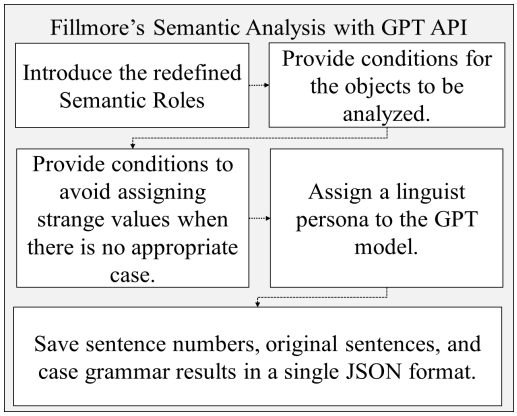
(Figure 3) Chomsky's Syntactic Analysis Process

The preprocessed sentences are stored as a single list, which is automatically read line by line and tokenized. Each token is assigned a POS tag, and the tokens along with their tags are stored in a dictionary format. Based on the results, the sentences are analyzed semantically. For semantic analysis, Fillmore's Semantic Roles theory is applied. Fillmore's theory examines the relationship between verbs and nouns, assigning appropriate roles to nouns [9]. The roles defined in the existing Fillmore theory have various interpretations [10]. Our study is to generate state diagrams through natural language in the cartoon domain. Therefore, we reduced the types of roles proposed by Fillmore and redefined the meanings as shown in Table 3.

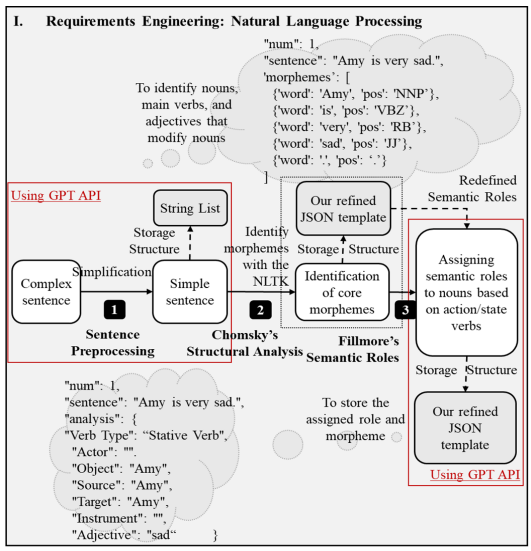
(Table 3) Semantic Roles for Generating State Diagram

Semantic Roles	Definition
Actor	The instigator of the event/action.
Object	The entity that changes or exists is under consideration.
Source	The entity performing the action..
Target	The entity receiving the action
Instrument	The implement used in carrying out an event/action.

To assign a role to a noun, check the JSON file of the structural analysis result to see what kind of structure the sentence has and what the main verb is. As shown in Table 3, appropriate semantic roles are assigned to nouns in positions such as the subject or object, depending on the sentence structure and type of verb. Based on this structural information, the GPT prompt is configured for automation, as shown in Figure 4.



(Figure 4) Fillmore's Semantic Roles Process



(Figure 5) Natural Language Analysis Process

The semantic roles (actor, object, source, target, and instrument) from Fillmore's theory are described within the GPT prompt. A condition is applied to assign a role only when the POS tag begins with 'NN', and another condition connects nouns with underscores ('_') to form a single word. Additionally, a constraint is added to prevent assigning a value when a noun does not correspond to an appropriate semantic role. In this semantic analysis process, GPT is assigned the persona of a linguist. Once the semantic analysis is complete, it is returned as a JSON file. The JSON

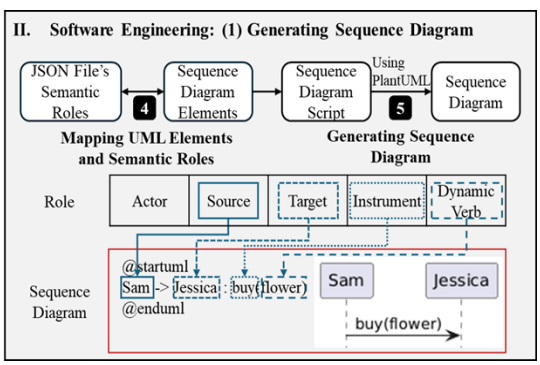
file consists of the analysis target sentence, the sentence order number, and the Main Verb, Emotional Adjective, and Fillmore Roles, which are the analysis results. Figure 5 shows the entire process of analyzing natural language. We applied the GPT API in the natural language analysis step and output the results in JSON format for each step.

3.2 UML Diagram Generation

To analyze the emotional state changes of the main character, we create both a UML Sequence Diagram and a State Diagram. To achieve this, the analyzed data from the sentence analysis are utilized. Refined semantic roles(actor, object, source, target, and instrument) are applied to generate sequence diagrams and state diagrams, particularly in sentence containing dynamic verbs. The dynamic verb of the sentence is the key part of the noun relationship within the sentence. It will also illustrate how the main object, which is an actor in the sentence, interacts with other objects.

1) Sequence Diagram Generation:

A sequence diagram illustrates the interactions between two objects over time. This diagram clearly shows the flow of events that occur during the interaction of objects in Figure 6.

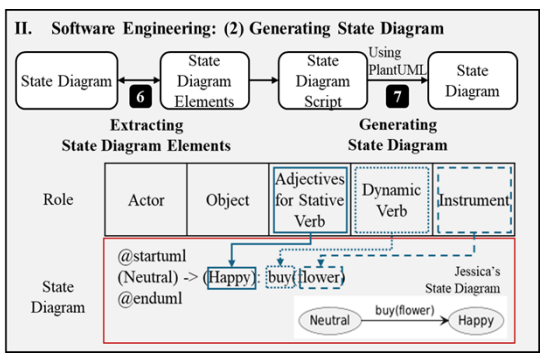


(Figure 6) Generating Sequence Diagram from Semantic Roles

Based on the natural language analysis information from the previous step, objects and their interactions are extracted and represented sequentially. The flow of events arranged in this way becomes the basic framework for the sequence of actions required to create a 3D model. However, because sequence diagrams represent only sequential flows and interactions between objects, the emotional changes between them are unknown.

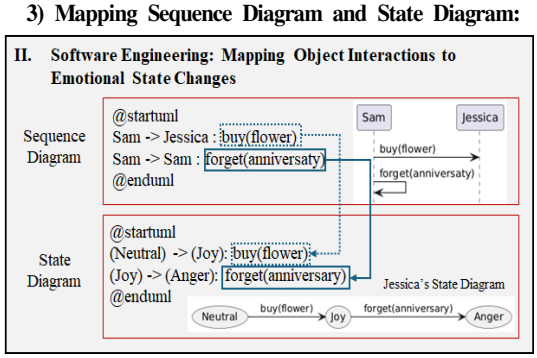
2) State Diagram Generation:

In comics, since the character's emotions change depending on specific actions, 'Action Verb' is used as the 'Event' of the state diagram. The diagram is completed by writing 'Instrument' used for actions as 'Event's Parameter' and adjectives expressing emotions as 'State'. Figure 7 shows the process of generating a State Diagram from natural language. Using the PlantUML library, the diagram script (.txt) and diagram picture (.png) files are generated.



(Figure 7) Generating State Diagram from Semantic Roles

3) Mapping Sequence Diagram and State Diagram:



(Figure 8) Sequence and State Diagram Mapping

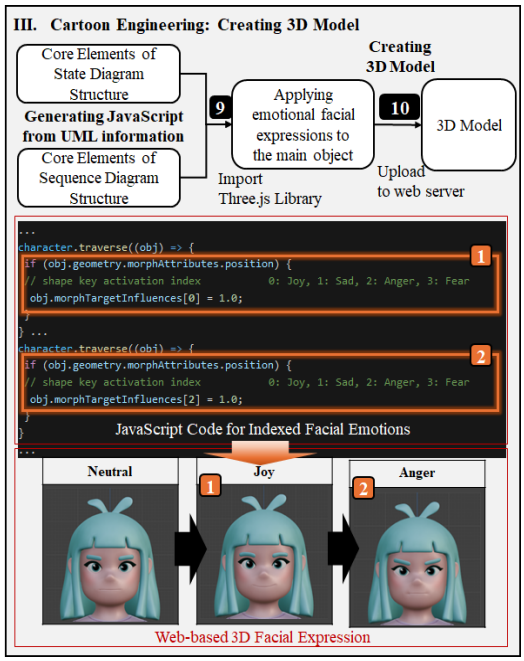
State diagrams and sequence diagrams have a

complementary relationship, with each one compensating for the limitations of the other. A sequence diagram extracts the interactions between objects from a sentence and clearly shows the sequential order of actions over time. However, this diagram alone has a limitation: it cannot identify how the internal emotional state of each object changes over time. Conversely, a state diagram is effective for visualizing the emotional changes of a specific object. However, it is difficult to determine what interaction caused that emotional change. To solve this problem, we link the two diagrams to fill in their missing parts. Figure 8 illustrates how to map object interactions to changes in emotional state.

The verbs representing events in the sequence diagram are mapped to the verbs connecting the transitions between states in the state diagram. This mapping allows us to verify and visualize the causal relationship between object interactions and state changes. As a result, when generating a 3D model, we can accurately implement not only the actions of the objects but also the emotional changes in a chronological and sequential flow.

3.3 A 3D Cartoon Extraction in the 3D Environment

To express the emotional state of the 3D model from both the sequence and state diagrams, we load the generated diagram script file. We extract the diagram information from the file and apply it as the character's expression. There are five expressions: Normal, Joy, Sadness, Anger, and Fear, and they are indexed in order. In previous research, we focused on expressing a single expression of a 3D object using a state diagram. However, by presenting a methodology for fusing sequence diagrams and state diagrams, we seek to explore the possibility of sequentially expressing the emotional changes of objects over time. This is important because in cartoons, a person's emotional line changes according to the situation and changes while interacting with other objects or environments. The emotional expressions are stored as morph targets transformed from the original shape of the 3D model, and the degree corresponding to each emotional state can be adjusted from 0 to 1. '0' means not applied at all, and '1' means fully applied. By making this change, various expressions corresponding to the emotional state can be implemented. Figure 9 illustrates the sequential change in an object's facial expression using a diagram. The transition from the 'Joy' to the 'Anger' state is confirmed through a state diagram. A sequence diagram is then used to identify the specific order and the message flow that causes this change. To express emotions sequentially, we apply an index to each corresponding emotion in order. In Figure 9, the index changes from 0 to 2, which represents the emotional shift from 'Joy' to 'Anger'.

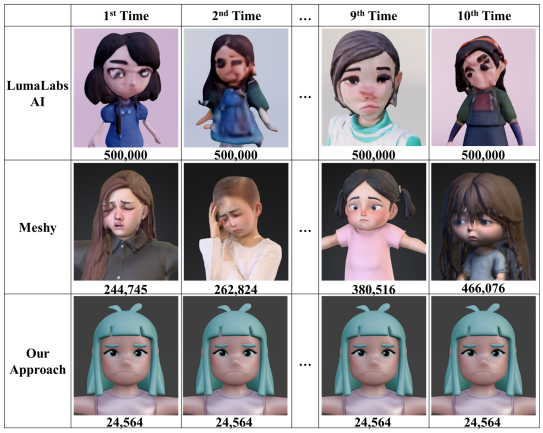


(Figure 9) Import 3D Models to the Web

The 3D models generated through the proposed process were compared with those produced by commercial tools. Figure 10 presents the comparison results of ten models generated using the proposed method, Meshy, and LumaLabs AI, evaluated in terms of polygon count and model quality.

LumaLabs AI failed to produce accurate 3D models in all trials and struggled to represent facial details, making it challenging to identify the intended sad expression. Although it produced the highest number of polygons, 50,000, the overall level of detail was low. Meshy demonstrated a relatively accurate representation of emotional expressions such as sadness; however, in the first trial, the model

exhibited unnatural results where the face overlapped with the hair region due to the smallest polygon count. In the second trial, the boundaries between the hands and face were indistinct. In general, AI-based model generation tools focus on producing new outputs, which limits their ability to apply multiple expressions to the same model. In contrast, the proposed process allows a single model to exhibit various facial expressions. This feature demonstrates the suitability of the proposed automated 3D modeling system for application in comic production.



(Figure 10) Comparison of 3D Models based on Commercial Tools and Our Research Process

The superiority of the proposed method is also evident in the quantitative comparison based on polygon count. In general, polygon count represents the geometric complexity of a model, and an increased number of polygons directly leads to higher GPU computational load and reduced rendering speed[11]. Excessively high polygon counts are particularly unsuitable for real-time applications such as games, VR, and web-based 3D environments. Typically, simple models contain approximately 1,000 ~ 5,000 polygons, mid-level models range from 10,000 to 50,000 polygons, and high-quality character models fall within the range of 50,000 ~ 200,000 polygons. However, in our experiments, LumaLabs generated models with approximately 500,000 polygons, and Meshy produced 244,745 ~ 466,076 polygons, resulting in overly complex and inefficient mesh structures for real-time 3D usage. In contrast, our method can represent all facial expressions with a lightweight model of approximately 25,000 polygons, whereas generative AI systems typically generate a separate high-polygon model for each expression. The proposed model successfully preserves both the accuracy of emotional expression and the structural stability of the geometry. Therefore, the proposed approach achieves the best balance between quality and performance, offering both efficiency and scalability for comic production pipelines and real-time 3D applications.

4. Conclusion

In this research, we aim to apply cartoon engineering principles to linguistic and software engineering, thereby enhancing reusability, consistency, and traceability. The application of these principles is expected to improve productivity and convenience in 3D cartoon development. The proposed study introduces a mechanism for generating 3D models from natural language and automates the process using the GPT API. We systematically analyze the natural language of comic scenes and convert them into 3D models. The GPT API is employed to process natural language that is difficult to handle using rule-based methods. Sequence and state diagrams are generated to clarify natural language sentences with low visibility. A significant contribution of this research is the visualization of low-visibility natural language sentences into sequence and state diagrams. These diagrams are crucial for maintaining consistency between natural language and 3D models by applying the information they contain to the expressions of 3D character entities. Specifically, the sequence diagram clearly shows the temporal flow of interactions between objects, which helps determine the order of movements and emotional transitions of the 3D model. Also the state diagram captures the emotional state transitions of a single object, enabling the concretization of expressions at specific points in time.

In future work, we plan to integrate the proposed cartoon creation process with Toonsquare Inc.'s "Tooning.ai.". In addition, we aim to completely automate the 3D model generation process, enabling the generation of 3D models with only natural language input.

Reference

[1] Fortune Business Insights, 3D Mapping & 3D Modeling Market Size, Share & Industry Analysis, 2024-2032, October 21 2024. Retrieved from https://www.fortunebusinessinsights.com/3d-mapping-and-modeling-market-106003

[2] M. H Jeoung, “A Study on the 3D Computer Graphics Application in Webtoons,” Smart Media Journal, vol.4, no.3, pp.31-37, 2015. https://www.kci.go.kr/kciportal/ci/sereArticleSearch/ciSereArtiView.kci?sereArticleSearchBean.artiId=ART002033176

[3] Kovanen, S., Functional Workflow in 3D Character Design, Karelia University of Applied Sciences, 2015. https://www.theseus.fi/bitstream/handle/10024/98327/Kovanen_Simo.pdf

[4] Y.J. Jin, C.Y. Seo, J.H. Kong, R. Y. C. Kim, “3D Object State Extraction Through Adjective Analysis from Informal Requirements Specs,” The Transactions of the Korea Information Processing Society (TKIPS), vol.13, no.10, pp.529-536, 2024. http://doi.org/10.3745/TKIPS.2024.13.10.529

[5] Y. Jin, C. Seo, & R. Y. C. Kim, “Generating 3D Models through Analyzing Natural Language Sentences with GPT API,” in Proc. of the 17th International Conference on Internet, vol.16, pp.278-282, 2024.

[6] H. Kim, J. Kim, J. Kong, K. Kim, & R. Y. C. Kim. “3D Object Extraction Mechanism from Informal Natural Language Based Requirement Specifications,” The Transactions of the Korea Information Processing Society, vol.13, no.9, pp.453-459, 2024. https://doi.org/10.3745/TKIPS.2024.13.9.453

[7] N. Chomsky, Syntactic structures, USA: Mouton de Gruyter, 2002.

[8] Betti, M. J., Sentence Patterns in English. University of Thi-Qar, 2021.

[9] C. J. Fillmore, The Case for Case, New York: HR&W, 1968.

[10] B. K. Park, and R. Y. C. Kim, “Effort estimation approach through extracting use cases via informal requirement specifications,” Applied Sciences, vol.10, no.9, p.3044, 2020. https://doi.org/10.3390/app10093044

[11] N. L. Webster, “High poly to low poly workflows for real-time rendering,” Journal of visual communication in medicine, vol.40, no.1, pp.40-47, 2017. https://doi.org/10.1080/17453054.2017.1313682

◐ Authors ◑

Yejin Jin

2023: Received the B.S. degree in Dept. of Software and Communications Engineering from Hongik University, South Korea.

2025: Received the M.S. degree in Dept. of Software and Communications Engineering from Hongik University, South Korea.

2024 ~ Present: Ph.D. Student in Dept. of Software and Communications Engineering from Hongik University, South Korea.

Research Interests (ongoing): Natural Language-based Code Generation, Text-to-Emotion Recognition, Software Validation, and Software Visualization.

e-mail: yejin_jin@g.hongik.ac.kr

Janghwan Kim

2019: Received the B.S. degree in Computer Science from Idaho State University, Pocatello, ID, USA.

2022: Received the M.S. degree in Dept. of Software and Communications Engineering from Hongik University, South Korea.

2024 ~ Present: Adjunct Professor, Department of Software Convergence, Hongik University, South Korea.

Research Interests (ongoing): AI Software Validation, Reinforcement Learning-based Software Validation, Software Visualization, and Requirements Engineering.

e-mail: lentoconstante@hongik.ac.kr

Hyunseung Son

2007: Received the B.S. degree in Dept. of Computer and Information Communications Engineering from Hongik University, South Korea.

2009: Received the M.S. degree in Dept. of Electronics and Computer Engineering from Hongik University, South Korea.

2015: Received the Ph.D. degree in Dept. of Electronics and Computer Engineering from Hongik University, South Korea.

2021 ~ Present: Assistant Professor at Dept. of Computer Engineering, Mokpo National University, South Korea.

Research Interests (ongoing): Software Engineering, Metamodel, Model Transformation, Testing

E-mail : hson@mnu.ac.kr

R. Young Chul Kim

1985: Received the B.S. degree in Computer Science from Hongik University, Republic of Korea.

2000: Received the Ph.D. degree in Software Engineering from the Department of Computer Science, Illinois Institute of Technology (IIT), Chicago, USA.

2000 ~ Present: Professor at Dept. of Software and Communications Engineering Hongik University, Seoul, South Korea

Research Interests (ongoing): Test maturity models, Model-based Testing, Metamodeling, Software Process Models, Software visualization, and Validation Techniques for Reinforcement Learning Software.

e-mail: bob@hongik.ac.kr