2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR) | 978-1-6654-9548-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/MIPR54900.2022.00025

Enabling Automatic Cinematography with Reinforcement Learning

Zixiao Yu[1], Chenyu Yu[2], Haohong Wang[3], Jian Ren[1]
[1]Department of ECE, Michigan State University, East Lansing, MI 48824-1226.
Email: {yuzixiao, renjian}@msu.edu
[2]Rensselaer Polytechnic Institute, 110 8th St, Troy, NY 12180
Email: charlesyu0618@gmail.com
[3]TCL Research American, 2025 Gateway Place, Suite 460, San Jose, CA 95110
Email: haohong.wang@tcl.com

***Abstract*—Rule-based automated cinematography has proven to be a very effective means of speeding up the filmmaking process and reducing costs by incorporating the rules of many camera shots into the optimization process. However, it is observed that many well-known directors developed their style of lens language and sometimes even violate the conventional rules. It is a very challenging task to exploit these directors' versatile styles into the automatic cinematography framework due to two main reasons: (i) it is not trivial to translate these styles into a library of rules; (ii) the data can be collected from existing films are very limited and the accuracy is insufficient for learning purposes. In this paper, we propose a novel unified Reinforcement learning-based Text to Animation (RT2A) framework that can apply reinforcement learning to automatic cinematography. In RT2A, the decisions by the director on the camera settings are recorded and can be utilized for the future auto cinematography agent training process. A well-designed reward functionality has been proposed that guides the algorithm to find the best policy and mimic the human director's decision-making process for the camera selection of each scene. The experimental results show that the proposed RT2A can effectively imitate the directors' usage of lens language patterns. Compared to the reference algorithm, RT2A can achieve a gain of up to 50% in camera placement acceptance rate and 80% in imitating the rhythm of camera switching.**

I. Introduction

Cinematography, the art of choosing camera shot types and angles in capturing the motion pictures, is an effective way to demonstrate its artistic charm in the film industry, and applying cinematography to content creation requires lots of training and knowledge. In the computer age, it becomes natural to explore the possibility of making robot directors and entrusting them with this challenging task, which is also known as automatic cinematography. Some successful efforts, such as [1], [2], have shown that this goal can be partially achieved by translating the rules from the standard cinematography guidelines into a number of cost functions and applying them during an optimization process. Such aesthetic rules-based robot directors can provide valuable references for an entry-level artist without much film experience. In [3] the aesthetic model is further extended to a hybrid model that considers the fidelity of the output video by comparing it with the textual content in the original script. It is reported in [4] that mere use of these rules for expression in the film is far from satisfactory. In the practical film industry, the human directors are experts in twisting these standard rules in cinematography and developing their own unique lens language to express human creativity and imagination [5]. Therefore, although creating robot directors based on common lens language and summarized rules is an interesting idea, it is practically very difficult to meet the expectation of film artists.

As a relevant exploration effort, [6] used a neural network to extract lens language from the existing film and imitated the human director's behavior in filmmaking. However, this effort suffered from an inevitable problem, that is the experimental data is inaccurate and incomplete because the data were estimated from two-dimensional frames with information loss in another dimension. Therefore, such an approach can only be applied in limited scenarios and is unable to comprehensively imitate the director's lens language. Another work [7] demonstrated that such a behavioral imitation problem with a limited amount of training data can be achievable by reinforcement learning (RL) techniques [8] with reward generated from human feedback. In order to develop a robot director that can truly learn human lens language and use them in various scenarios, it needs not only the precise data collection during cinematography but also the capacity to continuously improve the model based on external feedback. By using the current auto cinematography algorithms [3], [9], [10], it is possible to build a framework for directors to use so that their chosen camera settings can be recorded as training data for reinforcement learning algorithms to learn the lens language models.

This observation inspired us to extend our previous work [3] to the direction of using reinforcement learning because all the needed data for cinematography usage can be collected during the film production process without extra effort with our framework. We reused the structure of the filmmaking frame T2A proposed in [3], but used a reinforcement learning module for auto cinematography to replace the camera optimization process in T2A, thus the new framework is called RT2A. A director's feedback module is incorporated to correct the camera placements, and such feedback is valuable for the camera agent to learn and imitate the lens language of the

2770-4319/22/$31.00 ©2022 IEEE
DOI 10.1109/MIPR54900.2022.00025

target human director. In practical scenarios, the director's feedback can come from stored training data that is collected from real human directors during their routine filmmaking work using our tool [3]. With sufficient data and feedback, the RT2A can effectively learn to support such a robot director that has the potential to become the embodiment of a real director and produce animation with a similar lens language. To the best of our knowledge, this is the first work in the world that applies reinforcement learning to auto cinematography that can effectively imitate the human director's usage of lens language.

The rest of the paper is organized as follows: Section II overviews the related work in computational cinematography; Section III demonstrates the overall framework of RT2A, the problem formulation, and the solution; Section IV discusses the experimental results and the last section draws the conclusion about the work.

II. Related Work

The quality of lens language is a crucial factor that determines the quality of the final film. It requires acumen of artistic sense as well as great knowledge of cinematography when using shots going beyond the basic rules. Luring by the wish to use auto cinematography techniques to reduce the film production cost, there are a considerable amount of studies in this area, and they can be roughly categorized into two directions.

The first direction is the cinematography guideline rule-based approach. By defining multiple constraints according to various rules and formulating them into the corresponding loss functions, the optimal shot setup for the current situation can be calculated by minimizing the total loss. Different aesthetic constraints can be applied under separate scenarios, such as dialogue scene [11], cooking [12], and outdoor activities [13]. However, as discussed in [4], it is difficult to judge whether the use of lens language is sufficient under only aesthetic models. In [3], the fidelity model was introduced, which took the original script content into consideration in the optimization process to make sure that the generated video is fully aligned with the script. The advantage of the rule-based approach is that the results are perfectly consistent with the pre-defined constraints, thus guaranteeing that no rule-breaking will occur; however, it also severely limits the freedom and creativity of the artist.

The second direction is to learn the behavior model of these cameras directly from the existing movies [6]. The advantage of this approach is that it is a data-driven methodology instead of a rule-driven solution, therefore there is no need to create rules or constraints to guide the algorithm. By using reinforcement learning, such as deep Q-learning (DQN) [14], Trust Region Methods (TRPO), or policy optimization (PPO) [15], the solution can be found by providing sufficient training data. In literature, there are works in the field of drone photography controlling [7], [16], however, no work can be found in the literature using reinforcement learning in the auto cinematography for filmmaking, a possible reason is that to get sufficient precise data from existing movies like in the work of [6] is a non-trivial task.

III. Framework Design

The filmmaking process is demonstrated in the Fig.1 which contains four major modules, namely, *Action List Generation, Stage Performance, Auto Cinematography*, and *Video Generation*, and two additional modules (in orange color) to support the reinforcement learning workflow. The *Action List Generation* takes the textual script as input, analyzes with NLP techniques [17], and then generates the corresponding action list, which is a chronological list of action objects a_t and can be considered as a special format to represent the content of the original script. Each a_t contains the necessary information for the virtual characters to make the corresponding performance p_t in the *Stage Performance*. The entire script's story can be performed in the *Stage Performance* with a sequence of p_t that follows the order in the action list. At this phase, all the characters appearing in the scene are bound with multiple cameras that can be distinguished by the unique index. The frames captured by cameras also become a part of the p_t. The *Observation Extractor* takes the p_t to generate the corresponding observation o_t (details can be found in the section III.A) that is defined in the auto cinematography RL environment for the camera agent to calculate the current selected camera c_t^i (where i is the index of the camera). In case the initial training data is insufficient, the acceptance rate of c_t^i computed by the camera agent in the *Auto Cinematography* would be low, thus manual adjustment of c_t^i by the director is required to assure video quality. The modification process is accomplished in *Director Adjustment*. The revised c_t^i is then annotated as the ground truth camera gc_t and added together with o_t into the training data. The p_t and gc_t can be used by *Video Generation* to generate the current output video frame. With the growth of the number of desirable videos generated with this RT2A framework, the training data grows as well. With sufficient training data, *Auto Cinematography* will be able to properly train the camera agent and update its policy with the RL algorithm, thus the workload required for directors to adjust the camera placement would be significantly reduced.

In Fig.1, The camera agent training process of the auto cinematography module has been shown with red lines. An observation generator uses the training data to generate the single observation, o_t, at times t. The RT2A camera agent takes o_t as input to obtain the corresponding selected camera, c_t^i. The reward function calculates the reward, r_t, by comparing the c_t^i and the gc_t. The r_t is then used to update the RT2A camera agent policy and parameters by the RL algorithm.

A. Proposed Auto Cinematography

In the auto cinematography environment, the observation space contains all the information needed by the RT2A camera agent to select the optimal camera setting based on the current policy. The following aspects are included in the observation space.

Character Visibility Character visibility is determined by two factors: (i) the size of the character in the frame compared to the total frame size. (ii) the weights for various groups of characters and camera combinations during the calculation because multiple cameras are bound to different

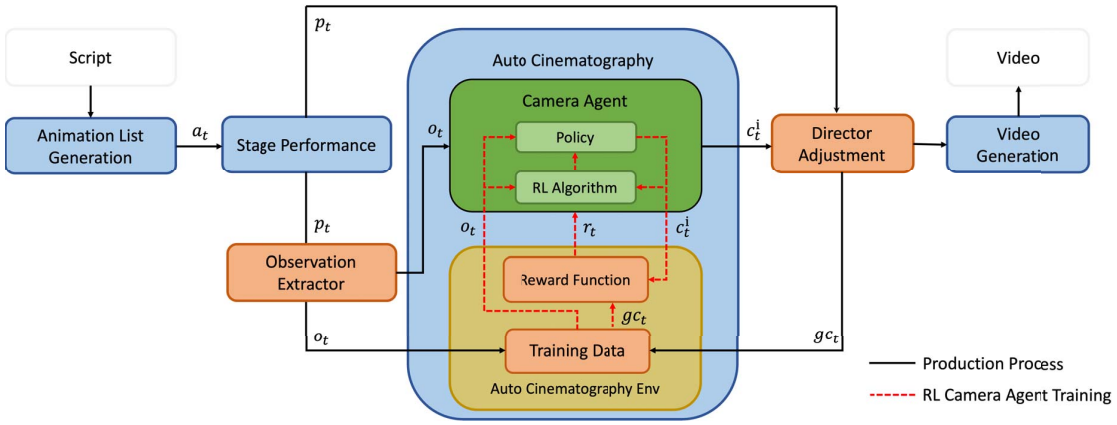

Fig. 1: Overview of our RT2A animation production framework

characters, which reflects the extent to which obstacles obscure the characters from the current camera view.

Camera Configuration When switching cameras, the configuration of the previous camera may also have an impact on the selection of the next camera, such as the shot-reverse shot commonly used in the dialogue scene. In such cases, the configuration of the previous camera will be included in the observations.

Left to Right Order (LRO) LRO is used to show the positional relationship of multiple characters in the shot frame, and the inclusion of this data has a very significant impact on the 180-degree rule, which is one of the most significant rules in cinematography. It is important to be aware that LROs at t taken by different cameras may not be the same.

Action Type Some character actions, such as "idle", do not affect the shot selection, while others may have a preference for shot selection. For example, the facial action of "anger" is more inclined to be expressed by a close shot. In our experiments, we divide the movements into categories of facial, upper limb, lower limb, whole body, and standby.

Action Start Time and Duration The start time of every character's action is crucial, and generally, the transition of the shot c_i is at the beginning of a certain action. On the other hand, the length of the action is also very crucial. In many cases, the action with a long performance time (e.g., more than 10 seconds) requires a combination of different shots to take it.

Dialogue Start Time and Duration The character's talking action in the dialogue scene is very special. During long conversations, the camera angle is switched between the interlocutors, and often an over-the-shoulder shot is used in such a scenario.

In our auto cinematography environment, the only action in the action space is camera index selection. There are various default cameras to shoot the characters from different distances and angles. The default cameras cover most of the basic camera settings in the cinematography guideline and each camera has been given a unique index for the agent to select it. the default camera setting can be described with 3 parameters:

1) $d(c)$: distance between the camera and the shooting character, which may include extreme close shot (ECU), close shot (CU), median shot (MS), full body shot (FS) and long shot (LS). By quantifying these distances to numeric representation from 0 to 4, we are able to calculate the differences between them.
2) $h(c)$: pan (horizontal) angle: form $0°$ to $360°$.
3) $p(c)$: pitch of the camera: form $15°$ to $-15°$.

Determining a meaningful reward function is very crucial for the RL algorithm. The reward function is construed only base on attributes of c_t^i and gc_t, where we need to consider the distinction between their settings in detail. The more similar the camera settings the higher the r_t.

The reward function for distance is defined as:

$$r_t^d = \begin{cases} 1 & \text{if } d(c_t^i) = d(gc_t) \\ 1 - \frac{|d(c_t^i) - d(gc_t)|}{4} & \text{otherwise.} \end{cases} \tag{1}$$

Similarly the reward function for pan and pitch of the camera are defined as:

$$r_t^h = \begin{cases} 1 & \text{if } h(c_t^i) = h(gc_t) \\ 1 - \frac{|h(c_t^i) - h(gc_t)|}{30} & \text{otherwise.} \end{cases} \tag{2}$$

$$r_t^p = \begin{cases} 1 & \text{if } p(c_t^i) = p(gc_t) \\ 1 - \frac{|p(c_t^i) - p(gc_t)|}{180} & \text{otherwise.} \end{cases} \tag{3}$$

When difference between the agent selected camera and the ground truth is less then a predefined threshold, δ, an extra reward is added to further boost the learning process. The larger the δ, the larger the deviation between c_t^i and gc_t that can be tolerated.

$$r_t^\delta = \begin{cases} 1 & \text{if } \frac{|d(c_t^i) - d(gc_t)|}{4} + \frac{|h(c_t^i) - h(gc_t)|}{30} + \frac{|p(c_t^i) - p(gc_t)|}{180} < \delta \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

The overall reward function is defined as the sum of all the previous rewards.

$$r_t = r_t^d + r_t^h + r_t^p + r_t^\delta. \tag{5}$$

B. Reinforcement Learning Algorithm

An agent starts with observation o_0 and selects the next camera index according to the strategy, *policy* $\pi(o)$, that agent

uses in pursuit of goals which in most cases is to maximize the reward, R.

The $\mathcal{R}$ for an episode with T steps is defined as:

$$\mathcal{R} = \sum_{0}^{T} \gamma^{t-1} r_t \qquad t = 0, 1, \cdots, T, \tag{6}$$

where r_t is immediate rewards at t and γ is a discount factor that defines the importance of r_t versus future rewards, where typically $0 \leq \gamma \leq 1$. The higher the γ value the more important the future rewards.

The aim of the RL algorithm is to find the optimal π^* which can maximize R.

$$\pi^* = \arg\max \mathbb{E}(R|\pi). \tag{7}$$

This process is accomplished by iteratively updating the parameter of the *policy*, π_θ, according to the loss function $\mathcal{L}(\centerdot)$ that measures the error between the reward estimation calculated by the current policy, π_{current}, and the previous policy, π_{old}. There are several methods to address the problem raised in Eq. 7, and we use PPO, a variant of an Advantage Actor-Critic (A2C) [18], which combines policy-based and value-based RL algorithms together. The actor neural network model takes state (or observation) and outputs the action according to the $\pi()$, and the critic neural network model maps each state to its corresponding quality of value the state (i.e., the expected future cumulative discounted return). The advantage $\hat{\mathcal{A}}$ (or discounted return) is used to indicate how good a camera selection is compared to the average camera selection for a specific observation. The $\hat{\mathcal{A}}$ at time t is defined as following:

$$\begin{aligned}\hat{\mathcal{A}} = & -V(o_t) \\ & + r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V(o_t),\end{aligned} \tag{8}$$

where the V is the learned state-value function and rt_t is the parameter changing ratio between the π_{current} and π_{old} at step t.

During the training process, PPO needs to update the parameters of both actor and critic neural networks by back-propagation according to two different loss function. Every update for π is designed to maximize the overall return (i.e. $\max[\mathbb{E}_t(rt_t\hat{\mathcal{A}})]$). However, changing the π_θ needs to be avoided in a single update. Thus, the $\mathcal{L}_{\mathsf{actor}}$ is defined as:

$$\mathcal{L}_{\mathsf{actor}}(\theta) = \min(rt(\theta)\hat{\mathcal{A}}_t,\ \mathsf{clip}(rt(\theta), 1-\epsilon, 1+\epsilon)\hat{\mathcal{A}}_t). \tag{9}$$

The $\mathsf{clip}(\centerdot)$ modifies the surrogate objective by clipping the probability ratio, which removes the incentive for moving rt outside of the interval $[1-\epsilon, 1+\epsilon]$. Regardless of the value of the positive feedback gained according to c_t^i, the PPO will only update the policy based on this result within this limited range. Thus, it is able to incrementally update π_θ with an appropriate value. However, the penalty based on the negative reward has no limitation.

The critic loss function $\mathcal{L}_{\mathsf{critic}}$ is defined as:

$$\mathcal{L}_{\mathsf{critic}} = \frac{1}{2}\hat{\mathcal{A}}^2. \tag{10}$$

IV. Experimental Results

In this section, the experiment set up and the performance details of our purposed RL auto cinematography method are demonstrated. We develop the auto cinematography environment and implement the RT2A camera agent using OpenAI Gym [19]. It took around 38 hours to finish the whole training process based on the NVIDIA GTX 2080 TI GPU. To evaluate the advantage of our proposed RL auto cinematography model, we compare RT2A with the reference methods: Aesthetic-based [10], and Aesthetic + Fidelity-Based [3]. The performance is evaluated from two aspects: (i) we compare the camera placement generated by the RT2A camera agent and by the reference algorithms; (ii) we compare the visual quality of videos produced by RT2A and reference algorithms.

Difference in Camera placement In the environment of the experiment, the cameras generated by different methods are sampled every second. By comparing the physical distance between the camera placement of the algorithm with the ground truth (placement manually by directors), and defining several acceptance thresholds (that is, the placement is accepted if the physical distance is less than the threshold), the performance of the algorithms can be measured by the percentage of the cameras being accepted, which is called acceptance rate. The calculation of the physical distance is based on the equations 1, 2, and 3. As the results are shown in Fig. 3, the proposed algorithm constantly outperforms the reference algorithm, and the gain in acceptance rate is up to around 50%. The visual comparison of the generated frames from various algorithms as shown in Fig. 2 also indicates that the proposed algorithm has a much higher similarity to the camera selected by directors than the reference methods. This evidence proves that the RT2A can effectively imitate the behaviors of the director's camera usage model after training.

Number of shot A complete shot is a continuous view through a single camera without interruption. The number of shots (i.e., average shot duration) used in a single scene is also an important indicator to reflect the shooting style of the director. Table. I shows the number of shots of our proposed auto cinematography approach compared to the reference methods in a number of different scenes selected from the test data set. The results indicate that the number of shots used in each of the test scenes by our proposed method is much closer to the ground truth than the reference methods. Visual results also support the conclusion when the frames generated from the aesthetic model [10] are compared to the frames generated from RT2A. It is important to realize that although it may be possible for the rule-based approach like the aesthetic model algorithm to mimic the director by setting a number of rules to optimize when compared to the proposed RT2A algorithm, the challenges in adjusting the weights for various parameters and cost functions are much more complicated. In the following text, we will demonstrate how the reference model can achieve similar visual results by adding the corresponding cost functions or adjusting weights, hence it convincingly proved the advantages of the proposed RT2A algorithm using a data-driven methodology, instead of manually crafting many rules.

Fig. 2: Sample frames for the a scene maintaining chronological order from left to right. As shown, our approach better imitates the lens language used by the animators.

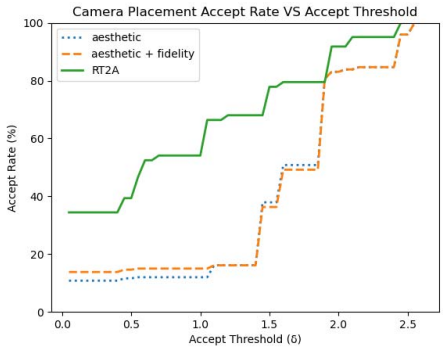

Fig. 3: Camera placement acceptance rate with different acceptance threshold

Particular shot selection As demonstrated in [20], *" the single long shot showing initial spatial relations became one portion of the scene, usually coming at the beginning. Its function then became specifically to establish a whole space which was then cut into segments or juxtaposed with long shots of other spaces."*, long shots as the establishing shots are used to set a particular tone and mood for what the audience is about to see. As shown in Fig. 4, compared to the reference approach, RT2A camera agent learns this lens language better and uses the establishing shot at the beginning of the scene. It is possible if the reference algorithm desires to achieve a similar outcome, that is, a new cost function determining whether the t represents the first few frames needs to be included in the optimization framework, and the characters captured by the LS shot will be required to occupy the least

TABLE I: The difference in the number of shot of our proposed and reference methods compared with the director's selected camera placement. The results show the number of shots (also the differences in percentage compared with the selected shot by director) used by different approaches in a single scene.

Script	Aesthetic	Aesthetic + Fidelity	RT2A	Director
1	32 (33%)	35 (46%)	27 (13%)	24
2	50 (150%)	43 (115%)	28 (40%)	20
3	45 (95%)	40 (74%)	31 (40%)	23
4	37 (146%)	39 (160%)	21 (35%)	15
5	15 (150%)	13 (116%)	8 (33%)	6

space in the frame compared to other shot types, which means that the weight of character visibility in cost function needs to be minimized.

Fig. 4: Captured establishing shot frames from the camera generated by aesthetic model (left) and the camera generated by RT2A (right)

The Over the Shoulder Shot is widely used in the dialogue for the audience to understand the relationship between the characters and to convey a dramatic tension to the viewers [21]. It is sometimes necessary to use it to assemble the

reverse shot in a dialogue scene. As shown in the Fig. 5, RT2A camera agent learns this lens language successfully and uses Over the Shoulder Shot in dialogue scenarios. If the reference model wants to achieve a similar result, a new cost function needs to be added to determine whether there is enough duration for the current "speak" action to switch between shots. In addition, the weight of the camera placement used to take the over-the-shoulder shot needs to be modified during the optimization process, to lose the requirement of capturing actions from the back of the character.

Fig. 5: Captured dialogue frames from the camera generated by aesthetic model (left) and the camera generated by RT2A (right)

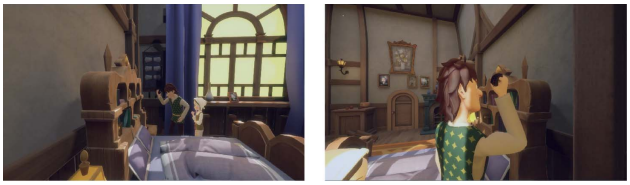

Fig. 6: Captured single action frames from the camera generated by aesthetic model (left) and the camera generated by RT2A (right)

Sometimes actions of particular characters need to be shot from a close distance and appropriate angle for the audience to better understand the content. As shown in the Fig. 6, the frames captured by the reference model selected camera do not properly illustrate the actions of "inspect the item" very well. In order to let the aesthetic model achieve a similar result, it is required to manually modify the weights of different camera configurations for some particular actions according to the interpretation of the story.

V. Conclusions and Future Works

The creativity and imagination demonstrated by the use of lens language in filmmaking requires tremendous professional knowledge and talent. It would benefit entry-level artist if the automatic cinematography algorithm can learn from the professional directors and thus mimic the camera languages used in successful films. However, a major obstacle to is that there is insufficient accurate training data in this area. In this paper, we presented RT2A framework, which produces accurate data for training the auto cinematography system with the support of directors. By learning the lens language from these directors, RT2A is able to select the right distance and camera angle for the automatic cinematography process. In the future, we will explore the possibility of personalizing auto cinematography by connecting the audiences viewing preferences with the filmmaking styles selection. In this way, the future films can be made in various styles and the audiences can get the one that fit into his/her personal taste most.

References

[1] Q. Galvane, *Automatic Cinematography and Editing in Virtual Environments.* PhD thesis, Université Grenoble Alpes (ComUE), 2015.

[2] A. Louarn, M. Christie, and F. Lamarche, "Automated staging for virtual cinematography," in *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, pp. 1–10, 2018.

[3] Z. Yu, E. Guo, H. Wang, and J. Ren, "Bridging script and animation utilizing — a new automatic cinematography model," in *Submitted to MIPR 2022*.

[4] M. Radut, M. Evans, K. To, T. Nooney, and G. Phillipson, "How good is good enough? the challenge of evaluating subjective quality of ai-edited video coverage of live events.," in *WICED@ EG/EuroVis*, pp. 17–24, 2020.

[5] G. Mercado, *The filmmaker's eye: Learning (and breaking) the rules of cinematic composition*. Routledge, 2013.

[6] H. Jiang, B. Wang, X. Wang, M. Christie, and B. Chen, "Example-driven virtual cinematography by learning camera behaviors," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 45–1, 2020.

[7] M. Gschwindt, E. Camci, R. Bonatti, W. Wang, E. Kayacan, and S. Scherer, "Can a robot become a movie director? learning artistic principles for aerial cinematography," *arXiv preprint arXiv:1904.02579*, 2019.

[8] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017.

[9] H. Subramonyam, W. Li, E. Adar, and M. Dontcheva, "Taketoons: Script-driven performance animation," in *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, pp. 663–674, 2018.

[10] L.Sun and H.Wang, "Director-hint based auto-cinematography," in *US Patent 11,120,638*, 2021.

[11] M. Leake, A. Davis, A. Truong, and M. Agrawala, "Computational video editing for dialogue-driven scenes.," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 130–1, 2017.

[12] A. Truong, F. Berthouzoz, W. Li, and M. Agrawala, "Quickcut: An interactive tool for editing narrated video," in *Proc. 29th Annual Symposium on User Interface Software and Technology*, pp. 497–507, 2016.

[13] I. Arev, H. S. Park, Y. Sheikh, J. Hodgins, and A. Shamir, "Automatic editing of footage from multiple social cameras," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–11, 2014.

[14] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, 2016.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[16] N. Passalis and A. Tefas, "Deep reinforcement learning for controlling frontal person close-up shooting," *Neurocomputing*, vol. 335, pp. 37–47, 2019.

[17] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. Peters, M. Schmitz, and L. S. Zettlemoyer, "Allennlp: A deep semantic natural language processing platform," 2017.

[18] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, pp. 1928–1937, PMLR, 2016.

[19] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[20] D. Bordwell, J. Staiger, and K. Thompson, *The classical Hollywood cinema: Film style & mode of production to 1960*. Columbia University Press, 1985.

[21] M. Svanera, S. Benini, N. Adami, R. Leonardi, and A. B. Kovács, "Over-the-shoulder shot detection in art films," in *2015 13th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pp. 1–6, IEEE, 2015.