iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80142

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Design of Hybrid SOCforan Intelligent Agricultural Deterrent System with an Embedded Vision Core

Vishnu Kanth G[1], Adeeba Fathima MD[2], Vyshnavi M[3], Shilpa B[4]
ECE Department, MVSR Engineering College

Abstract: ***Wildlife and domestic animal intrusion into agricultural land is a well-known problem that causes real financial damage to farmers and their families. Traditional methods of deterring animal intrusion into crops via direct observation of the crop field, such as physical guarding or using static devices , are ineffective because the animals typically adapt quickly to repetitive patterns of deterrence use. To address this problem, we developed an autonomous smart scarecrow system that utilizes two smartphone cameras as visual sensor nodes for detecting animal presence in the crop field, a YOLOv3 deep-learning detector hosted on a laptop, analysing live feeds from both smartphone cameras overlooking the same field area (i.e. one image from each camera), a pair of ESP32-CAM modules serving as wireless bridges between the cameras and the host laptop, a Nexys FPGA board (Artix-7 xc7a100tcsg324-1) programmed with Verilog for performing OR-based decision logic, and an ESP32-Dev module driving a df mini player and speaker, delivering randomized predator audio alongside LED flashes directed at detected intruders. Operating through a Sense–Process–Act loop, the system functions, whereupon detecting an animal (i.e. bird, dog, horse, bear, etc.) in real-time and in the absence of any human in the field of view of the cameras, upon which the FPGA triggers a randomly chosen predator sound through the speaker together with randomized LED flash patterns to drive away the animal.With this setup, the system spots animals as they appear and scares them off right away without causing any harm, keeping the crops safe.***

Keywords: YOLOv3, ESP32-CAM, FPGA, Artix-7, smart scarecrow, animal intrusion, crop protection, Verilog

I. INTRODUCTION

For many developing countries, farming is still the main source of income. Many nations face significant and often unaddressed issues regarding losses incurred due to the animal intruding up on agricultural land. Numerous animals such as birds , wild boar and other forms of wildlife often seek food sources in agricultural areas targeting crops during both the sowing and harvesting stages.

The quantities of crops destroyed during these invasions can happen in a matter of hours and create considerable financial difficulties and mental stress for farmers. For years, farmers have been using things like walking the fields themselves, putting up scarecrows, and building fences , but all of these take a lot of time and money, and none of them work around the clock of many fields and at the same time nighttime coverage and expansive acreage only compound the difficulty. Furthermore, over time many animals have habituated to the traditional methods. In this paper, we put together a system that combines deep learning object detection with FPGA hardware control.

Our setup uses two mobile phones as IP cameras that stream live video to a laptop. Both video feeds are checked at the same time by Python scripts running YOLOv3. If the script spots an animal but no person in the frame, it sends an HTTP signal over to the matching ESP32-CAM, which drives a digital GPIO signal to the FPGA. The Nexys FPGA board implements simple OR-based logic in Verilog; if either camera signal is active, it passes a trigger along to the ESP32-Dev, which tells the df mini player to play a random scary animal sound on the speaker while LEDs flash in different patterns at night.

The big advantage here is that the FPGA does the signal processing entirely in hardware, which means zero delay from software, while the ESP32-Dev drives the audio response.

II. LITERATURE REVIEW

Smart farming technology is moving towards ways of keeping wildlife away from crops without hurting them, rather than using harmful methods like electric fences and poisonous repellents. Using harmful methods brings up ethical questions and can hurt the local environment.

One of the key studies in this area put forward an IoT system that uses motion sensors and cameras to keep watch around the farm perimeter. Their cloud-based system used CNN and SVM algorithms, getting 96% and 90% detection accuracy. The primary reference for this project is taken from the IEEE paper titled "IoT based system for humane animal deterrence and sustainable crop management" During field testing, the original system cut crop damage by 97%. But cloud systems have their own issues, mainly network delays and bandwidth limits.

That earlier system runs everything on a regular CPU to collect and process sensor readings. Regular CPUs do things one at a time, which can slow down the whole data collection process. This becomes a big problem when you need to track fast-moving animals. This gap can be resolved through the use of Edge Computing and hardware accelerations.

From the perspective of the current literature review, FPGAs are particularly well suited to handle real-time processing. Indeed, using the control logic written using Verilog HDL allows performing true parallel computation, which means that so it can watch both camera feeds at once and set off the trigger in the same clock cycle. Furthermore, unlike the base article, which uses a CNN for classification purposes, the combination of real - time detection algorithms, such as YOLOv3 , alongside hardware-level optimizations leads to better and more efficient results. Such a solution helps to solve the dependency on the cloud infrastructure problem.

Although there have been numerous advancements within the field of edge computing, one of the current areas that lacks sufficient literature pertains to how IoT-based low-end nodes can be incorporated into higher-end hardware controllers. Existing ESP32-CAM publications predominantly address image capture and on-board object detection. Very little can be found regarding the use of GPIO signal interfacing in order to connect a Python program for object detection to an output control program on an FPGA device. Our project fills this gap by using the ESP32-CAM not just as a camera but also as a signal relay.

III. PROPOSED METHODOLOGY

A. *System Architecture and Data Flow*

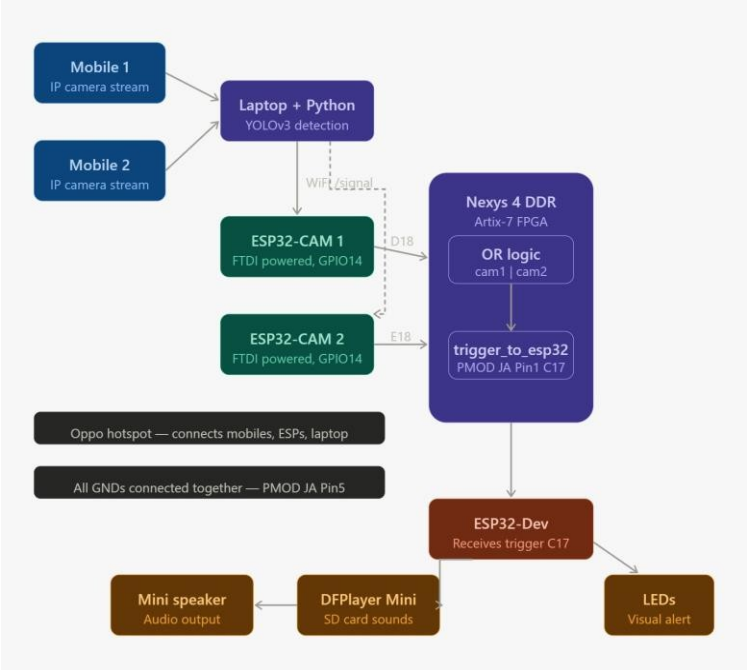


Fig.1Systemarchitecture

1) *Sensing Layer:*Two smartphones form the sensing layer. They capture video of the agricultural field from two differentangles.Running IP-camera applications, they stream footage in real time over a local Wi-Fi hotspot.
2) *Processing Layer:*A laptop serves as the processing layer, handling both video streams. It runs two independent Python scripts that each invoke YOLOv3 with pre-trained COCO weights. Each script inspects incoming video frames and checks for animals listed on a predefined alert roster.

3) *Signal Bridge Layer:*When an animal shows up, the Python code fires off an HTTP GET request to the right ESP32-CAM. The scripts format this request as a URL: /signal state=1 for an alert, or /signal state=0 for a safe status. The ESP32 - CAM responds by setting its GPIO14 signal HIGH for an alert and LOW for a safe status.
4) *Decision Layer:*The alert and safe signals from both ESP32-CAMs come into the FPGA through PMOD JA pins D18 and E18.A Verilog module applies OR logic to these inputs. If either incoming signal is HIGH, an alert signal goes out on pin C17.
5) *Actuation Layer:*When GPIO4 on the ESP32-Dev goes high. and subsequently commands the df mini player to play a randomly picked predator track from its SD card through the attached mini speaker. the module also flashes LEDs to add a visual scare on top of the sound.

B. *Detection Logic*

The Python detection scripts follow a three-rule decision structure:

1) A detected animal from the alert list will trigger sending a state = 1 to ESP32-CAM.
2) A detected human (farmer in the field) triggers sending state = 0 regardless of animal detected (safe/no alert).
3) Nothing detected = state = 0 (safe/no alert).

A state tracker is used to control the HTTP request so they are only sent when the detection state changes. That way the ESP32-CAM does not get bombarded with the same command over and over. Using the FPGA to provide logic via the implementation of Verilog of a simple combination logic design on the FPGA consists of a module for the smart scarecrow which compares the two input signals ; when triggered produces an outputsignal .

C. *FPGA Verilog Logic*

Eachinputvaluesareevaluatedinacasestatement:

1) 00(bothinputs)totheoutput=0
2) 01(anyoneoftheinputs)totheoutput=1
3) 10(anyoneoftheinputs)totheoutput=1
4) 11(bothinputs)totheoutput=1

We kept the logic as basic as possible so the output is always predictable and comes through with almost no delay.

D. *Audio Randomization*

The The audio playback device which uses a df mini player, has four different audio files located on the SD card: thunder, lion's roar, eagle's cry and firecrackers.

The audio playback device, through the use of the ESP32 dev kit and the random Seed algorithm, will trigger the appropriate audio file and will also randomly select which audio file is played back by using an event along with random led pattern. Because the sound changes every time, the animals cannot just ignore it the way they would a regular scarecrow.

IV. MATERIALS AND METHODS

A. *Hardware Components*

1) *Nexys FPGA Development Board:*We use the Nexys FPGA Development Board as the main brain of the system. We wrote the code for it in Verilog HDL using Vivado 2022.2. It uses the PMOD JA pins for inputs and outputs.
2) *ESP32-CAM Modules:*Every ESP32-CAM gets its power from a standard FTDI adapter. They link up to a WiFi hotspot. The Python detection code controls each module by sending HTTP requests that toggle GPIO14. The ESP32-CAM's GPIO14 pin connects straight to the FPGA's PMOD JA input pins. We also gave them static IP addresses so they are easier to find on the network: CAM1 uses 172.18.170.100, and CAM2 uses 172.18.170.101.
3) *ESP32-Dev Module:*When the FPGA sends a trigger, the ESP32-Dev catches it on its GPIO4 pin (routed from FPGA pin C17). It uses the UART protocol to command LEDs and the df mini player , which handles audio playback. at the same time, it triggers a visual alarm.
4) *Df mini player:*Thedf mini player ships with a FAT32-formatted MicroSD card.It holds four audio tracks. This player connects to the ESP32-Dev board using UART and directly powers a mini speaker.
5) *Mobile Phones*:We just run simple IP camera apps on mobile phones to send video over the Wi-Fi. On the laptop, our Python script just pulls the video directly from those URLs. We position the phones at various angles to monitor the farm land.

6) *Additional Hardware Components:*We also threw in some dedicated power supplies for field-deployed ESP32 boards. and used a basic breadboard with jumper wires to tie everything together. and a shared ground bus tying all modules together.

B. Softwarecomponents

1) *YOLOv3:* This object detection algorithm uses it's pretrained weights and a configuration file. It processes 416x416 pixel input images, applying a 0.7 confidence and 0.3 NMS threshold. It is set up to look for things like birds, dogs, horses, bears, and even humans.We implemented it on a laptop using OpenCV's DNN backend, running on the device's CPU.
2) *Python Detection Scripts:*We have two separate Python scripts running, one for each camera, to grab the video frames. These scripts pull frames from an IP camera on a device. They include a state machine that dispatches HTTP requests to an endpoint whenever the state changes. The scripts specifically send a GET request to the ESP32-CAM signal endpoint.
3) *Arduino IDE:*The ESP32-CAM manages the Wi-Fi connection, assigns IP addresses, and streams video through a URL.Meanwhile, the ESP32-Dev just sits there watching GPIO4 waiting for the FPGA to send a signal. and as soon as it gets it, it tells the df mini player to start making noise. The ESP32 cam board includes PSRAM and uses an APP 3MB partition type
4) *Vivado 2022.2:*We used Vivado 2022.2 to do all the synthesis, routing, and building the final bitstream for our Verilog code. The whole Verilog project really just breaks down into three main files: smart_scarecrow.v (design), smart_scarecrow.xdc (constraints), and tb_smart_scarecrow.v (testbench).

V. RESULTS

A. FPGALogicVerification

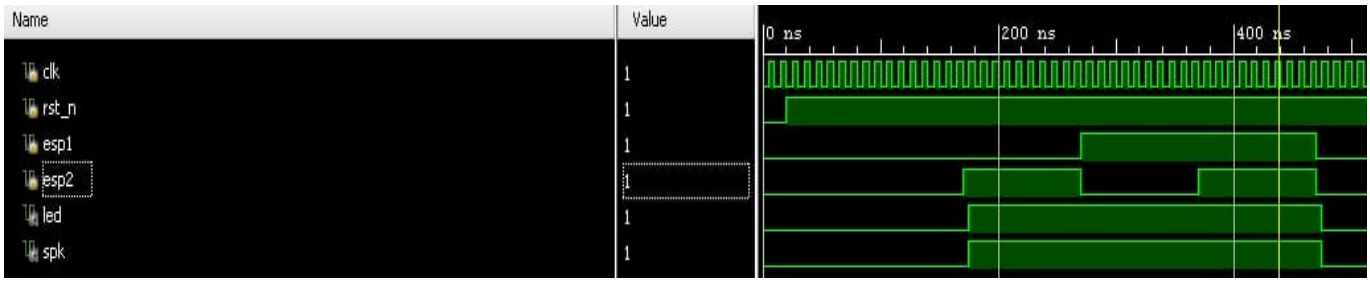


Fig.2Testbenchresult

Before actually building anything, we tested the logic with a behavioural simulation in Vivado. Four possible input scenarios (00, 01, 10, 11) were considered by the testbench, ensuring the correctness of the OR gate output. Running synthesis and implementation went smoothly without throwing any errors.

We pushed the generated bitstream down to the board using the normal JTAG cable, with the LED signal (LD0, H17) confirming proper operation.

B. Finalprototype

A final implementation of the hardware is presented in Fig 3. It consists of a combination of Nexys FPGA (Artix-7), which acts as a primary logic unit, connected to the ESP-32 CAM module and the ESP-32 Dev board through the use of a breadboard in order to distribute the signal effectively.

A df mini player paired with a compact speaker delivers predator sounds along with other lights, are used for visual scare tactics.

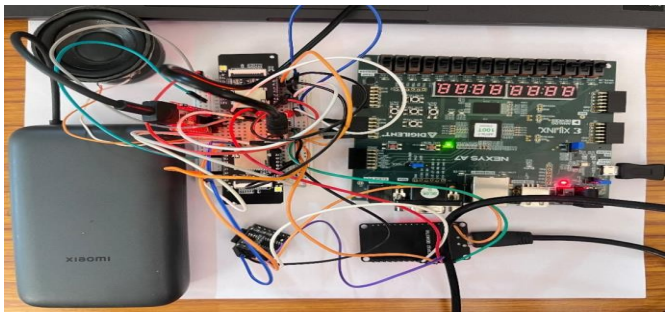


Fig.3Hardware prototype

VI. CONCLUSIONS

The system we have built provides a workable and modular design for smart crop protection. This is achieved with the use of the visual recognition system on a regular laptop, ESP32-CAM module to provide the signal between the Wi-Fi connection and the GPIO pins, an Artix-7 FPGA that takes care of the hardware-side logic and df mini player for audio deterrence through random selection of predator sounds. With basic OR logic coded in Verilog, the FPGA makes sure the deterrent response happens every single time to ensure that the deterrence is executed without any interruptions caused by software malfunctions. The randomisation of the sound through four different types of predator sounds guarantees that the animals will never get used to one single audio, — the principal weakness of conventional scarecrows.

Thekeyfeaturesthatweresuccessfullyachievedinclude:

1) *Modulararchitecture:*We can update one part without touching the rest.
2) *Human-safedeterrence:*If a farmer is in the picture, the system sees that and holds back the alarm.
3) *Adaptivedeterrence:*The sound selected randomly each time helps animals not to get habituated.

In the future, we plan to add a night-vision camera, SMS or mobile app notifications via the ESP32-Dev board and extension of FPGA logic to make decisions based on the identified species.

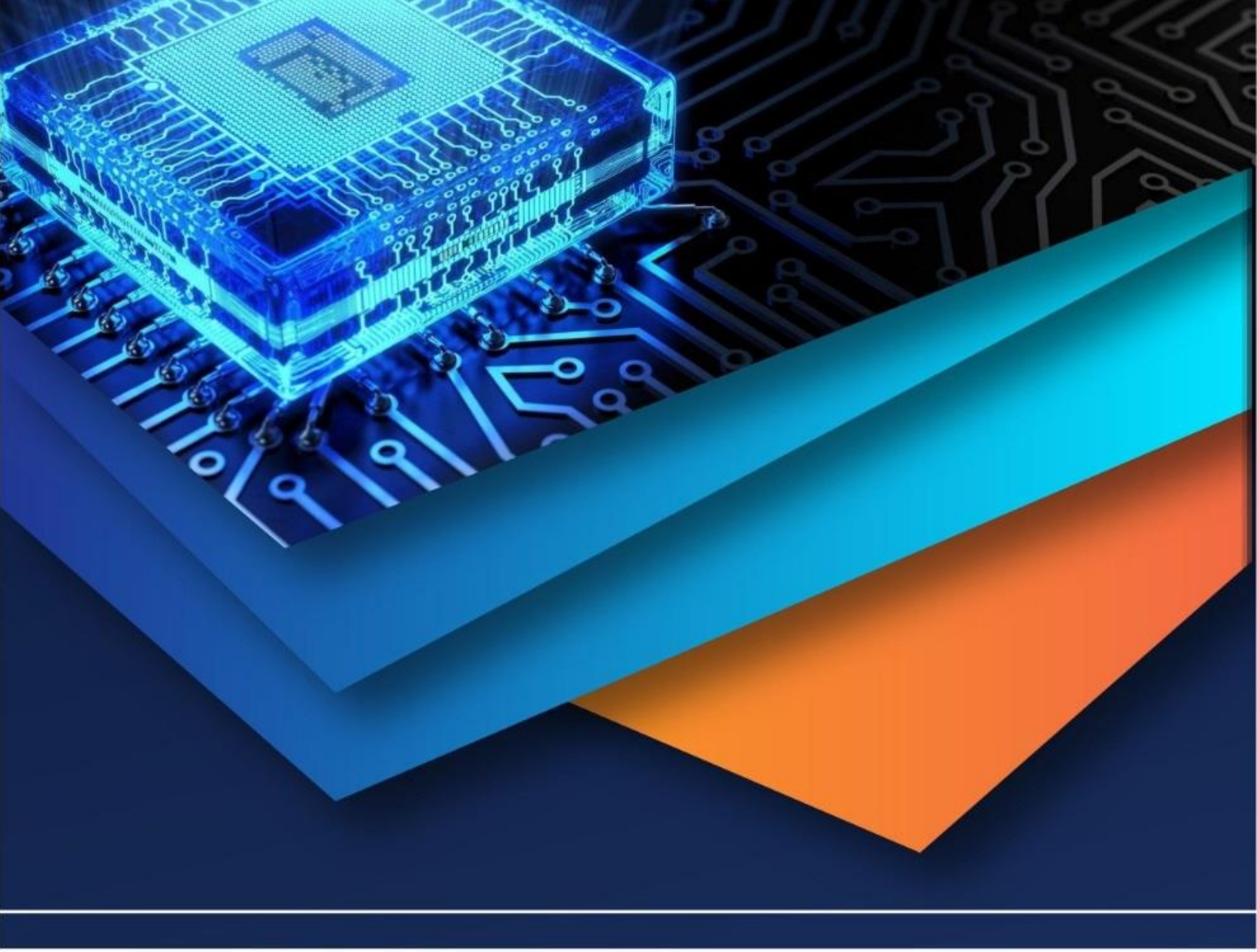
VII. ACKNOWLEDGMENT

We thank the ECE Department of MVSR Engineering College for giving us the tools and lab access we needed for this project. Special thanks go to our mentor Mr. Vishnu Kanth G whose guidance was instrumental in the project's completion.

We would also like to acknowledge the open-source contributors from IEEE, OpenCV and the ESP32 community without whose core tools, this prototype and humane means of protecting crops wouldn't have been possible.

REFERENCES

[1] S.Geerthik,etal.,"IoT-BasedSystemforHumaneAnimalDeterrenceandSustainableCropManagement," IEEEXplore,2023.(PrimaryBaseReference)
[2] RedmonandA.Farhadi,"YOLOv3:AnIncrementalImprovement,"arXiv:1804.02767,2018.
[3] ElectronicClinic,"ESP32-CAMwithPython,OpenCV,andYOLOv3 forObjectDetection,"(Online).Available: electroniclinic.com
[4] Digilent,"NexysA7 FPGAReferenceManual,"(Online).Available:digilent.com
[5] OpenCV,"DeepNeuralNetwork(DNN)ModuleDocumentation,"(Online).Available:opencv.org
[6] EspressifSystems,"ESP32-CAMAI-ThinkerModuleDatasheet,"2021.
[7] DFRobot,"DFPlayerMiniMP3PlayerDatasheet,"2018.
[8] AMDXilinx,"VivadoDesignSuite UserGuide(UG910),"2022.
[9] COCOConsortium,"CommonObjectsin Context(COCO)Dataset,"(Online).Available: https://cocodataset.org
[10] PythonSoftwareFoundation,"Python3.xLanguageDocumentation,"(Online).Available: python.org



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY