Pengyuan Zhu[1*]

Virtual Reality Animation Interaction Design using Bayesian Physics-Informed Neural Network with Archimedes Optimization Algorithm Based Scene Modelling

Abstract: - Virtual Reality Animation Interaction Design is a dynamic field at the intersection of technology and creativity, where immersive experiences come to life through the fusion of animation and interactive design. There are some challenges involves in design of virtual Reality animation. The main challenge is animation image error involves in the design. To overcome this issue, present a Virtual Reality Animation Interaction Design Using Bayesian Physics-Informed Neural Network with Archimedes Optimization Algorithm Based scene modelling (VRAID-BPINN-AOA) is proposed. Initially, the animation images are collected from animation dataset. Then, the animation images are fed to pre-processing segment. In pre-processing segment, the noise of the image is removing layer by layer by utilizing adaptive-noise Augmented Kalman Filter (ANAKF). Then the pre-processed animation image is given for Feature extraction process. In Feature extraction, spatiotemporal features such as Object Position, velocity, Optical flow and Crowd Density are extracted by utilizing Multi-Level 2-D Quantum Wavelet Transform (ML2DQWT). Finally the extracted feature attributes are given to Bayesian physics-informed neural network (BPINN) for the prediction of error in the animation images. In general, BPINN does not express some adaption of optimization strategies for determining optimal parameters to promise accurate prediction of error. Therefore, Archimedes Optimization Algorithm (AOA) is proposed to optimize the parameter of BPINN .The proposed technique is implemented and efficacy of VRAID-BPINN-AOA technique is assessed by support of numerous performances such as Bit Error Rate, Design Rate, End to End Delay, Latency, and Transmission Rate. Proposed VRAID-BPINN-AOA method attains 28.56%, 26.67% and 25.67% lower Bit error rate and26.42%, 25.67% and 23.67% lower Design rate are compared with existing such as Virtual Interactive Animation Design with Restricted Boltzmann machine (VI-AD-RBM),Generative deep learning for visual animation in landscapes design using Generative adversarial network(GDL-VALD-GAN), and Hand interface using deep learning in immersive virtual reality using convolutional neural network(HID-LVIR-CNN)respectively.

Keywords: Animation Data, adaptive-noise Augmented Kalman Filter, Archimedes Optimization Algorithm, Bayesian physics-informed neural network, Multi-Level 2-D Quantum Wavelet Transform.

I. INTRODUCTION

Virtual Reality Animation Interaction Design Technology combines the immersive capabilities of virtual reality (VR) with the dynamic world of animation and the principles of interaction design, offering a transformative and engaging user experience [1]. This multidisciplinary approach integrates cutting-edge technologies to create virtual environments where users can interact with animated elements in a natural and intuitive manner. VR animation interaction design technology allows for the creation of compelling narratives, simulations, and interactive experiences that go beyond traditional forms of media. Users can navigate through digital spaces, manipulate virtual objects, and participate in rich storytelling experiences; all while being surrounded by visually stunning and dynamically animated content. This fusion of technologies not only opens new frontiers for entertainment but also holds tremendous potential in fields such as education, training, and healthcare, providing users with immersive and interactive learning experiences [2-4]. As advancements in these technologies continue the possibilities for creative expression and practical applications are boundless, ushering in a new era of interactive and animated virtual reality experiences [5]. Existing deep learning methods in Virtual Reality Animation Interaction Design Technology face several drawbacks. One significant challenge lies in their dependence on large and meticulously labelled datasets, making the training process resource-intensive [6, 7]. The computational complexity of deep learning models, especially those with numerous layers, poses a hurdle for real-time applications in virtual reality, where low-latency interactions are paramount. Additionally, the lack of interpretability in these models hinders understanding their decision-making processes, crucial for refining user experiences. Adaptability to diverse inputs, such as gestures and voice commands, remains a challenge, impacting the naturalness of user interactions. Ethical concerns arise regarding biases in training data, as models may unintentionally perpetuate and exacerbate discriminatory patterns in virtual reality experiences [8-10]. Over fitting and generalization issues further limit the robustness of these methods, making it imperative for on-going research in order to get around these restrictions and enhance the effectiveness and inclusivity of Virtual Reality Animation Interaction Design Technology [11, 12].

[1*]Lecturer, School of Electronic and Computer Engineering, Yangling Vocational & Technical College, Xianyang, Shaanxi, 712100, China.
[*]Corresponding author e-mail: 13289399831@163.com

Overcoming the drawbacks of existing deep learning methods in Virtual Reality Animation Interaction Design Technology requires a multi-faceted approach [13]. To address data dependency issues, efforts should be directed towards developing more efficient data augmentation techniques and exploring transfer learning methods that can leverage pre-existing knowledge from related domains [14]. Managing computational complexity necessitates on-going advancements in hardware capabilities and optimization techniques, with a focus on creating models that balance performance and efficiency. Improving interpretability requires research into explainable AI techniques, enabling designers to better understand and fine-tune the generated interactions [15-17]. Enhancing adaptability to diverse inputs involves incorporating multimodal learning approaches that can effectively handle various forms of user input. Ethical concerns demand rigorous evaluation of training datasets for biases and the development of mitigation strategies, alongside increased transparency and accountability in the development process. Finally, addressing over fitting and generalization issues requires the exploration of more robust architectures, regularization techniques, and diverse training scenarios to ensure models can adapt effectively to new and unforeseen situations. A collaborative effort across researchers, developers, and ethicists is essential to steer advancements in Virtual Reality Animation Interaction Design Technology towards more inclusive, reliable, and user-friendly solutions [18-20].

The task of predicting errors in virtual reality animation images presents several challenges in its problem statement. Several existing methods are concentrated on developing the predicting mechanism such as bit error rate, end to end delay, latency design rate and transmission rate. To overawe the problems, specific solutions necessity to be put onward to fix this issue, the existing technique doesn't exactly perform the prediction of error in animation images. Such are inspired to do us this investigation work.

This paper's proposed approach seeks to lower the error of the animation image by ML2DQWT from the dataset. The proposed study uses aBPINN to predict the error in animation image. The AOA algorithm is used to optimize the parameter of the BPINN

The following are the research's primary contributions:

- The research presents a method that combines Bayesian Physics-Informed Neural Network (BPINN) and AOA. For an optimization of Virtual Reality Animation Interaction Design.
- The objective of the proposed method is to decrease the bit error rate, decrease the design rate, decrease the end to end delay, decrease the latency, and increase the transmission rate.
- The BPINN algorithm is utilized to predict the error in animation images.
- The AOA algorithm is utilized to optimize the parameter of the BPINN.

Rest of this manuscript is organized as below: segment 2 investigates literature review, proposed approach is designated in segment 3, outcomes with discussion are established in segment 4, and conclusion is presented in segment 5.

II. LITERATURE SURVEY

Many studies have previously presented on the literature which depends on a Virtual Reality Animation Interaction Design using deep learning.A portion of the works were analysed here.

Liu [21] have presented Studying Design Systems for Virtual Interactive Animation with a Foundation in Deep Learning Using the problem of style recognition in animation capture data, the fundamentals of the Restricted Boltzmann Machine are applied to create a semi-supervised spatio-temporal feature model. When the top model's parameters remain unchanged and only the bottom model requires retraining, the model is considered well parallelized. This is especially true when only the bottom features, such as overfitting or underfitting, are unable to accurately represent the animation features. This is achieved by using a layer-by-layer training approach. It attains higher transmission rate and it attains higher bit error rate.

Ardhianto et al. [22] have developed Deep learning generative techniques for designing animated landscapes. Deep learning using Contrastive Language-Image and Vector Quantized Generative Adversarial Network Alternative landscape designs were created using text prompts and assembled into an animation using pre-training. According to our experiment, it is possible to generate a single frame in approximately 3.636 ± 0.089 s, which is much faster than the traditional animation creation method. Additionally, our approach produced a high-quality image with an inception score evaluation of 3.2904. It attains lower end to end delay and it attains higher latency.

Kang et al. [23] have developed Immersion virtual reality hand interface utilizing deep learning. A controller was used to map a real hand gesture to a virtual hand in a comprehensible structure, enabling a real-to-virtual direct hand interface. Furthermore, a gesture-to-action interface can communicate the gesture-to-action process in real-time even in the absence of the graphical user interface (GUI) found in many contemporary interactive applications. The training process for image classification is applied in this interface, which involves capturing a 2D representation of a 3D virtual hand gesture model with a CNN deep learning model. It attains lower latency and it attains higher design rate.

Gan et al. [24] have developed a study on behavior control and role modeling for virtual reality animation systems in the IoT. In this work, single role models are generated using 3D modeling technology based on depth images, which are then combined and rearranged to form 3D scenes, using the development of the 3D animation interactive system as an example. The hybrid intelligent collision detection algorithm makes it possible to control character behavior in the interactive system. improves collision detection's effectiveness and accuracy by combining the differential and quantum behavior particle swarm optimization algorithms. It attains lower design rate and it attains higher end to end delay.

Paier, et al. [25] have developed an interactive deep neural network-based facial animation. Utilizing the latest developments in deep learning, a novel hybrid animation framework offers an engine for interactive animation that can be operated through an easy-to-use visualisation for editing facial expressions. The authors present an automatic pipeline for creating training sequences with both consistent three-dimensional face models and dynamic textures. To train a low-dimensional latent space of expressions on the face that was used for interactive face animation, they train a variation auto encoder using this data. It attains lower latency and it attains higher end to end delay.

Zhu and Lee [26] have developed Using Flash animation style and deep learning, 3D shape feature extraction is done. Combining deep learning with conventional 3D shape feature extraction techniques can help to increase the precision of 3D shape data classification and retrieval tasks and alleviate the bottleneck of non-deep learning techniques, particularly for non-rigid 3D shapes. This paper presents an accurate feature extraction scheme for flash animation that requires few training samples. Tests reveal that this paper's scheme had a higher success rate for accurate feature extraction than the most advanced techniques. It attains higher transmission rate and it attains higher bit latency.

Fukuda et al. [27] have developed Techniques for rendering virtual reality that can be used to analyze landscapes, train deep learning models, and avoid VR sickness. In addition to standard immersive rendering, create an annotated image dataset with paired foreground-background and semantically relevant images for use in landscape analysis and deep learning neural network training. Furthermore, develop a rendering method for camera velocity that uses a personalized method of segmentation rendering to calculate the angular and linear speeds of the VR camera at every frame in the virtual reality environment. Based on the velocity value, a color is overlayed on the screen. It attains lower latency and it attains higher bit error rate.

III. PROPOSED METHODOLOGY

In this section, Virtual Reality Animation Interaction Design using Bayesian Physics-Informed Neural Network with Archimedes Optimization Algorithm Based scene modelling(BPINN) is proposed. Block diagram of VRAID-BPINN-AOA is presented in figure 1. The five steps in this process are: data acquisition, feature extraction, pre-processing, classification, and optimization. Accordingly, detailed description of all step given as below,

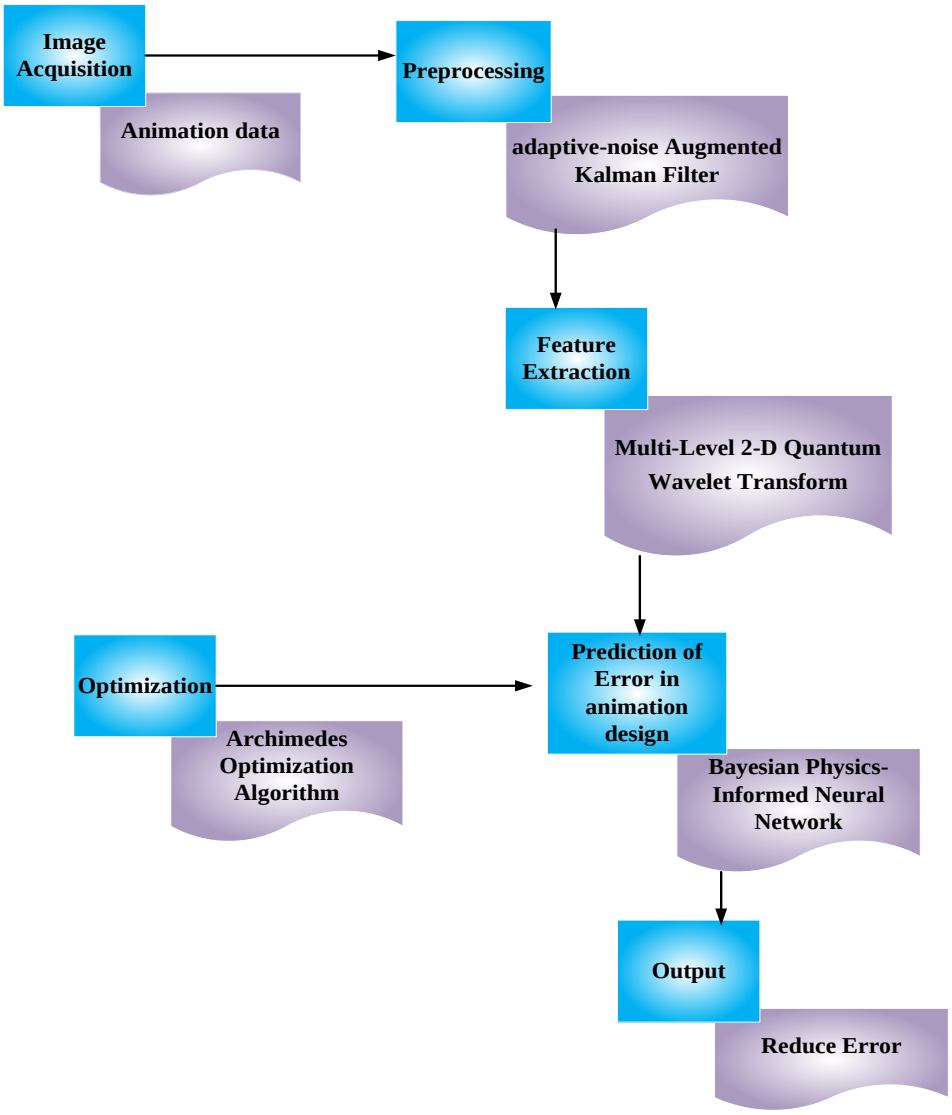

Figure 1: Block Diagram for proposed VRAID-BPINN-AOA method

A. Image Acquisition

Initially, image is collected form animation images dataset [28]. This dataset contains 1000 animated gifs split into test and train sets. An animation dataset is a collection of structured information specifically curated for training and testing purposes in the field of computer animation. These datasets typically include diverse sets of animations, covering various styles, characters, and movements to learn and generalize effectively.

B. Pre-processing using an adaptive-noise Augmented Kalman Filter

In this section, ANAKF [29] is used for pre-processing the animation image. ANAKF is used to remove the noise from the image. Based on the phase connection derivation among sub apertures, a unique ANAKF method is created. As a result, (1) provides the prediction error for the measured quantities.

$$E^{\circ} = \frac{1}{n_{\circ}} \sqrt{\sum_{l=1}^{n_{\circ}} \left(\arg\min_{\theta} \left\| \theta y_l^{\circ} - \left(y_l^{\circ} - \hat{y}_l^{\circ} \right) \right\|_2 \right)^2} \tag{1}$$

In the present window $(1 \le j \le N)$, the response obtained at the lth measured DOF, i.e., $1 \le l \le n_o$, is represented by $y_l^o = \left\lfloor y_{l_1}^o \quad \dots \quad y_{l_j}^o \quad \dots \quad y_{l_N}^o \right\rfloor$. In the same way, at the lth determined DOF in the current time batch at the N time-steps, The response that the AKF re-estimated is shown by $\hat{y}_l^o = \left\lfloor \hat{y}_{l_1}^o \quad \cdots \quad \hat{y}_{l_j}^o \quad \cdots \quad \hat{y}_{l_N}^o \right\rfloor$. For the ith measured response, where $y_l^o \in R^N$ and $\left(y_l^o - \hat{y}_l^o \right) \in R^N$, the term θ in Eq. (1) represents the scalar output of a system of linear equations solved using least squares $\theta y_l^o = \left(y_l^o - \hat{y}_l^0 \right)$. The vector y_l^o is utilized to

normalize this deviation and obtain a dimensionless estimate, the difference between the measured and re-estimated signals at each time-step is quantified by the latter vector. Compared to other estimation methods, such as calculating the mean value of instantaneous errors, the formulation of least-squares in Equation (1) has been favored because it enables a smoothed estimate of error throughout the N time-steps, guaranteeing consistent input and reaction forecasts for large q^{n_d}, q^{n_v} and q^{u} range bounds. For the variables that need to be tuned, this eliminates the requirement for the A-AKF user to choose ad hoc bounds. Next is the prediction error for the unmeasured quantities in (2).

$$E^{p} = \frac{1}{n_e}\sqrt{\sum\left(\arg\min_{\beta}||\beta y_l^e - \left(y_l^e - \hat{y}_l^e\right)\|_2\right)^2} \tag{2}$$

Where $\hat{y}_l^e = \left[\hat{y}_{l_1}^e \quad \ldots \quad \hat{y}_{l_j}^e \quad \ldots \quad \hat{y}_{l_N}^e\right]$ in the current window, the response is estimated by the AKF at the lth unmeasured DOF, or $1 \le l \le n_e$, and its actual value is y_l^e. If the latter quantity isn't available, an alternative can be a reference estimate. The AKF prediction error can then be calculated using the vector $\hat{z}_k^{\,p}$ of the responses estimated by CMS-ME, which is provided in (3).

$$E^{p} = \frac{1}{n_e}\sqrt{\sum_{l=1}^{n_e}\left(\arg\min_{\beta}||\beta \hat{z}_l^{\,p} - \left(\hat{z}_l^{\,p} - \hat{y}_l^e\right)\|_2\right)^2} \tag{3}$$

Where $\hat{z}_l^{\,p} = \left[\hat{z}_{l_1}^{\,p} \quad \cdots \quad \hat{z}_{l_j}^{\,p} \quad \cdots \quad \hat{z}_{l_N}^{\,p}\right]$ and $\hat{y}_l^e$ are the responses that, In the present batch time, $\left(1 \le j \le N\right)$, were calculated utilizing the AKF at the CMS-ME approach and lth unmeasured DOF, or $1 \le l \le n_e$. Next, the prediction error for the unidentified input is given in (4).

$$P = \begin{bmatrix} P^{nn} & P^{nu} \\ P^{un} & P^{uu} \end{bmatrix} \tag{4}$$

Where, $P^{uu} = E\left\{\left(u - \hat{u}\right)\left(u - \hat{u}\right)^T\right\} \in R^{n_i \times n_i}$ is the unknown input error's covariance. Equation (4)'s explicit time dependency has been removed for the purpose of clarity. Each unknown input estimation error's variance is represented by this matrix's diagonal elements. ANAKF is more challenging for datasets based on time series since any modification applied without domain expertise runs the risk of altering the temporal information in the animation images. It involves normalizing each image and removed the noise of the image. It is given in equation (5)

$$E^{u} = \sqrt{\frac{1}{N}\sum_{j=k}^{k+N}\frac{P_j^{uu}}{\hat{u}_j^2}} \tag{5}$$

For normalization, the matching estimated input's squared amplitude at the jth time-step in the window $\hat{u}_j^{\,2}$ has been utilized. The approach presented here assumes the presence of negligible cross-correlation terms and takes into account zero cross-correlation of noise sources, as P^{uu}. Finally, Animation images are pre-processed by ANAKF, which is removed the noise of every animation image layer by layer. These pre-processed animation images are fed into feature extraction segment.

C. Feature Extraction using Multi-Level2-D Quantum Wavelet Transform

In this section, ML2DQWT [30] is discussed for Feature Extraction. ML2DQWT is used to extract spatiotemporal features such as Object position, velocity, Optical flow and Crowd Density. The multilevel classical wavelet transform in two dimensions was given in (6).

$$W_{2^{n-(j-1)}} A_{j-1} W_{2^{m-(j-1)}}^{T} = \begin{bmatrix} A_j & B_j \\ E_j & G_j \end{bmatrix} \tag{6}$$

Where A_j, B_j, E_j, and G_j are the sub images, and $W_{2^{n-(j-1)}}$ and $W_{2^{m-(j-1)}}^{T}$ are the $2^{n-j+1} \times 2^{n-j+1}$ and $2^{m-j+1} \times 2^{m-j+1}$ matrices for wavelet kernels, respectively. Taking into account a $2^n \times 2^m$ image with an input

of $A_0 = \wedge_{2^n,2^m}$, the traditional 2-D wavelet transform with multiple levels of $\wedge_{2^n,2^m}$. First, define $T_{n,m}^{j}$ and $R_{n,m}^{j}$, which are provided in equations (7) and (8) to make the multilevel 2D-QWT possible.

$$T_{n,m}^{j} = Diag\left(W_{2^{n-j+1}} \otimes W_{2^{m-j+1}}, I_{2^{n+m-2j+3}}\right) \times \left(I_{2^2} \otimes P_{2^{m-j+1},2^{n-j+1}}\right)\left(I_2 \otimes P_{2^{n-j+1},2^{m-j+2}}\right) \tag{7}$$

$$R_{n,m}^{j} = \left(I_2 \otimes P_{2^{m-j+2},2^{n-j+1}}\right)\left(I_{2^2} \otimes P_{2^{n-j+1},2^{m-j+1}}\right) \tag{8}$$

Where A_j, B_j, E_j, and G_j are the sub images, and $W_{2^{n-(j-1)}}$ and $W_{2^{m-(j-1)}}^{T}$ are the $2^{n-j+1} \times 2^{n-j+1}$ and $2^{m-j+1} \times 2^{m-j+1}$ the corresponding matrix wavelet kernels. The traditional multilevel 2-D wavelet transform of $\wedge_{2^n,2^m}$ is performed, given an image $A_0 = \wedge_{2^n,2^m}$. The Object Position is the location or coordinates of an object with respect to a certain point or origin in a given reference frame are refers object position. It is a vector quantity expresses how far object has moved from a selected reference point. Position is frequently defined in physics in terms of direction and distance from a reference frame of reference point. The Object position is calculated in equation (9)

$$p(k) = p_0 + w_0 k + \frac{1}{2} c k^2 \tag{9}$$

Where, $p(k)$ represents the position at time k, p_0 denotes the initial position, w_0 indicates the initial velocity, c represents constant acceleration, k is the time. Velocity means speed at which something changes in relation to time. It is vector quantity that expresses the motion's direction and speed of an item. In mathematics, velocity is defined as the position of an object divided by its time. The velocity is formulated in equation (10)

$$w(k) = w_0 + ck \tag{10}$$

Here, w_0 indicates the initial velocity, $w(k)$ represents the velocity at time k, c represents constant acceleration, k is the time. The optimal flow is the movement or velocity of features or pixels between frames in a video sequence is represented by the optical flow field. It gives details about the movement of various parts of video, including its direction and intensity. Optimal flow is calculated in equation (11)

$$\frac{dv}{dk} + v\frac{dz}{dz} + w\frac{dv}{dx} = 0 \tag{11}$$

Where, v and w denotes horizontal with vertical components of optical flow vector, $\frac{dv}{dk}$ represents temporal derivatives of optical flow components, $\frac{dz}{dk}$ denotes the spatial gradients of the video. Crowd density is the concentration of people inside a specific area or place is referred to as crowd density. It is a measurement of the quantity of people or things in a given area. Then, the crowd density is given in equation (12)

$$Crowd\ Density = \frac{Number\ of\ People}{Area} \tag{12}$$

Here, $Number of people$ represents the count of individuals present in the area, $Area$ denotes the space over the crowd. Then the extracted anomalous spatiotemporal features such as Object position, velocity, Optical flow and BPINN is fed features related to Crowd Density in order to predict the error in the animation picture.

D. Prediction of error in animation design using Bayesian Physics-Informed Neural Networks

Using noisy data and partial differential equations (PDEs), a Bayesian physics-informed neural network (BPINN) [31] can solve forward and inverse nonlinear problems. Within this Bayesian framework, the variation inference (VI) or the Hamiltonian Monte Carlo (HMC) could function as an estimator of the posterior, while the BNN in conjunction serves as the prior for PDEs with a PINN. The general partial differential equation is given in (13)

$$\begin{aligned} N_x(u;\lambda) &= f, \quad x \in D, \\ B_z(u;\lambda) &= b, \quad x \in \Gamma, \end{aligned} \tag{13}$$

In this case, N_x represents a differential operator in general, D denotes the physical domain with dimension d, $u = u(x)$ represents the PDE's solution, and λ represents the PDE's parameter vector. Furthermore, the

operator for boundary conditions operating on the domain boundary Γ is B_x, and the forcing term is $f = f(x)$. Since λ is required for forward problems, our objective is to determine the distribution of u at any x. λ must also be inferred from the data in inverse problems. The noisy measurement is given in (14)

$$D = D_u \cup D_f \cup D_b, \tag{14}$$

Here, $D_u \left\{\left(x_u^{(i)}, \bar{u}^{(i)}\right)\right\}_{i=1}^{N_u}$, $D_f = \left\{\left(x_f^{(i)}, \bar{f}^{(i)}\right)\right\}_{i=1}^{N_f}$, $D_b = \left\{\left(x_b^{(i)}, \bar{b}^{(i)}\right)\right\}_{i=1}^{N_b}$.Assumes Gaussian distribution for each measurement that is independent and focused on the actual value that is hidden. The hidden value equation is given in (15)

$$\begin{aligned} \bar{u}^{(i)} &= u\left(x_u^i\right) + \in_u^{(i)}, \quad i = 1,2 \ldots N_u, \\ \bar{f}^{(i)} &= f\left(x_f^i\right) + \in_h^i, \quad i = 1,2 \ldots N_f, \\ \bar{b}^{(i)} &= b\left(x_b^{(i)}\right) + \in_b^{(i)}, \quad i = 1,2 \ldots N_b, \end{aligned} \tag{15}$$

Here $\in_u^{(i)}$, $\in_f^{(i)}$, and $\in_b^{(i)}$ are zero-mean independent Gaussian noises. Assume additionally that each sensor's fidelity is known, that is, the standard deviations of $\in_u^{(i)}$, $\in_f^{(i)}$, and $\in_b^{(i)}$ are known to be $\sigma_u^{(i)}$, $\sigma_f^{(i)}$, and $\sigma_b^{(i)}$, respectively. The BPINN model also predicts the error in the animation design. It is formulated in equation(16)

$$P(\theta, \lambda | D) = \frac{P(D|\theta, \lambda) P(\theta, \lambda)}{P(D)} \simeq P(D|\theta, \lambda) P(\theta, \lambda) = P(D|\theta, \lambda) P(\theta) P(\lambda), \tag{16}$$

In the last instance, the equality results from the independence of the priors for θ and λ. Note that although the PDE's parameter λ is a vector in the above problem configuration, the same framework could be used to represent the factor with a different surrogate model, such as a neural network, in situations where it is a field or fields that depend on x. Finally BPINN predicted the error in animation images. Due to its convenience, pertinence, AI-depend optimization strategy is taken to account in BPINN. The AOA is employed to enhance BPINN. Here, AOA is employed for optimizing the parameter of BPINN.

E. The Archimedes Optimization Algorithm

Here, step-by-step procedure for utilizing Archimedes Optimization Algorithm (AOA) [32] to get ideal BPINN values is explained. The AOA population based met heuristic algorithm, employing immersed objects as its population individuals.AOA initiates the search by generating an initial object population, similar to these other algorithms, all of which have random accelerations, densities, and volumes. These iterations are carried out by the algorithm until the termination condition is satisfied. The following presents a detailed mathematical expression of the AOA steps.

Step 1: Initialization

Set up the input parameters; in this case, they are the BPINN gain parameters, which are represent as N_x

Step2: Random Generation

After initialization, the input parameters are generated randomly in the form of a matrix. The AOA ought to choose the search phase prior to commencing operations. The function coefficient is calculated in (17)

$$MOA(C\_Iter) = Min + C\_Iter \times \left(\frac{Max - Min}{M\_Iter}\right) \tag{17}$$

Here, The function value at the t^{th} iteration, or $MOA(C\_Iter)$, is determined by applying Equation (21). The variable C_Iter signifies the current iteration, ranging from 1 to the maximum quantity of iterations (M_Iter). The minimum and maximum values of the accelerated function are indicated by the symbols Min and Max, correspondingly.

Step3: Fitness Function

Initialization values, result are random solution. Assessment of fitness values utilizes outcomes of weight parameter optimization N_x. It is given in equation (18)

$$fitness\ function = Optimizing(N_x) \tag{18}$$

Where N_x is the general differential operator.

Step 4: Update densities, volumes

Motivated by the principles of Archimedes' buoyancy, the Archimedes Optimization Algorithm, focuses on updating the density volume definition to enhance convergence. This algorithm dynamically adjusts the density estimation of solution spaces, optimizing computational efficiency. Using (19), the object's density and volume are updated for iteration $t+1$:

$$\begin{aligned} den_i^{t+1} &= den_i^t + rand \times \left(den_{best} - den_i^t\right) \\ vol_i^{t+1} &= vol_i^t + rand \times \left(vol_{best} - vol_i^t\right) \end{aligned} \tag{19}$$

Here the best object so far has been identified by its volume and density, vol_{best} and den_{best}, and a random number that is distributed evenly, $rand$.

Step5: Transfer operator and density factor

After initially colliding, the objects try to return to their initial state of equilibrium over time. The transfer operator TF assists in doing this in AOA, which is defined by (20) and moves search from exploration to exploitation.

$$TF = \exp\left(\frac{t - t_{\max}}{t_{\max}}\right) \tag{20}$$

Where transfer TF progressively rises over time to reach. The variables t and $t_{\max}$ represent the maximum and iteration number, respectively, in this case. Density decreasing factor D helps AOA with local and global search in a similar manner. Using (21): It gets smaller over time.

$$d^{t+1} = \exp\left(\frac{t_{\max} - t}{t_{\max}}\right) - \left(\frac{t}{t_{\max}}\right) \tag{21}$$

Where d^{t+1} gradually decreases, allowing convergence in the previously identified promising region. Keep in mind that managing this variable correctly will guarantee that exploration and exploitation in AOA are balanced.

Step6: Exploration Phase

In the exploration phase of the Archimedes Optimization Algorithm, the algorithm intelligently probes solution spaces with a diverse range of candidate solutions. By leveraging dynamic density adjustments, it adapts to the evolving landscape, emphasizing thorough exploration. When an object collides, choose a random material (mr) and use (22) to update the object's acceleration for iteration $t+1$.

$$acc_i^{t+1} = \frac{den_{mr} + vol_{mr} \times acc_{mr}}{den_i^{t+1} \times vol_i^{t+1}} \tag{22}$$

Whereas acc_{mr}, den_{mr} and vol_{mr} represent the acceleration, density, and volume of random material, den_i, vol_i and acc_i represent the density, volume, and acceleration of object I. Notably, exploration is guaranteed for one-third of the iterations if $TF \leq 0.5$. Figure 2 shows that the Flowchart for AOA optimizing BPINN.

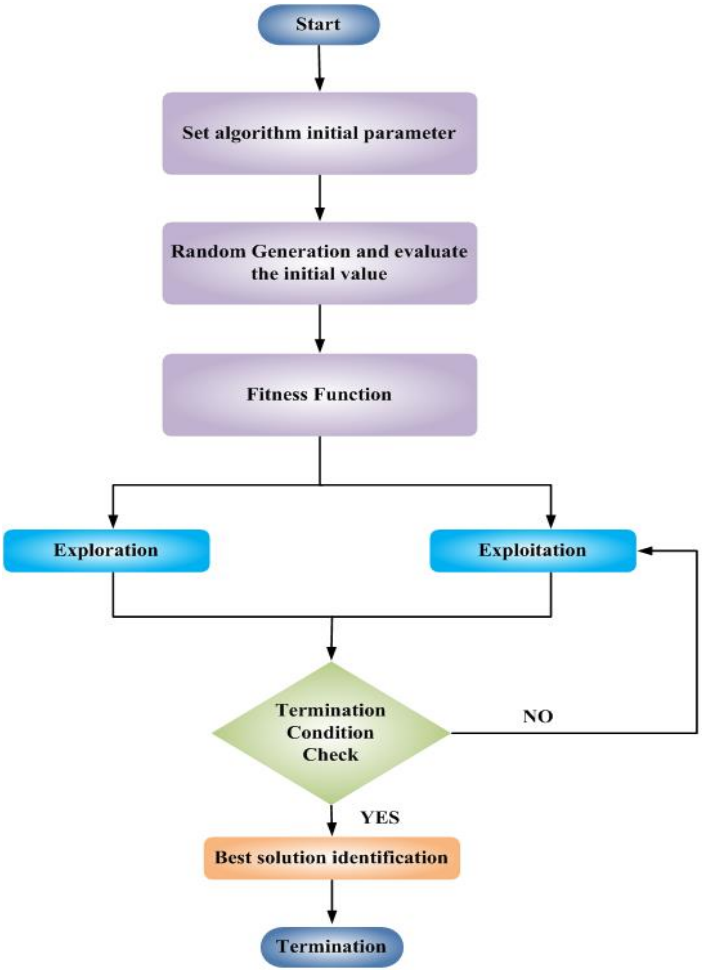

Figure 2: Flowchart for AOA optimizing BPINN

Step7: Exploitation Phase

During the exploitation phase of the Archimedes Optimization Algorithm, the focus shifts to refining promising solutions discovered in the exploration phase. This strategic exploitation phase enhances the algorithm's efficiency in converging towards optimal solutions in diverse optimization scenarios. If $TF > 0.5$ indicates that there are no object collisions, use (23) to update the object's acceleration for iteration $t+1$.I

$$acc_i^{t+1} = \frac{den_{best} + vol_{best} \times acc_{best}}{den_i^{t+1} \times vol_i^{t+1}} \tag{23}$$

Here the best object's acceleration is represented by acc_{best}.

Step8: Termination

The weight parameter values N_x from Bayesian physics-informed neural network are optimized with support of AOA, until the halting criteria are satisfied, iteratively repeat step 3. And lastly, VRAID-BPINN-AOA predicts the animation error with higher accuracy with less Error rate.

IV. RESULT WITH DISCUSSION

Investigational outcome of proposed technique is discussed. The proposed VRAID-BPINN-AOA method is implemented in MATLAB and evaluated under some metrics. Obtained results of HAD-TSV-ITF-CZNN technique are analysed with existing techniques likes VI-AD-RBM [21],GDL-VALD-GAN [22], HID-LVIR-CNN [23]correspondingly.

A. Performance measures

Selecting the best classifier requires taking this critical step. Performance metrics are evaluated in order to evaluate performance, including Bit Error rate, Design Rate, End to End Delay, Latency and Transmission Rate.

1) Bit Error Rate

BER, also known as BER, is the ratio of incorrectly received bits to total bits received. It is calculated by counting the number of errors and comparing the transmitted and received bit sequences. The bit error is calculated in (24)

$$BER = {N_{err}}/{N_{bits}} \tag{24}$$

Where, N_{err} is the number of error and N_{bits} number of bits?

2) Design Rate

Design rate refers to the pace at which neural network architectures and models are conceptualized and developed. The design rate is calculated by (25)

$$DR = \frac{(CP - IP)}{IP \times 100} \tag{25}$$

Where, CP is the current production and IP is the initial production

3) End to End Delay

The amount of time that data takes to travel through a neural network is referred to as the end-to-end delay, encompassing both inference and any pre-processing or post-processing steps. In (26) the End to End Delay is computed.

$$EED = \frac{N \times L}{R} \tag{26}$$

Where the packet length is L, the link is N, and the transmission rate is R.

4) Latency

Latency in deep learning denotes the time delay between initiating a task, such as making an inference, and receiving the corresponding output. The latency is calculated in the (27)

$$Latency = TransmissionDelay + processDelay + QueueingDelay \tag{27}$$

5) Transmission Rate

Transmission rate in deep learning pertains to the speed at which data is transferred between components, such as GPUs or distributed nodes, during training or inference. The Transmission Rate of the data is calculated by (28)

$$TR = D/T \tag{28}$$

Where, D is the data and T is the time

B. Performance Analysis

Figure 3 to 7 depicts the simulation results of proposed VRAID-BPINN-AOAmethod. Then, the proposed VRAID-BPINN-AOAmethod is likened with existingVI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNNmethods respectively.

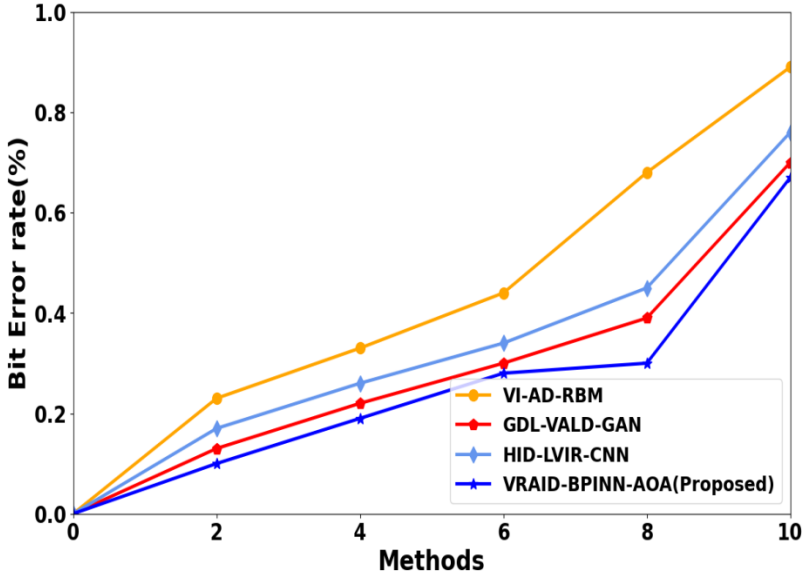

Figure 3: Performance Analysis of Bit Error rate

Figure 3 portrays Bit error rate analysis. Here, VRAID-BPINN-AOA method attains 26.48%, 23.56% and 21.16% lower Bit Error Rate at method of 2, 25.24%, 23.14% and 22.58% lower Bit Error Rate at method of 6; 28.56%, 26.67% and 25.67% lower Bit Error Rate at method of 10; when evaluated to existing VI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNN method respectively.

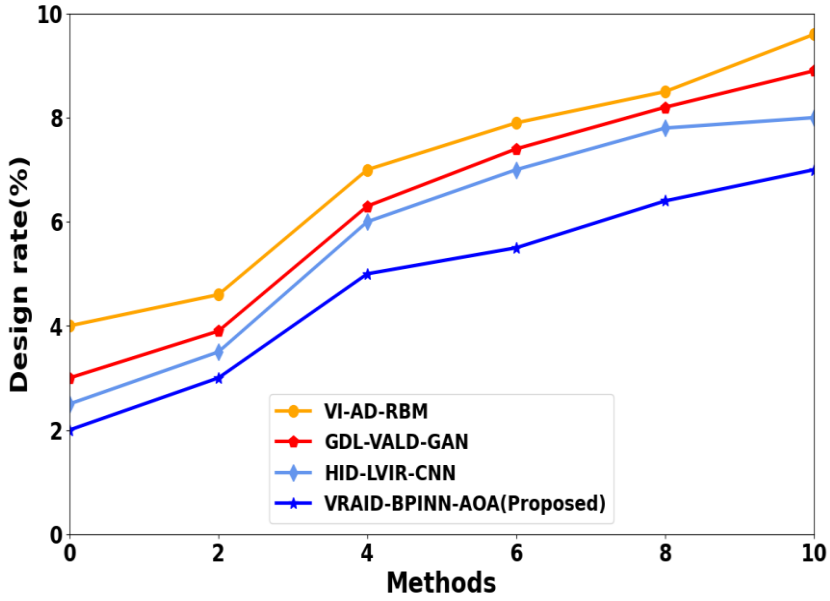

Figure 4: Performance Analysis of Design Rate

Figure 4 portrays Design rate analysis. Here, VRAID-BPINN-AOA method attains23.36%, 26.57% and 21.61% lower design rate at method of 4, 24.28%, 21.17% and 23.42% lower design rate at method of 6 and26.42%, 25.67% and 23.67% lower design rate at method of 10; when evaluated to existing VI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNN method respectively.

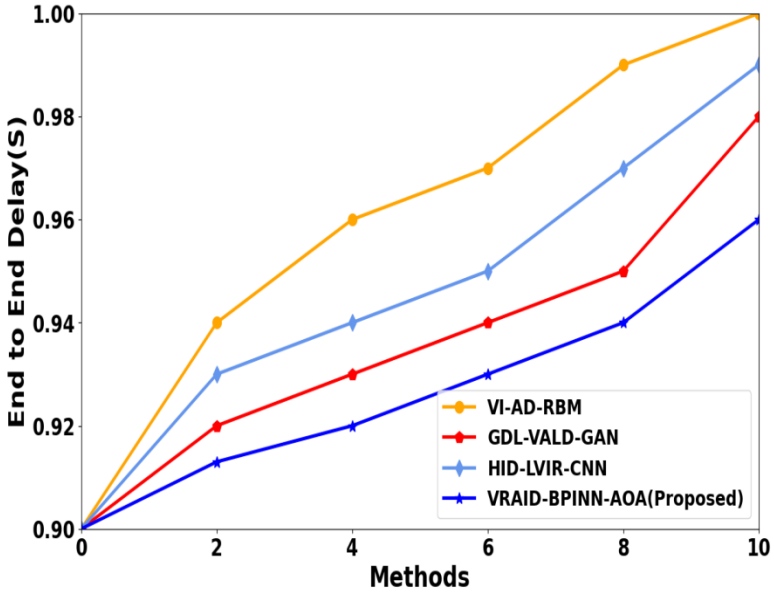

Figure 5: Performance Analysis of End to End Delay

Figure 5 portrays end to end delay analysis. Here, VRAID-BPINN-AOA method attains26.52%, 25.18% and 23.63% lower end to end delay at method of 4;27.22%, 25.29% and 24.43% lower end to end delay at method of 8 and28.56%, 24.67% and 26.67% lower end to end delay at method of 10; when evaluated to existing VI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNN method respectively.

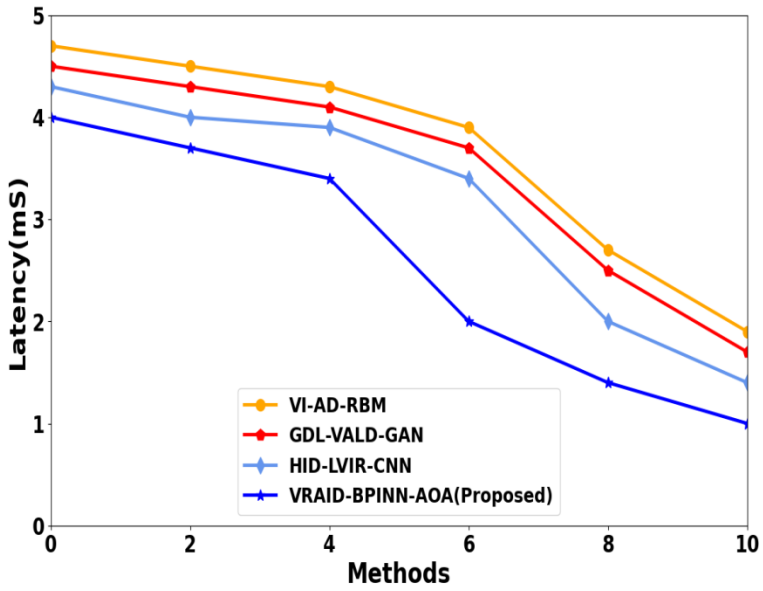

Figure 6: Performance Analysis of Latency

Figure 6 portrays latency analysis. Here, VRAID-BPINN-AOA method attains24.66%, 21.78% and 23.97% lower latency at method of 2; 26.52%, 24.68% and 22.32% lower latency at method of 4; 28.56%, 26.67% and 25.67% lower latency at method of 10; when evaluated to existing VI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNN method respectively.

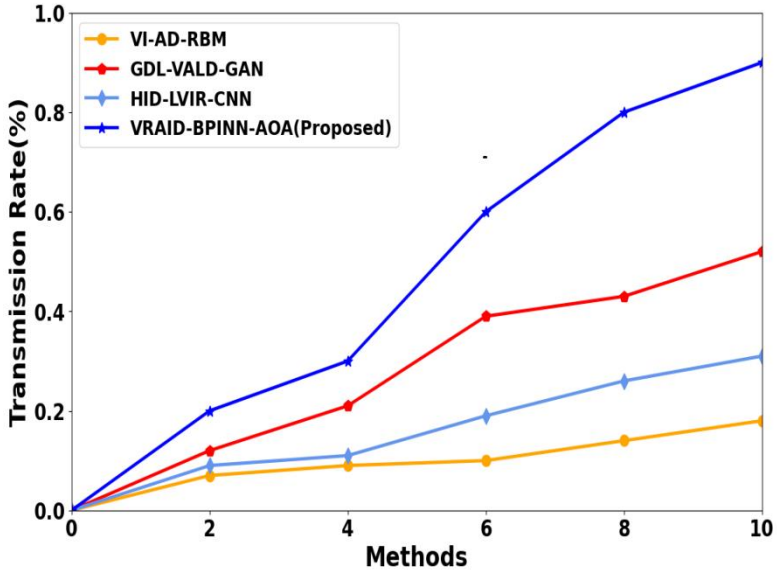

Figure 7: Performance Analysis of Transmission Rate

Figure 7 portrays Transmission Rate analysis. Here, VRAID-BPINN-AOA method attains 25.31%, 28.78% and 21.65% higher Transmission Rate at method of 2, 24.42%, 22.92% and 24.64% higher Transmission Rate at method of 6 and 26.22%, 22.67% and 25.67% higher Transmission Rate at method of 10; when evaluated to existing VI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNN method respectively.

C. Discussion

Major goal of this research project is to decrease the bit error rate, decrease the design rate, decrease the final delay, cut down on the latency, and increase the transmission rate. BPINN method for predicting the error in the animation images in the real world is presented in this paper. The thorough analysis of the literature also demonstrates when it comes to extracting features by ML2DQWT from animation images, BPINN techniques outperform manual ones. The proposed research offers an improved, pre-trained BPINN architecture that works better than the methods that have been previously published. From animation image, both spatiotemporal features are effectively extracted using this ML2DQWT model. Additionally, this work has shown how

important ANAKF is to improve training outcomes for architectures. The animation dataset is subjected to the proposed technique. The Bit error rate values of VRAID-BPINN-AOA are 28.56%, 26.67% and 25.67%lowerthan existing methods such as VI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNN respectively. Similar to this, the Transmission rate of proposed method is 96.94% analysed with average Transmission rate comparison techniques of 82.54%. The proposed method VRAID-BPINN-AOA has lower Bit error rate and higher Transmission rate evaluation metrics than existing methods. Therefore, the comparative methods are expensive than the proposed technique. As a result, the proposed technique Predict the error more effectively and efficiently.

V. CONCLUSION

This paper proposes deep learning models to predict the error in the virtual reality animation images with BPINN. Pre-processing algorithms are used to remove the noise in the images using an ANAKF. BPINN prediction algorithm is feature extractors to make prediction for spatiotemporal features such as Object Position, velocity, Optical flow and Crowd Density. The performance of the VRAID-BPINN-AOA technique has been examined using the MATLAB platform, and a comparative analysis has been conducted with other existing methods. The proposed technique is assessed through various scenarios, encompassing optimal and random scheduling, along with the utilization of a sophisticated AOA algorithm for evaluation. Performance of proposed VRAID-BPINN-AOA approach contains 28.56%, 24.67% and 26.67% lower end to end delay, 28.56%, 26.67% and 25.67% lower latency and 26.22%, 22.67% and 25.67% higher Transmission Rate is analysed with existing methods likes VI-AD-RBM, GDL-VALD-GAN and HID-LVIR-CNN respectively.

REFERENCES

[1] Debarba, H. G., Chagué, S., & Charbonnier, C. (2020). On the plausibility of virtual body animation features in virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, *28*(4), 1880-1893.

[2] Li, L., & Li, T. (2022). Animation of virtual medical system under the background of virtual reality technology. *Computational Intelligence*, *38*(1), 88-105.

[3] Hu, Z., & Liu, L. (2023). Research on the application of virtual reality technology in 3D animation creation. *Optik*, *272*, 170274.

[4] Çakıroğlu, Ü., Aydın, M., Özkan, A., Turan, Ş., & Cihan, A. (2021). Perceived learning in virtual reality and animation-based learning environments: A case of the understanding our body topic. *Education and Information Technologies*, *26*(5), 5109-5126.

[5] Chen, Y. C., Chang, Y. S., & Chuang, M. J. (2022). Virtual reality application influences cognitive load-mediated creativity components and creative performance in engineering design. *Journal of Computer Assisted Learning*, *38*(1), 6-18.

[6] Tian, Z. (2020). Dynamic visual communication image framing of graphic design in a virtual reality environment. *Ieee Access*, *8*, 211091-211103.

[7] Wang, Q., Li, C., Xie, Z., Bu, Z., Shi, L., Wang, C., & Jiang, F. (2020). The development and application of virtual reality animation simulation technology: take gastroscopy simulation system as an example. *Pathology & Oncology Research*, *26*, 765-769.

[8] Zhang, K. (2021). Animation virtual reality scene modeling based on complex embedded system and FPGA. *Microprocessors and Microsystems*, *80*, 103632.

[9] He, Z., Du, R., & Perlin, K. (2020, November). Collabovr: A reconfigurable framework for creative collaboration in virtual reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (pp. 542-554). IEEE.

[10] Raya, L., García-Rueda, J. J., López-Fernández, D., & Mayor, J. (2021). Virtual reality application for fostering interest in art. *IEEE Computer Graphics and Applications*, *41*(2), 106-113.

[11] Unsworth, L. (2020). A multidisciplinary perspective on animation design and use in science education. *Learning from animations in science education: Innovating in semiotic and educational research*, 3-22.

[12] Liu, X. (2020). Three-dimensional visualized urban landscape planning and design based on virtual reality technology. *IEEE access*, *8*, 149510-149521.

[13] Tsai, Y. T., Jhu, W. Y., Chen, C. C., Kao, C. H., & Chen, C. Y. (2021). Unity game engine: Interactive software design using digital glove for virtual reality baseball pitch training. *Microsystem Technologies*, *27*, 1401-1417.

[14] Guo, Z., Zhou, D., Zhou, Q., Mei, S., Zeng, S., Yu, D., & Chen, J. (2020). A hybrid method for evaluation of maintainability towards a design process using virtual reality. *Computers & Industrial Engineering*, *140*, 106227.

[15] Gao, G., & Li, W. (2022). Architecture of visual design creation system based on 5G virtual reality. *International Journal of Communication Systems*, *35*(5), e4750.

[16] Amat, A. Z., Zhao, H., Swanson, A., Weitlauf, A. S., Warren, Z., & Sarkar, N. (2021). Design of an interactive virtual reality system, InViRS, for joint attention practice in autistic children. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *29*, 1866-1876.

[17] Albus, P., Vogt, A., & Seufert, T. (2021). Signaling in virtual reality influences learning outcome and cognitive load. *Computers & Education*, *166*, 104154.

[18] Peng, L., Yu, L., Shen, H., & Pi, J. (2021). RETRACTED: 3D Garden landscape planning visualization system based on FPGA processor and virtual reality.

[19] Wu, H., Cai, T., Liu, Y., Luo, D., & Zhang, Z. (2021). Design and development of an immersive virtual reality news application: a case study of the SARS event. *Multimedia tools and applications*, *80*, 2773-2796.

[20] Zhang, Z., Wu, Y., Pan, Z., Li, W., & Su, Z. (2022). A novel animation authoring framework for the virtual teacher performing experiment in mixed reality. *Computer Applications in Engineering Education*, *30*(2), 550-563.

[21] Liu, B. (2022). Research on virtual interactive animation design system based on deep learning. *Computational Intelligence and Neuroscience*, *2022*.

[22] Ardhianto, P., Santosa, Y. P., Moniaga, C., Utami, M. P., Dewi, C., Christanto, H. J., & Chen, A. P. S. (2023). Generative deep learning for visual animation in landscapes design. *Scientific Programming*, *2023*.

[23] Kang, T., Chae, M., Seo, E., Kim, M., & Kim, J. (2020). DeepHandsVR: Hand interface using deep learning in immersive virtual reality. *Electronics*, *9*(11), 1863.

[24] Gan, B., Zhang, C., Chen, Y., & Chen, Y. C. (2021). Research on role modeling and behavior control of virtual reality animation interactive system in Internet of Things. *Journal of Real-Time Image Processing*, *18*(4), 1069-1083.

[25] Paier, W., Hilsmann, A., & Eisert, P. (2020). Interactive facial animation with deep neural networks. *IET Computer Vision*, *14*(6), 359-369.

[26] Zhu, M., & Lee, J. H. (2022). Deep Learning-Based 3D Shape Feature Extraction on Flash Animation Style. *Wireless Communications and Mobile Computing*, *2022*.

[27] Fukuda, T., Novak, M., Fujii, H., & Pencreach, Y. (2021). Virtual reality rendering methods for training deep learning, analysing landscapes, and preventing virtual reality sickness. *International Journal of Architectural Computing*, *19*(2), 190-207.

[28] https://www.kaggle.com/datasets/lukedraht/animation-dataset

[29] Vettori, S., Di Lorenzo, E., Peeters, B., Luczak, M. M., & Chatzi, E. (2023). An adaptive-noise Augmented Kalman Filter approach for input-state estimation in structural dynamics. *Mechanical Systems and Signal Processing*, *184*, 109654.

[30] Li, H. S., Fan, P., Peng, H., Song, S., & Long, G. L. (2021). Multilevel 2-d quantum wavelet transforms. *IEEE Transactions on Cybernetics*, *52*(8), 8467-8480.

[31] Yang, L., Meng, X., & Karniadakis, G. E. (2021). B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *Journal of Computational Physics*, *425*, 109913.

[32] Hashim, F. A., Hussain, K., Houssein, E. H., Mabrouk, M. S., & Al-Atabany, W. (2021). Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems. *Applied intelligence*, *51*, 1531-1551.