A Robotic Simulation for Aerial Monitoring and Disease Detection of Gladiolus Field

Arisha Saeed Akkas[1], Nimra Ejaz[1], Kanwal Atif[2], Maria Anjum[1], Syed Atif Mehdi[2*]
[1] Lahore College for Women University, Lahore, Pakistan
[2] University of Central Punjab, Lahore, Pakistan
*** Correspondence.** syed.atif@ucp.edu.pk, https://orcid.org/0000-0002-6102-9937

Citation | Akkas. A. S, Ejaz. N, Atif. K, Anjum. M, Mehdi. S. A, "A Robotic Simulation for Aerial Monitoring and Disease Detection of Gladiolus Field", IJIST, Vol. 7 Issue. 1 pp 550-565 March 2025
Received| Feb 11, 2025 **Revised|** March 4, 2025 **Accepted|** March 8, 2025 **Published|** March 12, 2025.

Agriculture is an essential sector that is witnessing the integration of advanced technologies to improve productivity and efficiency. Aerial crop monitoring using drones has surfaced as a pivotal technology for precision agriculture, allowing farmers to collect detailed data regarding crop health, soil conditions, and pest infestations. A robotic farm monitoring system in simulation can provide an initial platform to test various automated services before deploying them in the real field. This paper presents an agricultural robotic simulator currently developed for the gladiolus field. Simulation has been designed using V-REP (now known as CoppeliaSim) and Robot Operating System (ROS). Autonomous path planning and navigation are achieved through Hector Simultaneous Localization and Mapping (SLAM) and Rapidly Exploring Random Trees (RRT). One of the most common and fatal diseases of the gladiolus plant named 'Fusarium yellow' has been successfully detected through image processing. This simulation is specifically designed to save resources and reduce the time and cost of developing and testing real-time autonomous aerial robotic systems and test algorithms for crop monitoring. Usability evaluation of the developed system through user survey shows positive results.

Keywords. **S**imulation, UAV, Robot Operating System, Disease Detection, Precision Agriculture

Introduction.

The world population is increasing, according to UN statistics, it will ascend from 7.3 billion today to 9.1 billion in 2050 [1]. Farmers are progressively under pressure to yield high crop production, however, the disease rate in crops is increasing every day [2]. Early and accurate detection and diagnosis of plant diseases are significant aspects of plant production. Traditional monitoring methods such as ground surveys are often time-consuming and labor-intensive [3] [4]. Aerial monitoring offers numerous advantages, including timely data collection, enhanced analysis of crop health, and better decision-making capabilities for farmers. Drones equipped with cameras and sensors can cover larger areas more efficiently, providing actionable insights based on real-time data. It significantly enhances crop management efficiency, enabling timely interventions for pest control, irrigation management, and yield prediction [5].

Robotic simulation environments have gained significant traction in recent years for their ability to enhance the development and testing of multi-purpose robotic systems. They integrate realistic physics and sensor simulations to emulate real-world conditions. However, there are limitations, and custom modifications are required when conventional robot simulators are used directly in agricultural settings. Therefore, there is a need for dedicated robot simulators in agricultural robotics research because it would help to create an agricultural environment conveniently [6]. Such environments facilitate the simulation of drone flight dynamics, sensor data collection, and crop health assessments under variable environmental conditions without the risk and cost associated with physical trials. Recent studies highlight the use of deep learning algorithms for processing simulated imagery to train autonomous navigation systems and identify crop diseases or nutrient deficiencies. Moreover, research showcases the integration of Internet of Things (IoT) technologies within these simulations, allowing for real-time data collection and analysis to enhance decision-making processes for precision agriculture.

The study in hand presents an agricultural simulation with a quadcopter equipped with various sensors including a vision sensor for visual data and a laser scanner for distance measurements. These sensors are simulated and linked with their respective ROS nodes. The implemented algorithms include localization via Hector SLAM and path planning via the RRT technique. The SLAM algorithm processes laser scanner input to generate a map and estimate the drone's position, while RRT identifies potential paths to traverse autonomously. Disease detection is performed on the imagery captured during the traversal.

Objectives.

The proposed system allows for effective testing of the drone under various conditions. Parameters such as crop density and sensor accuracy can be adjusted to evaluate the performance of the system. Simulation results can be systematically analyzed to optimize the algorithms and enhance the robustness of the drone before actual deployment in the field. The drone's localization and mapping error can be monitored over different trials to ensure reliable performance. The user interface of the simulation has been evaluated by the farmers/naïve users as well as developers to meet their requirements in terms of usability, reliability, and adaptability.

Literature Review.

The integration of robotics in agricultural practices is paving the way for smart farming. Autonomous drones can navigate through fields, collect data, and perform tasks like spraying fertilizers and pesticides. A related study detects fungus in the gladiolus field through RGB imagery captured by a quadcopter. They have used machine learning techniques to classify diseased plants with an accuracy of 91% [7]. However, to ensure that these systems operate effectively, thorough testing is required in various environments which can be achieved in a simulation.

There are several environments available, allowing the researchers to create their unique simulations. Numerous simulation tools have been researched to find an environment that supports the agricultural details of multiple crops while also facilitating the dynamics of aerial robots.

Gazebo [8] and AirSim [9] are among the most prominent simulation tools, providing 3D environments to simulate drone operations. Gazebo boasts realistic physics and a rich sensory suite, allowing for nuanced modeling of environmental interactions, though it can be resource-intensive and less user-friendly for beginners. AirSim, developed by Microsoft, focuses more on aerial vehicles, offering high-fidelity simulations of flight dynamics and sensor integration, but it may lack specific agricultural scenarios.

Flightmare [10] is a dynamic simulator made up of two main components. a physics model and a Unity-based rendering engine. Both parts can function independently and are made to be as flexible as possible. The distinct control of Flightmare's rendering and physics engines may make integration more difficult and raise the learning curve for novice users, despite the program's flexibility and independence in its constituent parts [11].

ROS (Robot Operating System) [12] is a popular and open-source software framework that provides a high-level abstraction for building robotic applications. When it comes to creating agricultural simulation, ROS can be a fantastic tool for developing a realistic and efficient simulation environment. Some of the reasons are as follows.

Modularity. ROS follows modular architecture, allowing the breakdown of simulation into smaller components. These smaller components are encapsulated in nodes and communicate with other software modules through topics. This flexibility is particularly useful in agricultural simulations, where different crops, soil types, and weather conditions require tailored approaches. ROS's modular architecture enables developers to create custom solutions for specific farming scenarios.

Tools and Packages. ROS comes with a vast library of tools and packages for various functions such as simulation, visualization, navigation, and perception. These packages can be easily integrated and reused, accelerating the development process. For instance, RVIZ (ROS-VIsualiZer) is a 3D visualization tool for ROS applications. It provides a visual interface to inspect the robot's sensor data, model, and environment.

Rapid Prototyping. ROS's high-level abstraction and vast ecosystem of tools allow us to rapidly prototype and test a simulation. This enables it to iterate quickly and refine the simulation based on field observations and feedback from stakeholders.

Community Support. ROS has a vast and active community, which means existing knowledge, examples, and pre-built packages can be leveraged to accelerate project development. This community support is particularly valuable for agricultural simulations, where fine-tuning and optimizing the simulation is crucial.

Simulation-to-Real-World Transparency. ROS provides a seamless transition from simulation to reality. Simulation can be directly deployed on real robots or platforms, reducing the need for manual data transfer and reconfiguration.

AgROS is a very good example of an agricultural simulator based on ROS, that has been designed with customized options of agricultural layout like crops and landscapes; however, it only supports Unmanned Ground Vehicles (UGV) to be imported into simulation [13].

Tools and Techniques.

The following frameworks have been selected for this proposed work.

Robot Operating System (ROS). A flexible framework for writing robot software, ROS provides libraries and tools essential for building robotic applications. It supports communication between the drone's various components and devices.

V-REP/CoppeliaSim. Recognized for its powerful simulation capabilities, V-REP (now CoppeliaSim) [14][15] enables the design, simulation, and testing of robotic systems in a 3D environment. It allows for visually rich simulations of aerial vehicles. It has been selected due to its largest collection of features such as a scene editor, 3D model importing, and mesh manipulation [16]

Hector SLAM. This technique is critical for navigating environments without GPS. It utilizes laser scanner data and other sensors to build a map of the surroundings while localizing the drone within that map [17].

Rapidly-exploring Random Trees (RRT). RRT is a motion planning algorithm that enables the drone to efficiently navigate through complex environments by exploring feasible paths [18]. RRT is more suitable for an aerial robot in outdoor agricultural fields due to the high dimensionality of the environment and unpredictable obstacles, whereas the A* algorithm works better for indoor smart farms [19]. Standard RRT algorithm has been used due to its promising results as well as to demonstrate the scalability of the simulation that it can run algorithms that have been developed outside of it and libraries can be integrated easily.

Global Thresholding. Global and adaptive thresholding are two common techniques used in image segmentation to separate objects from the background. While global thresholding is a simple, fast, and widely used method, adaptive thresholding techniques like Otsu's method and k-means clustering can provide improved results in certain situations. However, since adaptive thresholding techniques tend to be computationally expensive, global thresholding has been selected for this project to keep the simulation robust and computationally lightweight.

Methodology.

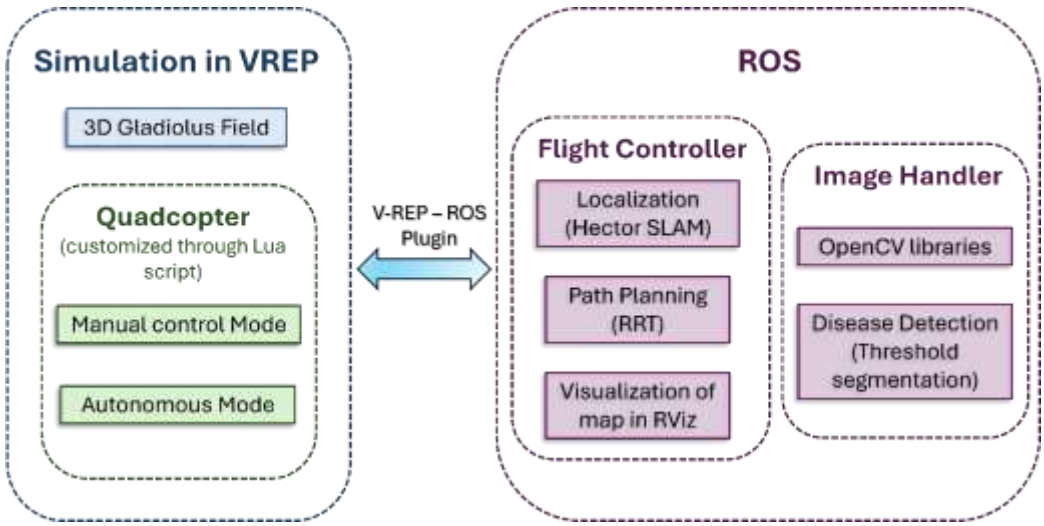

Figure 1. System Architecture

This study presents a robotic monitoring simulation, designed to test how aerial robots can be introduced into the crop fields for farm navigation and crop disease detection. The designed simulation environment consists of a gladiolus crop field. The age of the crop is about 3 months and is moderately affected by the Fusarium yellow's infection, a soil-borne fungal disease [20] A quadcopter equipped with a laser scanner and a vision sensor has been incorporated as well.

The architecture integrates these components to create a cohesive ecosystem. The ROS master node is responsible for the overall coordination of the system, managing nodes for sensor input, control algorithms, and communication with the simulation environment. The V-REP simulations work in tandem with ROS to visualize the drone's movements and

monitor the aerial crop settings. Hector SLAM is utilized for local mapping when navigating through densely planted areas or where GPS signals are unreliable. RRT aids in determining optimal paths, allowing the drone to maneuver between waypoints effectively, thereby ensuring efficient data collection. These images are segmented and then a threshold is applied to detect the amount of Fusarium yellow's infection that has affected each plant. The image processing is done using OpenCV libraries in ROS. Figure 1 represents all the elements that integrate to form the agricultural simulated environment.

The environment consisted of a crop field, which was designed using the soil object in V-REP. We imitated a real-life gladiolus field, by creating a 3-D gladiolus plant that was roughly 3 months old. The gladiolus object was planted on the whole soil; to prevent overlapping, the crops were planted row-wise with a considerable gap between them. The fact that some of the plants were all green suggested that they were in good health. To ensure that there are diseased plants in the field that can be found during the inspection procedure, other plants were given brown and yellow hues.

The quadcopter used for navigation over the field and capturing image frames is also a V-REP object. The quadcopter has been programmed by customizing the associated child script (Lua script) to achieve the desired functionalities. We also attached additional sensors with the quadcopter to fulfill the purpose of navigation and image capturing. The vision sensor object inside the V-REP was attached to acquire images. The sensor was used instead of the common RGB camera due to its significant role in the detection process. Moreover, V-REP provides an API through which the content of the vision sensor can be accessed, but the content of a camera cannot be later accessed.

In our project, we are using a perspective projection-type vision sensor along with the default cameras that are attached to the quadcopter model because a vision sensor has a fixed resolution while a camera has no specific resolution (i.e. it adjusts automatically to the view size) [14]. Data of the vision sensor is sent over to ROS for image processing through OpenCV libraries. The angle of the vision sensor is set to 60^{o} while the resolution is set to 512 * 512.

Figure 2. Manual Control of quadcopter in simulated gladiolus field

In addition to the vision sensor, a Laser Scanner (Hokuyo UTM-30LX-01 scanning laser rangefinder) is also attached. Data received from this sensor is used during the autonomous localization of the quadcopter.

Two control modes have been implemented for the quadcopter, manual and autonomous. If the user wants to control the flight of the quadcopter, they can self-guide it. The manual control mechanism was implemented inside V-REP by creating a remote control that allowed the user to take off the quadcopter, move it up, down, forward, backward, right, and left, and then land it in the desired position. However, the autonomous movement of the quadcopter was partially controlled by ROS and partially implemented inside V-REP.

Figure 2 shows a manually controlled quadcopter flying over the crop field with an active laser scanner and a vision sensor.

Quadcopter. A quadcopter is a multi-rotor aerial vehicle that is lifted and propelled by four rotors. The rotors control the lift, torque, and flight of the copter. It is operated by varying the speed and spin movement of its four rotors. The thrust from the rotors of the quadcopter plays an important role in maneuvering to keep the copter airborne. Quadcopters use two pairs of identical fixed propellers of which one is clockwise (CW), and the other pair is counter-clockwise (CCW) which results in its smooth movement [21].

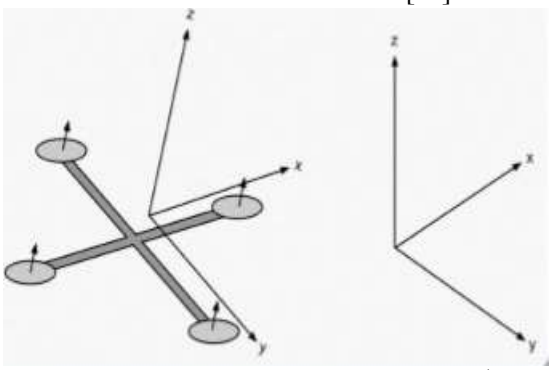

Figure 3. Body and Inertial frame[1]

Dynamics. The Quadcopter operates in two frames. body and inertial frame. The inertial frame is defined by the ground, with gravity pointing in the negative z direction. The body frame is defined by the orientation of the quadcopter, with the rotor axes pointing in the positive z direction and the arms (rotors) pointing in the x and y directions as shown in Figure 3.

Hover. Maintaining a constant state of altitude is called hovering. For hovering a balance of forces is needed. If we want the quadcopter to hover, the SUM of all forces on the copter (F_i) must be equal m*g (weight), where m is the mass, and g is the downward acceleration i.e. gravity.

$$\text{SUM}(F_i) = m * g <=> \text{hover}$$

Take-Off and Landing. Different kinds of movements can be achieved by different combinations of forces produced by each rotor. To achieve takeoff, all four rotors spin in a clockwise direction. The CW direction contributes positive net thrust (z-axis Body frame) on the quad-copter body, thereby enabling translational motion about the positive z-axis (Inertial frame). To achieve the landing, all the four rotors spin in a counterclockwise direction. The CCW direction contributes negative net thrust (z-axis Body frame) on the quad-copter body, thereby enabling translational motion about the negative z-axis if all rotors spin in the same direction with the same velocity.

Flight dynamics. For flight control of quadcopter, the navigation commands are written in the script. The virtual quadcopter inside V-REP uses the Eigen values and Eigen vectors for calculations of flight dynamics. Power, thrust, and torque play a key role in different movements during the flight which have been implemented mathematically through their standard equations [21].

V-REP and ROS.

The system collaboratively configures the V-REP simulation with the ROS environment. The first part in making this integration successful was to create a communication bridge between V-REP and ROS, so the data from V-REP could be used over ROS via APIs. V-REP provides a variety of APIs that are associated with almost all its objects. In our case, we created a plugin between V-REP and ROS since all the ROS messages that we required were directly supported. Through the plugin, the data from the V-REP is

[1] https://toglefritz.com/the-physics-of-quadcopter-flight/

published over ROS in the form of topics as soon as the simulation starts. These topics are then subscribed to the ROS nodes in our case, and the data is utilized as per the need.

Mapping System.

After the communication link between V-REP and ROS was established, the next step was mapping the quadcopter over the field. Mapping is often difficult to achieve in robots. Many techniques have been developed to improve the system of mapping such as Behavior-based navigation [22] fuzzy logic-based approach [23], and SLAM (Simultaneous Localization and Mapping) [24] SLAM tends to localize the robot while it's creating the map of its environment. The mapping results of SLAM are rather promising, but it is limited to lesser populated spaces, where the environment is predictable [25] Considering these characteristics of SLAM, its method of localization is best suited to our environment, which does not have any obstacles.

Hector SLAM. To implement SLAM, we utilized the Hector SLAM package, which is defaulted in ROS. The basic purpose of Hector SLAM is to combine the 2D SLAM system and the 3D navigation technique through robust scan matching using an inertial sensing system. Hector SLAM received data from a laser scanner which was facing downward towards the crops. It estimated the distance by calculating phase differences. Hector SLAM also requires odometrical information to work with a quadcopter and the transformation of the quadcopter as input; to localize the quadcopter to the environment and generate the desired map. The optimization of the map with the alignment of the laser beam endpoints results in the estimation of the 2D position of the quadcopter. This allowed the quadcopter to keep track of its location in an unknown environment. To complete the process of scan matching, the Gaussian-Newton equations were used. They helped in finding a transformation that allowed the best fit between laser beams and the map. The visualization of the map could be seen over the RViz. Changes had to be made to the nodes and the launch files of the Hector SLAM package as needed to get the desired results [26][27]. Figure 4 shows a sample of the map that has been generated for the field.

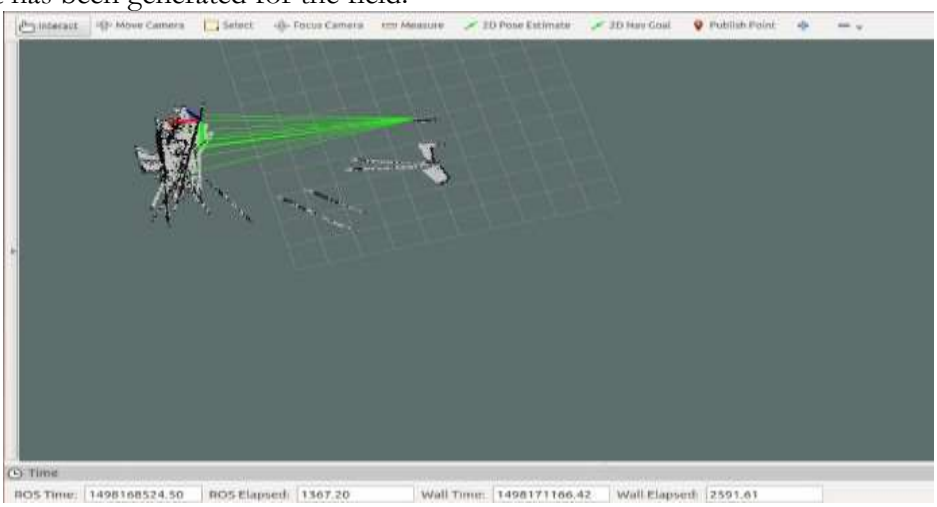

Figure 4. Mapping through Hector SLAM

Navigation System.

To make the quadcopter navigate over the field from a start position to an endpoint, the user can manually define a path. The path is computed in V-REP by using the built-in path planning module that utilizes the Rapidly-exploring Random Tree (RRT) Connect algorithm. The path planning module in V-REP allows convenient path planning in both 2D and 3D spaces for vehicles. The module enables the user to set the start and goal position as well as the markings of the obstacles that are to be avoided during navigation. The path that is created by linking the start and goal position can be in any configuration space using a specific number

of dimensions (X, Y, or Z) [14][15].

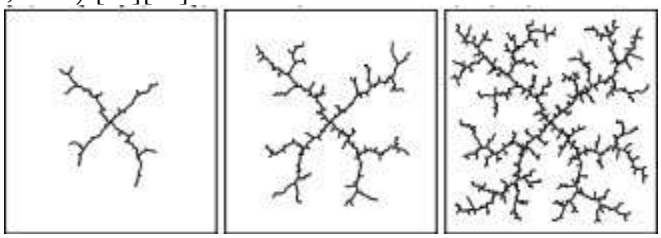

Figure 5. RRT expanding to explore a square [28]

Rapidly Exploring Random Trees. Researchers have investigated various path-planning algorithms such as Breadth First search (BFS), Depth First Search (DFS), A*, and Rapidly Exploring Random Tree Connect (RRT) in simulated agricultural setups [28]. RRTs have been widely used in autonomous robotic path and motion planning. The RRT is a randomized data structure used to solve path-planning problems in a defined search space for robots that have non-holonomic constraints. This algorithm finds paths in high dimensional spaces at interactive time rates. RRT algorithm efficiently computes a path from start to goal configuration in a given search space. The working of an RRT is usually faster than a probabilistic road map, based on the fact that it maintains a connected structure with the fewest number of edges. It works by incrementally building two rapidly exploring random Trees rooted at the start and the goal configurations. Both trees explore the space around them and advance towards each other using a simple greedy heuristic. The two trees expand towards each other until both connect at a certain point and the path from the start configuration to the goal configuration is achieved [18] One of the most important things when implementing path planning is to focus on collision detection. Due to the incremental building nature of RRT trees, it is highly suitable for incremental collision detection. Figure 5 shows how an RRT connects quickly to explore the corners of a square. For planning the path, it searches in a metric space say, X, from an initial state x_{init} to the final state x_{goal}. Algorithm 1 is an RRT from an initial state with K number of vertices, where τ represents a vertex [29].

Algorithm 1 Pseudocode for RRT

1. GENERATE_RRT(x_{init}, K, Δt)
2. τ.init(x_{init});
3. for k=1 to K do
4. $x_{rand} \leftarrow$ RANDOM_STATE();
5. $x_{near} \leftarrow$ NEAREST_NEIGHBOR(x_{rand}, τ);
6. u $\leftarrow$ SELECT_INPUT(x_{rand}, x_{near});
7. $x_{new} \leftarrow$ NEW_STATE(x_{near}, u, Δt);
8. τ.add_vertex(x_{new});
9. τ.add_edge(x_{near}, x_{new}, u);
10. RETURN τ

At the beginning, the vertex τ is at the initial state, as the iterations start a random state x_{rand} gets selected and then the closest vertex x_{near} to the random state x_{rand} is selected. Following this an input u is selected which minimizes the distance between x_{rand} and x_{near}. At the same time, it is ensured that the boundary is maintained. To evaluate a potential new state, a NEW_STATE procedure is called on each input. This new state x_{new} which is obtained after calling upon input u is added into the vertex τ. An edge is created between x_{rand} and x_{new} is also added to the vertex and the input is recorded along with the edge [29].

RRT has been selected due to the non-holonomic properties of the quadcopters as they only have 4 parallel force inputs that allow the control of 6 output coordinates, that is its position and orientation in the space. Start and goal states are provided to the system. The path computation is further refined to ensure that the quadcopter navigates over each row of

gladiolus plants in the field. Lua script was maintained for the quadcopter to guide it to reach its target. The base (Terrain) of our field has been defined as a search area for computing paths based on three dimensions X, Y, and Z.

Image Acquisition and Processing.

After taking off, the quadcopter starts capturing images of the field using the attached vision sensor. The acquired image frames of the crop field are processed with image processing techniques to analyze crop health in the Image Handler node in ROS. Stored images were then compiled into a video stream while removing duplicate frames.

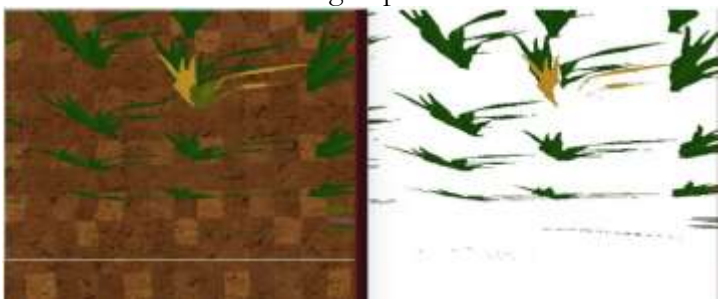

Figure 6. Foreground extraction through Grab-cut technique

Figure 7. Original and detected diseased region after threshold segmentation

Following this procedure, there are various ways through which the segmentation of the image can be achieved which include threshold-based, edge-based, region-based, and clustering-based segmentation [30] In our case, once the video stream is generated, it is subscribed by a ROS node which applies the Grab-cut algorithm on the image frames to extract the plants and remove the background as depicted in Figure 6. The segmented image frames are further refined with the scalar and InRange functions of OpenCV. Afterwards, global threshold segmentation has been applied because of its robustness, when dealing with segmentation based on the color. It partitions the image based on the intensity values of the pixels in the image [31]. Equation 1 depicts the global threshold with an appropriate threshold T.

$$g(x,y) = \begin{cases} 1, & if\ f(x,y) > T \\ 0, & if\ f(x,y) \leq T \end{cases} \quad (1)$$

A black-and-white image is generated by this thresholding process, where the black area represents the phase of every pixel outside of the intensity value range. The phase of every pixel within the intensity value range is indicated by the white portion. For a better user experience, two screens are shown in parallel. one with the original video and the other screen with the video highlighting the detected diseased part of the field in white color. Figure 7 shows the disease detection result produced by the image handler node and the image processing node.

Experiments & Results

A. Unit Testing.

Unit testing is performed manually by exercising the working of every functional component

of the system to determine accuracy, robustness and reliability of simulation, repeatedly. The activities such as fixing bugs and refactoring of errors are performed during the development of simulation. It is ensured that every unit implements the correct functionality and encapsulates the appropriate error handling. Multiple trials were performed in the designed simulation in both autonomous and manual modes. Functional components of simulation that are tested through unit testing are as follows.

- Graphical User Interface of the simulation.
  - The GUI provides a user-friendly interface for manual control of the quadcopter to navigate the crop field.
  - The interface includes a 3D visualization of the crop field with the quadcopter's current location.
  - A remote control is shown for manual control of the quadcopter's movement (take off, up, down, left, right, forward, backward, landing).
  - A camera view of the crop field from the quadcopter's perspective is displayed as well.
  - Appropriate warning windows pop-up during manual control mode to guide the user. For instance, a warning message will appear when quadcopter moves above the optimal height (Fig 8). A warning message will appear for safe and secure landing of quadcopter (Fig. 9).

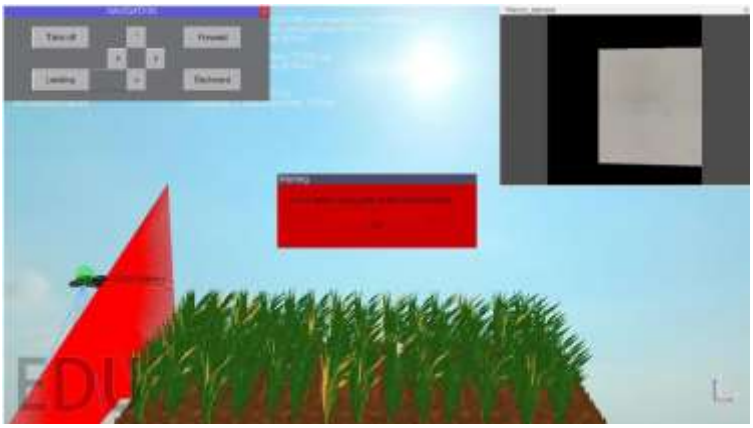

Figure 8. Height warning

- Autonomous navigation of quadcopter.
  - The quadcopter successfully navigated the crop field using laser sensor data, maintaining a consistent altitude and speed.
  - The autonomous navigation system ensured accurate coverage of the entire field, with minimal overlap and no gaps.
- Map generation of field.
  - The quadcopter generated a map of the crop field and its boundaries.

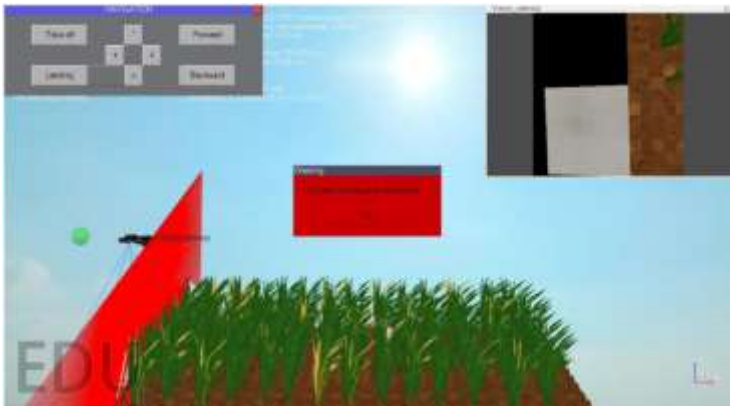

Figure 9. Landing warning

- The map was used to plan the optimal flight path for the quadcopter, ensuring efficient coverage of the entire field.

- Acquisition of image frames of crop field.
- The aerial robotics simulation environment successfully captured image frames of a crop field with a resolution of 512x512 pixels.
- The images were captured at a rate of 30 frame per second which were then converted into a video providing a smooth and continuous view of the field.
- The images were stored on the storage device for future reference.

- Application of image processing techniques on image frames of crop field.
- The pre-processing techniques enhanced the detection results.
- The processed images were analyzed for color features to identify potential disease symptoms.

- Analysis of fusarium yellow disease detection in gladiolus field.
- The algorithms identified areas of interest (diseased areas) with an accuracy of 95%.

Experiments indicated a significant improvement in the ability to detect variations in crop health through drone simulations compared to traditional methods. The analysis revealed that the drone could effectively identify areas of stress within the crop, highlighting regions requiring immediate attention. This capability not only allows for timely interventions but also minimizes resource wastage, as input applications can be targeted rather than uniformly applied.

Furthermore, the integration of drone technology reduced the time required for data collection from several hours of manual labor to approximately 15 minutes of flight. This stark contrast underscores the potential of UAVs to enhance operational efficiency in agricultural practices, allowing farmers to focus on strategic decision-making.

B. System Testing.

System testing is a level of testing in which a complete and integrated system is tested. The purpose of system testing is to verify that it meets specified requirements of a system. For system testing, black box testing approach is used which involves the external workings of the system from the user's perspective.

User's experience and satisfaction are trivial criteria to evaluate a simulation environment. The proposed system has been designed for developers to test their agricultural techniques in a simulated environment. Several experiments have been conducted to test the usability of the developed simulated environment through a survey. The purpose of this survey was to evaluate the efficiency and credibility of the proposed robotic simulation environment for crop

monitoring in accordance with the developer's expectations. The findings will assist developers and researchers to improve the functionality of the system according to user's requirements and ease of use.

C. Qualitative Analysis.

The questionnaire was formed by employing user experience (UX) analysis. UX provides insight into user's perception and user satisfaction that how users feel about a system, such as ease of use, perception of the value of the simulation, utility, and efficiency in performing tasks. The main goal of the User Experience Questionnaire was to allow a fast and immediate measurement of user experience of proposed system [31]. From user experience design coined by Peter Morville [32], the following evaluation factors have been selected.

- Usefulness - All the features of the system must fulfill needs and must be helpful.
- Usability - System must be efficient and easy to use.
- Findability - All the components of the graphical user interface of the system must be easily locatable.
- Credibility - Users must trust and believe all the information provided by the system.

We chose 15 users to run and analyze the simulation environment. Half of the users were experts on 3D simulations while the other half of the developers have no experience with such simulations.

Table 1. Results of qualitative analysis of user evaluation

Expert Users Evaluation				
	Usefulness	**Usability**	**Findability**	**Credibility**
Autonomous Motion	81%	73%	79%	77%
Manual Control	83%	81%	75%	84%
Image Capturing	84%	85%	87%	80%
Disease Detection	75%	80%	77%	76%
Naive Users Evaluation				
Autonomous Motion	81%	78%	81%	80%
Manual Control	80%	83%	92%	76%
Image Capturing	90%	74%	78%	80%
Disease Detection	86%	81%	72%	81%

The users were asked to test the simulation in both Autonomous and Manual modes. In Autonomous mode, they observe the take-off and landing process, autonomous path navigation and image acquisition by the quadcopter. Disease detection module was also evaluated in terms of how many infected plants have been detected implying the area successfully covered by the quadcopter. In manual mode, the quadcopter navigation through remote control, image acquisition and disease detection with regard to area coverage were analyzed.

Users were later asked to fill in a questionnaire. The answers to the questionnaire were based on a 5-point Likert scale. An average percentage was calculated based on the values generated under each section. Results depict a positive user experience with the developed simulation environment as shown in Table 1.

D. Quantitative Analysis.

The user's given points to each module of the system were statistically evaluated to give more meaning to the results. Likert scale points given by all users to each category were used to calculate Standard Deviation (s) and Confidence Interval (CI). Formulas used are as follows.

$$\bar{x} = \frac{\sum x}{n} \tag{2}$$

$$s = \sqrt{\frac{\Sigma(x-\bar{x})^2}{n-1}} \quad (3)$$

$$\mathrm{CI} = \bar{x} \pm Z * \frac{s}{\sqrt{n}} \quad (4)$$

Where $\bar{X}$ is the mean, s is the standard deviation of the sample and n is the sample size. CI is calculated with 95% confidence and Z as 1.960. Table 2 summarizes the results of this analysis.

Table 2. Results of quantitative analysis of the user evaluation

	Mean	Standard Deviation	Confidence Interval	
			Lower bound	Upper bound
Autonomous Motion	78.75	2.76	76.83	80.67
Manual Control	81.75	5.28	78.09	85.41
Image Capturing	82.25	5.20	78.64	85.86
Disease Detection	78.5	4.38	75.47	81.53

E. Discussion.

Experiments performed by both naïve and expert users vary in their experiences. This variation is evident in the higher values of standard deviation; for instance, in case of 'Manual Control' and 'Image Capturing' modules. It emphasizes the need of efficient autonomous control of the quadcopter which can be specially helpful for the farmers and other non-technical persons. Both qualitative and quantitative analysis performed by the users advocate the following key observations.

- Image capturing is the task which the naive users find most difficult to handle, especially in manual mode.
- All the users experience better take-off and landing of the quadcopter in autonomous mode.
- Naive users value the 'disease detection' feature the most.
- Expert users prefer to control the quadcopter manually.

The findings highlight the potential of drone technology in enhancing agricultural practices through the automation of health assessments. The use of ROS within a simulated environment provided a flexible and efficient platform for testing and refining health analysis methodologies. Despite the promising results, several opportunities for enhancement were noted during the experiments. The accuracy of the health assessments may be influenced by factors such as lighting conditions and the resolution of sensor data. Additionally, the dependency on simulations may overlook certain real-world variables, including soil conditions and pest infestations that can affect plant health.

Conclusion.

The developed simulation environment is an ideal playground for testing and evaluating the automation of various agricultural processes. In this work, an autonomous mapping and navigation system for a quadcopter has been designed to inspect and detect Fusarium yellow disease in gladiolus crop field. The work paves the way for further research and development, aiming to refine the algorithms and enhance the capabilities of drones for precision agriculture. Future studies will focus on integrating machine learning techniques to automate data interpretation, offering farmers even deeper insights into crop management and health optimization. Further agricultural machinery can be added in the proposed system to develop a collaborative environment where heterogeneous autonomous machines will be working in the field to achieve a mutual goal.

Acknowledgement. We are grateful to Green Works[2] for supporting the project and making their farms available for performing experiments on gladiolus crops.

Author's Contribution. A. S. Akkas. Resources, Methodology, Software; N. Ejaz. Data curation, Validation, Methodology; K. Atif. Validation, Writing, Review & editing; M. Anjum. Resources, Conceptualization, Supervision; S. A. Mehdi. Conceptualization, Supervision, Project administration.

Conflict of interest. The authors have no conflicts of interest to declare that are relevant to the content of this article.

References.

[1] Food and Agriculture Organization, "How to Feed the World in 2050," *Exec. Summ. Expert Meet. How to Feed World 2050, Rome, Italy*, 2009, [Online]. Available. https.//www.fao.org/fileadmin/templates/wsfs/docs/expert_paper/How_to_Feed_the_World_in_2050.pdf

[2] A.-K. Mahlein, "Plant Disease Detection by Imaging Sensors – Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping," vol. 100, no. 2, pp. 241–251, Jan. 2016, doi. 10.1094/PDIS-03-15-0340-FE.

[3] Y. Fang and R. P. Ramasamy, "Current and prospective methods for plant disease detection," *Biosensors*, vol. 5, no. 3, pp. 537–561, 2015, doi. 10.3390/bios5030537.

[4] K. Steddom, M. Bredehoeft, M. Khan, and C. Rush, "Comparison of Visual and Multispectral Radiometric Disease Evaluations of Cercospora Leaf Spot of Sugar Beet," *Plant Dis.*, vol. 89, no. 2, pp. 153–158, 2005, doi. 10.1094/PD-89-0153.

[5] M. Mahmud, M. Abidin, and Z. Mohamed, "Crop identification and navigation design based on probabilistic roadmap for crop inspection robot," *Int. Conf. Agric. Food Eng.*, vol. 23, p. 25, 2016, [Online]. Available. https.//www.researchgate.net/publication/314152363_Crop_identification_and_navigation_design_based_on_probabilistic_roadmap_for_crop_inspection_robot

[6] H. Mansur, S. Welch, L. Dempsey, and D. Flippo, "Importance of Photo-Realistic and Dedicated Simulator in Agricultural Robotics," *Eng.*, vol. 15, pp. 318–327, 2023, doi. 10.4236/eng.2023.155025.

[7] M. H. Zaheer, S. A. Mehdi, H. E. Keen, and K. Berns, "Detection of Fungus in Gladiolus Fields Using a Quadcopter," *Mech. Mach. Sci.*, vol. 102, pp. 127–134, 2021, doi. 10.1007/978-3-030-75259-0_14.

[8] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," *2004 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 3, pp. 2149–2154, 2004, doi. 10.1109/IROS.2004.1389727.

[9] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim. High-Fidelity Visual and Physical Simulation for Autonomous Vehicles," *Springer Proc. Field Serv. Robotics.*, vol. 5, pp. 621–635, 2018, doi. 10.1007/978-3-319-67361-5_40.

[10] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza, "Flightmare. A Flexible Quadrotor Simulator," *Proc. Conf. Robot Learn. (CoRL), PMLR*, pp. 1147–1157, 2021, [Online]. Available. https.//rpg.ifi.uzh.ch/docs/CoRL20_Yunlong.pdf

[11] M. Nikolaiev, M. Novotarskyi, "Comparative Review of Drone Simulators," *Information, Comput. Intell. Syst.*, vol. 4, 2024, doi. https.//doi.org/10.20535/2786-8729.4.2024.300614.

[12] A. Koubaa, *Robot Operating System (ROS)*, vol. 1. in Studies in Computational Intelligence, vol. 625. Cham. Springer International Publishing, 2016. doi. 10.1007/978-3-319-26054-9.

[2] www.facebook.com/GreenworksFloricultureAndHorticultureSolution/

[13] N. Tsolakis, D. Bechtsis, and D. Bochtis, "AgROS. A Robot Operating System Based Emulation Tool for Agricultural Robotics," *Agronomy*, vol. 9, no. 403, 2019, doi. https.//doi.org/10.3390/agronomy9070403.

[14] CoppeliaSim, "CoppeliaSim User Manual," Version 4.9, Accessed. Feb 27, 2025. [Online] Available. https.//manual.coppeliarobotics.com/index.html

[15] E. Rohmer, S. P. N. Singh, and M. Freese, "V-REP. A versatile and scalable robot simulation framework," *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 1321–1326, 2013, doi. 10.1109/IROS.2013.6696520.

[16] B. Van De Walker, B. Byrne, B. Purdie, et al. "Developing a Realistic Simulation Environment for Robotics Harvesting Operations in a Vegetable Greenhouse," *Agronomy*, vol. 11, no. 9, p. 1848, 2021, doi. https.//doi.org/10.3390/agronomy11091848.

[17] S. Kohlbrecher, "Hector SLAM," Accessed.Feb 27, 2025. [Online] Available. https.//wiki.ros.org/hector_slam/Tutorials

[18] S. M. LaValle, J. J. Kuffner, "Rapidly-Exploring Random Trees. Progress and Prospects" *Algorithmic Comput. Robot.*, pp. 303–307, Apr. 2001, doi. 10.1201/9781439864135-43.

[19] J. Pak, J. Kim, Y. Park, and H. I. Son, "Field Evaluation of Path-Planning Algorithms for Autonomous Mobile Robot in Smart Farms," *IEEE Access*, vol. 10, pp. 60253–60266, 2022, doi. 10.1109/ACCESS.2022.3181131.

[20] M. F. Heimann and G. L. Worf, "Gladiolus disorder. Fusarium yellows and bulb rot,", Accessed. 27 Feb, 2025. [Online] Available. https.//barron.extension.wisc.edu/files/2023/02/Gladiolus-Disorder-Fusarium-Yellows-and-Bulb-Rot.pdf

[21] M. K. N. Shah, M. B. J. Dutt, and H. Modh, "Quadrotor – An Unmanned Aerial Vehicle", *Int. Jour. Eng. Dev. Res.*, vol. 2, no. 1, pp. 1299-1303, March 2014, [Online] Available. https.//rjwave.org/IJEDR/papers/IJEDR1401231.pdf

[22] S. A. Mehdi, C. Armbrust, J. Koch, and K. Berns, "Methodology for robot mapping and navigation in assisted living environments," *Proc. 2nd Int. Conf. PErvasive Technol. Relat. to Assist. Environ.*, pp. 1–6, 2009, doi. 10.1145/1579114.1579176.

[23] H. Omrane, M. S. Masmoudi, and M. Masmoudi, "Fuzzy logic based control for autonomous mobile robot navigation," *Comput. Intell. Neurosci.*, 2016, doi. 10.1155/2016/9548482.

[24] J. Zhao, S. Liu, and J. Li, "Research and Implementation of Autonomous Navigation for Mobile Robots Based on SLAM Algorithm under ROS," *Sensors*, vol. 22, no. 11, p. 4172, 2022, doi. 10.3390/s22114172.

[25] H. Taheri and Z. C. Xia, "SLAM; definition and evolution," *Eng. Appl. Artif. Intell.*, vol. 97, p. 104032, Jan. 2021, doi. 10.1016/J.ENGAPPAI.2020.104032.

[26] J. M. Santos, D. Portugal, and R. P. Rocha, "An evaluation of 2D SLAM techniques available in Robot Operating System," *2013 IEEE Int. Symp. Safety, Secur. Rescue Robot. SSRR 2013*, 2013, doi. 10.1109/SSRR.2013.6719348.

[27] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," *9th IEEE Int. Symp. Safety, Secur. Rescue Robot. SSRR 2011*, pp. 155–160, 2011, doi. 10.1109/SSRR.2011.6106777.

[28] J. P. Vásconez, F. Basoalto, I. C. Briceño, J. M. Pantoja, R. A. Larenas, J. H. Rios, and F. A. Castro, "Comparison of path planning methods for robot navigation in simulated agricultural environments," *Procedia Comput. Sci.*, vol. 220, pp. 898–903, 2023, doi. 10.1016/j.procs.2023.03.122.

[29] S. LaValle, "Rapidly-exploring random trees . a new tool for path planning," *Annu. Res. Rep. 9811*, 1998, Accessed. 27 Feb, 2025. [Online] Available.

https://lavalle.pl/papers/Lav98c.pdf
[30] H. H. A. Kadouf and Y. M. Mustafah, "Colour-based Object Detection and Tracking for Autonomous Quadrotor UAV," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 53, 2013, doi. 10.1088/1757-899X/53/1/012086.
[31] A. K. Chaubey, "Comparison of The Local and Global Thresholding Methods in Image Segmentation," World Journal of Research and Review, vol. 2, no. 1, 2016. Accessed. Feb. 27, 2025. [Online]. Available. https://www.academia.edu/29433086/Comparison_of_The_Local_and_Global_Thresholding_Methods_in_Image_Segmentation
[32] D. Benyon, "Designing User Experience," Pearson UK, 2019, [Online] Available. https://books.google.com.pk/books?id=MXqFDwAAQBAJ
[33] P. Morville, "User Experience Design," Semantic Studios. Accessed. Feb. 27, 2025. [Online]. Available. https://semanticstudios.com/user_experience_design/