High Level Design (HLD)

UGV (Unmanned Ground Vehicle) based Agriculture solution

Revision Number: 4.0
Last date of revision: 30/06/2021

Arvind Chauhan
Hemanth Ganesh

Document Version Control

Date Issued	Version	Description	Author
06th June 2021	1.0	First Version of Complete HLD	Arvind Chauhan Hemanth Ganesh
20th June 2021	2.0	Updated Version of Complete HLD Added KPI	Arvind Chauhan Hemanth Ganesh
25th June 2021	3.0	Added details about ROS and some images for tomato crop disease images	Arvind Chauhan Hemanth Ganesh
30th June 2021	4.0	Added list of figures	Arvind Chauhan Hemanth Ganesh

Contents

List of Figures

Abstract

UGVs can be operated with or without human presence. UGVs can be used in different areas including agriculture. Here we are focusing only on agriculture. There are a variety of purposes where it can be helpful in agriculture such as soil sampling, irrigation management, precision spraying, mechanical weeding, crop harvesting, and disease detection, etc.

Here, we will be considering one use case that is tomato crop disease detection i.e., we will collect the different data samples for healthy and infected crops, then we will use those image samples for training our deep learning model. Once the model is ready, we will deploy that model using UGV which will collect the images from the field and classify the crops into various categories i.e., whether that crop is healthy or infected with a certain disease. The other one is weather prediction, this will forecast the weather of that area which will help farmers to give the information about temperature, moisture, rain forecast, etc., and will help to make decisions accordingly.

1 Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions before coding and can be used as a reference manual for how the modules interact at a high level.

The HLD will,

- Present all of the design aspects and define them in detail
- Describe the user interface is implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project
- List and describe the non-functional attributes like:
    - Security
    - Reliability
    - Maintainability
    - Portability
    - Reusability
    - Application compatibility
    - Resource utilization
    - Serviceability

1.2 UGV Background

As the world population keeps increasing, it needs a corresponding increase in food production with efficient and safe processes to the ecosystem. Going forward, the availability of manual labour will be minimal. So, machines powered with AI are used for various tasks in agriculture like soil sampling, irrigation management, mechanical weeding, crop harvesting, etc. To meet these requirements, we need unmanned ground vehicles.

The idea of an Unmanned Vehicle was developed after the concept of a working model of a remote-controlled car. The car was unmanned and controlled wirelessly via radio; it was thought the technology could be used in agriculture too. Thus, Unmanned Ground Vehicle was introduced in Agriculture. UGVs are one type of agricultural robot. UGVs operate on the ground without an onboard of man. They can be operated around the clock.

UGVs combine information and communication technologies, robots, artificial intelligence, big data, and the internet of things. Agricultural UGVs are highly capable, and their use has expanded across all areas of agriculture, including pesticide and fertilizer spraying, seed sowing, and growth assessment and mapping. Accordingly, the market for agricultural UGVs is expected to continue growing with the related technologies.

Generally, the vehicle will have a set of sensors to observe the environment, and will either autonomously make decisions about its behaviour or pass the information to a human operator at a different location who will control the vehicle through teleoperation.

Based on its application, unmanned ground vehicles will generally include the following components: platform, sensors, control systems, guidance interface, communication links, and systems integration features.

1.2.1 Platform

The platform can be based on an all-terrain vehicle design and includes the locomotive apparatus, sensors, and power source. Tracks, wheels, and legs are the common forms of locomotion

1.2.2 Sensors

A primary purpose of UGV sensors is navigation, another is environment detection.

1.2.3 Control systems

Unmanned ground vehicles are generally considered Remote-Operated and Autonomously.

1.2.4 Remote operated

A remote-operated UGV is a vehicle that is controlled by a human operator via an interface.

1.3 Problem Statement

To speed up manual work or manage manpower shortages, robots are used in farm activities such as Weed identification and identifying diseased tomato crops among many others. This can lead to productivity gains with minimization of errors, and consistency of work quality.

Our mission is to build a smart machine that is designed to detect, identify & make critical decisions around each & every plant in an agricultural field with the help of Computer Vision Techniques.

Nowadays, technology is growing so rapidly and is implemented in many major areas of interest. One of the most important areas is agriculture, which is given less importance as compared to other engineering activities. Good care of crops leads to better hygiene. It is important to yield good crops which are free from any diseases for human growth.

It is essential to identify any crop diseases before it gets wasted or before it has been accidentally consumed. Crop diseases are sometimes difficult to identify for the naked eye. Thus, it is a very important task to identify crop diseases and AI comes in handy here. AI systems are proved to identify crop diseases with accuracy sometimes greater than human-level accuracy.

1.3.1 Use Cases

Identify the diseased tomato crop among the field and notify the farmer with a proper solution to resolve the issue.

We use the Object Detection Technique to identify the diseased tomato crop by providing image analysis in real-time. This can be used by the farmers for disease detection by pre-processing images of leaves.

1.3.2 Application Flow for Use Case

Fig. 01 describe the application flow for disease detection for tomato crop.

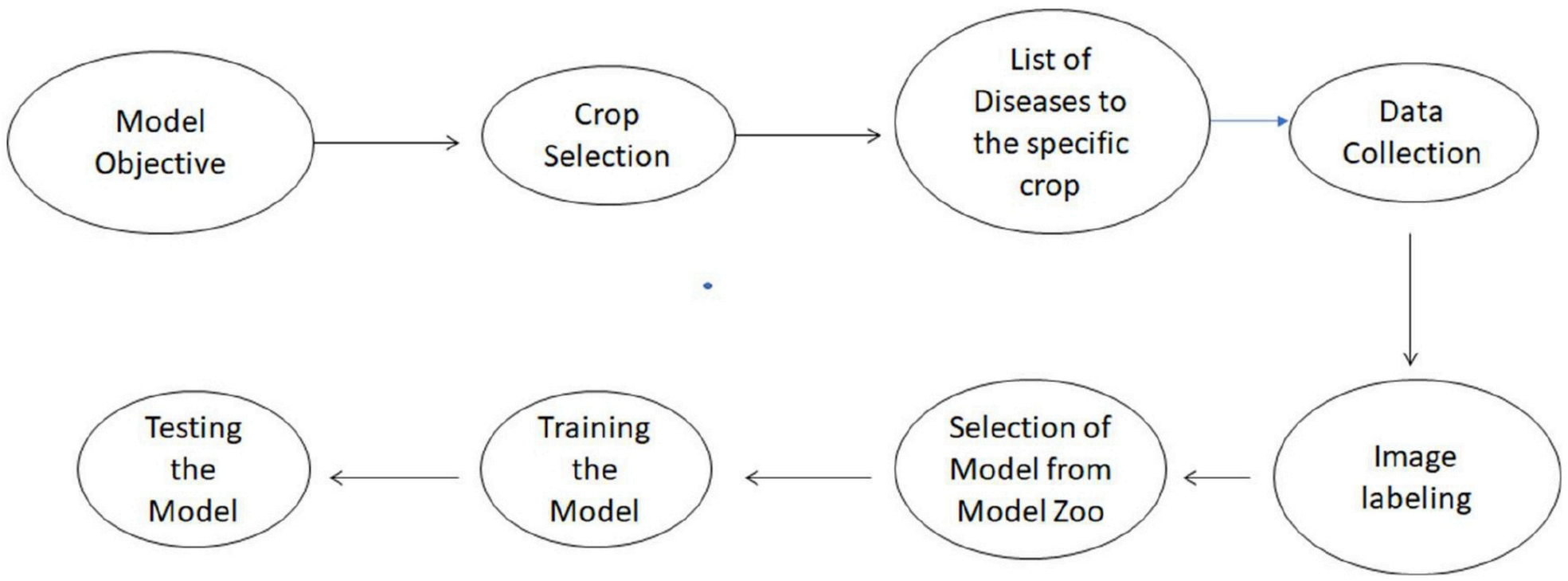

Fig. 01: Application Flow for Disease Detection Use Case

Another use case is weather prediction, this will forecast the weather prediction that will help farmers for giving the information about temperature, moisture, rain forecast, etc., and will help to take decisions accordingly.

1.4 Definitions

Term	*Description*
UGV	Unmanned Ground Vehicle
Database	Collection of all the information monitored by this system
IDE	Integrated Development Environment
AWS	Amazon Web Services

2 General Description

2.1 Product Perspective

The UGV based agriculture solution system is a deep learning-based object detection model which will detect the diseases of tomato crop and will intimate to farmers about the diseases. Also, this will forecast the weather prediction that will help farmers for giving the information about temperature, moisture, rain forecast, etc., and will help to make decisions accordingly.

2.2 Technical Requirements

Different crops require different ways of farming. This document addresses the requirements for the tomato crop. Mobile platforms like Ground robots should be used for this purpose. Ground Robots can be based on wheels, tracks, or legs.

- UGV should be able to move between the rows of tomato crops without causing damage to the crop. Wheel robots or Wheel-Legged robots would better fit this tomato crop.
- These UGVs' should include many sensors like stereo vision cameras, panoramic cameras, thermal and infrared detecting systems, to name a few, to assess the crop properties.
- These can be battery-powered or solar-powered.
- UGVs' should be equipped with proper computing power to process the images of the crops taken.

2.3 Data Requirements

There are numerous image file formats out there so it can be hard to know which file type best suits your image needs (on your requirement).

    - **TIFF** – Tagged image file format
    - **BMP** – Bitmap image file format
    - **JPEG** - Joint photographic experts groups
    - **GIF** - graphics interchange format
    - **PNG** – portable network graphics
    - **EPS** – encapsulated post script
    - **RAW** image files
- Tiffs are great for printing. These are lossless image files meaning they don't need to compress or lose any image quality or information. These format images are high-quality images.

- BMP format developed by Microsoft for windows. There is no compression or information loss; this format is generally recommended for high-quality scans.
- JPEG is a lossy format meaning that the image is compressed to make a smaller file but this loss is not noticeable.
- JPEG is a very popular format for digital cameras.
- GIFs are widely used for web graphics because they are limited to only 256 colors, can allow for transparency, and can be animated. These types of files are typically small in size and very portable.
- PNG is a lossless image format; these files can handle up to 16 million colors unlike the 256 colors supported by GIF.
- EPS is a common vector-type file.
- RAW images that are unprocessed have been created by a camera or scanner. Digital cameras can shoot in raw, mostly used in photography.

If the data is in video format like (MP4) convert it into images based on FPS (no. of frames displayed per second) in real-time processing. There is a number of tools to convert videos into images. Using cv we can convert video into images.

2.3.1 Tomato Crop Dataset

We are taking tomato crop data for detecting the disease leaf from the dataset. We are considering four diseases as Bacterial Spot, Late Blight, Septoria Leaf Spot, and Yellow Curved. These diseases are detecting by our proposed model.

2.3.1.1 Bacterial Spot

The symptoms consist of numerous small, angular to irregular, water-soaked spots on the leaves and slightly raised to scabby spots on the fruits. The leaf spots may have a yellow halo. The centers dry out and frequently tear. Fig.02 has sample dataset for Bacterial Spot,

Fig. 02: Sample data for Bacterial Spot Disease

2.3.1.2 Late Blight

Late blight is especially damaging during cool, wet weather. Young leaf lesions are small and appear as dark, water-soaked spots. These leaf spots will quickly enlarge, and a white mold will appear at the margins of the affected area on the lower surface of the leaves. Complete defoliation (browning and shrivelling of leaves and stems) can occur within 14 days from the first symptoms. Fungal spores are spread between plants and gardens by rain and wind. Fig.03 has sample dataset for Late Blight,

Fig. 03: Sample data for Late Blight Disease

2.3.1.3 Septoria Leaf Spot

This destructive disease of tomato foliage, petioles, and stems. Infection usually occurs on the lower leaves near the ground, after plants begin to set fruit. Numerous small, circular spots with dark borders surrounding a beige-colored center appear on the older leaves. Tiny black specks, which are spore-producing bodies, can be seen in the center of the spots. Severely spotted leaves turn yellow, die, and fall off the plant. Fig.04 has sample dataset for Septoria Leaf Spot,

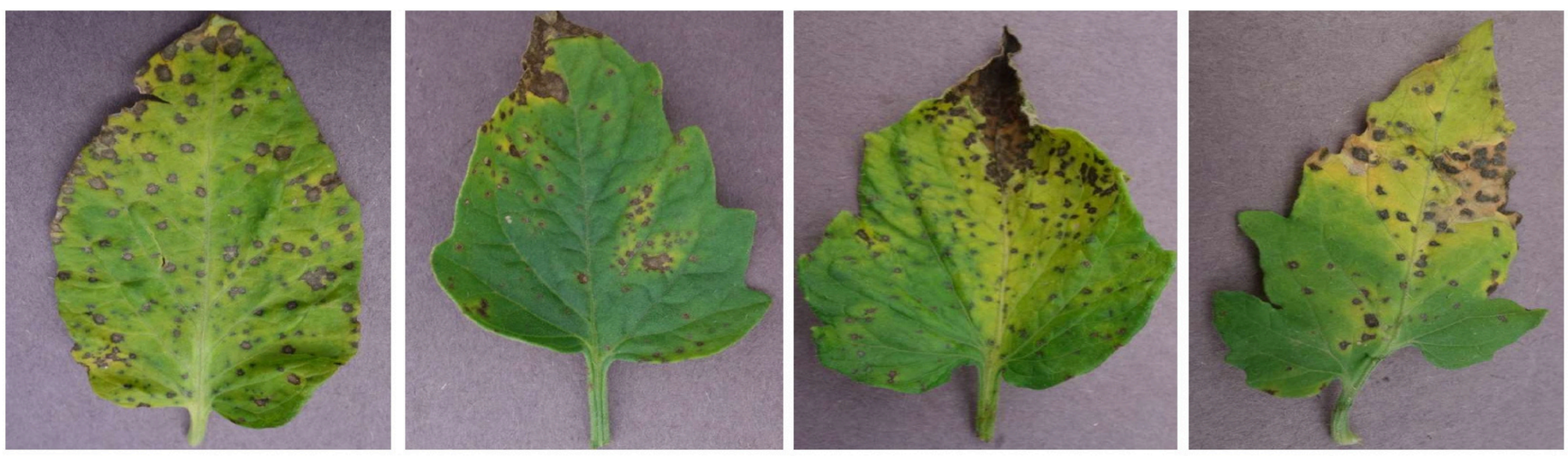

Fig. 04: Sample data for Septoria Leaf Spot Disease

2.3.1.4 Yellow Curved

The symptoms include severe stunting, marked reduction in leaf size, upward cupping, chlorosis of leaf margins, mottling, flower abscission, and significant yield reduction. Symptoms in common bean include leaf thickening, leaf crumpling, upward curling of leaves,

abnormal lateral shoot proliferation, deformation, and reduction in the number of pods. Fig.02 has sample dataset for Yellow Curved,

Fig. 05: Sample data for Yellow Curved Disease

2.4 Tools used

Python programming language and frameworks such as Numpy, Pandas, Scikit-learn, TensorFlow, Keras are used to build the whole model.

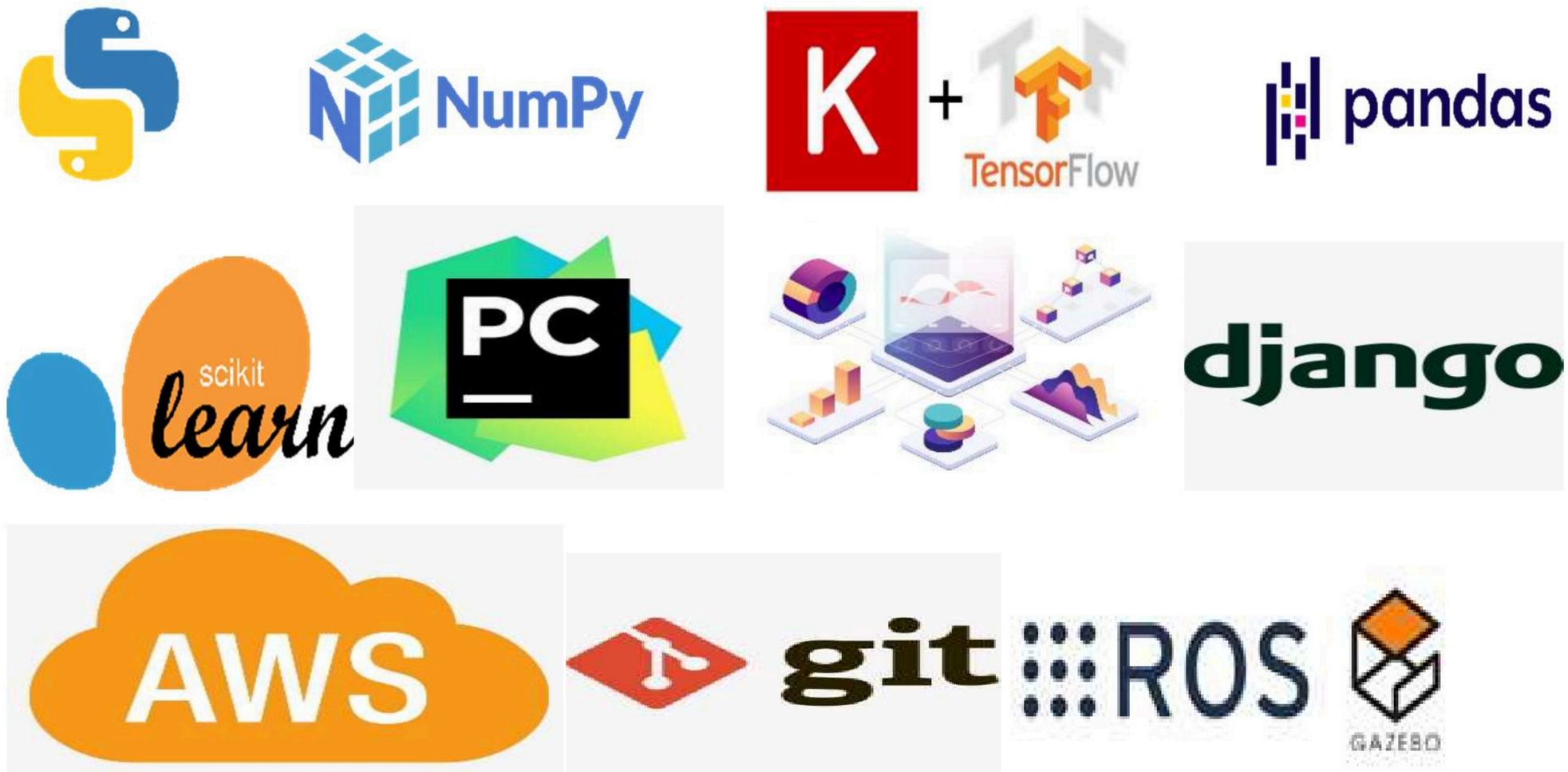

Fig. 06: Tools Used

- PyCharm is used as IDE.
- For visualization of the plots, Matplotlib, Seaborn, and Plotly are used.
- AWS/GCP is used for the deployment of the model.
- Tableau/Power BI is used for dashboard creation.
- MySQL/MongoDB is used to retrieve, insert, delete, and update the database.
- Front end development is done using HTML/CSS
- Python Django/Flask is used for backend development.
- GitHub is used as a version control system.

2.4.1 ROS (Robotic Operating System)

Robot Operating System is an open-source robotics middleware suite. Although ROS is not an operating system but a collection of software frameworks for robot software development, it provides services designed for a heterogeneous computer cluster such as hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management.

OS is a framework on top of the O.S. that allows it to abstract the hardware from the software. This means you can think in terms of software for all the hardware of the robot.

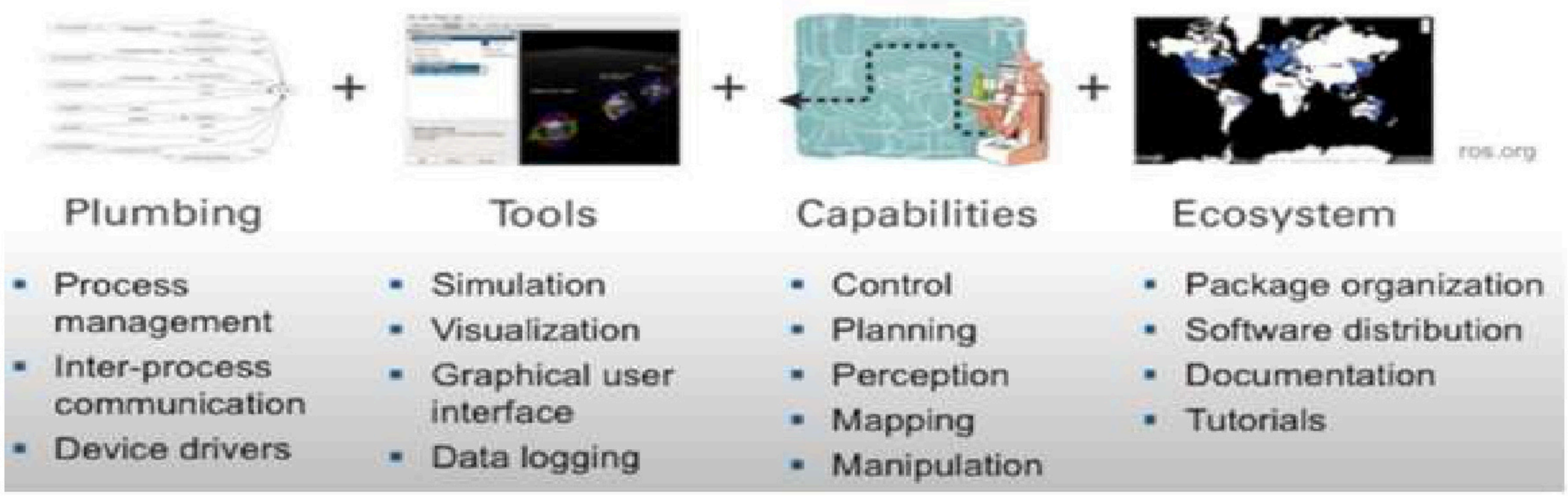

Fig. 07: ROS Framework

2.4.2 ROS Navigation

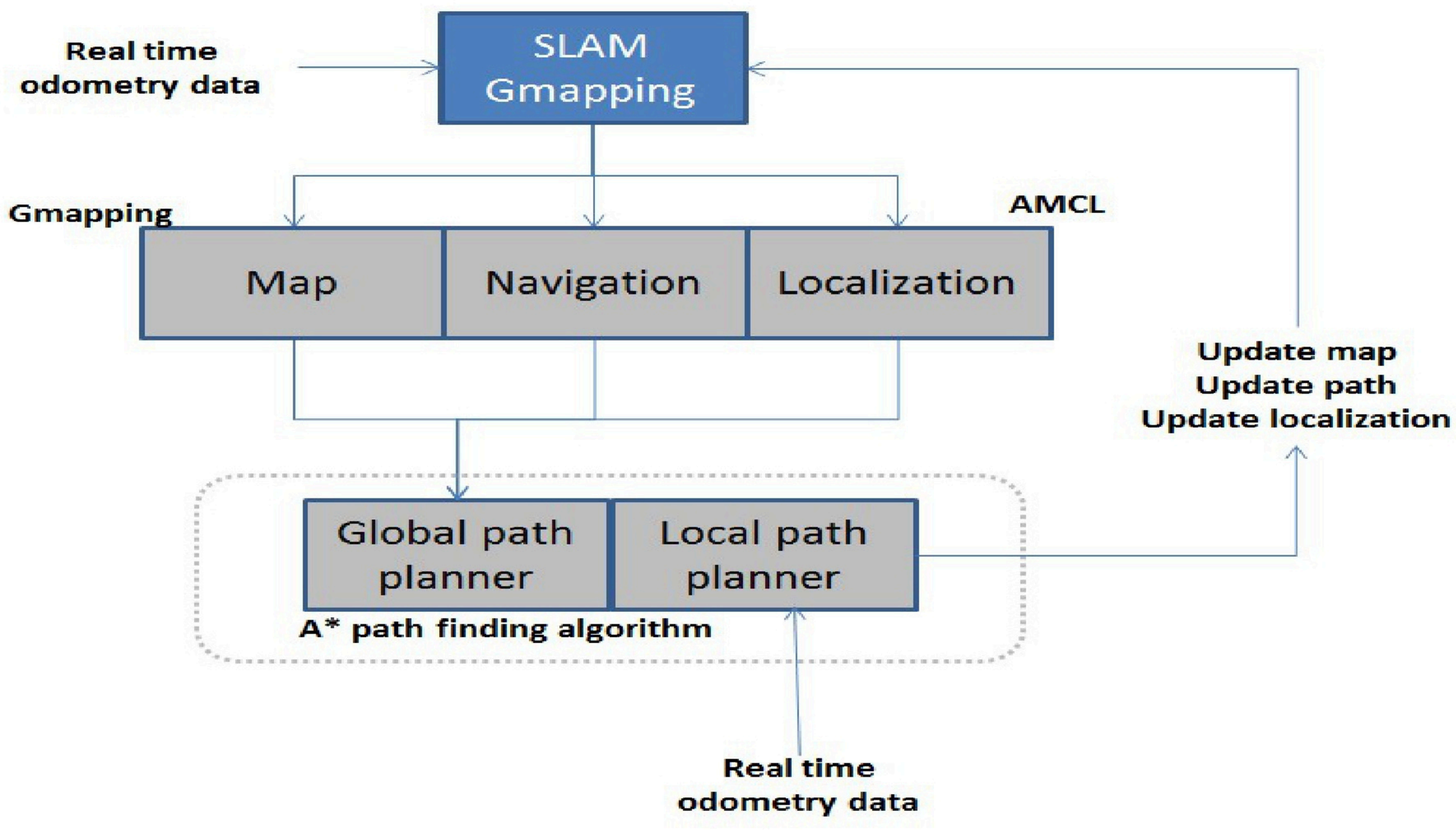

Fig. 08: ROS Navigation

2.4.3 Gazebo

The Gazebo is a 3D simulator, while ROS serves as the interface for the robot. Combining both results in a powerful robot simulator.

With Gazebo you can create a 3D scenario on your computer with robots, obstacles, and many other objects. The Gazebo also uses a physical engine for illumination, gravity, inertia, etc. You can evaluate and test your robot in difficult or dangerous scenarios without any harm to your robot. Most of the time it is faster to run a simulator instead of starting the whole scenario on your real robot.

2.4.4 Hardware

- Camera
- UGV

2.4.5 Robot Specific Features

In addition to the core middleware components, ROS provides common robot-specific libraries and tools that will get your robot up and running quickly.

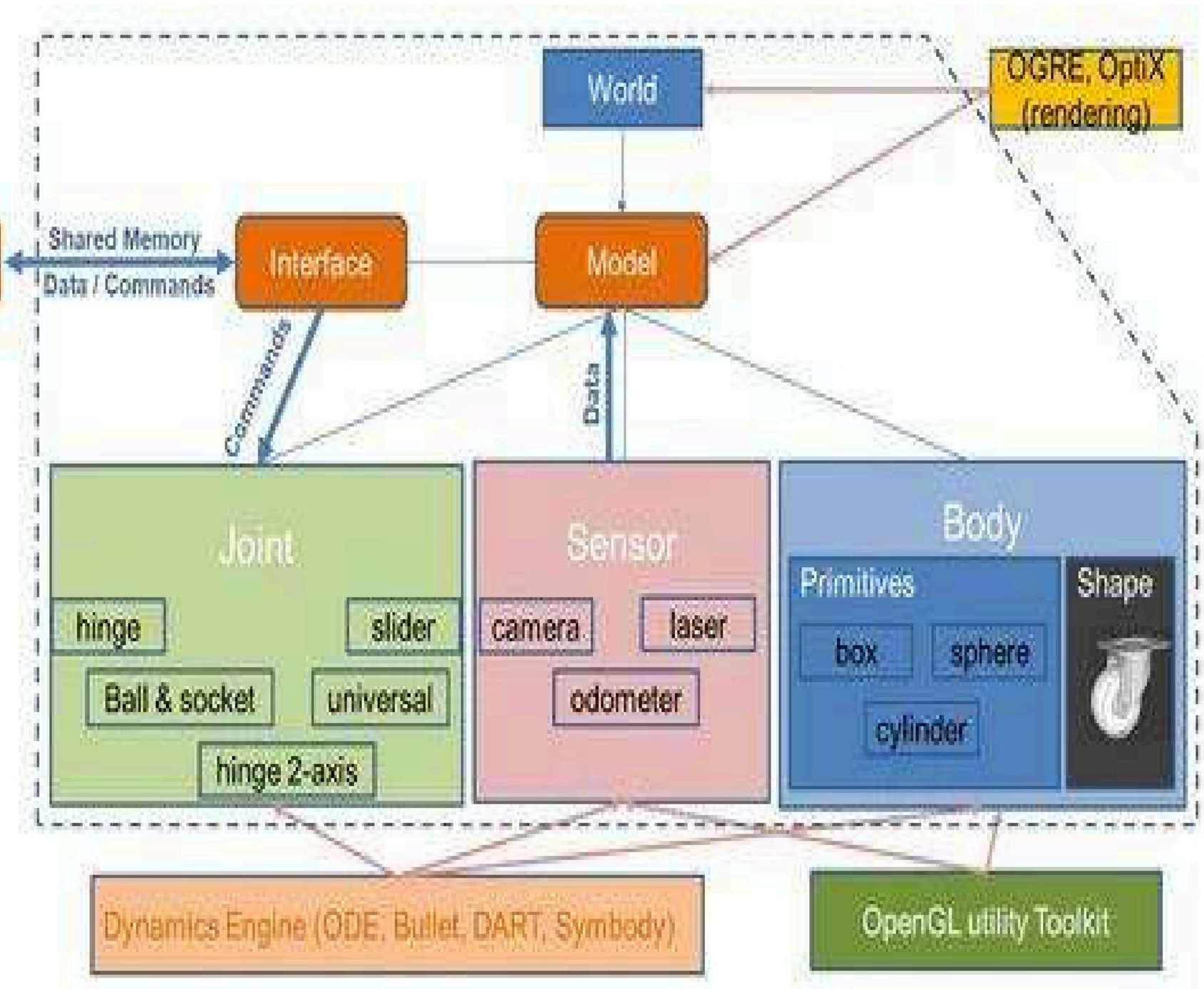

Fig. 09: ROS Components

Here are just a few of the robot-specific capabilities that ROS provides:

- Standard Message Definitions for Robots
- Robot Geometry Library
- Robot Description Language
- Preemptable Remote Procedure Calls
- Diagnostics
- Pose Estimation
- Localization
- Mapping
- Navigation

2.5 Constraints

The UGV based agriculture solution system must be user-friendly, as automated as possible and users should not be required to know any of the workings.

2.6 Assumptions

The main objective of the project is to predict the tomato crop disease for the new dataset that comes through the UGV vehicle which has a camera installed for capturing the live videos. Deep Learning based object detection model is used for detecting tomato crop diseases based on the input data. It is also assumed that all aspects of this project can work together in the way the designer is expecting. Initially, we are targeting for four diseases as Bacterial Spot, Late Blight, Septoria Leaf Spot, and Yellow Curved.

2.7 Objective

The main objective of the project is to detect tomato crop disease. Initially, we are considering the tomato crop, and then after proposed model will be used for 2-3 more crops.

- To design such a system that can detect crop disease and pests accurately.
- To help the farmers to detect the disease before spreading.
- To provide a remedy for the disease that is detected.
- To mark the location of the plant, where the infected plant is detected.

3 Design Details

3.1 Process Flow

For identifying crop diseases, we will use a deep learning-based model. Below is the process flow diagram for identifying the diseased crops.

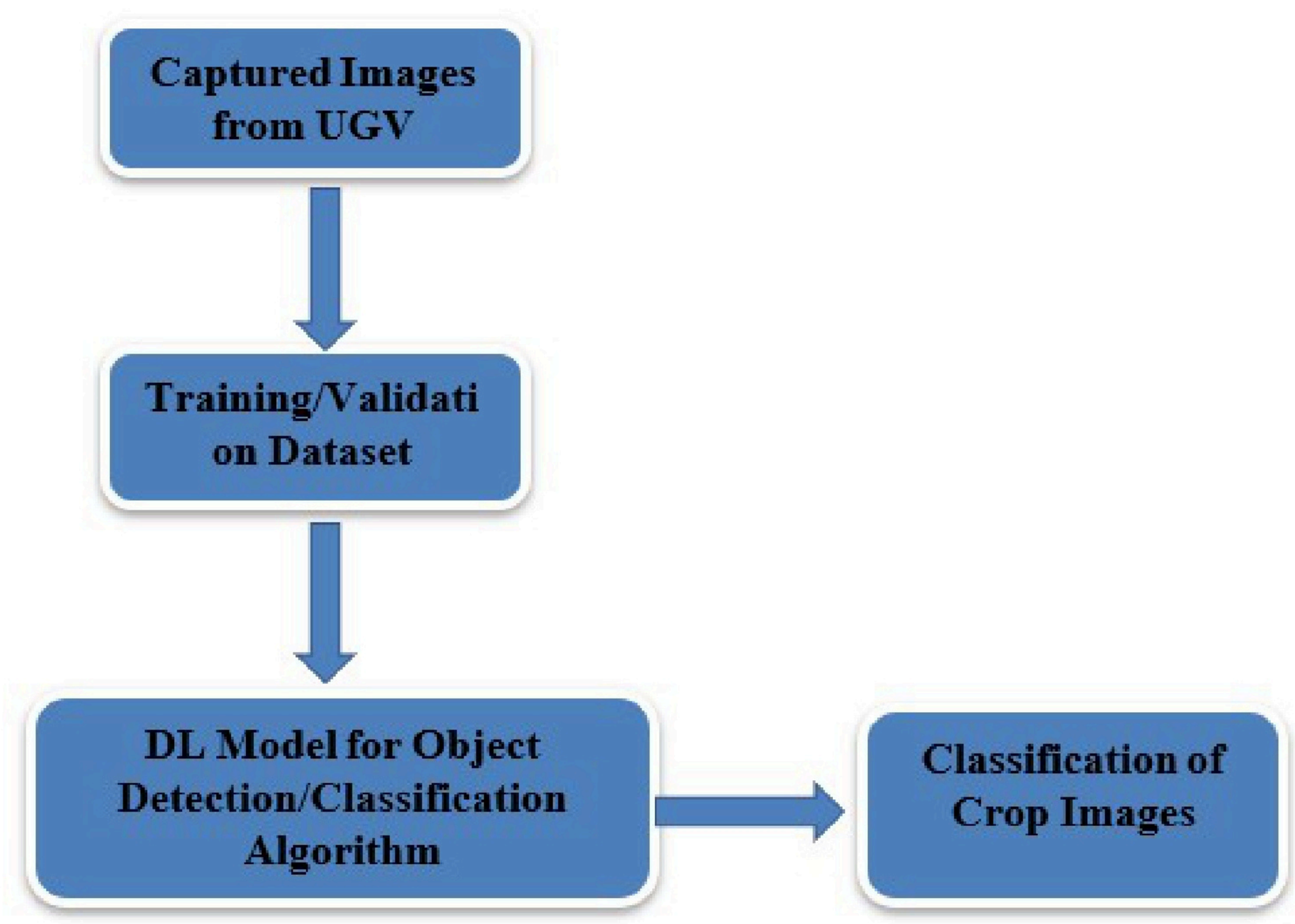

Fig. 10: Proposed methodology

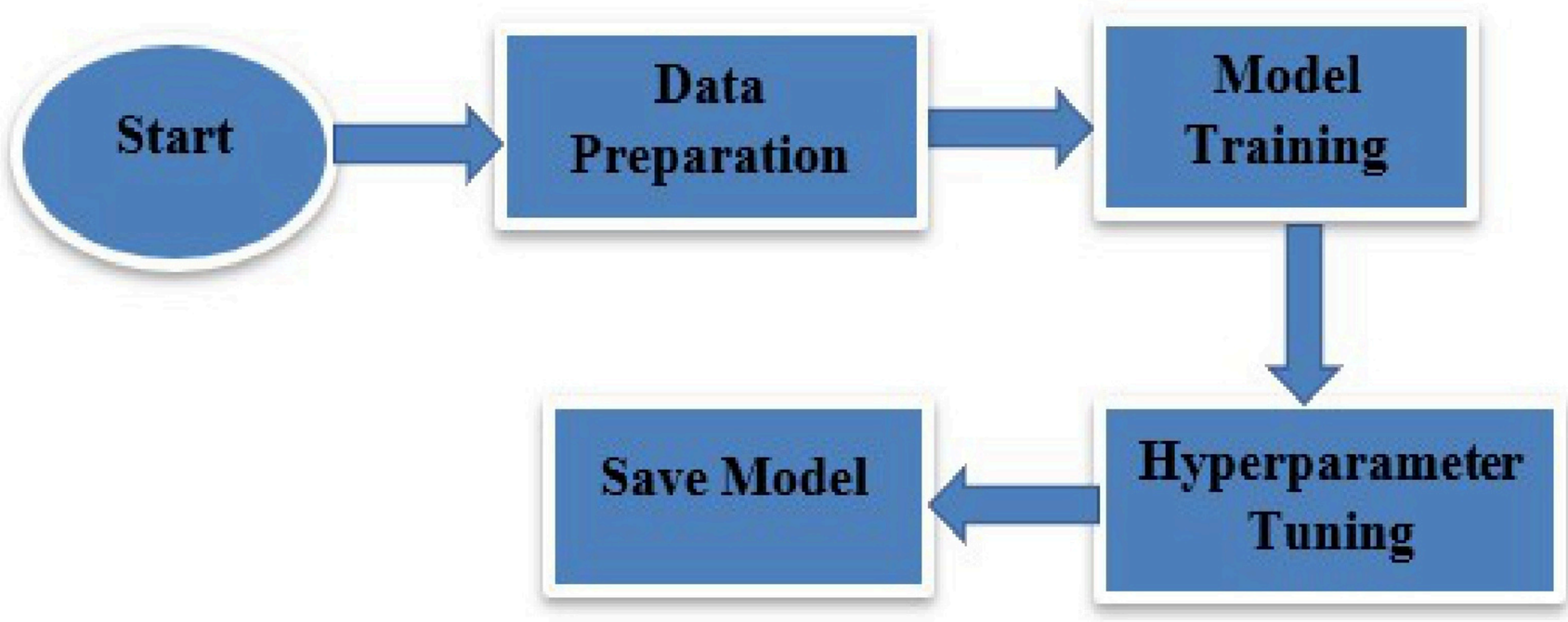

Fig. 11: Model Training and Hyperparameter Tuning

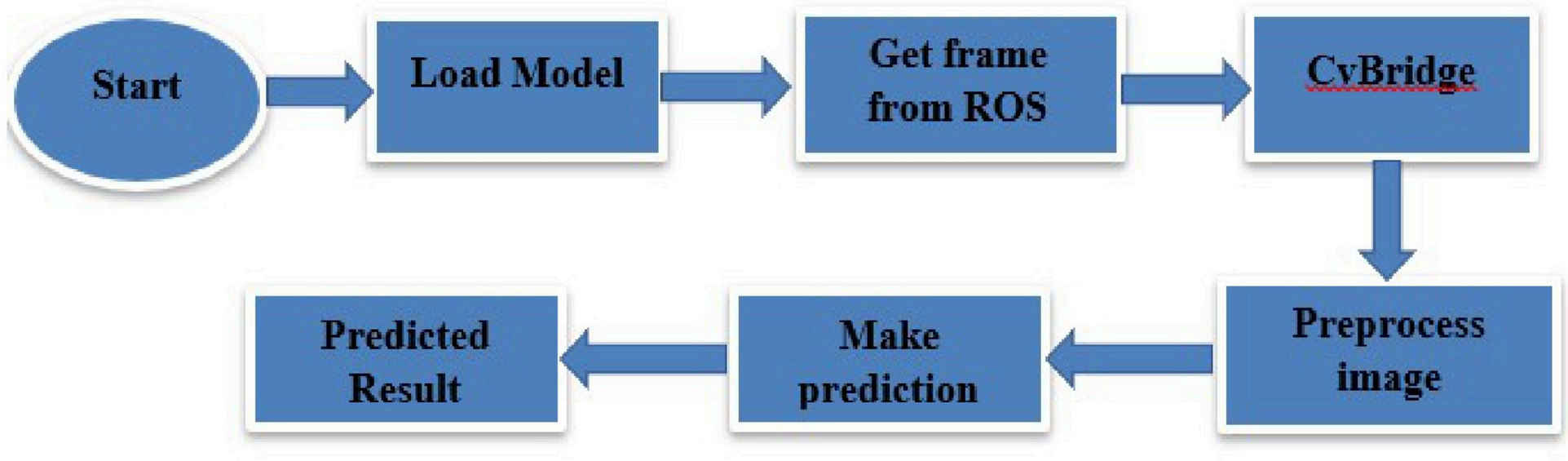

Fig. 12: Deployment Process

3.2 Event Log

The system should log every event so that the user will know what process is running internally.

Initial Step-By-Step Description:

1. The System identifies at what step logging required
2. The System should be able to log each and every system flow.
3. The developer can choose the logging method. You can choose database logging/ File logging as well.
4. The system should not hang even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.

3.3 Error Handling

Should errors be encountered, an explanation will be displayed as to what went wrong? An error will be defined as anything that falls outside the normal and intended usage.

3.4 Performance

The UGV based agriculture solution is used for predicting the disease whenever UGV will move in the field and accordingly it will intimate to the farmers of the tomato crop disease, it should be as accurate as possible. So that it will not mislead the farmers. Also, model retraining is very important to improve performance.

3.5 Reusability

The code written and the components used should have the ability to be reused with no problems.

3.6 Application Compatibility

The different components for this project will be using Python as an interface between them. Each component will have its task to perform, and it is the job of Python to ensure the proper transfer of information.

3.7 Resource Utilization

When any task is performed, it will likely use all the processing power available until that function is finished.

3.8 Deployment

Fig. 13: Cloud Deployment Technology

4 Dashboards

Dashboards will be implemented to display and indicate certain KPIs and relevant indicators for the tomato crop disease and weather forecasting.

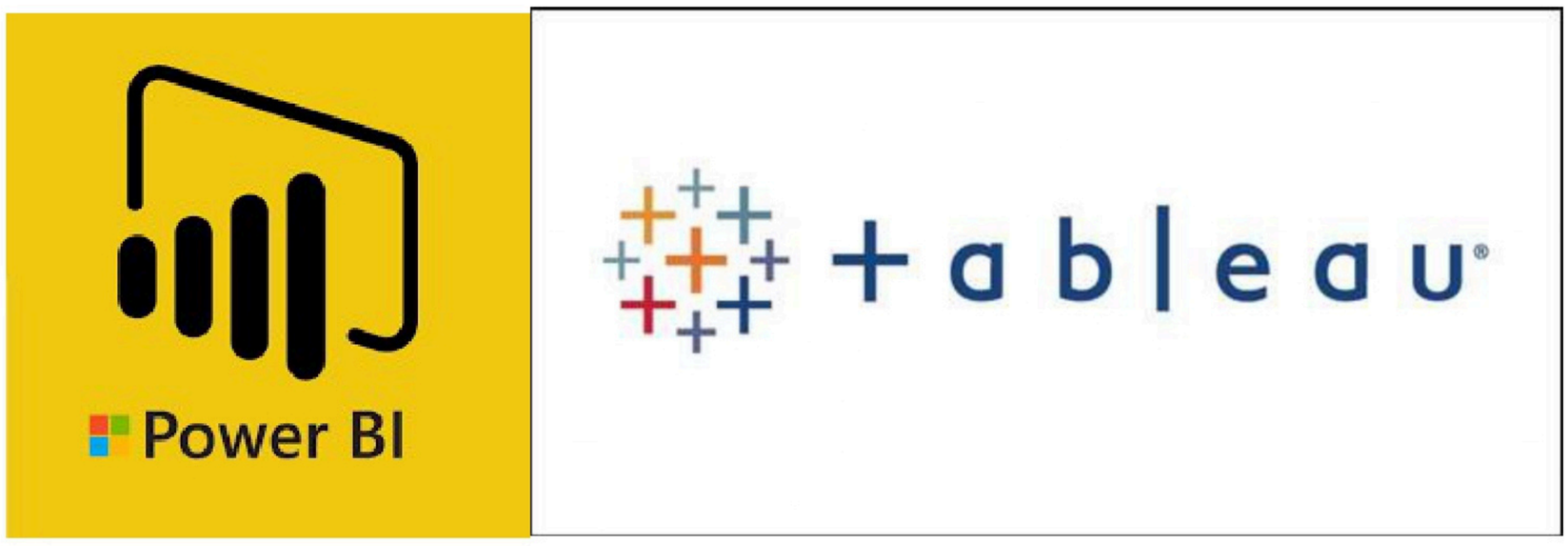

Fig. 14: Dashboards Tools

As and when the system starts to capture the historical/periodic data for a user, the dashboards will be included to display charts over time with progress on various indicators or factors

4.1 KPIs (Key Performance Indicators)

Key indicators displaying a summary of the tomato crop disease as compared to a healthy crop.

1. Time and workload reduction using the UGV based agriculture solution model.
2. Comparison of accuracy of model prediction and farmer's manual prediction.
3. Area of the field for respective crop covered by UGV based agriculture solution model.
4. The time between symptom onset and detection of disease.
5. Timely identification of various crop diseases using UGV based agriculture solution model.
6. UGV based agriculture solution model will also forecast the weather prediction/climate changes for helping farmers to take the necessary action accordingly.
7. The number of images is captured by UGV in every 10 minutes.
8. Probability of the crop disease.
9. Display of battery life and percentage of UGV.
10. Distance travelled by UGV.
11. Evaluation of individual disease affecting the crop.

5 Conclusion

The Designed UGV (Unmanned Ground Vehicle) will analyze the crops based on the crop data trained using our algorithm. We can identify the crop diseases in the early stages and transmit the information so that farmers can take necessary action to increase the yield of the crop.

6 References

1. Stephanie Bonadies, Alan Lefcourt, and S. Andrew Gadsden "A survey of unmanned ground vehicles with applications to agricultural and environmental sensing", Proc. SPIE 9866, Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping, 98660Q (17 May 2016)
2. https://en.wikipedia.org/wiki/Unmanned_ground_vehicle
3. https://blog.generationrobots.com/en/robotic-simulation-scenarios-with-gazebo-and-ros/
4. https://plantix.net/en/library/plant-diseases/200036/tomato-yellow-leaf-curl-virus
5. https://dengarden.com/gardening/Yellow-leaves-on-tomato-plants-Get-rid-of-yellow-tomato-leaves
6. https://www.sciencedirect.com/topics/agricultural-and-biological-sciences/tomato-yellow-leaf-curl-virus