한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 32, No. 3, P. 121~131

가상 프로덕션에서의 시각 언어 모델을 이용한 각본 기반 카메라 촬영 자동화

한지민° 김영준*

컴퓨터공학과, 이화여자대학교

jimin.han@ewha.ac.kr° kimy@ewha.ac.kr*

Script-driven Autonomous Cinematography in Virtual Production using Vision Language Models

Jimin Han° Young J. Kim*

Dept. of Computer Science and Engineering, Ewha Womans University

요 약

LED 월 기반 가상 프로덕션은 실시간 가상 배경과 촬영 환경을 통합하는 제작 방식으로 최근에 대두되고 있다. 그러나 자연어로 표현된 촬영 대본을 촬영 의도에 맞게 해석하는 촬영 자동화는 가상 프로덕션 제작 분야에서 아직 어려운 문제이다. 본 논문에서는 가상 프로덕션 환경에서 시각 언어 모델을 활용한 각본 기반 촬영 자동화 시스템을 제안한다. 제안 시스템은 대규모 언어 모델을 이용하여 자연어 각본으로부터 구조화된 촬영 계획을 생성하고, 목표 피사체 선택, 피사체 추적, 로봇 카메라 제어를 통해 계획된 카메라 움직임을 실제 촬영으로 실행한다. 또한 촬영 결과가 각본 의도를 충분히 반영하지 못하는 경우, 시각 언어 모델을 이용해 결과를 관찰하고 촬영 계획을 갱신한다. 본 연구에서는 LED 월, 카메라 추적 시스템, 로봇 카메라 시스템을 통합한 가상 프로덕션 환경에서의 세 가지 시연을 통해, 제안된 시스템이 자연어 각본의 촬영 의도를 반영한 촬영 자동화를 구현할 수 있음을 검증하였다.

Abstract

LED Wall-based virtual production has emerged, integrating real-time background rendering with physical filming environments. However, camera automation for interpreting natural-language scripts according to cinematic intent in virtual production remains challenging. We propose a VLM-based script-driven camera automation system for virtual production. The proposed system generates a structured shot plan from a given natural-language script using an LLM, selects and tracks the target subject, and controls a robotic camera system according to the planned camera movement. Moreover, when the executed result does not sufficiently reflect the script's intention, a VLM observes the result video and updates the shot plan. Through three script-based demonstration scenarios in a virtual production environment with camera tracking and a robotic camera system, the results demonstrate that the proposed system can automate camera control while reflecting the cinematic intent described in a natural-language script.

키워드: 영화 촬영 자동화, 각본 기반 카메라 제어, 시각 언어 모델, 가상 프로덕션
Keywords: Autonomous Cinematography, Script-driven Camera Control, Vision-Language Models, Virtual Production

*corresponding author: Young J. Kim / Dept. of Computer Science and Engineering, Ewha Womans University (kimy@ewha.ac.kr)

Received : 2026.05.29./ Review completed : 1st 2026.06.07. 2nd 2026.06.25./ Accepted : 2026.06.29.
DOI : 10.15701/kcgs.2026.32.3.121
ISSN : 1975-7883(Print)/2383-529X(Online)

1. Introduction

LED Wall-based virtual production enables virtual backgrounds to be rendered in real time and captured directly by a physical camera on set [1]. This approach allows virtual environments, physical lighting, and actors' performances to be observed together during filming. Virtual production is a complex system that integrates LED walls, real-time rendering, camera tracking, and robotic camera control [2]. Despite these technical advances, camera operation still largely depends on the expertise of skilled human operators.

Previous studies on autonomous cinematography have mainly focused on virtual camera placement, camera trajectory generation, and camera movement planning [3, 4, 5, 6]. Drone and robotic camera studies have demonstrated the feasibility of automatic physical camera control [7, 8], but they generally assume a given real-world background and focus on controlling camera viewpoints or trajectories. Recent Large-Language Model (LLM)- and Vision-Language Model (VLM)-based film production studies have shown the potential of generating shot compositions, scene layouts, and camera settings from textual inputs [9, 10, 11]. However, relatively little attention has been given to an integrated system that connects script interpretation, camera tracking, robotic camera execution, and result observation within an actual LED Wall-based virtual production environment.

Meanwhile, translating the cinematic intention in a natural-language script into physical camera motion requires more than simply moving a camera. The system must interpret which subject is important in the script, identify and track the corresponding subject in the actual scene, and control the physical camera based on the intended shot and the subject's motion. In this process, a gap remains between natural-language cinematic intent, the visual subjects in the scene, and executable camera control commands. To bridge this gap, LLMs and VLMs are useful because they can interpret the language instructions and connect them to visual evidence in images [12, 13].

Mascelli [14] insists that "every shot must serve the story" as a key principle of filmmaking. In this sense, camera shooting can be understood as a choice for narrative delivery. Bordwell and Thompson [15] explain that "a film employs cues in order to involve us." They also distinguish a set of narrative elements which constitute the film's story, from stylistic elements which includes how the camera moves. Based on this view, script-driven automatic camera control should be organized around the process of conveying narrative elements through cinematographic decisions: the system must interpret the intended subject and action from the script, ground them in the actual scene, execute camera movement to present them visually, and verify whether the resulting video makes them understandable.

In this paper, we propose a script-driven autonomous cinematography system for LED Wall-based virtual production. Given a natural-language script, the proposed system generates a structured shot plan that includes subject and camera movement. The shot plan serves as a system-level intermediate representation shared by the subject selection, visual tracking, and robotic camera control modules. Based on this representation, the system selects the target subject in the camera image, tracks it over time, and converts the tracking results into pan, tilt, and linear motion commands for robotic camera execution in a virtual production environment. After execution, the resulting video is analyzed to verify whether it sufficiently reflects the script's intention, and the shot plan is updated if it does not. We implement the proposed system using an LED Wall, a game engine, camera tracking, and a robotic camera system to demonstrate this system-level feasibility.

The main contributions of this system are as follows. First, we propose a script-driven autonomous camera control system that connects shot plan generation, subject selection, tracking, robotic camera control, and post-execution result observation in an LED Wall-based virtual production environment. Second, we use a structured shot plan as an executable interface that connects script-level cinematic intent with target subject selection, tracking, and robotic camera control and shot plan revision. Third, we demonstrate the feasibility of the proposed system in an actual virtual production environment consisting of an LED Wall, Unreal Engine, camera tracking, and a robotic camera system.

2. Related Works

2.1 Virtual Production and Camera Control

Virtual production has evolved from chroma-key-based compositing [16] and simulcam [1] workflows toward LED Wall-based in-camera VFX. In LED Wall-based virtual production, virtual backgrounds rendered by a real-time game engine are displayed on LED panels and captured directly by a physical camera. This allows the actor, lighting, and virtual environment to be integrated during filming, while the camera pose is tracked and reflected in the virtual scene [17, 18]. Therefore, an LED Wall-based virtual production stage should be regarded as a physical production system that integrates LED Wall configuration, camera tracking, multi-display synchronization, and camera operation.

Although virtual production technologies have improved production efficiency and visual consistency [19, 20], camera operation in such environments still requires skilled human operators. For small teams or non-expert users, controlling the camera while

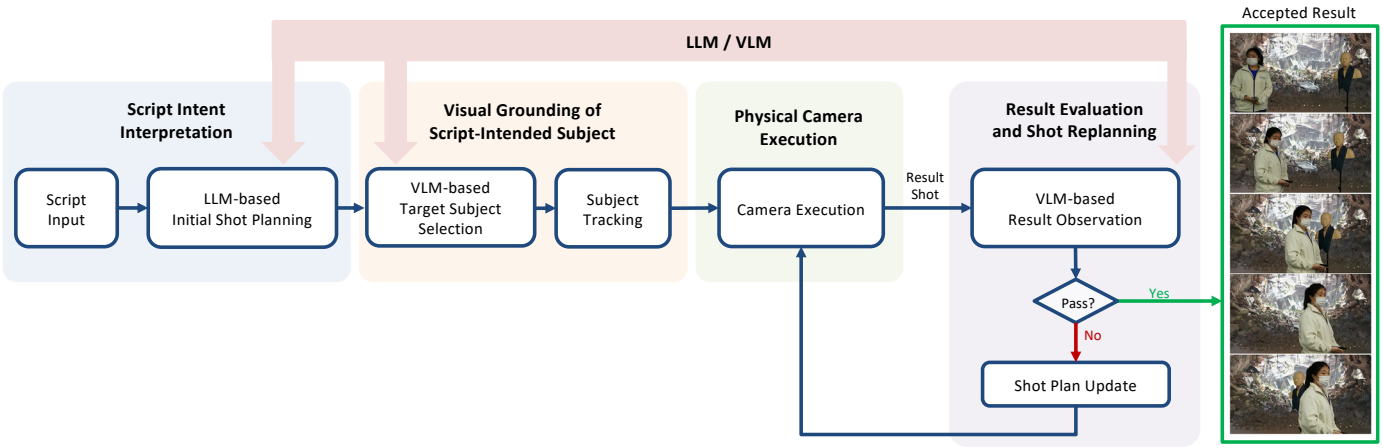

Figure 1: Pipeline of the autonomous cinematography system for virtual production.

also managing virtual production-specific components remains difficult. Therefore, camera automation in virtual production requires a system-level approach that connects cinematic intent, physical camera tracking, robotic camera control, and the virtual production rendering environment.

2.2 Autonomous Cinematography

Classical cinematography literature defines camera placement, shot size, framing, and camera movement as important elements for conveying visual and narrative intention [21]. Based on these principles, autonomous cinematography has been studied in virtual cinematography, and robotic camera control. Early work on virtual camera control formulated camera control as a problem of computing camera parameters, viewpoints, or movements that satisfy user-specified visual or narrative constraints [22, 23]. Subsequent studies introduced camera representations and control methods such as toric space [4], example-driven virtual cinematography [5], and camera keyframing with style and control [6]. These approaches contributed to generating plausible camera placements, trajectories, and camera motions in virtual environments.

Another line of work has explored physical camera automation in drone and robotic camera systems. Drone cinematography studies have planned camera paths that follow a subject while satisfying cinematographic constraints such as shot size, composition, and visibility [7, 24]. Other robotic camera systems have used visual feedback or user-provided references to adjust physical camera viewpoints and improve image composition [25, 26].

These studies show that camera placement and trajectory generation can be automated under predefined visual objectives. However, most prior work assumes that the target subject and the desired camera behavior are already given and primarily focuses on generating viewpoints or trajectories. In contrast, we start from a natural-language script, infer the target subject and camera movement, and convert them into a structured shot plan for physical camera execution.

2.3 LLM-based Film Production

Recent advances in LLMs have enabled text-based planning and script interpretation for film and video production. Prior studies such as VideoDirectorGPT [10], and Anim-Director [11] use LLMs to decompose textual inputs into scene layout, visual composition, and cinematographic instructions. FilmAgent [9] further shows that LLM-based agents can generate cinematography-related plans in a virtual filmmaking environment. Our preliminary study also explored script-driven camera automation in a physical virtual production setup by converting natural-language scripts into structured shot plans and executing them through a robotic camera system [27]. However, these approaches primarily focus on generating or executing camera plans and offer limited mechanisms for verifying execution results and revising the shot plan when the initial plan does not adequately present the intended subject or action.

3. Proposed System

3.1 System Overview

Fig. 1 shows the overall pipeline of the proposed script-driven camera automation system for virtual production. The system connects natural-language script understanding to physical camera execution through shot planning, target subject selection, visual tracking, robotic camera control, and executed-video observation. Given a script, the system generates a structured shot plan and uses it as the main interface for target subject selection, visual tracking, and robotic camera control. Through this pipeline, script-level cinematic intent is translated into executable camera behavior, includ-

3.2.3 Tracking System and Inner Frustum Setup

For camera tracking, we used a Vive Mars-based tracking system to transmit the position and rotation of the physical camera to Unreal Engine in real time. A Vive tracker was attached to the camera rig, and two base stations were installed to ensure stable tracker recognition. The tracking data were sent to Unreal Engine through LiveLink and applied to the virtual camera. As the physical camera moves, the virtual camera pose and the inner frustum are updated every frame to keep the virtual background on the LED Wall aligned with the physical camera viewpoint.

3.2.4 Camera-Tracker Calibration

Accurate virtual production shooting required consistent alignment among the coordinate systems of the physical camera, tracker, LED Wall, and Unreal Engine scene. Since the tracker was mounted above the camera body, we calibrated the fixed offset between the tracker's reference point and the camera's optical center. The center of the LED Wall floor was then set as the reference point for the tracking space and aligned with the nDisplay root coordinate system. After that, the reference anchor, tracker object, and virtual camera were arranged in a parent-child hierarchy.

Through this configuration, the position and rotation of the physical camera were transferred into the nDisplay coordinate system, allowing the physical camera movement and the Unreal Engine virtual camera movement to maintain the same viewpoint relationship. The field of view of the physical camera lens and the stage layout information were also reflected in the Unreal Engine settings to ensure spatial consistency between the virtual background and the captured image.

3.2.5 Robotic Camera System

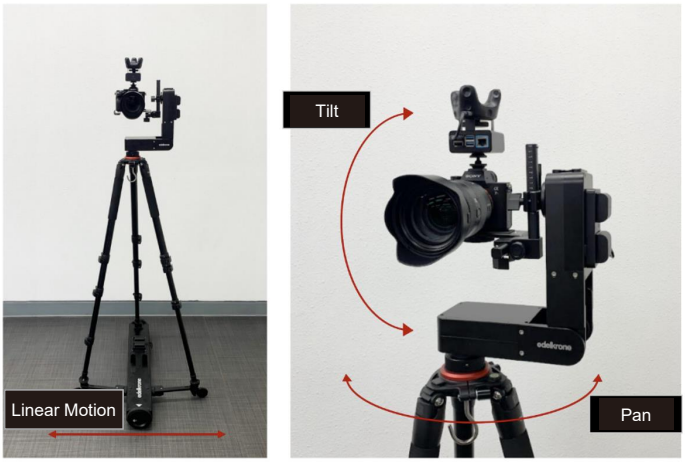

Figure 4: Robotic camera system used for physical camera execution.

The robotic camera is shown in Fig. 4. We used an Edelkrone motion control system[3] that supports pan, tilt, and linear motion. The system receives wireless commands via an HTTP-based control protocol, and the proposed system sends Python commands to control each motion axis in real time. The pan and tilt axes are used to adjust the subject position within the frame, while the linear motion axis is used to reproduce planned camera movements and adjust the camera-subject distance. The motion speed and easing parameters of each axis were empirically adjusted to obtain smoother camera movement in the physical filming environment.

3.3 Script-driven Shot Planning

The first stage in Fig. 1 converts a natural-language script into a structured shot plan. Before shot planning, the language model is initialized with a planning prompt that includes cinematography domain knowledge, predefined shot-plan fields, allowed camera movement categories, output-format instructions, and example input-output pairs. In the current implementation, the shot plan consists of two fields: `subject` and `camera_movement`.

The purpose of this stage is to convert the script into an executable representation that can be directly used by the following modules. The subject selection module uses the `subject` field to identify the target in the image and initialize the tracking subject. The camera control module uses the `camera_movement` field to determine the robotic camera motion mode.

The `subject` field is represented as a natural-language description of the target subject, such as `A woman`, `the person wearing earphones`, or `Person B`. This field is used as a query for selecting the target subject in the initial camera frame. The `camera_movement` field is selected from the camera movement candidates: `static`, `track`, `dolly_in`, `dolly_out`, `truck_left`, and `truck_right`. These candidates are defined based on the controllable motion axes of the robotic camera system. The shot-planning stage outputs a JSON-formatted shot plan, such as {`"subject": "A woman", "camera_movement": "track"`}. During post-execution replanning, only the `subject` field or the `camera_movement` field is revised based on the observed failure evidence, and the updated shot plan is re-executed.

3.4 Visual Grounding of the Target Subject

The visual grounding stage in Fig. 1 connects the subject described in the shot plan with the corresponding visual subject in the camera image. The `subject` field is extracted from the shot plan and

[3] `https://edelkrone.com`

used as a text query for target subject selection. This stage is necessary because the script-level subject description, such as `A woman` or `the person wearing earphones`, does not directly indicate which visual subject in the camera image should be tracked.

In the current implementation, the system first captures the initial camera frame and detects human candidates using YOLO [28]. Each detected candidate is assigned a target ID, and the initial frame is annotated with the candidate bounding boxes and target IDs. The VLM receives the annotated frame and the `subject` field from the shot plan, and outputs the target ID that best matches the script-intended subject.

The selected target ID is used as the tracking target in the YOLO and DeepSORT [29] tracking module. During camera execution, the tracker follows the selected subject and provides the target bounding box for camera control. If the selected subject is found to be inconsistent with the script intention through post-execution observation, the subject field can be revised during shot replanning and the target subject can be selected again in the next attempt.

3.5 Physical Camera Execution

The camera control stage in Fig. 1 executes the camera movement specified in the shot plan using the tracking result and the robotic camera system. During execution, the system tracks the selected subject using YOLO and DeepSORT and obtains the target bounding box for each frame. The control policy depends on the `camera_movement` field in the shot plan.

For `track`, the system uses the center error between the target subject's bounding box and the image frame to keep the target subject near the center of the frame. Let (x_c, y_c) denote the center of the target subject's bounding box, and let W and H denote the frame width and height. The horizontal and vertical errors are computed as

$$e_x = x_c - \frac{W}{2}, \quad e_y = y_c - \frac{H}{2}.$$

The pan axis is controlled using the horizontal error e_x, while the tilt axis is controlled using the vertical error e_y.

For `dolly_in` and `dolly_out`, the system controls the linear motion axis based on the subject size in the frame. In the current implementation, shot-size adjustment is defined between `full shot` and `medium shot`. The face bounding box size is used to estimate whether the subject is closer to the target shot size. The camera moves forward for `dolly_in` and backward for `dolly_out` until the target shot-size condition is reached. In the current setup, the linear motion axis is configured for one direction per execution. Therefore, `dolly_in`/`dolly_out` and `truck_left`/`truck_right` are treated as separate linear-motion modes and are not executed simultaneously.

For `static`, all motion axes are set to zero velocity, and the camera remains fixed. For `truck_left` and `truck_right`, the robotic camera moves along the linear axis with a predefined speed in the left or right direction. These movements do not use tracking-based speed adjustment in the current implementation.

The resulting velocity commands are sent to the robotic camera system through the HTTP-based command interface, while the tracked physical camera pose is reflected in the Unreal Engine camera actor and the LED Wall inner frustum.

3.6 Result Evaluation and Shot Replanning

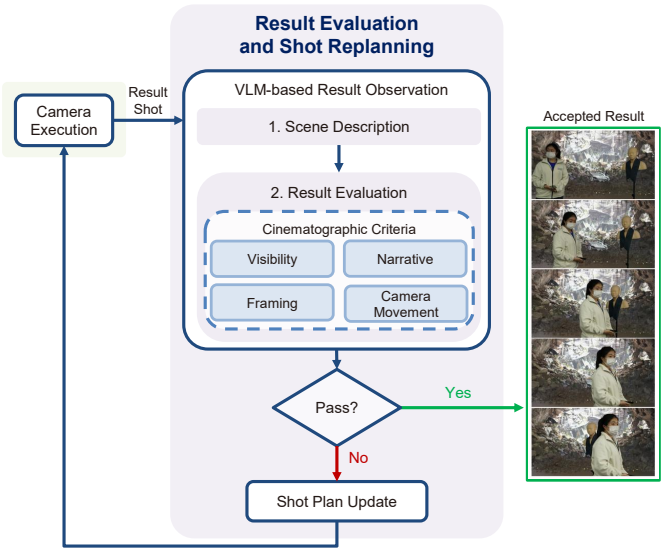

Figure 5: Detailed view of the Result Evaluation and Shot Replanning stage in Fig. 1.

After camera execution, the recorded result video is observed to verify whether the script intention is visually presented in the executed shot. As shown in Fig. 5, this stage consists of three steps: scene description, result evaluation, and shot replanning.

In the scene description step, the VLM receives sampled frames from the executed video and describes the visual result in a structured dictionary format. The description is organized into four categories: visibility, framing, narrative emphasis, and camera movement. These perspectives are organized based on prior discussions in cinematography and autonomous cinematography [30, 15]. The visibility field describes which subjects and actions are visible in the scene. The framing field describes where the subject is located in the frame and how the subject moves over time. The narrative emphasis field describes which subject is visually dominant and why it appears dominant. The camera movement field describes how the target subject appears over time and whether the camera

motion maintains the subject and action within the frame.

This process follows a VQA-style evaluation [31], where the VLM answers whether the executed video satisfies each cinematographic criterion based on the sampled frames. In the result evaluation step, the VLM receives the script, the current shot plan, sampled frames, and the scene description. The output is a structured pass/fail result for four cinematographic criteria: visibility, framing, narrative emphasis, and camera movement. Visibility checks whether the main subject and action described in the script are visible. Framing checks whether the main subject and action are appropriately positioned within the frame. Narrative emphasis checks whether the visually dominant subject corresponds to the script-relevant target. Camera movement checks whether the camera motion maintains the visibility of the main subject and action over time. The shot is accepted only when all four criteria are evaluated as successful.

If any criterion fails, the system performs VLM-based shot replanning. The VLM receives the script, the current shot plan, the scene description, the result evaluation, and the attempt history, which includes previous shot plans and reasoning results. To isolate the effect of each revision, only one major field is updated in each attempt: either `subject` or `camera_movement`. The revised `subject` is represented as a natural-language target description, while the revised `camera_movement` is selected from the predefined camera movement candidate set. The updated shot plan is then passed back to the execution stage for the next attempt. The maximum number of attempts is set to five.

4. Experiments

4.1 Experimental Setup

Experiments were conducted using the LED Wall-based virtual production system described in Section 3.2. The setup included a Sony A7R IV camera, Vive Mars CamTrack, Unreal Engine with nDisplay, and an Edelkrone robotic camera system supporting linear translation, pan, and tilt. GPT-4.1[4] was used as the language and vision-language model for shot planning, subject selection, executed-video observation, result evaluation, and shot replanning.

The experiments used two virtual backgrounds: a cave[5] and a subway train[6]. The cave scene contained a relatively large number of detailed background objects, with an asset size of approximately 17GB, whereas the subway train scene was lighter, with approximately 800MB asset size. The LED Wall output was configured at 7680×2160 resolution through nDisplay. During the experiments, the rendering frame time was observed to be approximately 16.7ms per frame, corresponding to the target 60 fps output.

We used three shot-level test scripts that describe a target subject, subject action, and, when specified, an intended camera movement. These scripts were used to examine whether the system can connect script-level intent to physical camera execution and result-based shot replanning in the virtual production setup.

4.2 Demonstration

We tested the system using three script-based demonstration scenarios. Since the goal of this study is to verify whether script-level intent can be connected to physical camera execution and result-based shot replanning in an actual virtual production system, we used shot-level scripts rather than long multi-shot sequences. Script 1 and Script 2 were designed to examine whether the system can execute planned camera movements from natural-language scripts, while Script 3 was designed to examine whether the executed video can be used for shot replanning when the initial result does not sufficiently maintain the intended subject.

4.2.1 Script 1

The first scenario uses the script, “The person is still standing. His/her expression turns serious. Camera Dolly in.” This script explicitly describes both the target subject’s action and the intended camera movement. From this script, the system generates a shot plan with the target subject and `dolly_in` movement. As shown in Fig. 6, the camera moves closer to the subject over time, and the subject is progressively framed in a tighter shot.

4.2.2 Script 2

The second scenario uses the script, “A person walks into the cave. Camera track.” This script describes the target subject’s action and specifies a tracking movement. From this script, the system generates a shot plan with the target subject and `track` movement. As shown in Fig. 7, the camera follows the subject as the subject walks into the cave scene, and the subject remains visible across the captured frames.

4.2.3 Script 3

The third scenario uses the script, “Person B walks past A and heads forward.” This script describes the intended action at an abstract level, but the visual identity of Person B must be grounded in the actual scene. From this script, the system generates an initial shot plan and executes the planned camera movement. As shown

[4] `https://openai.com`
[5] `https://fab.com/s/c9a039f679f8`
[6] `https://fab.com/s/466470e6868a`

Figure 6: Result for Script 1. The planned Dolly-in movement is executed, and the subject is gradually framed in a tighter shot.

Figure 7: Result for Script 2. The planned track movement is executed while the subject walks into the cave scene and remains visible across the captured frames.

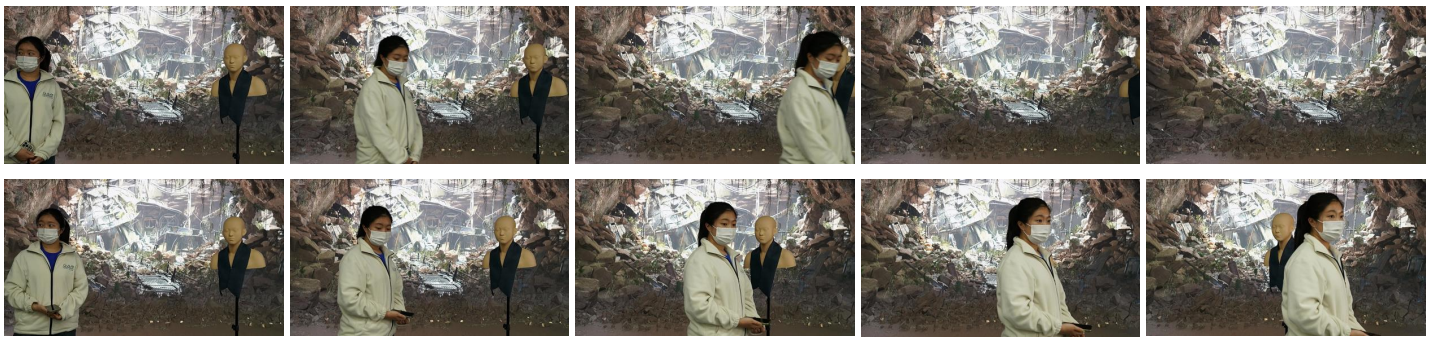

Figure 8: Initial and final results for Script 3. The top row shows the initial attempt, where the intended subject moves out of the frame. The bottom row shows the final attempt after camera replanning, in which the subject is more consistently visible and better framed.

in the top row of Fig. 8, the initial execution does not sufficiently maintain the intended subject within the frame. The executed video is used for result evaluation and shot replanning. The revised shot plan is executed again, and the final result, shown in the bottom row of Fig. 8, presents the intended subject and action more clearly.

Table 1: System evaluation for each criterion.

Script #	Vis.	Fram.	Narr.	Cam Move.
1 (Initial)	Succ.	Succ.	Succ.	Succ.
2 (Initial)	Succ.	Succ.	Succ.	Succ.
3 (Initial)	Succ.	Fail	Succ.	Fail
3 (Replanned)	Succ.	Succ.	Succ.	Succ.

4.3 Discussions

We evaluate whether the system conveys the script-level intention through the generated shot plan and the executed video.

4.3.1 Categorical Evaluation

The evaluation considers whether the intended subject and camera movement are presented in accordance with the four cinematographic criteria used in the result evaluation stage: visibility, framing, narrative emphasis, and camera movement. A result is considered successful only when all four criteria are satisfied.

For Scripts 1 and 2, the generated shot plans correctly select the intended subject and the camera movement specified in the scripts. In Script 1, the planned `dolly_in` movement is visually reflected as the camera moves closer to the subject over time and produces a tighter shot, as shown in Fig. 6. In Script 2, the planned `track` movement keeps the intended subject visible within the frame, as shown in Fig. 7. Therefore, both results satisfy all four criteria, as shown in Table 1.

For Script 3, the initial execution fails to maintain the intended subject within the frame. As shown in the top row of Fig. 8, the subject moves out of the frame over time. The VLM-based result observation stage reports a framing failure because the intended subject is partially cut off on the right side of the frame, and a camera movement failure because the camera does not follow the intended subject as shown in Table 1. Based on this failure evidence, the system revises the shot plan and re-executes the shot. After replanning, the final execution maintains the intended subject and action more stably within the frame, as shown in the bottom row of Fig. 8. Table 1 shows that the result satisfies all four criteria.

4.3.2 Metric Evaluation

As shown in Table 2, we evaluate target visibility and framing stability in the executed results. Script 1 and Script 2 achieve high target-in-frame ratios in the initial shot plans, indicating that the target subjects remain visible within the frame. They also show low mean center errors, indicating that the target bounding box centers are maintained close to the frame center. In Script 3, the final shot plan executed after the replanning loop achieves a higher target-in-frame ratio than the initial shot plan. It also reduces the mean center error, indicating that the target bounding box center is maintained closer to the frame center. The mean center error was normalized by dividing the distance between the target bounding box center and the frame center by the maximum center-to-corner distance in a 1920×1080 frame.

Table 2: System evaluation using different metrics.

Script #	Target-in-frame ratio	Mean center error
1(Initial)	1.00	0.05
2(Initial)	1.00	0.27
3(Initial)	0.63	0.58
3(Replanned)	1.00	0.34

These results show that the proposed system generally maintains the target subject visible and close to the frame center in successful executions. In particular, the improvement in Script 3 after replanning indicates that the planning loop helps recover target visibility and framing stability when the initial execution fails.

5. Conclusion and Future Work

In this paper, we propose a VLM-based script-driven camera automation system for LED Wall-based virtual production. The proposed system connects natural-language script understanding to physical camera execution by generating a structured shot plan, visually grounding the target subject in the camera image, tracking the subject, and controlling a robotic camera system according to the planned camera movement. In addition, the system observes the executed video, evaluates whether the intended subject and action are visually understandable, and updates the shot plan when the result does not sufficiently reflect the script's intention.

The current system still has several limitations. First, the robustness of the proposed pipeline should be further evaluated across more diverse scripts, visual scenes, subject configurations, and camera movements. Second, the current demonstrations are limited to short shot-level scenarios. Future work will extend the system toward long-horizon multi-shot filming, where multiple subjects, actions, and camera movements must be planned and executed sequentially. In addition, we plan to further develop and evaluate the feedback process so that the system can more reliably verify execution results, identify failure cases, and revise the shot plan for subsequent camera execution.

Acknowledgement

This work was supported in part by the ITRC/IITP Program (IITP-2026-RS-2020-II201460) and the NRF (NRF-2022R1A2B5B03001385) in South Korea.

References

[1] M. Kavakli and C. Cremona, "The virtual production studio concept–an emerging game changer in filmmaking," in *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 2022, pp. 29–37.

[2] A. Azzarelli, N. Anantrasirichai, and D. R. Bull, "Intelligent cinematography: a review of ai research for cinematographic production," *Artificial Intelligence Review*, vol. 58, no. 4, p. 108, 2025.

[3] M. Christie, P. Olivier, and J.-M. Normand, "Camera control in computer graphics," in *Computer graphics forum*, vol. 27, no. 8. Wiley Online Library, 2008, pp. 2197–2218.

[4] C. Lino and M. Christie, "Intuitive and efficient camera control with the toric space," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, pp. 1–12, 2015.

[5] H. Jiang, B. Wang, X. Wang, M. Christie, and B. Chen, "Example-driven virtual cinematography by learning camera behaviors." *ACM Trans. Graph.*, vol. 39, no. 4, p. 45, 2020.

[6] H. Jiang, M. Christie, X. Wang, L. Liu, B. Wang, and B. Chen, "Camera keyframing with style and control," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 6, pp. 1–13, 2021.

[7] X. Xiong, J. Feng, and B. Zhou, "Automatic view finding for drone photography based on image aesthetic evaluation," in *International Conference on Computer Graphics Theory and Applications*, vol. 2. SCITEPRESS, 2017, pp. 282–289.

[8] I. Mademlis, N. Nikolaidis, A. Tefas, I. Pitas, T. Wagner, and A. Messina, "Autonomous uav cinematography: A tutorial and a formalized shot-type taxonomy," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–33, 2019.

[9] Z. Xu, J. Wang, L. Wang, Z. Li, S. Shi, B. Hu, and M. Zhang, "Filmagent: Automating virtual film production through a multi-agent collaborative framework," in *SIGGRAPH Asia 2024 Technical Communications*. Association for Computing Machinery, 2024, pp. 1–4.

[10] H. Lin, A. Zala, J. Cho, and M. Bansal, "Videodirectorgpt: Consistent multi-scene video generation via llm-guided planning," *arXiv preprint arXiv:2309.15091*, 2023.

[11] Y. Li, H. Shi, B. Hu, L. Wang, J. Zhu, J. Xu, Z. Zhao, and M. Zhang, "Anim-director: A large multimodal model powered agent for controllable animation video generation," in *SIGGRAPH Asia 2024 Conference Papers*, 2024, pp. 1–11.

[12] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," *Advances in neural information processing systems*, vol. 36, pp. 34 892–34 916, 2023.

[13] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, and E. Chen, "A survey on multimodal large language models," *National Science Review*, vol. 11, no. 12, p. nwae403, 2024.

[14] J. V. Mascelli, *The five C's of cinematography*. Grafic Publications Hollywood, 1965, vol. 1.

[15] D. Bordwell and K. Thompson, *Film Art: An Introduction*. New York, NY: McGraw-Hill Education, 2020.

[16] N. Jacobs, M. Munro, and C. Broodryk, "Green screen: The actor's challenge," in *2013 DEFSA conference Design Cultures: Encultured Design*, 2013.

[17] F. Pires, R. Silva, and R. Raposo, "A survey on virtual production and the future of compositing technologies," *Avanca Cinema Journal*, vol. 21, no. 692-9, 2022.

[18] J. Swords and N. Willment, "The emergence of virtual production–a research agenda," *Convergence*, vol. 30, no. 5, pp. 1557–1574, 2024.

[19] D. Silva Jasaui, A. Martí-Testón, A. Muñoz, F. Moriniello, J. E. Solanes, and L. Gracia, "Virtual production: Real-time rendering pipelines for indie studios and the potential in different scenarios," *Applied Sciences*, vol. 14, no. 6, p. 2530, 2024.

[20] C. Cremona and M. Kavakli, "The evolution of the virtual production studio as a game changer in filmmaking," in *Creating digitally: Shifting boundaries: Arts and technologies—Contemporary applications and concepts*. Springer, 2023, pp. 403–429.

[21] D. Arijon, "Grammar of the film language," *Hastings House Publishers*, 1976.

[22] M. Gleicher and A. Witkin, "Through-the-lens camera control," in *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, 1992, pp. 331–340.

[23] D. Amerson and S. Kime, "Real-time cinematic camera control for interactive narratives," in *Proceedings of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*. Stanford, CA: AAAI Press, 2001, aAAI Technical Report SS-01-02.

[24] Q. Galvane, J. Fleureau, F. Tariolle, and P. Guillotel, "Automated cinematography with unmanned aerial vehicles," in *Proceedings of the Eurographics Workshop on Intelligent Cinematography and Editing*, 2016, pp. 23–30.

[25] P. Carr, M. Mistry, and I. Matthews, "Hybrid robotic/virtual pan-tilt-zoom cameras for autonomous event recording," in *Proceedings of the 21st ACM international conference on Multimedia*, 2013, pp. 193–202.

[26] O. Limoyo, J. Li, D. Rivkin, J. Kelly, and G. Dudek, "Photobot: Reference-guided interactive photography via natural language," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 2479–2486.

[27] J. Han, "대규모 시각 언어 모델을 이용한 가상 제작 기반 카메라 촬영 자동화," Master's thesis, Ewha Womans University, 2026.

[28] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[29] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 3645–3649.

[30] Q. Galvane, "Automatic cinematography and editing in virtual environments." Ph.D. dissertation, Université Grenoble Alpes, 2015.

[31] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "Vqa: Visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2425–2433.

〈저자소개〉

한 지 민

- 2022이화여자대학교 수학교육과 학사
- 2026 이화여자대학교 컴퓨터공학과 석사
- 관심분야: Virtual Production, Autonomous Cinematography, Computer Graphics
- https://orcid.org/0009-0008-4031-4502

김 영 준

- 1993 서울대학교 계산통계학과 학사
- 1996 서울대학교 계산통계학과 석사
- 2000 Perdue University, Computer Science 박사
- 2003 ~ 현재 이화여자대학교 컴퓨터공학과 교수
- 관심 분야: Computer Graphics, Intelligent Robotics, Virtual Reality, Haptics, Computer Games
- https://orcid.org/0000-0003-2159-4832