Augmented Reality-Based Visualization and Simulation of Autonomous Unmanned Aerial Vehicle Control System

Mokhles M. Abdulghani
Jackson State University
Jackson, Mississippi
mokhles.m.abdulghani@students.jsums.edu

Wilbur L. Walters
Jackson State University
Jackson, Mississippi
wilbur.l.walters@jsums.edu

Khalid H. Abed
Jackson State University
Jackson, Mississippi
khalid.h.abed@jsums.edu

***Abstract*—Although the incorporation of sensing, actuators, and controllers make robots powerful systems, simulating them is a big challenge. While it is possible to assemble a simulator from the innumerable graphics, kinematics, physics and libraries, the architecture and control procedure are crucial to shaping the interaction of these elements and consequently the general performance of the robotic system. This paper reviews the capability of two different robotic simulation platforms to simulate different robotic systems that allow for direct interface of various control techniques implemented in MATLAB. This direct interface with Augmented Reality (AR) and simulation models allows more accessibility to test and evaluate the designed control systems to a general-public by reducing the model operation complexity. Using AR simulators reduces the cost, strengthens productivity, and increases the quality of design by offering a flexible diagnostic and evaluation method using AR.**

Index Terms*—*Augmented reality, Robot Simulation, CoppeliaSim, V-Rep, Unity 3D, MATLAB, Simulink, 3D simulation, Real-time Control.

I. Introduction

The virtual environment is an environment in which simulations activities are executed with the purpose of analyzing and testing the stability of the simulated system. The virtual environment was initially developed for application in video games and entertainment; nowadays, virtual environments can be used to simulate different applications in robotics. Augmented Reality (AR) offers simulation for various types of real-time control systems and provides tools, functionalities, and conceptualization for the basic robotic system and its complexity since this system specificity cannot be predicted. AR is beneficial in testing and evaluating the performance of various robotic systems where robotics generally is classified according to their field of application, service robotics and industrial robotics [1], [2]. Furthermore, AR simulation offers flexible controller approach that can be portable, easily designed and maintained, applicable and scalable to various robotic models. There are currently several robot simulation platforms available, for instance Gazebo [3] Open HRP [4], or WebOS [5]. While some offer efficient competing functionality, many other platforms fail in offering complementary programming techniques, and their simulation models and controllers are only partially portable. For example, controller recompilation on a different hardware or platform is often necessary, or the simulation model and controller need to be carefully matched since they represent at least two distinct files. When scaling is supported, it is done via relatively obscure hard-wired mechanisms. The Virtual Robot Experimentation Platform [6] offers a robotic simulation environment using the concept of AR. Unity 3D is another platform that offers 3D visualization and simulation for robotic control systems. It is commercial software developed by Unity 3D Technologies. For successful AR application, a deep understanding of the implemented tasks is required, a challenging task that includes object tracking, multiple sensor fusion, object detection and tracking, Unmanned Aerial Vehicles (UAVs) navigation. UAV can be used for a large range of tasks and having them as a part of the communication infrastructure. Using UAVs to autonomously accomplish complex tasks from the ground, the water, or the sky could include several challenges.

In this paper, we implemented AR-based visualization and simulation of autonomous navigation of 2WD robot and drone using direct interface between MATLAB-CoppeliaSim and MATLAB-Unity 3D platforms. Section II discusses the related work in the field of 3D robot simulation using CoppeliaSim and Unity 3D. Section III of this paper describes using CoppeliaSim as a 3D simulation virtual environment for UAVs. Section IV of this paper describes using Unity 3D as a 3D simulation virtual environment for UAVs. Section V of this paper illustrates the direct interface technique between MATLAB and the AR platforms to test and visualize UAVs control system using Unity 3D platform and CoppeliaSim virtual environment. Finally, section VI presents a conclusion for this paper.

II. Related Work

Robotics simulators are one of the best methods to simulate, visualize, analyze, the performance of robotic control systems in AR. CoppeliaSim (formerly V-Rep), is a virtual robotic simulation platform with an integrated development environment based on an allocated and distributed control architecture [7]. The most used approach to interface the MATLAB-Simulink environment with CoppeliaSim is by using the ROS communication libraries for prototyping of robot control algorithms [8], [9].

Unity 3D simulation works through functions defined using programming languages and provides three-dimensional manipulation. The platform capable of making high quality 3D animation, design and develop 3D social network gaming platforms [10]. In our previous research, we used Unity 3D with single-agent reinforcement learning. The design and training of the model of the exploring agent that could explore the environment and avoid the obstacles using the Proximal Policy Optimization (PPO) algorithm has been implemented using only Unity 3D platform. The designed model was tested and evaluated in the same platform, and we could minimize the collisions for the agent by increasing the hidden units and decreasing the training steps [11]. Another work we implemented was to train multiple agents in different environments to minimize collisions and avoid obstacles and an excellent result was achieved by reducing the collisions and reducing the execution time [12]-[13].

III. UAV Simulation Using CoppeliaSim and MATLAB

Almost every aspect of the robot simulation in CoppeliaSim can be customized. This simulator can be customized and molded to perform exactly as desired. This is offered throughout an intricate Application Programming Interface (API). More than six coding techniques are supported in CoppeliaSim, each can fit on a certain robotic control system. One of the most used methods to initiate a MATLAB-CoppeliaSim communication is through the Robot Operating System (ROS). MATLAB version maters in terms of the compatibility for tools and applications with its different versions. In the case of ROS, it is compatible with MATLAB-SIMULINK version 2018 and earlier. Fig. 1 shows the CoppeliaSim Robotic Simulation Models.

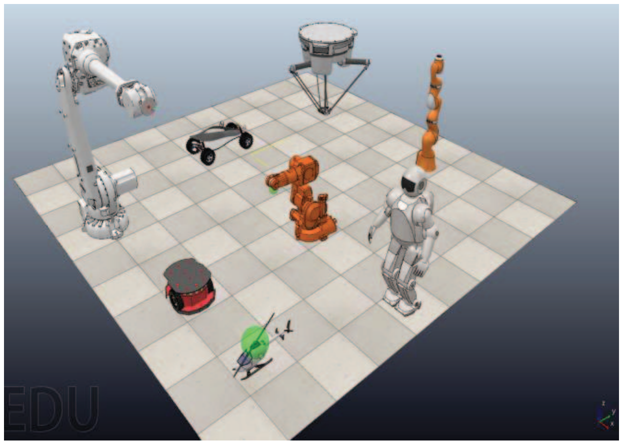


Fig. 1. CoppeliaSim Robotic Simulation Models.

IV. UAV Simulation Using Unity 3D and MATLAB

Unity 3D is a cross-platform robot simulator designed to support and develop 2D and 3D visualization, virtual reality, simulations for computers, consoles, and mobile devices platform. In addition, 3D visualization, functions and attributes, intelligence and metric measurements can be done with Unity 3D encoding. Fig. 2 shows the Unity 3D Robotic Simulation Models.

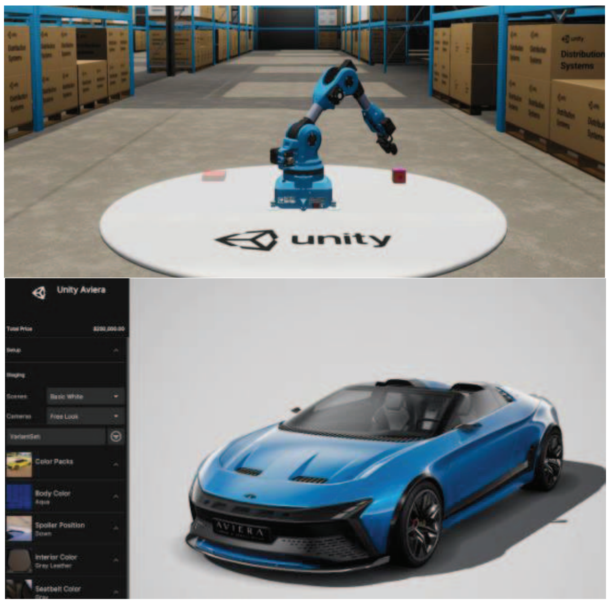


Fig. 2. Unity 3D Robotic Simulation Models.

V. Direct Interface Between MATLAB and AR Platforms

MATLAB is a programing platform with independent coding language and advance engineering environment that offers the efficient skill of data processing and matrix manipulation. Implementing the designed algorithm on a certain machine while simulating it on a different machine could represent a distinct machine or a robot, connected to the simulator machine via a specific network such as, serial port or socket. Using this approach allows the control algorithm to remain native and executed on the original hardware. Moreover, a noticeable reduction in the computing load on the simulation machine will be offered. In this section we will illustrate the implementation of direct interface between MATLAB and the two chosen AR-based robotic simulation platforms: CoppeliaSim and Unity 3D. Both CoppeliaSim and Unity 3D, with integrated development environment, is simulating the chosen model with a distributed control architecture: each object/model can be separately controlled through a remote API client, an embedded script, a ROS node, a plugin, or a custom solution. This makes them extremely resourceful and flexible for multi-agent applications as we implemented in our previous work [12], [13]. The controller algorithms can be written in MATLAB, Python, C/C++, Lua, Java, or Octave. CoppeliaSim is used for autonomous UAVs and automation simulations, fast prototyping, efficient algorithm development, and verification, robotics related education, remote monitoring, safety double-checking, as digital twin, and much more. Fig. 3 shows the mechanism of interface between MATLAB and AR platforms. CoppeliaSim can be communicated directly with MATLAB as we have previously presented in [6].

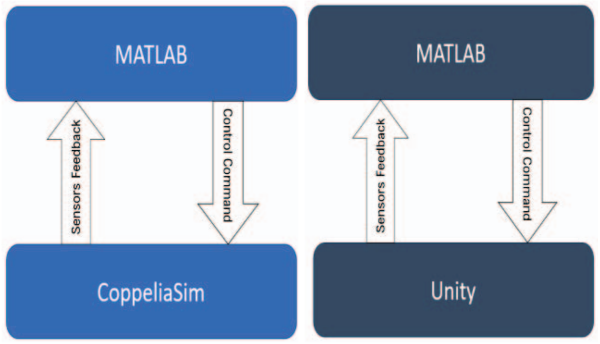


Fig. 3. The mechanism of interface between MATLAB and AR platforms.

A. Direct Interface Between CoppeliaSim and MATLAB

The remote API interface in CoppeliaSim offers interaction with AR environment or a simulation, controlled via an external entity through socket communication. It is comprised of remote API clients and remote API server services. The client side can be set in as a small footprint code in AR representing any hardware including real robots. It allows calling of remote functions and quick data streaming back and forth. On the client side, functions are called almost as regular functions, with two exceptions however: remote API functions accept an additional argument which is the operation mode and return the same error code. The operation mode offers calling the function as blocking and then wait the feedback from the server.

Fig. 4 presents the implemented ANFIS controller of autonomous 2WD mobile robot in our previous work [6] using MATLAB-Simulink performing collision avoidance tasks in AR using CoppeliaSim. Each of the MISO ANFIS controllers (for the left and right motors) will receive feedback signals from six ultrasonic sensors provided with the 2WD Pioneer robot and produce the output according to the training data. The sensors reading received in meters from CoppeliaSim are converted to centimeters in the MATLAB-Simulink model using Gain and MATLAB function blocks for each input. The ANFIS controller produces the output as a pulse-width modulation (PWM) signal then it will be converted to a rotation speed in round per minute (RPM) to be sent to the CoppeliaSim Pioneer model.

MATLAB starts the communication with CoppeliaSim and reads data from the robot sensors through a set of commands written in script code. The remote API function in the CoppeliaSim software has been used to define the required variables to be received and sent from CoppeliaSim to MATLAB-Simulink. In case of connecting the MATLAB script code to the CoppeliaSim, the asynchronous mode should be selected which means that MATLAB and CoppeliaSim will run in parallel by sending data and commands back and forth.

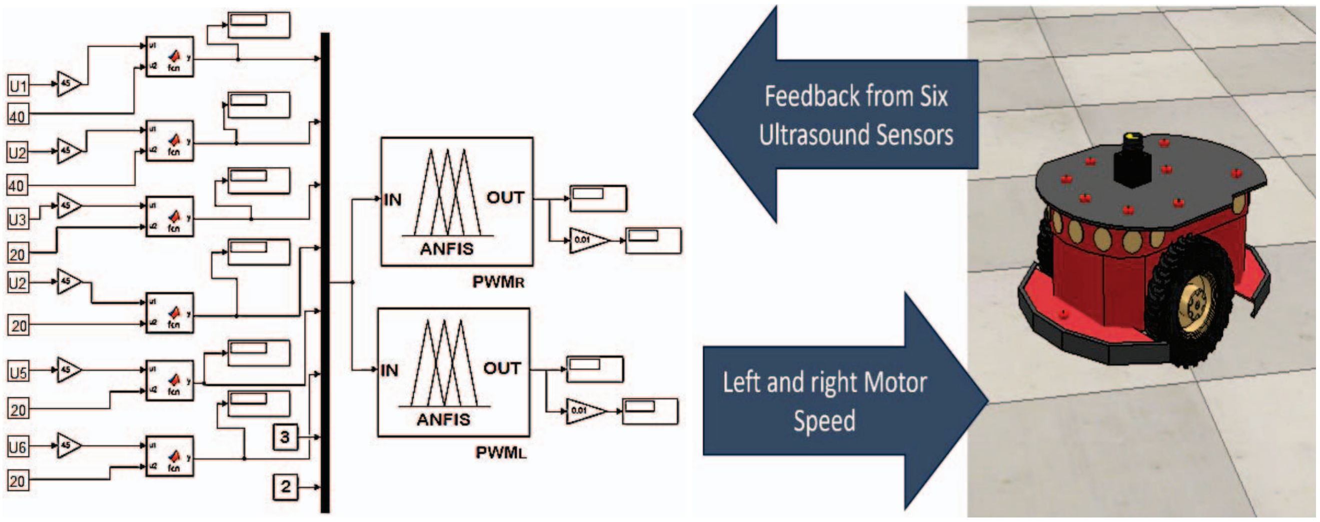
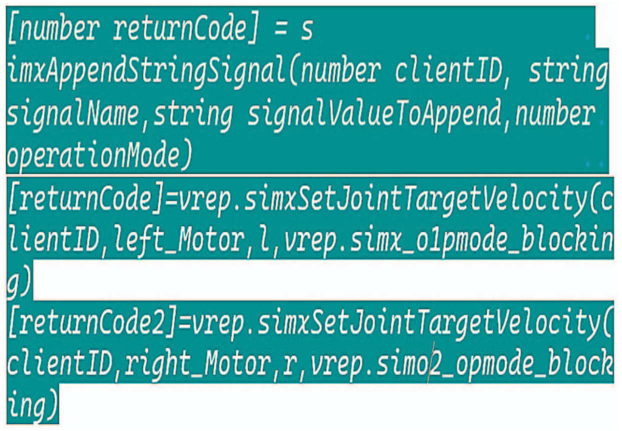

U1
40
U2
40
U3
20
U2
20
U5
20
U6
20
3
2
IN
OUT
ANFIS
PWMR
IN
OUT
ANFIS
PWML
Feedback from Six Ultrasound Sensors
Left and right Motor Speed

Fig. 4. Direct Interface of MIMO ANFIS MATLAB-Simulink model to control autonomous 2WD mobile robot in AR using CoppeliaSim.

The following function has been used to define the CoppeliaSim variables needed to communicate with the MATLAB-Simulink model:



```
[number returnCode] = s
imxAppendStringSignal(number clientID, string
signalName,string signalValueToAppend,number
operationMode)
[returnCode]=vrep.simxSetJointTargetVelocity(c
lientID,left_Motor,l,vrep.simx_o1pmode_blockin
g)
[returnCode2]=vrep.simxSetJointTargetVelocity(
clientID,right_Motor,r,vrep.simo2_opmode_block
ing)
```

The (*l*) in the above code represents the left motor speed signal and (*r*) represents the right motor speed signal that were sent as PWM signal. The same sampling rate has been chosen (0.01 second) for each output to keep the system working in asynchronously. Fig. 5 shows the simulation of the 2WD mobile robot using the CoppeliaSim platform.

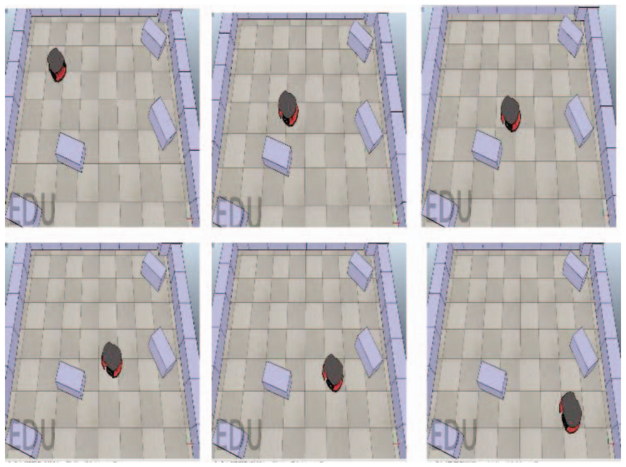


Fig. 5 Simulation of autonomous 2WD mobile robot using the CoppeliaSim platform.

B. Direct Interface Between CoppeliaSim and Unity 3D

Similar to CoppeliaSim platform, Unity 3D platform offers the possibility of implementing a direct interface with MATLAB and MATLAB-Simulink. This can be done through bidirectional communication where the windows Inter Process Communication (IPC) mode is offered in Unity 3D. IPC can be enabled in the operating systems on which information can be exchanged through an active communication which enables data exchange and synchronization in the desired control system. The bidirectional data communication between MATLAB-Unity3D is performed by a dynamic link library (dll), in which the Shared Memory method is implemented through Unity Distribution Portal (UDP). This is a one-way communication between MATLAB and Unity 3D meaning that Unity will not manipulate the simulation, only visualize what is decided by the control model in MATLAB. We implemented a drone flight control system and the direct interface between MATLAB-Simulink and Unity 3D model of Trello Drone which we used in our previous work [14]. Fig. 6 shows the Simulation of autonomous Drone model using the Unity 3D platform and MATLAB-Simulink controller.

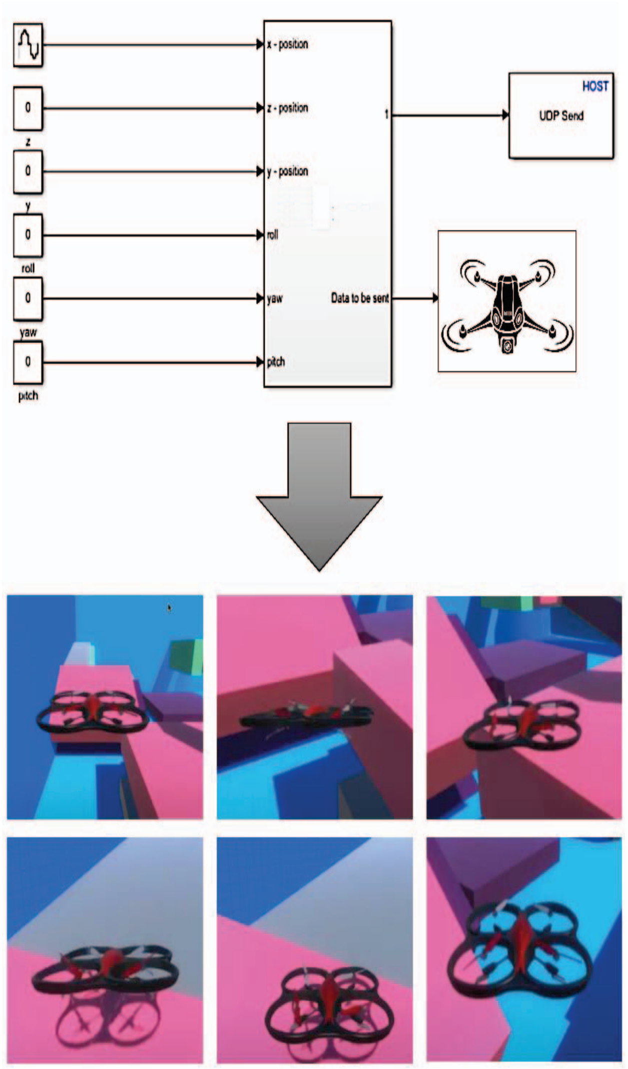


Fig. 6 Simulation of autonomous Drone model using the Unity 3D platform.

VI. Conclusion

MATLAB-CoppeliaSim direct simulation was introduced as an efficient and scalable simulation in AR environment. By offering a multitude of different coding approaches for the desired controllers along with the possibility of embedding the control algorithm in the simulation models, it reduces the complexity for the users and increases the reproducibility. MATLAB-Unity 3D interface has been implemented to simulate a drone control system in AR. This simulator considers the bilateral communication between MATLAB-Unity3D through a dll to enable synchronized control and visualization.

Acknowledgments

This work was supported in part by United States Air Force *Research Institute for Tactical Autonomy (RITA)* University Affiliated Research Center (UARC), and in part by The United States Air Force Office of Scientific Research (AFOSR) contract FA9550-22-1-0268, entitled: "I*nvestigating Improving Safety of Autonomous Exploring Intelligent Agents with Human-in-the-Loop Reinforcement Learning*," and in part by Jackson State University.

References

[1] Andersen, R.S.; Bogh, S., Moeslund, T.B.; Madsen, O. "Intuitive task programming of stud welding robots for ship construction", IEEE Industrial Technology (ICIT), 2015 IEEE International Conference on, pp 3302 – 3307, March 2015.

[2] Ying J.L; Peng J.S; Qi Z; Chang C.L; Yong H.; "The Review of Workpiece Loading and Unloading Robot in the Catenary Shot Blasting"; Research and Design of Machinery, Equipment and Technological Processes in Mechanical Engineering; Applied Mechanics and Materials, Vols. 496-500, pp. 578-581, Jan. 2014

[3] N. Koenig, and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in Proc of Int. Conf. on Intelligent Robots and Systems, pp. 2149-2154, Sendai, Japan, Sept.-Oct. 2004.

[4] F. Kanehiro, H. Hirukawa, and S. Kajita, "Open HRP: Open Architecture Humanoid Robotics Platform," Int. J. of Robotics Research, vol 23, pp. 155-165, 2004.

[5] O. Michel³Webots: professional mobile robot simulation´Int. J. Adv. Robot. Syst., vol. 1, pp. 39-42, 2004. [4] V-REP simulator: http://www.coppeliarobotics.com

[6] Abdulghani, M. M., Al-Aubidy, K.M. "Design and evaluation of a MIMO ANFIS using MATLAB and VREP" 11th International Conference on Advances in Computing, Control, and Telecommunication Technologies, ACT 2020; Virtual, online; 28 August 2020 through 29 August 2020, ISBN: 978-171381851-9.

[7] Spica R. Interfacing Matlab/Simulink with V-REP for an Easy Development of Sensor-Based Control Algorithms for Robotic Platforms. Lagadic group Inria Rennes Bretagne Atlantique & IRISA; 2014; p. 1-10.

[9] S. K. Thiem, S. Stark, D. Tanneberg, J. Peters, E. Rueckert ."Simulation of the underactuated Sake Robotics Gripper in V-REP", 17th International Conference on Humanoid Robotics (IEEE-RAS, Humanoids 2017).

[10] Bae J.H and A.H. Kim, 2014. Design and Development of Unity 3D Game Engine-Based Smart SNG (Social Network Game). International Journal of Multimedia and Ubiquitous Engineering. Vol. 9, No. 8 (2014), p.p 261-266.

[11] Abdulghani M. Abdulghani, Mokhles M. Abdulghani, Wilbur L. Walters, and Khalid H. Abed, "AI Safety Approach for Minimizing Collisions in Autonomous Navigation", Tech Science Press Journal on Artificial Intelligence (Accepted on April 10, 2023).

[12] Christopher C. Rosser, Wilbur L. Walters, Abdulghani M. Abdulghani, Mokhles M. Abdulghani, and Khalid H. Abed, "Implementation of Strangely Behaving Intelligent Agents to Determine Human Intervention During Reinforcement Learning", Tech Science Press Journal on Artificial Intelligence (Accepted on March 21, 2023).

[13] Mokhles M. Abdulghani, Abdulghani M. Abdulghani, Wilbur L. Walters, Khalid H. Abed, "Obstacles Avoidance Using Reinforcement Learning for Safe Autonomous Navigation" The 25th International Conference on Artificial Intelligence (ICAI'23), Las Vegas, Nevada, July 24-27, USA, 2023. (Accepted on May 19, 2023).

[14] M. M. Abdulghani, A. A. Harden and K. H. Abed, "A Drone Flight Control Using Brain Computer Interface and Artificial Intelligence," 2022 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2022, pp. 246-250, doi: 10.1109/CSCI58124.2022.00047.