e-ISSN: 2550-164X

Available online at
https://jmeche.uitm.edu.my/

Journal of Mechanical Engineering 22(2) 2025, 47 – 57.

Journal of Mechanical Engineering

Augmented Reality Monitoring System for Cross-Belt Conveyor in Advanced Automation Line

Sukarnur Che Abdullah[1], Mohamad Dzulhelmy Amari[1*], Mohammad Azzeim Mat Jusoh[1], Irfan Syahmi Rusyaidi Amirudin[1] and Ashraf Zainalabidin[1]

[1]*School of Mechanical Engineering, College of Engineering, Universiti Teknologi MARA, 40450, Shah Alam, Selangor, Malaysia*

ARTICLE INFO

Article history:
Received 22 August 2024
Revised 3 November 2024
Accepted 25 November 2024
Online first
Published 15 May 2025

Keywords:
Augmented reality
Conveyor
Simulation
Manufacturing

DOI:
https://doi.org/10.24191/jmeche.v22i2.2933

ABSTRACT

Remote monitoring systems are increasingly adopted as one of the control strategies in contemporary industrial operations. However, the integration of Augmented Reality (AR) within industrial applications remains limited due to a deficit in comprehensive research. In alignment with Industry 4.0 principles, the deployment of AR in automation introduces a spectrum of possibilities for manufacturing sectors, particularly in critical production processes like material handling. This motivates our initiative to develop a simulation of a cross-belt conveyor system. The algorithms for the conveyor system were implemented using C# within Visual Studio and Unity environments. The algorithm's efficacy was validated through its application to a basic material handling process, specifically a barcode sorting conveyor. This project was realized by embedding AR monitoring capabilities within the conveyor system, facilitating real-time simulation synchronisation between a virtual conveyor model and its counterpart. Data transmission was orchestrated via a Python script, sensor_app.py' which communicated the data to a dedicated web interface. An Android application, "AR Simulation" was developed to deliver AR-based simulation and visualization. Within the application interface, the quantities of items sorted into respective containers were displayed dynamically. The monitoring algorithm's functionality was verified by deploying the AR Simulation app and evaluating its performance. Moreover, the system periodically logs data into a CSV file for archival and analytical purposes. The proposed system is also designed to support a web-based interface, enabling remote access to the data across multiple devices via wireless networks. The findings indicate that the AR Simulation application is a robust tool, with AR's interactive features effectively rendering data under the defined algorithms. This research enhances the current understanding of AR technology's role in industrial applications, specifically in the context of remote monitoring within manufacturing environments.

[1*] Corresponding author. *E-mail address*: mdzulhelmy@uitm.edu.my
https://doi.org/10.24191/jmeche.v22i2.2933

INTRODUCTION

A study conducted by Carmigniani (2011) explained that AR technology offers enrichment to the interaction between objects and computers. Even though people began exploring AR in the early 2000s, the technology was still in its early stages because its purpose was not clear at that time (Carmigniani et al., 2011). In accordance, limited integration of AR technology in industrial applications, specifically within the context of remote monitoring in advanced automation lines. The challenges include the lack of comprehensive research on AR's industrial applications, the complex integration of AR systems and the high cost associated with AR technology, which has made its implementation less common in industries like manufacturing. Advancements in pursuing a fully automated production line by using robotics and IoT systems create a window of opportunity for AR monitoring systems to be implemented in the automation line. The material handling system of the production line is vital in transporting items with the use of industrial conveyor belts. Among a variety of conveyor belt systems, cross-belt conveyors are widely popular among parcel delivery companies. In this project, a simple cross-belt conveyor prototype will be used to move wooden boxes with barcodes designated into three different containers. First, the barcode will be scanned, and its identity will be designated for which container it will drop into. At the same time, the data will be shared into an application software via a cloud server. Accessing the application allows an AR visual of the prototype to be viewed and interacted with. A clear advantage of such systems allows operators to better understand the production line, and emergencies can be handled more effectively when the operators are far from the site. In addition, the efficiency of the production line would also increase as operators have easy access to the site with an AR monitoring system.

Recent development in IR4.0 shows the promising application of an IoT control system that handles the communication between smart sensors and the introduction of Extended Reality (XR) technology, which mainly revolves around Virtual Reality (VR) and Augmented Reality (AR) into on-site applications. The increase in data transfer of sensors should be controlled efficiently into an easy-to-digest form when it reaches the operators of the production line (Kim et al., 2016). Rather than searching through a large collection of cluttered data, it is more effective to implement AR technology (Tezer et al., 2019). AR technology, with the aid of a production algorithm, can visualize important data for operators (Amin et al., 2015). However, there have only been a few AR application that allows remote monitoring. AR monitoring Apps can visualize data much more easily with more appealing and interactive graphics, and operators can manage the production line even when they are not on site. This allows for quicker and more precise management of the production line (Merz et al., 2020). IoT monitoring is still implemented widely in manufacturing industries because there is a solid project foundation in that technology, whereas AR technology has only recently been developed. This naturally led to AR technology being an expensive option. Hence, superior implementation in this research provides an interactive, immersive AR experience for operators, which allows them to monitor, visualize, and control the material handling process remotely, even when they are not physically present. This includes remote monitoring where AR offers the ability for off-site operators to not only monitor but also interact with the system visually (Wang et al., 2020).

There have been very few studies on the application of AR technology in the manufacturing industries. According to Chen (2019), the advancement in display devices offers a great medium to promote AR advancements. Nonetheless, other than technical hurdles, AR technology itself suffers from rapid growth, which does not cover all industries, mainly concerning this project, which is the manufactory industry. Although AR technology has made great progress in the past 20 years, many technical problems still exist (Chen et al., 2019). One of the main reasons why AR technology has not seen implementations in the manufacturing industries is that companies would require a large sum of funds for projects and development, leaving small and medium industries behind, as said by Bottani & Vignali (2019). To this end, the development of the AR solution should be regarded as an investment, meaning that once implemented, its usage will continue for some years; hence, investment evaluation is probably the most appropriate approach to assess whether and how fast the invested funds return (Bottani & Vignali, 2019).

https://doi.org/10.24191/jmeche.v22i2.2933

Furthermore, Tezer et al. (2019) mentioned that there were 343 project publications in 2019 on AR technology, which is more than a 50% increase from 2018, which mainly highlights AR implementation in education. The main reason why the issue is being considered is the integration of AR applications with regard to education (Tezer et al., 2019). This brings into this project the realization to implement AR technology into an automated production line, specifically a material handling system. A prior project conducted by Jaafar et al. (2021 and 2023) and Rajalah et al. (2023) in developing an algorithm for the monitoring system of an AR-IoT application, "AR-Simulasi", came into several limitations, such as the lack of interactive features that an AR technology should have (Jaafar et al., 2021; Rajalah et al., 2023; Jaafar et al., 2023).

As there is a lack of research conducted in the area of AR technology as a monitoring system in an advanced automation line, this shows that there are research gaps to conduct a proper implementation of AR technology (Ko et al., 2013; Wang et al., 2010; Linowes & Babilinski, 2017). Some of the constraints or gaps faced by industries are due to integration complexities, precision and data security concerns, which limit the full potential of AR technology. A few approaches that can be implemented, such as strategic planning, technological expertise, and comprehensive user training, can pave the way for successful implementation and foster a transformative impact on modern manufacturing that may enhance the process involved.

Future research could conduct surveys on why AR developers are not keen on entering the manufacturing industry. Such a project can create more awareness towards tech companies to invest more labour and resources in the manufacturing industries, which leads to more competitive pricing that companies would be willing to invest in. Therefore, this project aims to explore the potential of an AR application of the cross-belt conveyor simulation with real-time monitoring through a Python-built data acquisition system.

METHODOLOGY

Development of Cross-Belt Conveyor System Simulation

In this project, the first approach is to design the cross-belt conveyor system components; the conveyor, the base frame that houses other components and sensors and lastly the storage box that will collect the objects to be sorted. By using CATIA V5 to design the components, we can obtain a precise model with engineering standards. By implementing the reverse engineering method on an existing full-scale cross-belt conveyor system, a prototype of the system was designed as shown in Fig 1. The conveyor system consists of a cross-belt conveyor, which transports objects on it and a base frame that houses mechanical components that powers the conveyor belt system.

The CAD design is then imported into Blender where we will change the format from .stl to .obj which then will be used inside Unity 3D, shown in Fig 2. For this project, Unity 3D is selected as the development software to develop both simulation and AR applications with C# coding language. As mentioned by Buyuksalih et al. (2017), Unity 3D provides three-dimensional manipulation and simulation through functions defined using programming languages (Buyuksalih et al., 2017).

https://doi.org/10.24191/jmeche.v22i2.2933

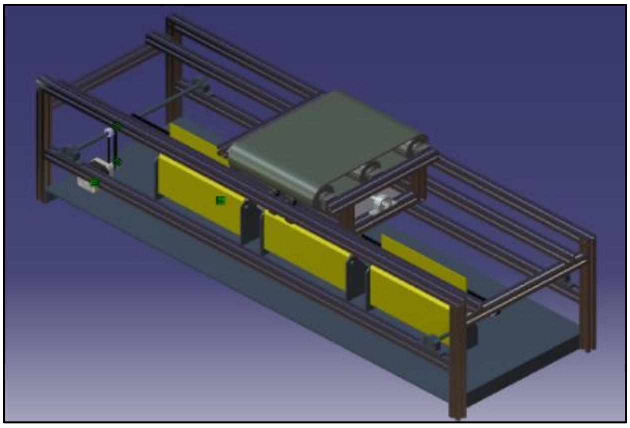

Fig. 1. Cross-belt conveyor system 3D CAD model [Scale 1:2].

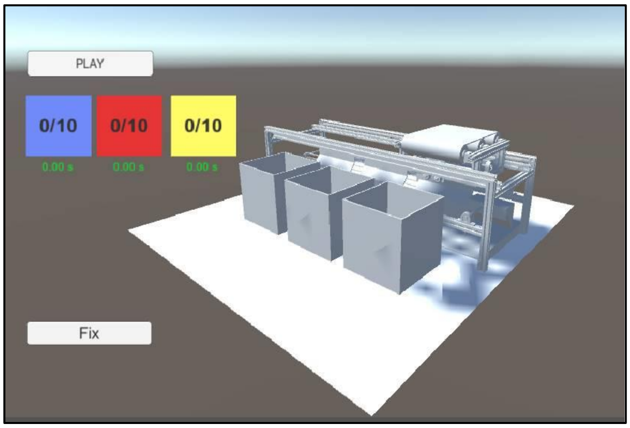

Fig. 2. Full assembly of components in Unity 3D.

Development of Sorting Algorithm

The algorithm needed for the sorting system was written in Visual Studio. Fig 3 shows the sorting algorithm functions as such; firstly, an object with randomized colour from three predetermined colours, which are blue, red and yellow, will be dropped down into the conveyor. Then a sensor will check its iteration, timestamp, material (in this case, its colour) and defect. After that, the conveyor will move the object into its respective storage box colour in the order of blue, red and yellow. Another sensor will detect if the object enters the storage box and update the counter. If an object with a defect enters the box, the box will change colour to red and notify that a defect has entered the box. The user would then need to press a fix button that will remove the defect from the box and update the counter. In the case that the counter reaches 10 (pre-determined target counter), the box will turn green to notify the user and execute the next process for the object. In the actual working environment, the objects will be transferred to other workstations, either to the rework station for defective objects or the next station for non-defective objects.

https://doi.org/10.24191/jmeche.v22i2.2933

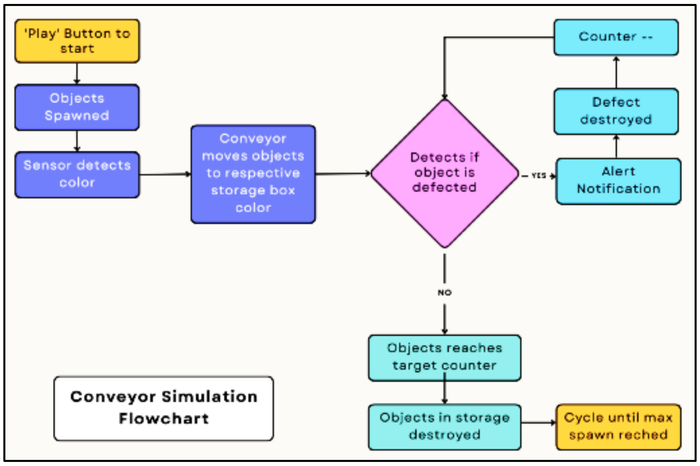

Fig. 3. Flowchart of the sorting algorithm.

Development AR Simulation

The main components to create a simulation inside Unity 3D are models that will be visualized, scripts that contain algorithms and Unity elements that calculate the physics of material and a Graphical User Interface (GUI) shown in Fig 4.

The GUI displays the set of buttons that will play the simulation and apply the fix function. Then there are three coloured panels that represent the boxes that will contain the coloured parcel, from the right side to the left, blue, red and yellow. The numbers inside the panels represent the number of parcels inside the box over the target number pre-determined for the box to hold. Below the panels are the timestamps for each respective storage box.

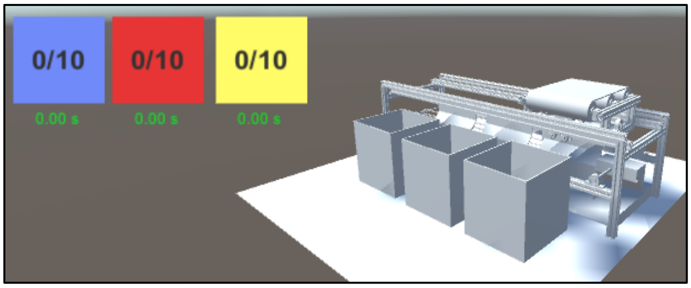

Fig. 4. Simulation scene with GUI setup.

Development of AR Application

Using geometrical representation of 3D objects, we can manipulate their position and orientation, matching them to their counterparts in the field of view (Mekni et al., 2014). Hence, using the model as a guide, we need to project to entire model through the smartphone camera where the surrounding data is collected and then transmitted by overlapping the model onto the environment data. The development of AR applications requires the installation of an Android Software Development Kit (SDK) to build the application for Android devices. Then it needs to be accompanied by ARCore by Google and Unity AR Foundation with AR Tracking. By installing the essential AR SDK and plugins, we can develop AR

https://doi.org/10.24191/jmeche.v22i2.2933

functionality. In addition, the ‘Lean Touch’ plugin is installed so that the user can apply translation and rotation to the models from the previous simulation.

Fig 5 shows the simulation being projected onto the real-world environment that can be seen on the device’s screen. The AR Simulation Application functions as intended with its AR features, which enable an immersive experience.

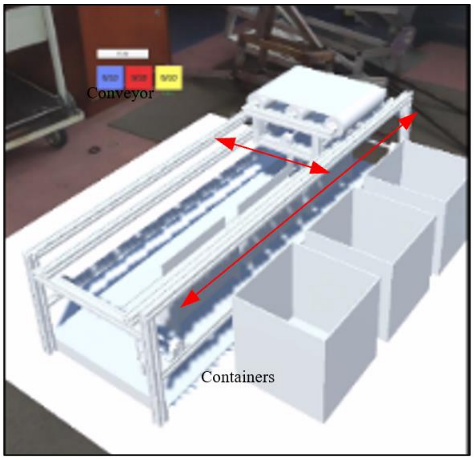

Fig. 5. AR environment in AR application ‘AR Simulation’.

Data Transfer from Simulation to Website using Python

When the simulation is set to run, scripts from the simulation will generate .csv file that is edited in real time and by allowing a separate Python script to access the .csv file, we can have a simple data acquisition function for this system (Hughes, 2010). The data acquired by the data acquisition system is accessible in real time through a web browser. The system hosts a website in Python using the Flask library (Emilio, 2013). A client web browser connecting to the website obtains data in real-time at an interval of 1 second, but can be adjusted. To connect to the website, type the system’s IP address into the address bar, for example, in this case, the system’s IP address is 192.168.0.155:2525. The website can only be accessed if the client's computer is connected to the same local area network as the system. Otherwise, both the system and the client computer must first establish a connection with a virtual private network program such as LogMeIn Hamachi for the system’s website to be accessible to the client computer from a different network, as shown in Fig 6.

The background_thread() function runs in a separate thread and continuously generates random sensor values every second. It also reads data from a CSV file using the read_csv(). The updated sensor data, along with the current time, is sent to the client using socketio.emit() with the event name 'updateSensorData.

https://doi.org/10.24191/jmeche.v22i2.2933

```python
def get_readings():
    reading = random.randint(0, 5)
    return reading

def read_csv():
    csv_path = "C:/Users/NITRO/Desktop/FYP 2/Unity/Aplha test/exported_data(1).csv"
    data = pandas.read_csv(csv_path)
    material_name = data["MaterialName"]
    print(material_name)

def get_current_datetime():
    now = datetime.now()
    return now.strftime("%H:%M:%S")


def background_thread():
    print("Generating random sensor values")
    while True:
        current_time = get_current_datetime()

        for chart in charts:
            read_csv()
            charts[chart]["reading"] = get_readings()

        socketio.emit('updateSensorData', {'value': charts, "date": current_time})

        socketio.sleep(1)
```

Fig. 6. Python coding of the data acquisition system of the simulation.

RESULTS AND DISCUSSION

To test the credibility of the simulation, 10 sets of test run were conducted with the following parameters as shown in Table 1.

Table 1. Simulation parameters

Maximum number of spawn	30
The target number of objects inside the storage box	10 each
Material of objects	Blue, red, yellow
Delay between spawn	3 seconds
Defect object spawning probability	10%

The data acquisition of the data was carried out by a script, 'sensor_app.py' that collects the data in real-time into a website that collects the data transmitted and converts it into a graphical representation in Fig 7.

The graph shows that parcel object generation is updated in real time by allowing a Python script to read the data collected during the simulation. There are three-line graphs plotted, each to count their respective material colour. The simulation starts with no materials, hence why all three plots start at 0. But over time after the material is detected and moved into its container, it updates its data file to increment the material count by colour. Thus, the graph's increments in material count in its plot as the data file gets updated until the simulation ends after all 30 materials have been sorted, in this case, 11 of them are red, 10 of them are yellow, and 9 are blue.

The probability of defect spawned is set to 0.1 with the frequency of one in 10 sets of simulation cycles. We need to evaluate the significance of the proposed defect rate. The defect rate refers to the actual frequency of defect occurrences observed across multiple simulation cycles. In this context, the defect rate is evaluated by comparing the observed frequencies of defect occurrences with the expected frequency, which is based on the specified defect probability of 0.1. Nevertheless, the defect probability is defined as

https://doi.org/10.24191/jmeche.v22i2.2933

the likelihood that a defect will occur during a single simulation cycle which is predetermined to be 0.1. This implies that each simulation cycle has a 10% probability of generating a defect.

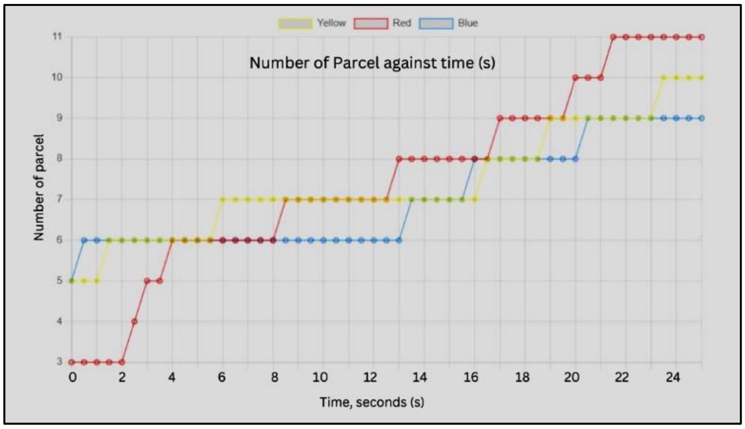

Fig. 7. Data acquisition of simulation.

A statistical analysis using the chi-squared test for goodness of fit compares the observed frequencies of defect values with the expected frequencies based on the pre-determined defect rate. Under the hypotheses of: Null Hypothesis (HO) and Alternative Hypothesis (Ha), where HO means that the observed defect percentages follow the expected defect rate of 0.1. While Ha represents that, the observed defect percentages do not follow the expected defect rate of 0.1. The significance level was set to 0.05, which we compared with the value of chi-squared from this simulation.

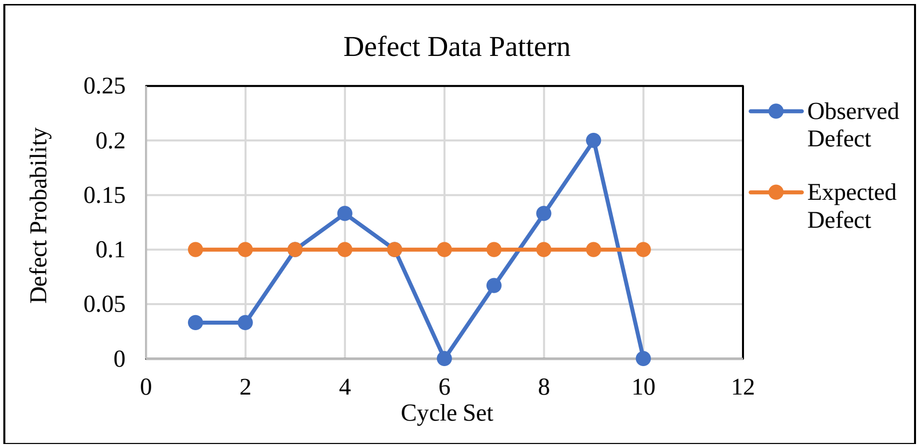

Fig. 8. Defect data pattern.

Fig 8 shows that the defect for each cycle barely aligns with the expected defect of 0.1. The highest value of defect probability can be observed in the ninth cycle while the lowest defect probability of zero has a frequency of 2 in the sixth and tenth cycle.

https://doi.org/10.24191/jmeche.v22i2.2933

Table 2. Chi-squared fitness analysis for defect spawn in simulation

Cycle set	Observed defect probability	Frequency of defect	Actual defect squared	Expected defect probability	Frequency of expected defect	Chi-squared (X^2)
1	0.033	0	0.0011	0.1	1	1
2	0.033	0	0.0011	0.1	1	1
3	0.1	1	0.01	0.1	1	0
4	0.133	1	0.0177	0.1	1	0
5	0.1	1	0.01	0.1	1	0
6	0	0	0	0.1	1	1
7	0.067	1	0.0045	0.1	1	0
8	0.133	1	0.0177	0.1	1	0
9	0.2	2	0.04	0.1	1	1
10	0	0	0	0.1	1	1
					Total	5

The calculation of the P-value with chi-squared requires the value of the degree of freedom, which is the value of the cycle minus one. Then the calculation also requires the value of chi-squared exhibited in Table 2. The observed defect probability, which is less than 0.1, will be considered as ‘0’ frequency, and the observed value from 0.1 until less than 0.2 will be considered as ‘1’, where 0.2 will be ‘2’.

$$x^2 = \sum \frac{(Observed - Expected)^2}{Expected} \tag{1}$$

By using the chi-squared formula in Equation 1, where chi-square is the summation of the squared difference between observed minus expected frequencies over the expected. The final value of the chi-squared was obtained, that is, five.

Then, using the ‘P-value Calculator for Chi-Square Distribution’ developed by the University of Illinois, Department of Statistics. We obtained a right-tailed p-value of 0.8343, which is higher than the significance level. Hence, the result is not statistically significant as there is not enough evidence to reject the null hypothesis, suggesting that there is not enough evidence to conclude a significant difference. Moreover, with a small sample size, random fluctuations can make it harder to detect statistically significant differences. Increasing the number of cycles, such as from 10 to 100, will give more robust data. A larger dataset provides more power to detect small deviations from the expected defect rate.

The analysis can be interpreted as such; insufficient sample size is considered small for this scale of analysis, as shown by the huge margin between the p-value and the significance level, 0.05. Another interpretation is that the HO holds or is likely true that the defect percentage of the simulation is 0.1. With no proper conclusion that can be drawn, collecting a larger dataset can improve this analysis, providing better results. For example, increasing the total cycle to 100 and the maximum number of spawns to 100 from 30.

CONCLUSIONS

A cross-belt conveyor system simulation and the AR application, AR Simulation, of the system were developed. Besides that, the data transfer process of the simulation with Python from the system to the website was verified. The statistical analysis of the data obtained from the system was carried out by determining the fitness test of the simulation’s defect probability, but we did not obtain any concrete evidence to support any conclusion. On the other hand, the analysis does not rule out the credibility of the

https://doi.org/10.24191/jmeche.v22i2.2933

simulation itself. Hence, the analysis result for this scale of project is acceptable regardless. It is also to be noted that the result can be evaluated by increasing the sample size.

This project intends to contribute to areas related to AR simulation development where off-site monitoring devices are lacking. Thus, this project on AR simulation will help future studies and analyses to improve and determine the prospects of AR as a future technology for various applications, especially in the manufacturing industry. In addition, other approaches can be used to improve the result, such as applying a Monte Carlo simulation.

ACKNOWLEDGEMENT/ FUNDING

The authors would like to express sincere appreciation to the College of Engineering, Universiti Teknologi MARA (UiTM), for the support provided throughout this project. The availability of research facilities, technical tools, and a conducive academic environment greatly contributed to the successful completion of this work. The authors also acknowledge the assistance and cooperation of the academic and technical staff involved.

CONFLICT OF INTEREST STATEMENT

The authors agree that this research was conducted in the absence of any self-benefits, commercial or financial conflicts and declare the absence of conflicting interests with the funders.

AUTHORS' CONTRIBUTIONS

The authors collectively contributed to the planning, execution, and completion of this research. This includes the development of the research framework, design of the methodology, data acquisition, analysis, and interpretation of findings. The authors also collaborated in writing, reviewing, and refining the manuscript to ensure it accurately reflects the research outcomes.

REFERENCES

Amin, D., & Govilkar, S. (2015). Comparative study of augmented reality SDKs. International Journal on Computational Science & Applications, 5(1), 11-26.

Bottani, E., & Vignali, G. (2019). Augmented reality technology in the manufacturing industry: A review of the last decade. Institute of Industrial and Systems Engineers (IISE) Transactions, 51(3), 284-310.

Buyuksalih, I., Bayburt, S., Buyuksalih, G., Baskaraca, A. P., Karim, H., & Rahman, A. A. (2017). 3D modelling and visualization based on the unity game engine–advantages and challenges. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 4, 161-166.

Carmigniani, J., Furht, B., Anisetti, M., Ceravolo, P., Damiani, E., & Ivkovic, M. (2011). Augmented reality technologies, systems and applications. Multimedia Tools and Applications, 51, 341-377.

Chen, Y., Wang, Q., Chen, H., Song, X., Tang, H., & Tian, M. (2019). An overview of augmented reality technology. Journal of Physics: Conference Series (p. 022082). IOP Publishing.

https://doi.org/10.24191/jmeche.v22i2.2933

Emilio, M. D. P. (2013). Data acquisition systems: From fundamentals to applied design. Springer Science & Business Media.

Hughes, J. M. (2010). Real world instrumentation with Python: Automated data acquisition and control systems. O'Reilly Media, Inc.

Jaafar, S. A., Abdullah, S. C., Jusoh, M. A. M., Azmat, F., & Jaffar, A. (2021). AR simulasi: An augmented reality real-time cloud-based simulation for off-site monitoring in industrial manufacturing application. International Transaction Journal of Engineering, Management, & Applied Sciences & Technologies, 12(9), 1-9.

Jaafar, S. A., Che Abdullah, S., Mat Jusoh, M. A., Rajalah, H. H., & Muhamad Affandi, M. A. H. (2023). Algorithm of remote AR-Monitoring 3D visualization simulation. AIP Conference Proceedings (p. 050003). AIP Publishing.

Ko, S. M., Chang, W. S., & Ji, Y. G. (2013). Usability principles for augmented reality applications in a smartphone environment. International Journal of Human-Computer Interaction, 29(8), 501-515.

Linowes, J., & Babilinski, K. (2017). Augmented reality for developers: Build practical augmented reality applications with Unity, ARCore, ARKit, and Vuforia. Packt Publishing Ltd.

Mekni, M., & Lemieux, A. (2014). Augmented reality: Applications, challenges and future trends. Proceedings of the 13th International Conference on Applied Computer and Applied Computational Science (pp. 205-214). World Scientific and Engineering Academy and Society Press.

Merz, R., Hoch, R., & Drexel, D. (2020). A cloud-based research and learning factory for industrial production. Procedia Manufacturing, 45, 215–221.

Rajalah, H. H., Mat Jusoh, M. A., Jaafar, S. A., & Che Abdullah, S. (2023). Optimization of the material handling system in adapting the real-time AR simulation. AIP Conference Proceedings (p. 050009). AIP Publishing.

Tezer, M., Yıldız, E. P., Masalimova, A. R., Fatkhutdinova, A. M., Zheltukhina, M. R., & Khairullina, E. R. (2019). Trends of augmented reality applications and project throughout the world: Meta-analysis of theses, articles and papers between 2001-2019 years. International Journal of Emerging Technologies in Learning (iJET), 14(22), 154-174.

Wang, R. X., Wang, R., Fu, P., & Zhang, J. M. (2020). Portable interactive visualization of large-scale simulations in geotechnical engineering using Unity3D. Advances in Engineering Software, 148, 102838.

Wang, S., Mao, Z., Zeng, C., Gong, H., Li, S., & Chen, B. (2010). A new method of virtual reality based on Unity3D. 18th International Conference on Geoinformatics Proceedings (pp. 1-5). IEEE Publisher.

https://doi.org/10.24191/jmeche.v22i2.2933