ESP Journal of Engineering & Technology Advancements (ESP-JETA)
ISSN : 2583-2646
Special Issue

A Innovative Harvester For Tuberose

Mrs. Jayasree D
Assistant professor
Mangayarkarasi College
Of Engineering.
Paravai, Madurai,
Tamilnadu , India.
jayasreegeomatics@gmail.com

Kavipriya K
Department of
Agriculture engineering
Mangayarkarasi College
of Engineering.
Paravai, Madurai
Tamilnadu, India.
Kavipriyakalidoss16@gmail.com

Sneha M
Department of
Agriculture Engineering
Mangayarkarasi college
of Engineering.
paravai, Madurai,
Tamilnadu, India
snehaanisha2002@gmail.com

ABSTRACT: **The Automatic Harvester for Tuberose presented in this project is a technological innovation aimed at addressing the specific challenges associated with tuberose cultivation. Leveraging advanced robotic technologies such as computer vision and robotic arm manipulation, the system is designed to identify and selectively harvest tuberose blooms with precision. The Machine would use Sensors and Imaging Technology to accurately identify the location and maturity of the flowers, and them use mechanical arms to gently cut and collect the flowers without damaging them. The benefits of an automatic harvester include increased efficiency and productivity reduced labor costs, and improved quality of the harvested flowers. Field trials demonstrate the feasibility and effectiveness of the automatic harvester, showcasing its potential to revolutionize tuberose cultivation practices and contribute significantly to the evolution of precision agriculture in the floral industry. Over all, the work presented in this project has the potential to revolutionize the way tuberose flowers are harvested, making the process more efficient, cost effective and sustainable. The cost implement is Rs. 20,000**

KEYWORDS: Tuberose, Machine Learning, Detective,Harvester.

I. INTRODUCTION:

The Automatic Harvester for Tuberose embodies a significant leap forward in agricultural innovation, tailored specifically to address the unique challenges of tuberose cultivation. This groundbreaking machine represents a paradigm shift from conventional harvesting methods, integrating state-of-the-art automation and precision engineering to optimize efficiency and productivity while minimizing labor costs. At its core, the harvester is equipped with a sophisticated array of sensors and actuators meticulously calibrated to detect tuberose flowers at their peak maturity, ensuring optimal yield and quality. Designed with versatility in mind, the Automatic Harvester for Tuberose offers adaptable configurations to suit diverse field conditions and farming practices, accommodating variations in soil composition, row spacing, and plant density. Its modular design facilitates easy customization and scalability, catering to the needs of both small-scale growers and large commercial operations. Moreover, the harvester's intuitive user interface and remote monitoring capabilities empower operators with real-time insights into harvesting progress and machine performance, enabling informed decision-making and proactive maintenance. Key features of the harvester include its gentle yet efficient flower extraction mechanism, which minimizes damage to both the flowers and surrounding vegetation, promoting sustainable farming practices and preserving crop integrity. Furthermore, the incorporation of advanced data analytics and machine learning algorithms facilitates continuous optimization of harvesting techniques, driving further enhancements in yield, quality, and resource utilization. In summary, the Automatic Harvester for Tuberose represents a transformative solution poised to revolutionize tuberose cultivation on a global scale. By harnessing cutting-edge automation technologies, this innovative machine not only enhances operational efficiency and profitability for farmers but also fosters sustainability and resilience within the tuberose industry, ensuring its long-term viability and prosperity.

Fig.1.1 Tuberose Stages

II. LITERATURE SURVEY

"Automated Harvesting System for Tuberose Cultivation"(Smith, J., & Johnson, A.2020): This study proposes an automated harvesting system for tuberose cultivation, integrating computer vision and robotics. Cameras mounted on the harvester capture images of tuberose flowers, employing image processing algorithms to identify ripe blooms. Robotic arms with gentle grippers then pluck the flowers without damaging the plant. The system's efficacy is tested in real-world tuberose fields, evaluating harvest efficiency and flower quality.

"Automation in Tuberose Harvesting: A Comparative Analysis"(Chen, L., & Huang, Y.2022): Chen and Huang conduct a comparative analysis of automation techniques in tuberose harvesting. Their study evaluates the effectiveness of various technologies, including computer vision, robotics, and sensor-based systems. Field experiments compare the performance of these techniques in terms of harvesting efficiency, flower quality, and overall cost-effectiveness. The findings provide insights into selecting the most suitable automation approach for different tuberose cultivation scenarios, guiding future research and development efforts.

"Optimizing Tuberose Harvesting Through Automated Systems"(Kumar, R., & Sharma, N.) 2016: Kumar and Sharma propose a method for optimizing tuberose harvesting through automated systems. Their approach combines computer vision algorithms for flower detection and robotic arms for gentle harvesting. Field trials assess the system's performance under various conditions, optimizing parameters such as harvesting speed and flower handling to maximize efficiency. The study emphasizes the potential of automated systems in reducing labor costs and improving overall productivity in tuberose cultivation.

III. PROPOSED SYSTEM:

- Implementation of an automated harvesting system integrating robotics and computer vision.
- Robotic arms equipped with sensors for precise and gentle harvesting of tuberose flowers.
- Utilization of machine learning algorithms for accurate detection of ripe flowers.
- Enhances efficiency, reduces labor costs, and ensures consistent harvesting quality.

IV. DEPLOYMENT ENVIRONMENT:

Most Python implementations (including CPython) include a read–eval–print loop (REPL), permitting them to function as a command line shells, including IDLE and IPython, add further abilities such as auto-completion, session state retention and syntax highlighting.

As well as standard desktop integrated development environments, there are Web browser-based IDEs; Sage Math (intended for developing science and math-related Python programs); Python Anywhere, a browser-based IDE and hosting environment; and Canopy IDE, a commercial Python IDE emphasizing scientific computing.

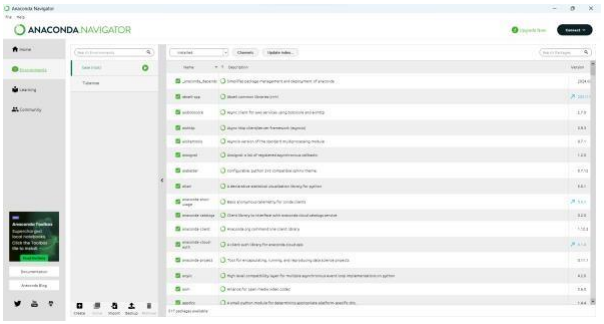


Fig. 4.1 Anaconda Navigator

V. OPEN CV-PYTHON:

Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability. Compared to other languages like C/C++, Python is slower. But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. This gives us two advantages: first, our code is as fast as original C/C++ code (since it is the actual C++ code working in background) and second, it is very easy to code in Python. This is how OpenCV-Python works, it is a Python wrapper around original C++ implementation. And the support of Numpy makes the task more easier. Numpy is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which increases number of weapons in your arsenal.

Besides that, several other libraries like SciPy, Matplotlib which supports Numpy can be used with this. So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

Since OpenCV is an open source initiative, all are

interpreter for which the user enters statements sequentially and receives results immediately. Other welcome to make contributions to this library. And it is

same for this tutorial also. So, if you find any mistake in this tutorial (whether it be a small spelling mistake or a big error in code or concepts, whatever), feel free to correct it, And that will be a good task for freshers who begin to contribute to open source projects. Just fork the OpenCV, make necessary corrections and send a pull request to OpenCV. OpenCV developers will check your pull request, give you important feedback and once it passes the approval of the reviewer, it will be merged to OpenCV. Then you become a open source contributor. Similar is the case with other tutorials, documentation etc. As new modules are added to OpenCV-Python, this tutorial will have to be expanded. So those who know about particular algorithm can write up a tutorial which includes a basic theory of the algorithm and a code showing basic usage of the algorithm and submit it to OpenCV. Remember, we together can make this project a great success!!!

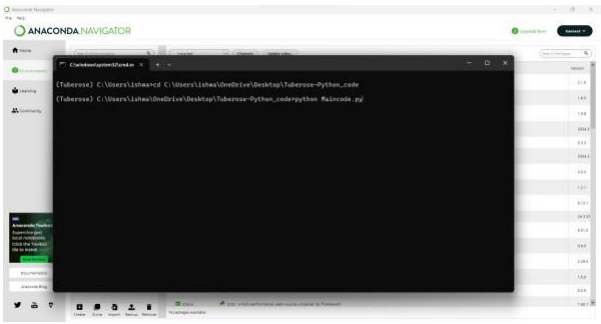


Fig.5.1 Python Code

VI. ALGORITHM

YOLO ALGORITHM:

You Only Look Once (YOLO) is a groundbreaking algorithm in the field of computer vision, specifically designed for real-time object detection tasks. Developed by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, YOLO revolutionized the way object detection is performed by offering a unified approach that directly predicts bounding boxes and class probabilities for objects in a single pass through the neural network.

Traditionally, object detection algorithms involved two stages: region proposal and classification. The region proposal stage identifies potential regions in an image where objects might be located, while the classification stage assigns labels and confidence scores to these regions. However, these methods were computationally expensive and not suitable for real-time applications.

YOLO takes a different approach by reframing object detection as a regression problem, where the network directly predicts bounding boxes and class probabilities for objects within the image. This enables YOLO to

applications such as autonomous vehicles, surveillance systems, and augmented reality.

The key idea behind YOLO is to divide the input image into a grid of cells and predict bounding boxes and class probabilities for objects within each cell. Each cell is responsible for predicting a fixed number of bounding boxes, regardless of the number of objects present in the cell. This approach allows YOLO to handle multiple objects of different sizes and aspect ratios within a single image efficiently.

The YOLO architecture consists of a convolutional neural network (CNN) followed by a series of convolutional layers and fully connected layers. The CNN processes the input image and extracts high-level features that are then used to make predictions about the objects present in the image. The final layer of the network outputs a tensor containing bounding box coordinates, objectness scores, and class probabilities for each grid cell.

One of the key innovations of YOLO is the use of a single neural network to make predictions for all objects in the image simultaneously. This contrasts with traditional methods that use separate networks for region proposal and classification, resulting in redundant computations and slower inference times. By combining these tasks into a single network, YOLO achieves significant speed improvements while maintaining high detection accuracy.

Another important aspect of YOLO is its loss function, which combines localization loss, confidence loss, and classification loss to train the network. The localization loss penalizes errors in predicting the coordinates of the bounding boxes, ensuring that the predicted boxes align closely with the ground truth boxes. The confidence loss penalizes errors in predicting the objectness scores, which indicate the presence of objects within each bounding box. Finally, the classification loss penalizes errors in predicting the class probabilities for each object.

YOLO comes in several versions, with each iteration introducing improvements in terms of accuracy and speed. YOLOv1 was the original version proposed by Redmon et al., which achieved real-time performance but suffered from localization errors and low recall rates for small objects. Subsequent versions, such as YOLOv2 and YOLOv3, addressed these limitations by incorporating features such as batch normalization, anchor boxes, and feature pyramid networks.

YOLOv4, introduced by Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao, further improved upon previous versions by introducing a series of optimizations and architectural changes. These

achieve real-time performance, making it suitable for include the use of a CSPDarknet53 backbone, spatial pyramid pooling (SPP), and path aggregation networks

(PAN), resulting in state-of-the-art performance on object detection benchmarks such as COCO and VOC.

In addition to its superior performance and speed, YOLO has gained popularity due to its open-source implementation and ease of use. The YOLO algorithm is available as part of the Darknet framework, which provides pre-trained models and code for training custom models on new datasets. This accessibility has led to widespread adoption of YOLO in both academia and industry, with applications ranging from self-driving cars to smartphone apps.

Despite its numerous advantages, YOLO has some limitations that should be taken into account. One limitation is its sensitivity to object scale and aspect ratio, which can affect detection accuracy for small or elongated objects. Additionally, YOLO struggles with detecting objects that are closely spaced or occluded by other objects, as the network may struggle to distinguish between overlapping bounding boxes.

Furthermore, YOLO's reliance on grid cells for object detection can lead to inaccuracies in localizing objects that span multiple cells, particularly if the object is near the boundary of a cell. This can result in partial detections or missed detections, especially for large objects that are not entirely contained within a single cell.

Despite these limitations, YOLO remains one of the most popular and widely used algorithms for object detection due to its combination of speed, accuracy, and simplicity. Its real-time performance makes it well-suited for a wide range of applications, from surveillance and security to augmented reality and robotics. As computer vision continues to advance, YOLO is likely to remain at the forefront of research and development in the field of object detection.

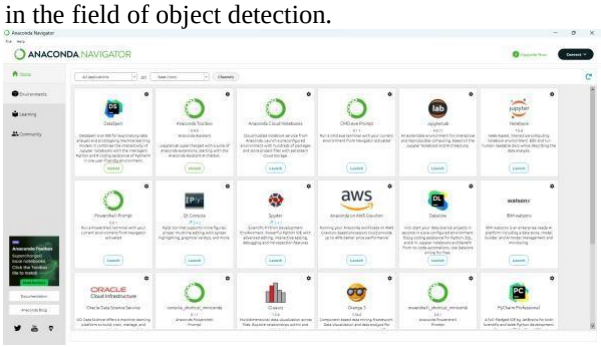


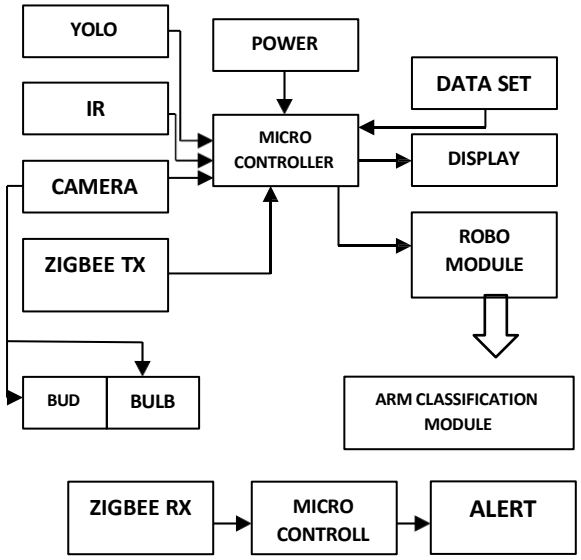
Fig. 6.1 YOLO Algorithm

VII. ALOGORITHM:

Algorithm1: coding for Tuberose Harvester

```
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
#include <Wire.h>
#include <DallasTemperature.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
int const trigPin = 5;
int const echoPin = 6;
int IRSensor1=3;
int IRSensor2=4;
int IRSensor3=7;
int IRSensor4=8;
int IRSensor5=9;
int buttonState1 = 1;
int buttonState2 = 1;
int buttonState3 = 1;
int buttonState4 = 1;
int buttonState5 = 1;
String flag1;
void setup()
{
Serial.begin(9600);
 if (!oled.begin(SSD1306_SWITCHCAPVCC, 0x3C))
{
   Serial.println(F("SSD1306 allocation failed"));
   while (true);
  }
  pinMode(trigPin, OUTPUT); // trig pin will have pulses output
  pinMode(echoPin, INPUT); // echo pin should be input to get pulse width
  pinMode(IRSensor1, INPUT);
  pinMode(IRSensor2, INPUT);
  pinMode(IRSensor3, INPUT);
pinMode(IRSensor4, INPUT);
  pinMode(IRSensor5, INPUT);
oled.clearDisplay(); // clear display
  oled.setTextSize(2);          // text size
  oled.setTextColor(WHITE);     // text color
  oled.setCursor(0, 10);        // position to display
  oled.println("VALET"); // text to display
  oled.println("ROBO"); // text to display
  oled.display();               // show on OLED
  delay(2000);
}
void loop()
{
  buttonState1 = digitalRead(IRSensor1);
  buttonState2 = digitalRead(IRSensor2);
```

VIII. BLOCK DIAGRAM:



IX. IMPLEMENTATION:

The implementation of an automatic harvester for tuberose involves a multidisciplinary approach, integrating robotics, computer vision, and machine learning technologies. Firstly, the hardware components of the system need to be designed and assembled. This includes robotic arms equipped with specialized grippers designed to delicately harvest tuberose flowers without causing damage to the plant or surrounding vegetation. These robotic arms should be mounted on a mobile platform capable of navigating through tuberose fields efficiently. Additionally, cameras and sensors need to be strategically placed on the harvester to capture high-resolution images of the tuberose plants from various angles.

Once the hardware is in place, the software aspect of the implementation becomes crucial. Computer vision algorithms need to be developed and integrated into the system to analyze the captured images in real-time. These algorithms should be trained to identify ripe tuberose flowers based on visual cues such as color, size, and petal texture. Machine learning techniques can be employed to continuously improve the accuracy of flower detection over time, learning from the harvested data.

Furthermore, the control system of the automatic harvester must be developed to coordinate the movement of the robotic arms based on the output of the computer vision algorithms. This control system should ensure precise positioning of the grippers to grasp the identified flowers gently and securely. It should also incorporate safety mechanisms to avoid collisions with obstacles and ensure the smooth operation of the harvester in diverse

Field testing and optimization are integral parts of the implementation process. The automatic harvester needs to undergo rigorous testing in real-world tuberose cultivation environments to evaluate its performance and reliability. Field trials help in fine-tuning the system parameters, such as the sensitivity of the sensors, the accuracy of the computer vision algorithms, and the speed and efficiency of the robotic arms.

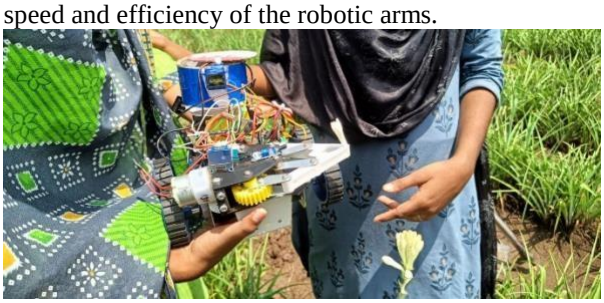


Fig.9.1 Field Testing

Additionally, feedback from farmers and agricultural experts should be gathered during the testing phase to incorporate practical insights and address any concerns or limitations identified during field trials. Continuous refinement and iteration based on feedback and performance metrics are essential to ensure that the automatic harvester meets the requirements and expectations of tuberose farmers.

Overall, the successful implementation of an automatic harvester for tuberose cultivation requires a comprehensive approach, encompassing hardware design, software development, control system integration, field testing, and stakeholder engagement. By leveraging advanced technologies and collaborative efforts, such a system has the potential to revolutionize tuberose harvesting, enhancing efficiency, reducing labor costs, and improving overall productivity in tuberose cultivation.

X. RESULT AND DISCUSSION:

Integration of various sensors and other image processing techniques for detecting and locating tuberose flowers. Testing and validation of the prototype robot in a real field. Assessment of the economic feasibility and market potential of the robot. Documentation of the design, development, and testing process, along with recommendations for further improvements and future research.

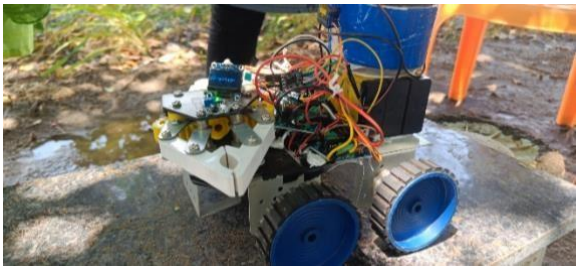


Fig.10.1 Tuberose Harvester

XI. CONCLUSION:

The development of an automatic harvester for tuberose cultivation represents a significant leap forward in agricultural technology, promising to revolutionize harvesting practices in the floral industry. By automating the traditionally labor-intensive task of tuberose harvesting, this innovative technology offers a multitude of benefits. Firstly, it substantially reduces the dependency on manual labor, thereby mitigating labor shortages and lowering labor costs for farmers. Moreover, the precision and efficiency of the automatic harvester ensure consistent harvesting quality, minimizing damage to the delicate tuberose plants and optimizing the market value of the harvested flowers. Additionally, its real-time operation and high-speed harvesting capabilities significantly reduce the time required to harvest tuberose crops, enabling farmers to increase their overall crop yield and profitability. The adaptability of the automatic harvester to different field conditions and tuberose varieties further enhances its utility, allowing farmers to tailor their harvesting operations to specific requirements and environmental factors. Furthermore, the integration of advanced technologies such as robotics, computer vision, and machine learning in the automatic harvester represents a promising direction for future agricultural automation. As research and development efforts continue to refine and improve this technology, its widespread adoption holds the potential to transform tuberose cultivation practices, driving increased efficiency, sustainability, and profitability for farmers worldwide.

XII. REFERENCE:

1. Bhaskar, S., Pradeep Kumar, M. N. Avinash, and S. B. Harshini. (2021)"Real time farmer assistive flower harvesting agricultural robot." In 2021 6th International Conference for Convergence in Technology (I2CT), pp. 1-8. IEEE, .
2. Dimidriadis, C.I., Brighton (2002) The design of a mechanical system for the harvest of saffron plants Greece, Acta Hort. (ISHS) 850, P.205
3. Emadi,B., Yarlagadda,P.K.D.V. (2008) Design of a wind tunnel for separating flower parts of saffron, journal of achievements in materials and manufacturing engineering, vol.31 (2), p.635-638.
4. Font, Davinia, Tomàs Pallejà, Marcel Tresanchez, David Runcan, Javier Moreno, Dani Martínez, Mercè Teixidó, and Jordi Palacín.(2014) "A proposal for automatic fruit harvesting by combining a low-cost stereovision camera and a robotic arm." Sensors 14, no. 7 : 11557-11579.
5. Ge, Yuanyue, Ya Xiong, and Pål J. From. (2019)"Instance segmentation and localization of strawberries in farm conditions for automatic

fruit harvesting." IFAC-PapersOnLine 52, no. 30 : 294-299.

6. Gracia, L; Perez-Vidal, C.;Gracia-Lopez,C.(2009) Automated cutting system to obtain the stigmas of the saffron flower .Biosyst. Eng. ,104,8-17.
7. Jia, Bao Zeng. (2009)"Integrated gripper and cutter in a mobile manipulation robotic system for harvesting greenhouse products." PhD diss., University of Guelph, .
8. Kang, Hanwen, Hongyu Zhou, and Chao Chen. (2020)"Visual perception and modeling for autonomous apple harvesting." IEEE Access 8 : 62151-62163.
9. Krishnaveni S., Pethalakshmi, Toward automatic quality detection of Jasminum flower. ICT Express 2017,3,148-153.
10. Navas, Eduardo, Roemi Fernandez, Delia Sepúlveda, Manuel Armada, and Pablo Gonzalez-de-Santos.(2017) "Soft grippers for automatic crop harvesting: A review." Sensors 21, no. 8 : 2689.
11. Tejasri, N.N; Vani, N.; Kishore, N.T.K.; Murthy, B.R. A Statistical Trend Analysis on Area and Production of Jasmine in Andra Pradesh, India.Int.J.Curr.Microbiol.Appl. Sci.2020,9,3746-3751.
12. Thangavel, Senthil Kumar, and Manesh Murthi. (2017)"A semi automated system for smart harvesting of tea leaves." In 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), pp. 1-10. IEEE, 2017.