Game Engine Based Data Augmentation with In-game Customization and Modeling for Malicious Behaviors Identification in Nuclear Security

Xingyu Song
The University of Tokyo

Zhan Li
The University of Tokyo

Shi Chen
The University of Tokyo

Kazuyuki Demachi
The University of Tokyo

ABSTRACT

Nuclear security is potentially seriously compromised by external and internal threats to nuclear facilities, such as stand-off attack and intrusion. Detection is the bottleneck of the Physical Protection System in nuclear security because delay and response will not work if detection fails. However, commonly employed detection systems rely on manual monitoring and extensive sensor deployment, which is time-consuming and costly. To this end, we introduce deep learning for vision-based automatic malicious behaviors identification, which benefits from its automaticity, generalizability, and diversity. The performance of deep learning models is highly dependent on training datasets. However, existing benchmark datasets are limited by small quantity and variety of scenes, objects, actions, and characters related to nuclear security. Besides, creating datasets from real-world filming is environmentally restrictive. Therefore, we propose to carry out data augmentation with game engine. First, we customize the buildings and interiors in the adopted game engine with a 3D map editor to simulate the environment of nuclear facilities. As objects involving malicious behaviors are not included in the resource library, we create various types of entities using modeling software with customized colors and textures. We further enhanced the conditions faced by simulated nuclear facilities by customizing environmental settings like time, weather, and season. The in-game simulated nuclear facilities allow characters with various clothes and appearances to perform multiple complex actions. All these customizations assure the diversity and quantity of datasets. Finally, we invoke the in-game camera in arbitrary distance of panoramic view, which ensures efficient and automated dataset generation. To validate the data augmentation approach, we feed the generated dataset to action recognition models for malicious behaviors identification against nuclear facilities. The experimental results demonstrate that the game engine augmented dataset allows an improved performance of malicious behaviors identification and helps get a better understanding in behavior identification.

INTRODUCTION

Nuclear security refers to the measure and practices designed to prevent the theft, diversion,

sabotage, or unauthorized access of nuclear material, facilities, and technology. The aim of nuclear security is to prevent nuclear and radiological terrorism and to ensure that nuclear materials and facilitates are used only for peaceful purposes. Nuclear security involves a range of activities, including physical protection, personal security, material control and accounting, transportation security, border security, and information security.

Physical protection is a key component of nuclear security, which involves the safeguarding of nuclear facilities, nuclear material, and radioactive sources from unauthorized access, theft, sabotage, or other malicious behaviors. There are mainly four stages in the Physical Protection System (PPS): deterrence, aiming to discourage potential intruders or attackers from attempting to breach the security measures in place, like fences and security personnel; detection, which helps to identify and respond to potential security threats, using surveillance cameras and alarm systems; delay, which helps to buy some time for response forces to arrive and respond to potential security threat, like security patrols and security checkpoints; response, which involves tacking action to address a security threat and prevent or minimize the consequences of an incident, like trained response forces and panic alarms. Detection is a critical part of PPS as it is the first line of defense against potential security threats. Without effective detection measures, security personnel would not to be able to identify and respond to potential threats in a timely and effective manner.

However, the existing methods of detection in nuclear security mainly contain only in two approaches, manual monitoring, and extensive sensor deployment. Manual monitoring involves security personnel visually inspecting and monitoring areas for potential security threats, which may suffer from the human error caused by fatigue or distraction, limited coverage, highly cost, delayed response, and inconsistency (different security personnel may have different levels of judgments). Similarly, extensive sensors [1] like radiation detectors and X-ray and gamma-ray imaging systems can also have some potential disadvantages for the detection in nuclear security. The most serious problems can be the highly cost, due to the continuing maintenance requirements of sensors to ensure they are functioning correctly, and the large amount of sensors under engaging, not to mention the vulnerability to destroying and hacking.

Therefore, we proposed to adopt the automatic malicious behaviors identification system using vision-based deep learning models. We have used object detection models like Yolox [2], to detect the malicious objects like RPG-launcher or wire-cutter; human pose estimation models like HRNet [3] to obtain the human skeleton keypoints; and action recognition models, 2D-CNN pipline models like TSN [4], TSM [5], and 3D-CNN models like I3D [6] and I3D Non-Local [7]. However, existing benchmark datasets are limited by small quantity and variety of scenes, objects, actions, and characters related to nuclear security, which really limiting the performance of the models. Besides, creating datasets from real-world filming is environmentally restrictive.

In nowadays, with the development of game industry, video games become more and more photorealistic. In addition, the flexibility of game modification increased as well. Accordingly, many games have been adopted to benefit the computer vision research [8, 9]. Specifically, an

action-adventure game with high customizability and scalability, called Grand Theft Auto V (GTAV), has been used in collecting datasets for deep learning model [10, 11]. GTAV supports multidimensional modification including game scenes, characters, animations, and scripting. Thus, GTAV provides a suitable platform for producing high-diversity data for computer vision.

Accordingly, we propose to carry out data augmentation with game engine. Game engine is a software framework designed to faciliate the creation and development of video games. It typically provides developers with a set of tools and functionalities that they can use to create games without having to build everything from zero. With the use of game engine, scenarios of malicious behaviors in nuclear security can be simulated with high customizability.

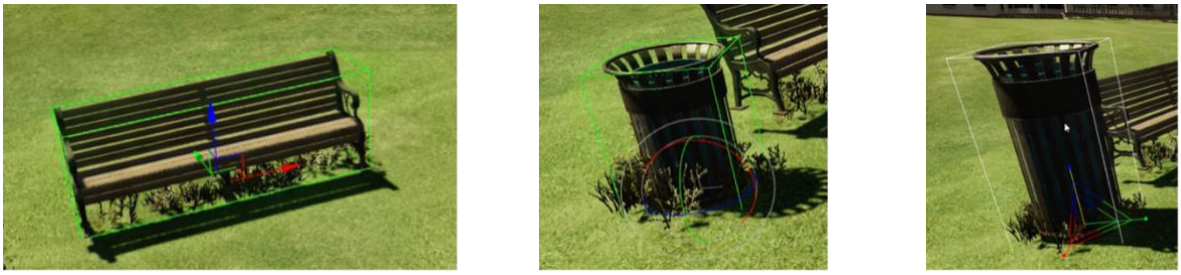
In this paper, we first customize the buildings and interiors in the adopted game engine of GTAV with a 3D map editer called CodeWorker [12] to simulate the environment of nuclear facilities. Secondly, with the objects involving malicious behaviors but are not included in the resourse library, we use a 3D modeling software called 3Ds Max [13] to create various entities with customized colors and textures. Third, we further enhanced the conditions faced by simulated nuclear facilities by customizing environmental settings like time, weather, and season. Fourth, the in-game simulated nuclear facilities allow characters with various clothes and appearances to perform multiple complex actions. Finally, we also propose the Random-Camera-Moving (RCM) algorithm to ensure the uncertainty of camera in arbitrary distance of panoramic view. All these customizations assure the diversity and quantity of datasets. The experimental results demonstrate that the game engine augmented dataset allows a rewarding performance of malicious behaviors identification.

METHOD

We use the game engine of GTAV for data augmentation by the modification of the game. All the modifications are accomplished in a FiveM [14] server we set up. FiveM is a modification platform for GTAV enabling for player to play multiplayers on customized dedicated server. The modifications on FiveM server we have conducted contain five parts, map editing, entity modeling, environmental customization, character customization, and camera customization.

Map Editing

Map editing is used for simulating actions presented in different kinds of locations. We use the CodeWorker [12] to edit the map at entity level, as shown in Figure 1, and editing the properties of the structures consisting of entities, which is shown in Figure 2. Additionally, we use the CodeWorker to construct a virtual nuclear power plant (NPP) equipped with physical protection, the overview of NPP is shown in Figure 3.



(a) Moving entity (b) Rotating entity (c) Changing the scale

Figure 1: Editing map in CodeWorker at entity level

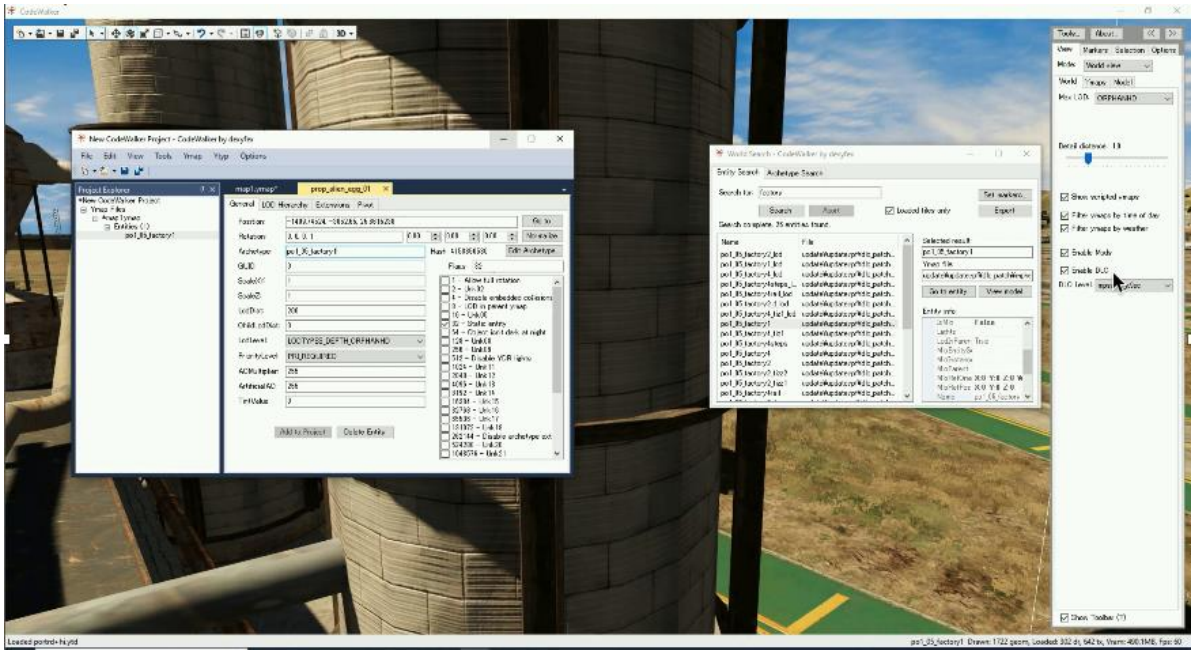


Figure 2: Editing properties of the structure in CodeWorker

Entity Modeling

Entity modeling is used to add objects which are not included in the resource library of CodeWorker. We first use the 3D modeling software 3Ds Max to model the malicious objects, as shown in Figure 4. After importing the customized model into the resource library of CodeWorker using OpenIV, which is a popular modification tool for GTAV, we edit the customized object in the CodeWorker, as shown in Figure 5.

Figure 3: The overview of the virtual nuclear power plant constructed by CodeWorker

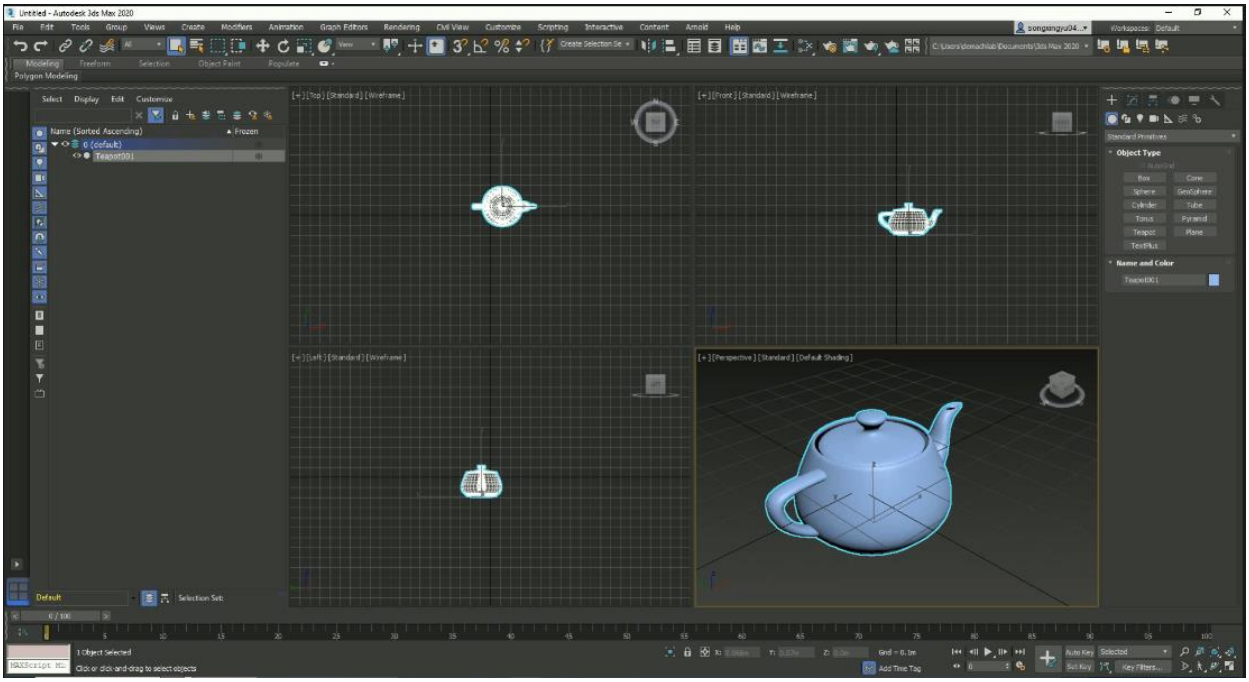


Figure 4: Modeling object in 3Ds Max

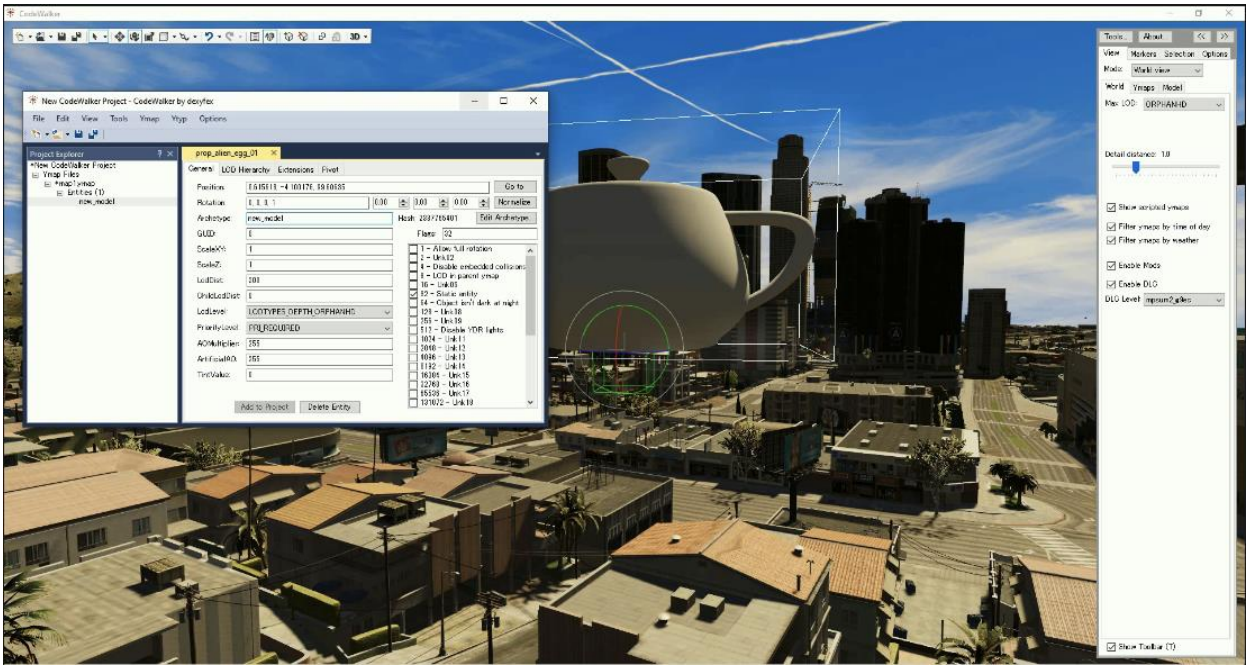


Figure 5: Editing customized object in CodeWorker

Environmental Customization

Environmental customization is used to simulate actions presented in different environmental situations. The environmental customization including the adjustment of the weather, and the time of the day. In this part, we adopt FiveM scripting to accomplish random in-game weather change, and clock time change. We involve 14 types of weather change (shown in Table 1) and 4 different time changes, corresponding to 4 specific periods of the day (shown in Table 2). With the customization of weather and clock time, different natural condition in game, including illumination, shadow, will be presented.

Character Customization

Character customization is used to simulate the actions presented by different characters. GTAV includes over 827 alternative build-in pedestrian [14] to simulate people with different age, gender, and occupation in the real world, the samples of some of the build-in pedestrians are shown in Figure 6.

Table 1: Optional weather and examples

Weather	Example	Weather

BLIZZARD

NEUTRAL

CLEAR

OVERCAST

CLEARING

RAIN

CLOUDS

SMOG

EXTRASUNNY

SNOW

FOGGY

SNOWLIGHT

XMAS  THUNDER

Table 2: Optional times, periods, and examples

Time	Period	Example
5:30	Dawn	
12:00	Noon	
20:00	Evening	
23:00	Midnight	

Figure 6: Samples of pedestrian

Camera Customization

Camera customization allows collecting action videos data automatically by the automatic moving in-game camera in arbitrary distance of panoramic view. The position of camera can be set in any place in the three-dimensional vector space, controlling by the scripts. In addition, the orientation of the camera can be set in arbitrary direction, and even can be fixed to the character or object. The movement of camera is controlled by the Random-Camera-Moving (RCM) algorithm, which is an algorithm controls camera movement between the recording of a specific action in different perspectives. According to the RCM, the initial coordinate of camera is set to be the same coordinate of character, also defined as origin (point o in Figure 7). In each movement, the camera moves a random distance with a random angle in the horizontal plane, and a random distance in vertical axis. Thus, if the movement between two adjacent points Pos_i and Pos_{i+1} is expressed in vector as:

$$\overrightarrow{Pos_{i+1}} = \overrightarrow{Pos_i} + \Delta\overrightarrow{Pos} \tag{1}$$

Where $\overrightarrow{Pos_{i+1}}$ stands for the vector of camera in position i, $\Delta\overrightarrow{Pos}$ stands for the change of position vector. Then $\Delta\overrightarrow{Pos}$ can be expressed by:

$$\Delta\overrightarrow{Pos_{xy}} = \Delta\overrightarrow{Pos} - \Delta\overrightarrow{Pos} \cdot \overrightarrow{i_z} \tag{2}$$

$$\Delta magnitude_{xy} = ||\Delta\overrightarrow{Pos_{xy}}|| \tag{3}$$

$$\Delta\theta = arccos\frac{\Delta\overrightarrow{Pos_{xy}}\cdot\overrightarrow{\iota_x}}{||\Delta\overrightarrow{Pos_{xy}}||} \tag{4}$$

$$\Delta z = \left\|\Delta\overrightarrow{Pos}\cdot\overrightarrow{\iota_z}\right\| \tag{5}$$

Where $\Delta\overrightarrow{Pos_{xy}}$ is the projection of $\Delta\overrightarrow{Pos}$ on xOy plane, $\overrightarrow{\iota_z}$ and $\overrightarrow{\iota_x}$ is the unit vector along the z-axis and x-axis. The xOy plane is a two-dimensional plane in the Cartesian coordinate system, which is formed by two perpendicular number lines, the horizontal x-axis, and the vertical y-axis. $\Delta magnitude_{xy}$ is the magnitude of $\Delta\overrightarrow{Pos_{xy}}$, which stands for the move distance on the horizontal plane xOy. $\Delta\theta$ is the angle between $\Delta\overrightarrow{Pos_{xy}}$ and $\overrightarrow{\iota_x}$, which stands for the change of the angle of camera. Δz is the magnitude of the projection of $\Delta\overrightarrow{Pos}$ on z-axis, which stands for the move distance on z-axis.

In RCM, we use $\Delta magnitude_{xy}$, $\Delta\theta$ and Δz to control the movement of camera between two adjacent positions:

$$\Delta magnitude_{xy} = U(a_m, b_m) \tag{6}$$

$$\Delta\theta = U(a_\theta, b_\theta) \tag{7}$$

$$\Delta z = U(a_z, b_z) \tag{8}$$

Where $U(a_m, b_m)$ is the random equation follows a uniform distribution between the lower limit a and upper limit b. a_m, b_m, a_θ, b_θ, a_z, b_z are the lower and upper limits of $\Delta magnitude_{xy}$, $\Delta\theta$ and Δz respectively. The whole schematic diagram of RCM is shown in Figure 7. The pink dots are the camera positions derived by RCM. The purple lines are the movement of camera between adjacent positions. o is the initial point of camera and the position of character as well. $\Delta magnitude_{xy}$ and $\Delta\theta$ indicate the random distance and angle change in horizontal plane xOy. Δz indicates the random height change in vertical z-axis.

With each change of camera position within the same action presenting location, $\Delta magnitude_{xy}$, $\Delta\theta$ and Δz will be randomized. With each change of action presenting location, the start point (origin o) of camera will be reset to the new coordinate of character in the new location.

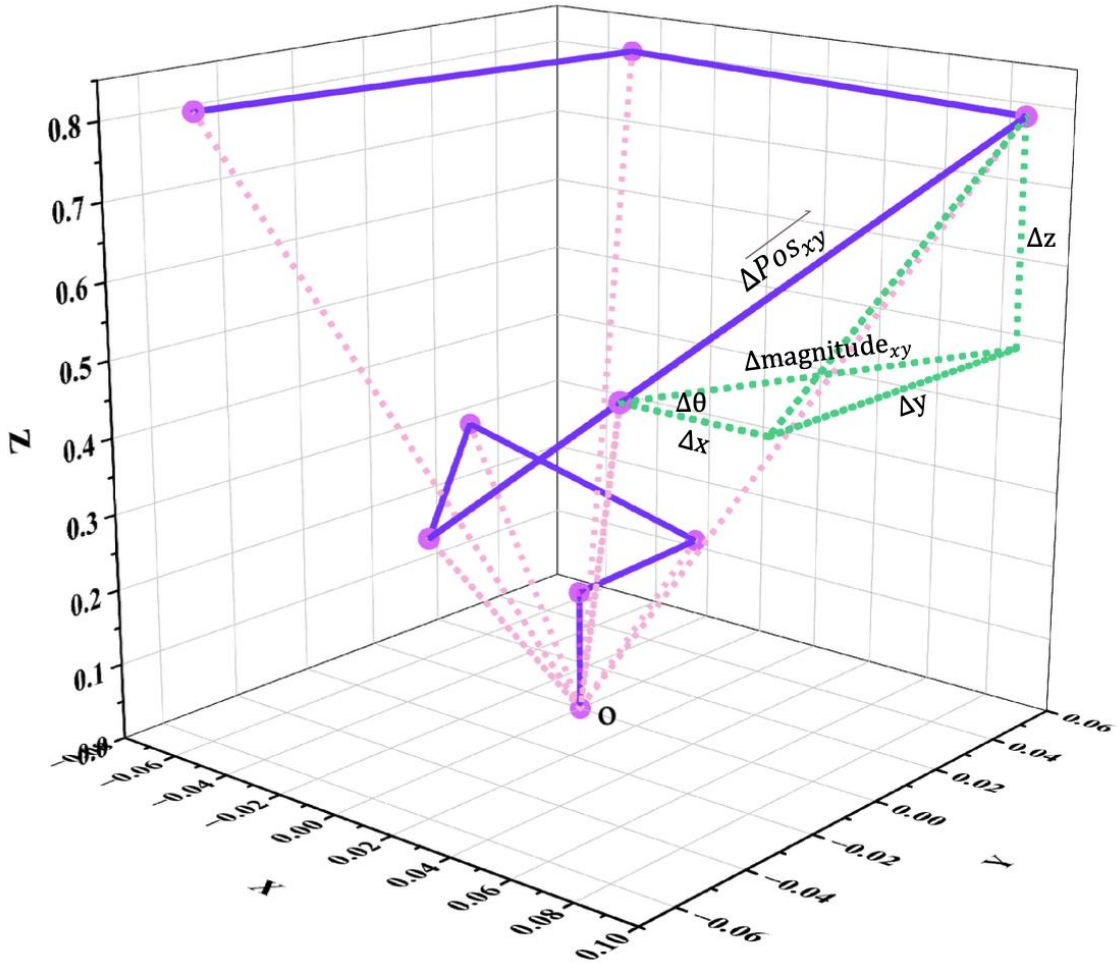


Figure 7: Schematic diagram of RCM

Table 3: The action list of sample dataset generated from the game engine, with different action type, video quantity and frame quantity

Action Type	Action	Video Number	Frame Number	Video Quantity	Frame Quantity
Normal	Celebrating	399	24767	Small	Small
Malicious	Punching	899	32701	Medium	Small
Normal	Walking	884	44706	Medium	Small
Malicious	Climbing Ladder	899	23142	Medium	Small
Normal	Clapping	899	24375	Medium	Small
Malicious	Smoking	899	152061	Medium	Large

EXPERIMENTS AND RESULTS

To evaluate the action video data generated by game engine, we first generate a sample dataset with 3 normal actions and 3 malicious actions with different video number, frame number, video quantity and frame quantity, which is shown in Table 3. Then, we establish experiments using

representative action recognition methods, the results of both mean accuracy and top-1 accuracy are shown in Table 4. We include the action recognition methods both in 2D-CNN and 3D-CNN. We also conduct the across-evaluation of each action with different distribution, the results are shown in Table 5.

We obtain three groups of actions, celebrating against punching, walking against climbing ladder, clapping against smoking. The purpose of the first group is to gain a better understanding of the upper body movement on distinguishing malicious behavior from normal behavior. The purpose of the second group is to gain a better understanding of the hole body movement. The third group is for the interaction of objects.

Table 4: Results of accuracy in different methods

Method	Type	Modality	Mean	Top-1
TSN [4]	2D-CNN	RGB	99.56	87.81
I3D [6]	3D-CNN	RGB	47.11	85.62
NL I3D [7]	3D-CNN	RGB	49.56	90.00
TSM [5]	2D-CNN	RGB	82.22	89.38
TIN [16]	2D-CNN	RGB	82.44	86.67
TPN [17]	2D-CNN	RGB	83.11	75.62
TANet [18]	2D-CNN	RGB	96.00	80.00

Table 5: Results of cross-evaluation in each class

Action	TSN [4]	I3D [6]	NL I3D [7]	TSM [5]	TIN [16]	TPN [17]	TANet [18]
Celebrating	98	30	6	66	70	56	94
Punching	100	40	72	88	84	78	98
Walking	100	40	72	88	84	78	98
Climbing Ladder	100	58	64	96	100	98	100
Clapping	98	30	8	58	66	78	86
Smoking	98	92	98	60	62	88	92

The results in Table 4 show that comparing to 3D-CNN methods (I3D, NL I3D), 2D-CNN methods (others) obtain the better performance in the dataset generated by game engine. Among all the methods, TSN and TANet obtain the best mean accuracy.

The results of cross-evaluation (Table 5) show that, by comparing with-object and non-object actions, there is no significant gap between accuracy of with-object action, like climbing ladder and smoking, and non-object actions, like punching and walking. But with-object actions show a better accuracy on 3D-CNN than non-object actions. As for the effect of video clip quantity, comparing between “celebrating” and others, it gets a lower average accuracy. According to the

results of small frame amount action, like “smoking” and others, it shows a better accuracy.

CONCLUSION

In this paper, we propose to carry out data augmentation with game engine. First, we customize the buildings and interiors in the adopted game engine with a 3D map editor to simulate the environment of nuclear facilities. As objects involving malicious behaviors are not included in the resource library, we create various types of entities using modeling software with customized colors and textures. We further enhanced the conditions faced by simulated nuclear facilities by customizing environmental settings like time, weather, and season. The in-game simulated nuclear facilities allow characters with various clothes and appearances to perform multiple complex actions. All these customizations assure the diversity and quantity of datasets. Finally, we invoke the in-game camera in arbitrary distance of panoramic view, which ensures efficient and automated dataset generation. To validate the data augmentation approach, we feed the generated dataset to action recognition models for malicious behaviors identification against nuclear facilities. The experimental results demonstrate that the game engine augmented dataset allows an improved performance of malicious behaviors identification, especially in the method of TSN and TANet, helps get a better understanding in behavior identification.

REFERENCES

[1] Kim, S. H., & Lim, S. C. (2018). Intelligent intrusion detection system featuring a virtual fence, active intruder detection, classification, tracking, and action recognition. Annals of Nuclear Energy, 112, 845-855.

[2] GE, Zheng, et al. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021.

[3] SUN, Ke, et al. Deep high-resolution representation learning for human pose estimation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019. p. 5693-5703.

[4] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[5] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[6] Joa˜o Carreira and Andrew Zisserman. Quovadis,action recognition? A new model and the kinetics dataset. *CoRR*, abs/1705.07750, 2017.

[7] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. *CVPR*, 2018.

[8] Victoria Bloom, Dimitrios Makris, and Vasileios Argyriou. G3d: A gaming action dataset and real time action recognition evaluation framework. In *2012 IEEE Computer society conference on*

computer vision and pattern recognition workshops, pages 7–12. IEEE, 2012.
[9] Alina Roitberg, David Schneider, Aulia Djamal, Constantin Seibold, Simon Reiß, and Rainer Stiefelhagen. Let's play for action: Recognizing activities of daily living by learning from life simulation video games. *CoRR*, abs/2107.05617, 2021.
[10] Sandy Ardianto and Hsueh-Ming Hang. Nctu-gtav360: A 360° action recognition video dataset. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–5, 2019.
[11] Zhe Cao, Hang Gao, Karttikeya Mangalam, Qi-Zhi Cai, Minh Vo, and Jitendra Malik. Long-term human motion pre- diction with scene context. *CoRR*, abs/2007.03672, 2020.
[12] Dexyfex. Codeworker. https://de.gta5- mods.com/tools/codewalker-gtav-interactive-3d-map. Accessed March 8, 2023.
[13] Autodesk. 3ds max 2023. https://www.autodesk.co.jp/products/3ds- max/overview?term=1-YEAR&tab=subscription. Accessed March 8, 2023.
[14] Fivem. Fivem docs. https://docs.fivem.net/docs/game-references/ped-models/#story. Accessed March 8, 2023.
[15] 2008-2022 OpenIV. Openiv. https://openiv.com. Accessed March 8, 2023.
[16] Hao Shao, Shengju Qian, and Yu Liu. Temporal interlacing network. AAAI, 2020.
[17] Ceyuan Yang, Yinghao Xu, Jianping Shi, Bo Dai, and Bolei Zhou. Temporal pyramid network for action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
[18] Zhaoyang Liu, Limin Wang, Wayne Wu, Chen Qian, and Tong Lu. Tam: Temporal adaptive module for video recognition. arXiv preprint arXiv:2005.06803, 2020.