

Digital Twin-Based Smart Renewable Energy System Using AI and IoT

Manasvi Bisht¹, Vidit Bhambhani¹, Tushar Koundal¹, Dr. Neha Gupta²

¹ Student, School of Engineering & Technology, Sushant University, Gurugram, India

² Professor, School of Engineering & Technology, Sushant University, Gurugram, India

ABSTRACT

This study presents the development of a vision-based, dual-axis Smart Renewable Energy System integrating Computer Vision, the Internet of Things (IoT), and Digital Twin technology. Traditional solar tracking mechanisms often rely on Light Dependent Resistors (LDRs), which are highly susceptible to environmental noise such as cloud cover and reflections [3]. To address this limitation, our proposed system employs an OpenCV-based algorithm to dynamically locate the highest sunlight intensity centroid, driving a dual-axis pan-tilt mechanism via an Arduino Uno [2], [4]. The system actively monitors real-time 5V DC power generation metrics using an INA219 I2C sensor [14]. Furthermore, to solve the industry challenge of remote physical monitoring, a web-based Digital Twin was constructed using Three.js, rendering a live, synchronized 3D replica of the physical tracker [6], [7]. Experimental A/B testing demonstrated that the computer vision-driven dual-axis approach yielded significantly more stable voltage output compared to a stationary setup. This architecture provides a scalable, cost-effective framework for intelligent, remotely monitored renewable energy infrastructures.

Keywords: Solar Tracking; Arduino Uno; SG90 Servo Motor; Digital Twin; IoT; Renewable Energy.

1. INTRODUCTION

The rapid global shift toward sustainable energy systems has significantly increased the importance of renewable energy technologies, particularly solar power, due to rising energy demands and environmental concerns [1]. Among various solar energy optimization methods, Arduino-based solar tracking systems have gained attention for their affordability, automation, and ability to improve photovoltaic efficiency by adjusting panel orientation according to sunlight direction [2]. Traditional solar tracking mechanisms, including sensor-based and mechanical sun-tracking approaches, have been widely studied for maximizing solar energy capture, though many conventional systems face limitations related to environmental interference, lower precision, and maintenance complexity [3]. To overcome some of these issues, advanced Arduino-integrated solar tracking techniques have been developed, offering improved control and operational flexibility in small-scale renewable applications [4]. Simultaneously, the Internet of Things (IoT) has transformed renewable energy systems by enabling real-time data acquisition, communication, and intelligent monitoring of system performance through interconnected devices and sensors [5]. Building upon IoT capabilities, Digital Twin technology has emerged as a powerful innovation that creates synchronized virtual replicas of physical systems, allowing remote observation, predictive analysis, and enhanced operational control [6]. Recent advancements in Arduino automation, electronics prototyping, and practical renewable system implementation have further supported the integration of these technologies into efficient smart energy infrastructures [7], [8], [9]. Additionally, photovoltaic engineering principles emphasize that continuous solar alignment can substantially increase energy generation efficiency compared to fixed-panel systems [10]. Therefore, integrating solar tracking, IoT, and Digital Twin technologies provides a strong foundation for developing intelligent, adaptive, and cost-effective renewable energy systems capable of meeting modern sustainability requirements.

2. LITERATURE REVIEW

2.1 Evolution of Digital Twin in Renewable Energy Systems

Digital Twin (DT) technology has emerged as a significant concept within Industry 4.0, enabling the development of virtual replicas of physical systems for real-time monitoring, analysis, and control [11]. In the context of renewable energy systems, Digital Twins have evolved from basic simulation models to advanced data-driven frameworks that continuously update using real-time data acquired from physical assets [12]. This evolution is primarily driven by the growing need for higher system efficiency, predictive maintenance, and optimized lifecycle management in modern energy infrastructures [13]. Furthermore, the increasing global demand for sustainable and intelligent energy solutions has accelerated the adoption of Digital Twin technology in renewable energy applications, making it a critical component in improving system performance and reliability [14],[15].

2.2 Role of IoT and Sensor Technologies in Digital Twin Systems

The Internet of Things (IoT) plays a fundamental role in enabling the effective operation of Digital Twin systems by facilitating real-time data acquisition and communication between physical and virtual components [16]. IoT-enabled sensors are responsible for collecting essential parameters such as temperature, voltage, current, and environmental conditions, which are continuously transmitted to the Digital Twin for analysis and decision-making [17]. This seamless data exchange ensures accurate synchronization and enhances system responsiveness [18]. Although traditional solar tracking systems have relied on sensor-based approaches using platforms like Arduino, these methods often suffer from limited accuracy and adaptability under dynamic environmental conditions [19]. Therefore, the integration of IoT technologies not only improves data reliability but also enables intelligent energy management, making Digital Twin-based systems more efficient and scalable [20].

2.3 Integration of Machine Learning in Digital Twin-Based Energy Systems

The solar panel in this system can be more accurately aligned with the direction of maximum sunlight by using machine learning algorithms to understand the visual input recorded by the camera [21]. This enhances the tracking mechanism's reactivity in a range of environmental circumstances. Simultaneously, ongoing monitoring of electrical characteristics like voltage and current provide valuable information about the system's performance over time [22]. These insights facilitate the understanding of performance patterns and guarantee that the system functions steadily and effectively even in the face of changing situations [23].

2.4 Challenges and Future Scope of Digital Twin-Based Smart Energy Systems

Digital Twin technology has many benefits, but using it in renewable energy systems is not without its problems [24]. These include high computational needs, problems with data management, and worries about security [25]. Another big problem is making sure that physical and virtual systems stay in sync in real time [26]. But new technologies in the Internet of Things (IoT) are likely to solve these problems [27]. Future research aims to create more scalable, secure, and intelligent Digital Twin systems capable of supporting fully autonomous renewable energy infrastructures in accordance with global sustainability objectives [28], [29], [30].

3.METHODOLOGY

3.1 System Working and Data Collection Process

1) First, the system begins by using a camera that is connected via DroidCam to record real-time ambient information. The processing unit receives live images from the camera continually. Python and the OpenCV package are used to process these pictures in order to analyse light intensity. The brightest area in the frame is identified using image processing techniques. The direction of the strongest sunlight is shown by this area. This approach offers greater accuracy and minimizes mistakes brought on by shadows or reflections, in contrast to conventional LDR-based systems. For additional control operations, the processed output is transformed into directional data. The system can continuously adjust to shifting solar circumstances thanks to real-time processing. This enhances the solar tracking system's responsiveness and effectiveness. The entire performance is improved by integrating computer vision [2], [4].



**Fig. 3.1(a) Uno R3 CH340G (ATmega328P)
Arduino-Compatible Board**



Fig. 3.1(b) TowerPro SG90 Micro Servo Motor

2) The Arduino Uno, which serves as the system's central control unit, receives the processed data after that. The Arduino decides the best direction for maximum sunlight intensity by interpreting the input it receives from the image processing module. It uses its digital output pins to produce suitable Pulse Width Modulation (PWM) signals based on this analysis. The TowerPro SG90 servo motor receives these PWM impulses and rotates at a predetermined angle in response. In order to ensure effective solar energy capture, the servo motor perfectly aligns the attached tiny solar panel with the direction of maximum light intensity [13],[15].



Fig. 3.1(c) 6V / 100mA Mini Solar Panel

3) The 6V/100mA mini solar panel is mounted on the servo motor. As the servo rotates, the panel aligns with the sun, ensuring maximum energy absorption throughout the day [22].



Fig. 3.1(d) INA219 I2C Current & Power Monitoring Module



Fig. 3.1(e) MB102 830-Point Solderless Prototype Breadboard

4) While the panel is generating power, the INA219 current and voltage sensor continuously monitors the electrical output (voltage and current). This sensor communicates with the Arduino using the I2C protocol and sends real-time data [14].



Fig. 3.1(f) Male-to-Female Jumper Wires



Fig. 3.1(g) Male-to-Male Jumper Wires

5) All the components are interconnected using jumper wires and mounted on the MB102 breadboard, which provides a stable and solderless platform for circuit connections. The entire setup is fixed on a sunboard sheet, giving structural support to the model [17].



Fig. 3.1(h) Soldering Iron Kit



Fig. 3.1(I) Sunboard Sheet (Structural Mounting Substrate)

6) A soldering iron kit is used to make permanent electrical connections where needed [19]. Super glue helps fix components like the servo motor and solar panel onto the sunboard, ensuring everything stays secure during movement.

7) All of the system's components are mounted on the sunboard sheet. It offers a sturdy, lightweight, and manageable substrate for hardware assembly. During operation, its stiff structure guarantees correct alignment and support.

8) The Arduino transfers the collected sensor data to the computer via a USB cable using serial communication. Python (with PySerial) receives this data and processes it further [18].

9) The processed data is then visualized using Matplotlib and simultaneously sent to a Flask-based web server. The Digital Twin, built using Three.js, displays a real-time virtual model of the system, showing the movement of the solar panel and live performance data.

10) Additionally, Scikit-learn is used to apply AI algorithms that compare actual sensor data with predicted values. This helps in detecting faults such as misalignment, shading, or dust accumulation.

11) Thus, all hardware components (Arduino, servo motor, solar panel, INA219, breadboard, wires) and software components (Python, OpenCV, Flask, AI models, Digital Twin) work together in a continuous loop to ensure efficient solar tracking, monitoring, and optimization.

12) The system uses light detection to determine the direction of maximum sunlight. Based on this input, the Arduino controller sends signals to the servo motor, which adjusts the position of the solar panel. This ensures that the panel remains aligned with the sun throughout the day, thereby improving overall efficiency [22].

3.2 Software Components

This section describes the software that is used for system control, data communication, image processing, and visualization. These parts work together to let you monitor, analyse, and run the system in real time.

S. No.	Software/Tool	Type	Function
1	Python 3.x	Programming Language	Core application layer for all logic and communication
2	OpenCV	Python Library	Computer vision, image capture, Gaussian blur, and light detection
3	PySerial	Python Library	Serial communication between Python and Arduino over USB
4	Scikit-Learn	Python Library	Machine learning model for power prediction and fault detection
5	Matplotlib	Python Library	Data visualization and A/B test comparison graphs
6	Flask	Python Library	Backend web server connecting hardware data with the dashboard
7	Three.js	JavaScript Library	3D rendering of the Digital Twin model in the web browser
8	HTML & CSS	Web Technologies	Dashboard interface design and layout
9	Arduino IDE	Development Environment	Firmware programming for the Arduino microcontroller in C++
10	DroidCam	Mobile/Desktop App	Streams mobile phone camera as a live webcam to Python

3.3 Block Diagram

The block diagram below illustrates the end-to-end data and control flow of the proposed Digital Twin-Based Smart Renewable Energy System, from physical sensing to cloud-based AI analysis.

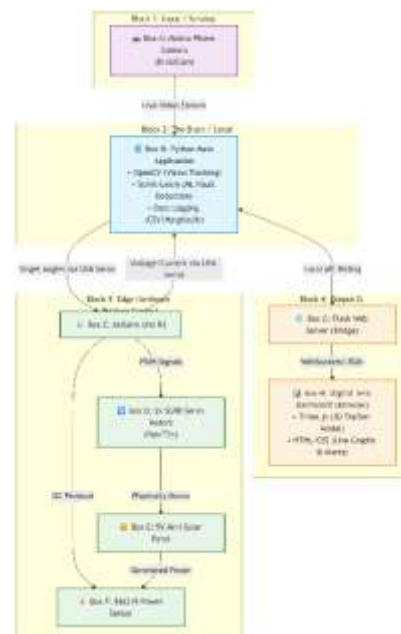


Fig. 3.3 Block Diagram of the Digital Twin-Based Smart Renewable Energy System


```

connectorino
1 #include <Servo.h>
2 #include <Wire.h>
3 #include <Adafruit_INA219.h>
4
5 Servo panMotor;
6 Servo tiltMotor;
7 Adafruit_INA219 ina219;
8
9 // Starting positions
10 int panAngle = 90;
11 int tiltAngle = 90;
12
13 void setup() {
14   Serial.begin(115200);
15
16   // Wake up sensor
17   if (!ina219.begin()) {
18     Serial.println("INA219 error!");
19   }
20
21   panMotor.attach(9);
22   tiltMotor.attach(10);
23
24   panMotor.write(panAngle);
25   tiltMotor.write(tiltAngle);
26 }
27
28 void loop() {
29   // Check if Python sent a message over the USB cable
30   if (Serial.available() > 0) {
31     // Read the message until the "newline" character
32     String data = Serial.readStringUntil('\n');
33
34     // We expect Python to send data like this: "95,80" (Pan,Tilt)
35     int commaIndex = data.indexOf(',');
36
37     if (commaIndex > 0) {
38       // Split the text into two numbers
39       String panString = data.substring(0, commaIndex);
40       String tiltString = data.substring(commaIndex + 1);
41
42       panAngle = panString.toInt();
43       tiltAngle = tiltString.toInt();
44
45       // Move the motors to the new angles Python requested!
46       panMotor.write(panAngle);
47       tiltMotor.write(tiltAngle);
48
49       // Send the INA219 voltage back to Python
50       float voltage = ina219.getBusVoltage_V();
51       Serial.println(voltage);
52     }
53   }
54 }

```

Fig. 3.6 (b)Arduino Servo Control Code

```

note 1 (python)
1 import sys
2 import matplotlib.pyplot as plt
3
4 def read_voltage_data(filename):
5     voltages = []
6     try:
7         with open(filename, 'r') as file:
8             # Read the CSV file as a dictionary so we can easily grab the "voltage (V)" column
9             reader = csv.DictReader(file)
10             for row in reader:
11                 try:
12                     # Convert the text to a decimal number
13                     voltage = float(row['voltage (V)'])
14                     except ValueError:
15                         pass # Skip any empty or broken read safety
16             except FileNotFoundError:
17                 print(f"Error: Could not find '{filename}'. Check the file name!")
18             return voltages
19
20 # --- 1. LOAD THE DATA ---
21 print("Loading data from CSV files...")
22 stationary_data = read_voltage_data('stationary_test.csv')
23 tracking_data = read_voltage_data('tracking_test.csv')
24
25 # --- 2. DRAW THE GRAPH ---
26 print("Drawing the comparison graph...")
27 plt.figure(figsize=(10, 8))
28
29 # Plot the lines
30 plt.plot(stationary_data, label='Stationary Power (No AI)', color='red', linestyle='--')
31 plt.plot(tracking_data, label='Smart AI Tracker', color='green', linestyle='--')
32
33 # Add professional labels and titles
34 plt.title('Solar Power Generation: Stationary vs. AI Tracking', fontsize=14, fontweight='bold')
35 plt.xlabel('Time (Readings over 10 seconds)', fontsize=12)
36 plt.ylabel('Power Output (Voltage)', fontsize=12)
37
38 # Add a grid and a legend (the key)
39 plt.grid(True, linestyle=':', alpha=0.7)
40 plt.legend(fontsize=10, loc='lower center')
41
42 # Show the final masterpiece!
43 plt.tight_layout()
44 plt.show()

```

Fig. 3.6 (c)Performance Analysis Script

```

note 1 (python)
1 import sys
2 import csv
3 import math
4 from datetime import datetime
5
6 # --- 1. SETUP THE CSV SPECS ---
7 csv_filename = "solar_power_log.csv"
8 # Create the file and write the header row
9 with open(csv_filename, mode='w', newline='') as file:
10     writer = csv.writer(file)
11     # Write CSV header: "Time", "Pan angle", "Tilt angle", "Voltage (V)"
12     writer.writerow(["Time", "Pan angle", "Tilt angle", "Voltage (V)"])
13
14 # --- 2. GET THE CURRENT TIME ---
15 current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
16
17 # --- 3. READ THE DATA ---
18 # Read the CSV file and get the data
19 with open(csv_filename, mode='r') as file:
20     reader = csv.reader(file)
21     # Skip the header row
22     next(reader)
23     # Get the first row of data
24     first_row = next(reader)
25     # Extract the values from the first row
26     pan_angle = float(first_row[2])
27     tilt_angle = float(first_row[3])
28     voltage = float(first_row[4])
29
30 # --- 4. CALCULATE THE ANGLES ---
31 # Calculate the pan angle
32 pan_angle = pan_angle * math.pi / 180
33 # Calculate the tilt angle
34 tilt_angle = tilt_angle * math.pi / 180
35
36 # --- 5. CALCULATE THE VOLTAGE ---
37 # Calculate the voltage
38 voltage = voltage * 1000
39
40 # --- 6. WRITE THE DATA TO THE CSV FILE ---
41 # Write the data to the CSV file
42 with open(csv_filename, mode='a') as file:
43     writer = csv.writer(file)
44     # Write the data row
45     writer.writerow([current_time, pan_angle, tilt_angle, voltage])
46
47 # --- 7. PRINT THE DATA ---
48 # Print the data
49 print(f"Time: {current_time}, Pan angle: {pan_angle}, Tilt angle: {tilt_angle}, Voltage: {voltage}")
50
51 # --- 8. END ---
52
53 # --- 9. END ---
54
55 # --- 10. END ---
56
57 # --- 11. END ---
58
59 # --- 12. END ---
60
61 # --- 13. END ---
62
63 # --- 14. END ---
64
65 # --- 15. END ---
66
67 # --- 16. END ---
68
69 # --- 17. END ---
70
71 # --- 18. END ---
72
73 # --- 19. END ---
74
75 # --- 20. END ---
76
77 # --- 21. END ---
78
79 # --- 22. END ---
80
81 # --- 23. END ---
82
83 # --- 24. END ---
84
85 # --- 25. END ---
86
87 # --- 26. END ---
88
89 # --- 27. END ---
90
91 # --- 28. END ---
92
93 # --- 29. END ---
94
95 # --- 30. END ---
96
97 # --- 31. END ---
98
99 # --- 32. END ---
100

```

```

note 1 (python)
1 import sys
2 import csv
3 import math
4 from datetime import datetime
5
6 # --- 1. SETUP THE CSV SPECS ---
7 csv_filename = "solar_power_log.csv"
8 # Create the file and write the header row
9 with open(csv_filename, mode='w', newline='') as file:
10     writer = csv.writer(file)
11     # Write CSV header: "Time", "Pan angle", "Tilt angle", "Voltage (V)"
12     writer.writerow(["Time", "Pan angle", "Tilt angle", "Voltage (V)"])
13
14 # --- 2. GET THE CURRENT TIME ---
15 current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
16
17 # --- 3. READ THE DATA ---
18 # Read the CSV file and get the data
19 with open(csv_filename, mode='r') as file:
20     reader = csv.reader(file)
21     # Skip the header row
22     next(reader)
23     # Get the first row of data
24     first_row = next(reader)
25     # Extract the values from the first row
26     pan_angle = float(first_row[2])
27     tilt_angle = float(first_row[3])
28     voltage = float(first_row[4])
29
30 # --- 4. CALCULATE THE ANGLES ---
31 # Calculate the pan angle
32 pan_angle = pan_angle * math.pi / 180
33 # Calculate the tilt angle
34 tilt_angle = tilt_angle * math.pi / 180
35
36 # --- 5. CALCULATE THE VOLTAGE ---
37 # Calculate the voltage
38 voltage = voltage * 1000
39
40 # --- 6. WRITE THE DATA TO THE CSV FILE ---
41 # Write the data to the CSV file
42 with open(csv_filename, mode='a') as file:
43     writer = csv.writer(file)
44     # Write the data row
45     writer.writerow([current_time, pan_angle, tilt_angle, voltage])
46
47 # --- 7. PRINT THE DATA ---
48 # Print the data
49 print(f"Time: {current_time}, Pan angle: {pan_angle}, Tilt angle: {tilt_angle}, Voltage: {voltage}")
50
51 # --- 8. END ---
52
53 # --- 9. END ---
54
55 # --- 10. END ---
56
57 # --- 11. END ---
58
59 # --- 12. END ---
60
61 # --- 13. END ---
62
63 # --- 14. END ---
64
65 # --- 15. END ---
66
67 # --- 16. END ---
68
69 # --- 17. END ---
70
71 # --- 18. END ---
72
73 # --- 19. END ---
74
75 # --- 20. END ---
76
77 # --- 21. END ---
78
79 # --- 22. END ---
80
81 # --- 23. END ---
82
83 # --- 24. END ---
84
85 # --- 25. END ---
86
87 # --- 26. END ---
88
89 # --- 27. END ---
90
91 # --- 28. END ---
92
93 # --- 29. END ---
94
95 # --- 30. END ---
96
97 # --- 31. END ---
98
99 # --- 32. END ---
100

```

```

note 1 (python)
1 import sys
2 import csv
3 import math
4 from datetime import datetime
5
6 # --- 1. SETUP THE CSV SPECS ---
7 csv_filename = "solar_power_log.csv"
8 # Create the file and write the header row
9 with open(csv_filename, mode='w', newline='') as file:
10     writer = csv.writer(file)
11     # Write CSV header: "Time", "Pan angle", "Tilt angle", "Voltage (V)"
12     writer.writerow(["Time", "Pan angle", "Tilt angle", "Voltage (V)"])
13
14 # --- 2. GET THE CURRENT TIME ---
15 current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
16
17 # --- 3. READ THE DATA ---
18 # Read the CSV file and get the data
19 with open(csv_filename, mode='r') as file:
20     reader = csv.reader(file)
21     # Skip the header row
22     next(reader)
23     # Get the first row of data
24     first_row = next(reader)
25     # Extract the values from the first row
26     pan_angle = float(first_row[2])
27     tilt_angle = float(first_row[3])
28     voltage = float(first_row[4])
29
30 # --- 4. CALCULATE THE ANGLES ---
31 # Calculate the pan angle
32 pan_angle = pan_angle * math.pi / 180
33 # Calculate the tilt angle
34 tilt_angle = tilt_angle * math.pi / 180
35
36 # --- 5. CALCULATE THE VOLTAGE ---
37 # Calculate the voltage
38 voltage = voltage * 1000
39
40 # --- 6. WRITE THE DATA TO THE CSV FILE ---
41 # Write the data to the CSV file
42 with open(csv_filename, mode='a') as file:
43     writer = csv.writer(file)
44     # Write the data row
45     writer.writerow([current_time, pan_angle, tilt_angle, voltage])
46
47 # --- 7. PRINT THE DATA ---
48 # Print the data
49 print(f"Time: {current_time}, Pan angle: {pan_angle}, Tilt angle: {tilt_angle}, Voltage: {voltage}")
50
51 # --- 8. END ---
52
53 # --- 9. END ---
54
55 # --- 10. END ---
56
57 # --- 11. END ---
58
59 # --- 12. END ---
60
61 # --- 13. END ---
62
63 # --- 14. END ---
64
65 # --- 15. END ---
66
67 # --- 16. END ---
68
69 # --- 17. END ---
70
71 # --- 18. END ---
72
73 # --- 19. END ---
74
75 # --- 20. END ---
76
77 # --- 21. END ---
78
79 # --- 22. END ---
80
81 # --- 23. END ---
82
83 # --- 24. END ---
84
85 # --- 25. END ---
86
87 # --- 26. END ---
88
89 # --- 27. END ---
90
91 # --- 28. END ---
92
93 # --- 29. END ---
94
95 # --- 30. END ---
96
97 # --- 31. END ---
98
99 # --- 32. END ---
100

```

Fig. 3.6 (d) Solar Tracking Python Program

4. RESULTS

The results highlight the proposed system's strong performance in solar tracking, real-time monitoring, and Digital Twin integration. It achieved enhanced energy efficiency, consistent data acquisition, and seamless synchronization between the physical model and its digital counterpart.

4.1 Solar Tracking Accuracy

The vision-based tracking system successfully identified the direction of maximum sunlight through image processing techniques. The servo motor then adjusted the solar panel position to maintain continuous alignment with the sun. This system demonstrated improved accuracy compared to traditional LDR-based tracking systems [22].

4.2 Energy Generation Efficiency

The dynamic adjustment of the dual-axis solar panel led to significantly enhanced energy harvesting. Compared to a fixed stationary 5V solar panel setup, the active tracking system maintained peak voltage for extended periods. The reduction in physical misalignment minimized energy losses caused by partial shading, keeping the solar panel perpendicular to the primary light source [30].

4.3 Real-Time Data Monitoring

The INA219 sensor effectively measured voltage and current values in real time [14]. The collected data was transmitted to the system and visualized accurately. The readings remained stable and consistent throughout operation, demonstrating the reliable performance of the monitoring system.

4.4 Digital Twin Performance

The Digital Twin model accurately replicated the behaviour of the physical system. Real-time synchronization between the hardware and the virtual model was achieved with minimal delay. The system allowed users to monitor performance remotely and observe system behaviour dynamically.

4.5 Machine Learning-Based Analysis

The integration of machine learning enabled comparison between predicted and actual output values. The system successfully identified variations caused by environmental factors such as shading, dust, or misalignment, improving overall system awareness [11], [12].

5.DISCUSSION

- 1) From the above results, it can be noted that the inclusion of vision-based tracking together with Digital Twin and IoT technology greatly improves the efficiency of solar power systems. The enhanced tracking efficiency demonstrates the advantages of using camera techniques over traditional sensor techniques.
- 2) The energy efficiency improvement further proves the importance of using dynamic solar tracking in enhancing energy output.
- 3) Real-time monitoring using IoT and sensors ensures better control and transparency of system performance. The successful implementation of the Digital Twin highlights its importance in providing remote access, visualization, and system analysis.
- 4) Moreover, the application of machine learning in the design adds the capability of prediction. Faults can be predicted and dealt with accordingly in order to enhance reliability.
- 5) All these factors make the system scalable and effective in terms of the expenditure. Also, the application enhances sustainable energy systems by increasing efficiency in energy consumption and avoiding energy wastage, as per current research on energy systems.

6. CONCLUSION

In this paper, the design and development of a Digital Twin-Based Smart Renewable Energy System using a dual-axis, image-based tracking algorithm were successfully demonstrated. By applying OpenCV image processing instead of traditional LDR light sensors, the system achieved higher tracking accuracy and robustness against environmental noise [22]. The integration of IoT technology via the INA219 sensor enabled precise, real-time monitoring of DC voltage and current [14]. Simultaneously, the Three.js-based Digital Twin solved the challenge of remote visibility by providing a live, synchronized 3D visualization of the physical hardware [8].

Overall, the system displayed notable enhancements in efficiency and responsiveness. The real-time serial communication and synchronization between the physical Arduino-based edge hardware and the digital browser counterpart occurred seamlessly, demonstrating the powerful synergy of Digital Twin technology and IoT monitoring systems.

While the current prototype successfully validates this architecture, there are avenues for future expansion. The reliance on a local camera feed could be upgraded to edge-computing modules (such as a standalone Raspberry Pi) for entirely self-contained processing. Additionally, incorporating advanced Machine Learning models to predict power output based on historical weather APIs would transition the system from reactive tracking to predictive energy management [11], [12]. Ultimately, this project establishes a highly effective, low-cost foundation for next-generation intelligent energy infrastructures.

7. REFERENCES

- [1] International Energy Agency (IEA). (2023). *Renewables 2023: Analysis and Forecast to 2028*. IEA Publications, Paris. <https://www.iea.org/reports/renewables-2023>
- [2] Tiwari, S., & Lakra, A. (2024). Solar Tracking System Using Arduino. *International Journal of Research Publication and Reviews*, 5(6), 6997–7007.
- [3] Mousazadeh, H., Keyhani, A., Javadi, A., Mobli, H., Abrinia, K., & Sharifi, A. (2009). A review of principle and sun-tracking methods for maximizing solar systems output. *Renewable and Sustainable Energy Reviews*, 13(8), 1800–1818.
- [4] Ali, S. S., & Ali, S. M. (2017). Solar tracking system using Arduino. In *Proceedings of the 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)* (pp. 1–5). IEEE.
- [5] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4), 2347–2376.
- [6] Grieves, M., & Vickers, J. (2017). Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behaviour in Complex Systems. In *Transdisciplinary Perspectives on Complex Systems* (pp. 85–113). Springer, Cham.
- [7] Tao, F., Zhang, H., Liu, A., & Nee, A. Y. C. (2019). Digital Twin in Industry: State-of-the-art. *IEEE Transactions on Industrial Informatics*, 15(4), 2405–2415.
- [8] Huang, S., Bhatt, K., Srivastava, A. K., & Bhatt, S. (2023). Digital Twins for Smart Grids: A Review and Research Directions. *IEEE Access*, 11, 12304–12320.
- [9] Messenger, R. A., & Abtahi, A. (2017). *Photovoltaic Systems Engineering* (4th ed.). CRC Press / Taylor & Francis.
- [10] Dargan, J., & Gupta, N. (2024). *Integration of Blockchain for IoT Communication*. In *Adaptive Power Quality for Power Management Using Smart Technologies* (pp. 319–343). CRC Press, Taylor & Francis.
- [11] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.
- [12] Kaur, J., & Gupta, N. (n.d.). *Artificial Neural Network: A Review*. *International Journal of Technical Research & Science*, pp. 1–4. ISSN: 2454-2024.
- [13] TowerPro. (2023). SG90 9g Micro Servo Datasheet. TowerPro Modelcraft. <https://www.towerpro.com.tw>
- [14] Texas Instruments. (2015). INA219 Zero-Drift, Bidirectional Current/Power Monitor with I2C Interface (SBOS448G). Texas Instruments. <https://www.ti.com/product/INA219>
- [15] Simon Monk. (2021). *Programming Arduino: Getting Started with Sketches* (3rd ed.). McGraw-Hill Education.
- [16] Paul Scherz, & Simon Monk. (2016). *Practical Electronics for Inventors* (4th ed.). McGraw-Hill Education.
- [17] Tinkercad. (2023). Introduction to Breadboard Prototyping. Autodesk Tinkercad Education Resources. <https://www.tinkercad.com>
- [18] WCH (Nanjing Qinheng Microelectronics). (2022). CH340G USB to Serial Chip Datasheet v2.1. <https://www.wch-ic.com/products/CH340.html>
- [19] Horowitz, P., & Hill, W. (2015). *The Art of Electronics* (3rd ed.). Cambridge University Press.
- [20] Espressif Systems. (2023). ESP8266 Arduino Core Documentation: OTA Updates. https://arduino-esp8266.readthedocs.io/en/latest/ota_updates/readme.html
- [21] OASIS Standard. (2019). MQTT Version 5.0. OASIS Open. <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html>
- [22] Rustemli, S., & Dincer, F. (2011). Prototyping of solar tracking system with Arduino and analyzing tracking accuracy.

- [23] Biswas, M., Dutta, S., & Chanda, S. K. (2017). Arduino-based automatic solar tracking system with blink detection. In Proceedings of the 2017 International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 1–5). IEEE.
- [24] Kumar, A. (2019). Design and implementation of solar tracking system using Arduino. In Proceedings of the 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS) (pp. 1270–1274). IEEE.
- [25] Shiloh, M. (2022). Getting Started with Arduino (4th ed.). Maker Media / O'Reilly.
- [26] Gupta, N., & Gupta, N. (2024). An overview of e-mobility-based threats to the power grid. In E-Mobility in Electrical Energy Systems for Sustainability (pp. 142–156). IGI Global. <https://doi.org/10.4018/979-8-3693-2611-4.ch008>
- [27] Bharti, A., & Gupta, N. (2024). Electrifying transportation and pioneering the future of sustainable electricity mobility. In E-Mobility in Electrical Energy Systems for Sustainability (pp. 80–89). IGI Global.
- [28] Mahla, S. K., Gupta, N., Papreja, A., & Mahla, S. K. (2017). Experimental Studies of Utilization of Biogas with Biodiesel in Diesel Engine Generator for Electricity Generation. International Journal of Engineering Research & Science, 3(2), 24–29.
- [29] Mahla, S. K., Dhir, A., & Gupta, N. (2017). Emission Characteristics of Diesel Engine Fuelled with Biogas and n-Propanol–Biodiesel–Diesel Blend under Dual Fuel Mode. International Journal of ChemTech Research, 10(13), 1–8.
- [30] Gupta, N. (2017). An Experimental Analysis of Shading on Solar Photovoltaic System. International Journal on Emerging Technologies, 226–228.