RESEARCH MANUAL

Detect and Identify Disease in Plants

Lecturer

Nigel Whyte

Date of Submission

09/11/2018

(Student) - Darran Gahan

C00098391@itcarlow.ie

Abstract

It is estimated that 39 percent of worldwide crops are lost to disease and insects. The ability to accurately detect and identify disease on plants is key to helping reduce this figure (Sfiligoj, 2018). The scientists in EnviroCORE are researching innovative environmental technologies and biotechnologies with a view towards enhancing the economic and social development in an environmentally friendly manner (envirocore, 2018). Researchers within EnviroCORE based at Institute of Technology Carlow (ITC) currently carry out visual inspections on plant leaves and make subjective opinions on the health of the plant. This project will produce an application that will remove this subjectivity from the process. This application will achieve this through the use of computer vision and machine learning.

Contents

1. Currently Available Apps

While conducting research for this project I looked for apps that are currently on the market. I found a wide range of apps that can detect the species of plant from an image and a few that detect disease in plants from an image. The apps I reviewed that detect species of plants and or disease are *"PictureThis"*, *"Pl@ntNet"*, "*PlantSnap*", *"Plantix" and "LeafDoctor"*. These apps all had there good and bad points.

1.1 PictureThis

"PictureThis" was launched in 2017 it is an English version of a Chinese application named "XingSe" which was launched in 2015. "PictureThis" has a very clear and easy to use UI. It allows users to either take a picture with the camera on the device or to use a picture already on the device. The landing page shows the user a simple and clear screen with three main buttons "Take Photo", "Photo Album" and "Plant Care". The first time you select "Take Photo" the application provides a screen with examples of good and bad photos. When taking a picture, the application display's an outline on the screen to help the user centre the picture. The application then attempt's to analysis the image straight away, if it cannot find a match it will provide a list of possible matches and allow the user to pick the correct one. If there are no correct matches the user can select "No Match" which will give the user, the option to either ask for help from the community or to suggest a name. While testing this app it was unable to identify a leaf off a rose bush.

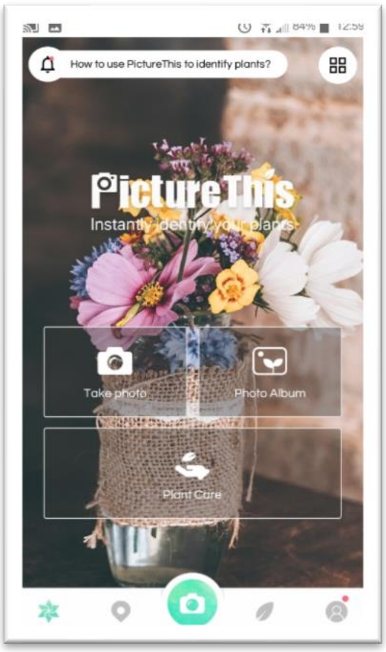
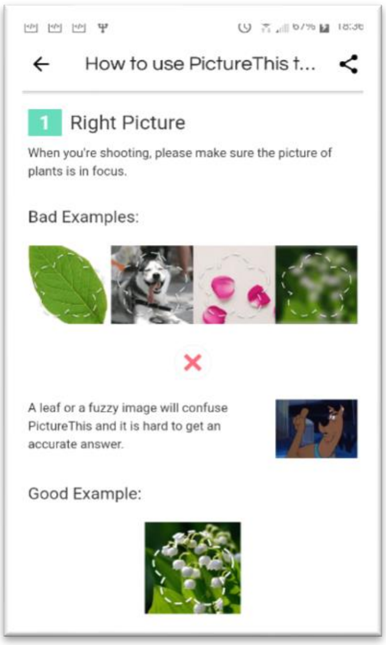

How to use PictureThis to identify plants?
PictureThis
Instantly identify your plants
Take photo
Photo Album
Plant Care

Figure 1.1 – Landing Screen


How to use PictureThis t...

1 Right Picture

When you're shooting, please make sure the picture of plants is in focus.

Bad Examples:

A leaf or a fuzzy image will confuse PictureThis and it is hard to get an accurate answer.

Good Example:

Figure 1.2 – Examples of good and bad pictures

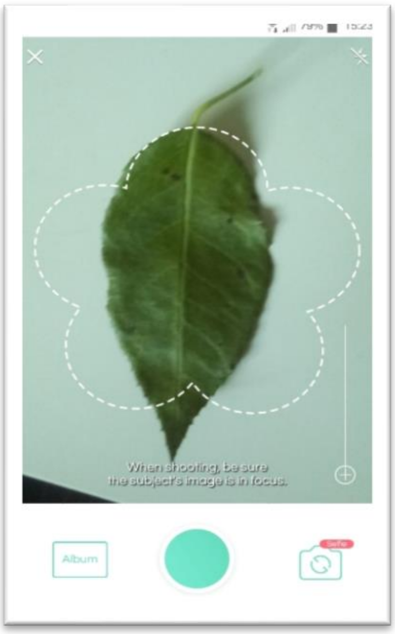


Figure 1.3 – Overlay on screen to assist the user

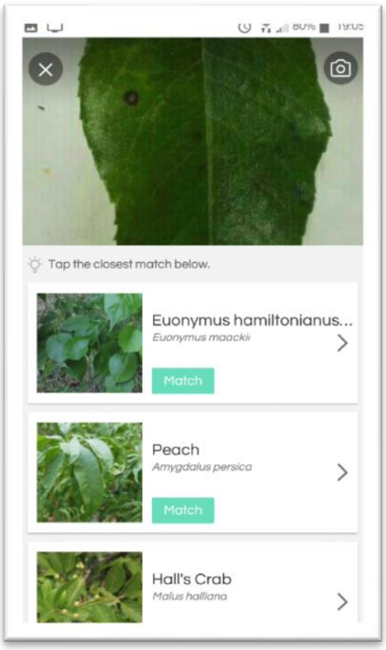


Figure 1.4 - Analysis results

1.2 Pl@ntNet

Pl@ntNet is a research and a citizen science project, it was initially supported by Agropolis Foundation, and developed since 2009. This project continues today through the Floris'Tic initiative funded by a Future investment program. The landing screen provides the user with images recently submitted by the community. There is a button at the bottom of the screen to take a picture. The user is given the option to take a picture or use a picture from their gallery. It provides a tutorial that shows the user examples of good and bad photos. It attempts to provide analysis of the species straight away, if it cannot find a match it will provide the user with a list of possible matches. However, while I was testing the app it could not identify a rose bush leaf and the list of possible matches it provides was very large, about 40 to 50 species.

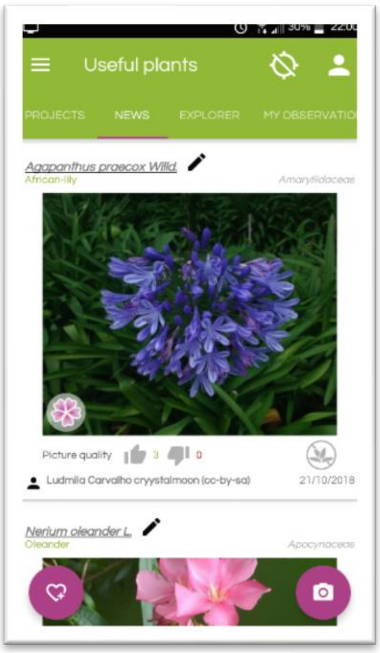


Figure 1.5 – landing screen

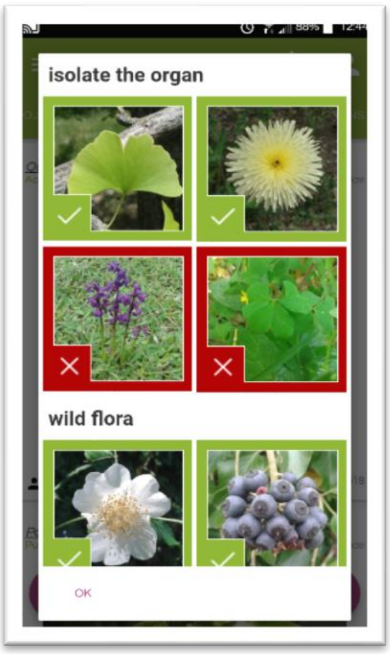


Figure 1.6 - examples of good and bad pictures

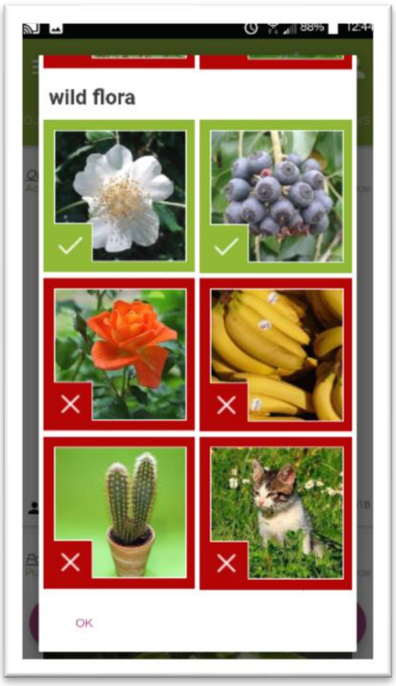


Figure 1.7 - Examples of good and bad pictures

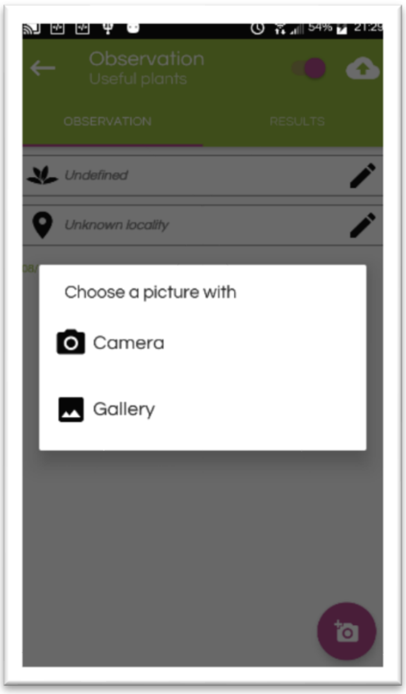


Figure 1.8 - option for camera or gallery

1.3 PlantSnap

The landing page for the application is the active camera screen where the user can take a picture if select one from their gallery. There is a button bar across the bottom, selecting the "More" button brings the user to the "More" screen here a user can view the help section which provides a number for tutorials on taking an appropriate picture, such as examples of good and bad images as well as videos on YouTube to help the user. The app attempts to analyse the image straight away and if it cannot find a match it will offer the user the option to send the image to be identified by a person. When testing this app, it could not identify a leaf from a rose bush.

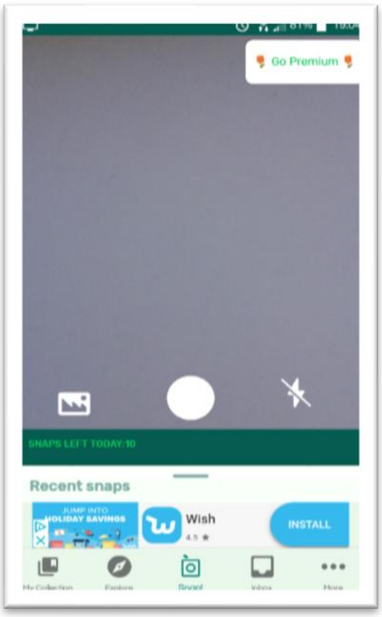


Figure 1.9 - Landing screen, places user straight into camera.

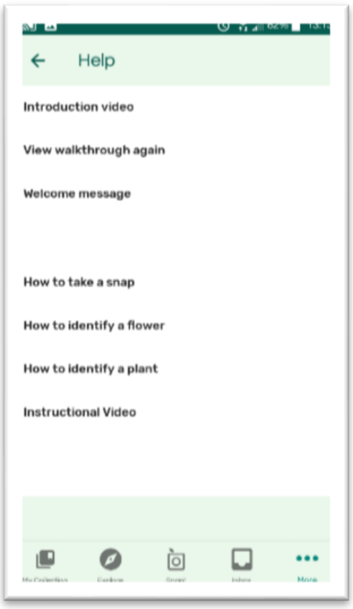
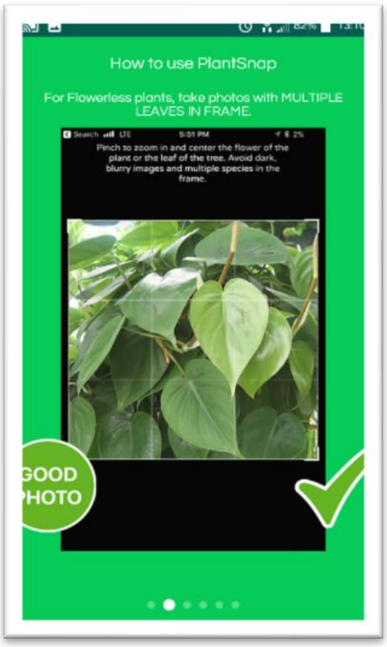


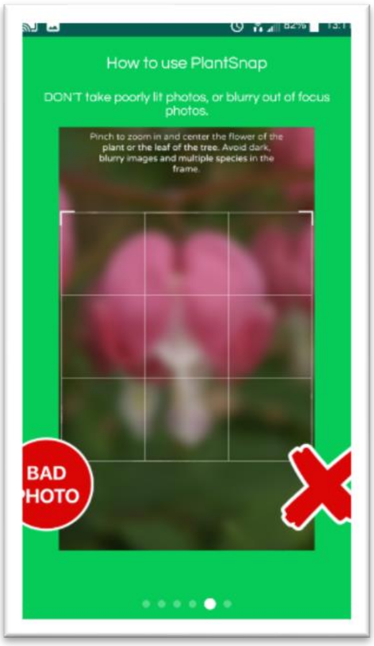
Figure 1.10 - Help screen showing tutorials


How to use PlantSnap

For Flowerless plants, take photos with MULTIPLE LEAVES IN FRAME.

Pinch to zoom in and center the flower of the plant or the leaf of the tree. Avoid dark, blurry images and multiple species in the frame.

Figure 1.11 - Example of good picture


How to use PlantSnap

DON'T take poorly lit photos, or blurry out of focus photos.

Pinch to zoom in and center the flower of the plant or the leaf of the tree. Avoid dark, blurry images and multiple species in the frame.

Figure 1.12 - example of bad picture

1.4 Plantix

Plantix is a mobile application developed by PEAT who were founded in Hanover, Germany 2015. PEAT define and maintain the global standard in image recognition for plant protection. *Plantix* aims to provide the user with the ability to detect and identify any disease that may be present in a plant. *Plantix* allows the user to take a picture of a leaf or analysis an image from the user's gallery. The landing screen provides the user with the last photo taken and weather information for the user's current location. There is a clearly labelled button in the bottom right hand corner that will take the user to the camera screen, so they can take a photo. The first time you go to the camera screen the application provides the user with an example of do's and don'ts when taking a picture. Once the picture is taken the application analysis the image and provides the user with any information on possible disease that may have been found in the given image.

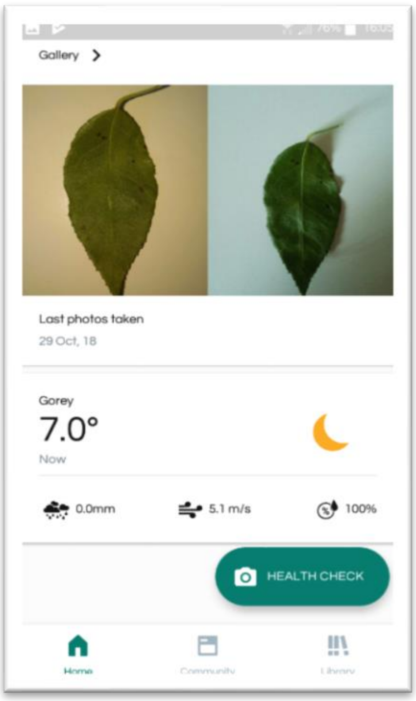


Figure 1.13 - landing screen

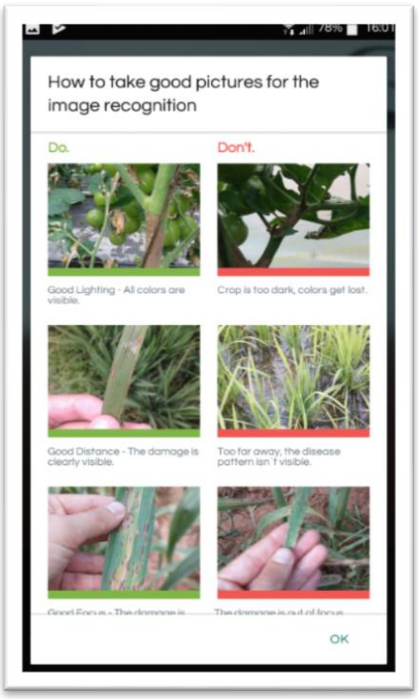

How to take good pictures for the image recognition
Do.
Don't.
Good Lighting - All colors are visible.
Crop is too dark, colors get lost.
Good Distance - The damage is clearly visible.
Too far away, the disease pattern isn't visible.
OK

Figure 1.14 - examples of good and bad pictures

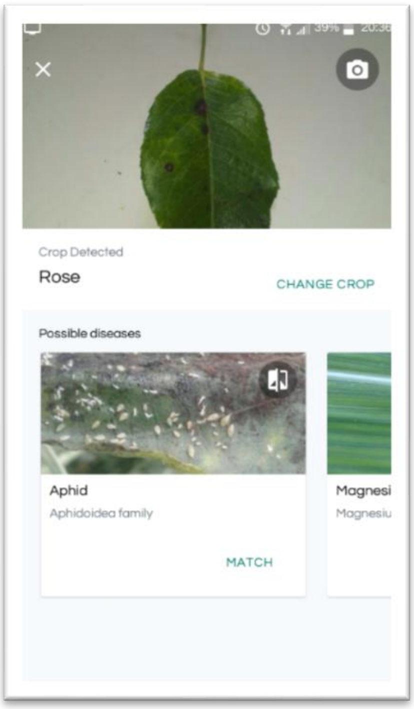


Figure 1.15 - result of analysis of leaf

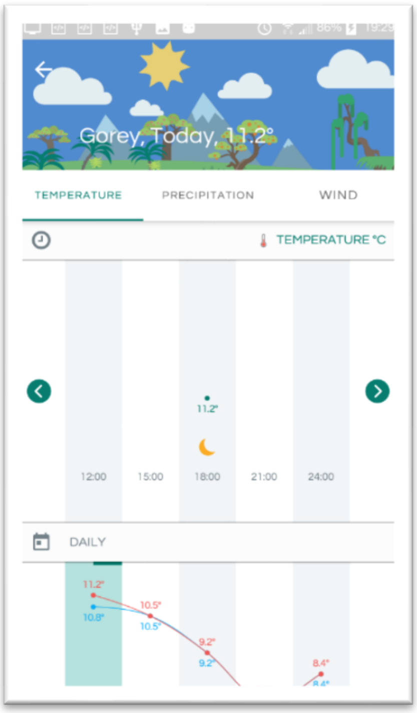


Figure 1.16 - Weather data provided

1.5 LeafDoctor

LeafDoctor is a mobile app to assess the level of disease in plants. The application allows users to take a photo or use an image from the user's gallery of a leaf and measure the percentage of diseased area. *LeafDoctor* has an easy to use UI. When opened the user is given the option to analysis a leaf, view their history, look for help and information about the app. When the user selects analyse, they are given the option to take a photo or load an image from their gallery. When the photo is taking, or image selected from the gallery, the user is provided with a slider to select the threshold for the image analysis. The application then moves to the results screen where the information about the percentage of disease detected on the leaf if presented.

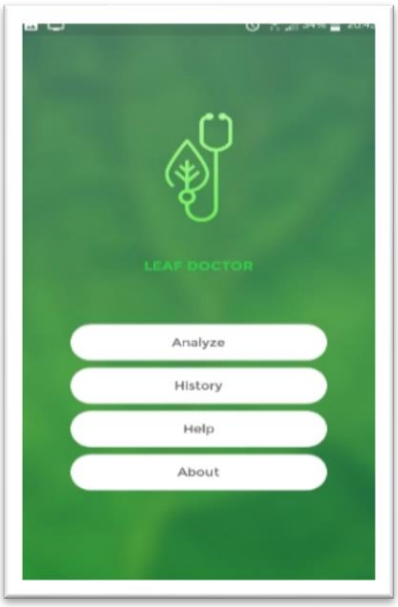


Figure 1.17 - landing screen

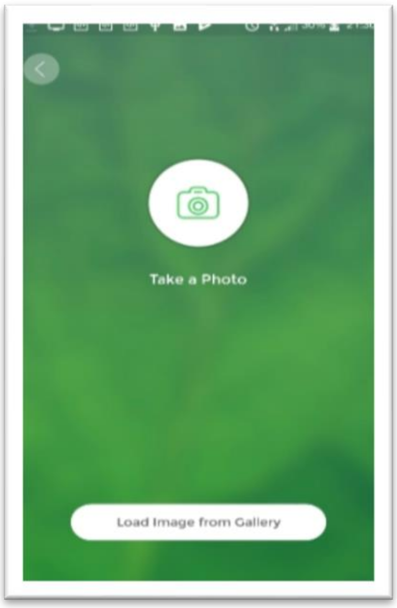


Figure 1.18 - Take photo or load image from gallery

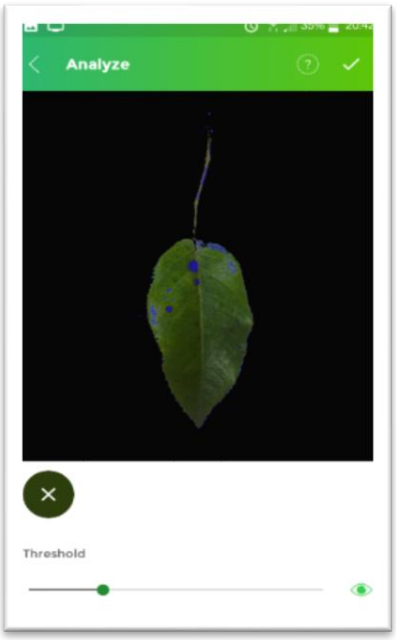
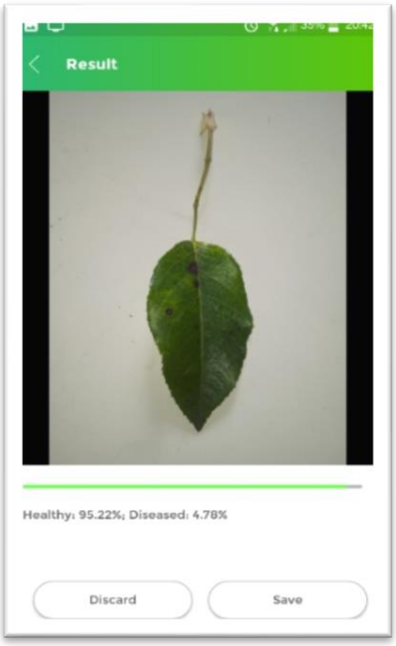


Figure 1.19 – Threshold slider for analysis



Healthy: 95.22%; Diseased: 4.78%

Figure 1.20 - result of analysis

Overall during my research, I found a few common good points and a few common bad points with currently available apps. Good points such as instant analysis, easy to use UI, tutorials on taking a good and bad photo and examples of good and bad photos, as well as providing a list of possible matches if a match is not found but limiting this list to a number that is not overwhelming. Using GPS location to get weather information for the image and to save the location of the image, and in turn the plant, making it easier for the user to track individual cases where disease is found.

2. Technologies

Most of this project will be written in the Java programming language. The reason for this is its ability to implement the OpenCV library and the fact it runs on Android, as there will be a mobile application. When looking at possible programming languages to use for the mobile application I considered using python, but upon researching python and OpenCV I found that implementing the two in the Android environment was a much harder task than using Java as the programming language. The Desktop application will be written Python using the Django framework. As one of the main features of the application will be to take and process images it is very important that the computer vision library used is of a high standard and proven to work as intended. OpenCV has been implemented by many companies around the world including Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota etc, and will be used in this project.

2.1 Computer Vision

The goal of computer vision is to make useful decision about real world objects and scenes based on sensed images (Shapiro & Stockman, 2001). Computer vision is closely tied to artificial intelligence because the computer must be able to process what it sees and then decide the outcome. Computer Vision has three main processing components that run one after the other. They are image acquisition, image processing, and image analysis and understanding. Image acquisition is the process of turning the world we live in into a binary data representation in the form of digital images. Image processing is the low-level processing of these images. Algorithms are applied to the obtained images. This can involve algorithms for things like Edge Detection, Segmentation, Classification, and Feature Detection and Matching. Image Analysis and Understanding is where the computer makes a decision about the image. High-level algorithms are applied using the image data and the low-level information obtained in the image processing such as, Object recognition and Object tracking (Kaiser, 2017).

2.1.1 Open CV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning library. OpenCV was developed to provide a common infrastructure for computer applications and to help speed up the use of machine perception. The library contains more than 2500 optimized algorithms which can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3d model of objects and many more. OpenCV is written in C++ but supports Java, Python and METLAB interfaces, it supports Windows, Linux, Mac OS, Android and iOS (opencv.org, 2018). OpenCV was initially developed by intel and later supported by Willow Grange, who are a robotics research lab based in Menlo Park California, and then later by Itseez, who were then acquired by intel in 2016 (opencv.org/intel-acquires-itseez.html, 2016).

2.1.2 Downsides

Computer vision can have its challenges when it comes to completing its task. Noisy or incomplete data can be a big problem as the focus of the image may not by accurately represented and can lead to false results. Real-time processing and limited resources i.e. power, memory (Kaiser, 2017) .

2.2 Edge Detection

Edge detection is a technique for processing images to find the boundaries of objects within the images. Edge detection works by detecting discontinuities in the brightness of an image. Edge detection can be used for data extraction and image segmentation in image processing, computer vision and machine learning. Some common edge detection algorithms are Canny, Sobel, Prewitt, and Roberts (uk.mathworks.com/discovery/edge-detection.html, n.d.).

2.2.1 Sobel

The Sobel operator is a discrete differential operator. It used two 3x3 kernels. One to calculate the gradient of the x-direction and one to calculate the gradient of the y-direction.

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Fig 2.1 - kernel for x-direction in sobel

$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

Fig 2.2 - Kernel for y- direction in sobel

The original image is convolved with both kernels to get the approximate derivatives in the horizontal and vertical change. For each given point the magnitude of the gradient can be approximated by using either of the formulas: G = $G\sqrt[G]{G_X^2 + G_Y^2}$, or $G = |G_x| + |G_y|$. The latter being a faster method of computing the gradient magnitude. The Sobel operator uses the smoothing effect Gaussian smoothing, because of this it is less sensitive to noise in the images. The smoothing affects the accuracy of the edge detection this means that the Sobel method does not produce images with high accuracy for edge detection, but it is more than adequate to be used in numerous applications (Kim, 2013).

2.2.2 Canny

The canny edge detector was developed by John F. Canny in 1986. The Canny operator consists of multiple steps. The first is a Gaussian filter that is used to smooth the image and in turn remove noise from the image. Second, the gradient magnitude is computed, this is done by use of a pair of 3x3 convolution masks. One is used to calculate the x direction and one to calculate the y direction.

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Figure 2.3 - Convolution mask for x direction in canny

$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

Figure 2.4 - Convolution mask for y-direction in canny

Thirdly the non-maximum suppression is applied. The algorithm removes pixels that are not part of an edge. The last step involves using hysteresis thresholding along the edges. Hysteresis Thresholding involves the use of two thresholds an upper threshold and a lower threshold. If a pixel is above the upper threshold the pixel is marked as an edge. If the pixel is below the lower threshold then the pixel will be discarded. If the pixel is between the upper threshold and the lower threshold only pixels that are connected to pixels above the upper threshold will be marker as edges (Kim, 2013).

2.2.3 Prewitt Operator

The Prewitt operator is credited to Dr. Judith Prewitt. The operator calculates the gradient of the image intensity at each point, given the direction of the largest possible change in the direction of the largest possible increase from light to dark and the rate of change in that direction. The result shows how abruptly or smoothly the image changes at that point and in turn how likely it is that part of the image represent an edge and how that edge is orientated. The magnitude of a calculation is more reliable and easier to interpret that the direction. The Operator uses two 3x3 kernels which are convolved with the original image to calculate the approximation of the derivatives.

$$\begin{pmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{pmatrix}$$

Figure 2.5 - kernel for x-direction in prewitt

$$\begin{pmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$$

Figure 2.6 - kernel for y-direction in prewitt

2.3 Object Recognition

Object recognition is a computer vision technique for identifying objects in images or videos. Object recognition is a key output of deep learning and machine learning algorithms. The goal is to teach a computer to do what comes naturally to humans, gain understanding of what an image contains. Objects can then be outlined with a boundary box. Object detection is similar to object recognition, but you only have two class of object classification which means object bounding boxes and non-object bounding boxes. Object recognition is a key technology behind driverless cars, it enables them to recognize a stop sign, or distinguish a pedestrian from street sign. It is also used applications such as disease identification in bioimaging, industrial inspection, and robotic vision.

2.4 Object Detection

Object detection focuses on identifying a particular type of object in a picture instead of all objects, like with object recognition, algorithms typically use extracted features and learning algorithms to detect instances of an object by a category. Image classification takes an image and predicts if an object is present in the image. If you build a cat-dog classifier, you would take an image and predict if there is a cat or dog present in the image, but if both where present in the image it would cause a problem. A multi-label classifier can be built to solve this problem, it will predict both classes in the image. This classifier will not be able to tell you the location of the objects. Identifying the location of an object in an image is called localization. Determining the class of an object and its location is known as object detection. In object detection the class of the object as well as a rectangle, known as a bounding box, to contain the image must be predicted.

Object Detection is modelled as a classification problem where windows of a fixed size from an input image are fed into an image classifier. The classifier predicts if there is or is not an object in the window. One problem we face is the object can be of varying sizes. To solve this an image pyramid is created by scaling the image. The way this works is by resizing the image at multiple scales and counting on the fact that at one of the scales the object will be completely contained within the chosen window size and thus help solve the problem of size and location. There are various algorithms for object detection such as Hog features and R-CNN. Some algorithms for object detection treat it as a regression problem, these include YOLO and SSD (Sachan, 2017).

2.4.1 Hog Features

Navneet Dalal and Bill Triggs developed Histogram of Oriented Gradients (HOG) feature in 2005. Hog features are computationally inexpensive and is good for solving many real-world problems. For Each window obtained the Hog features is calculated which is then fed to an SVM (Support vector machine), which is a supervised learning model used for classification, to create classifiers. Hog features can be run in real time on videos.

2.2.4 Region-based Convolution Neural Networks

Deep learning provides us with the option to replace the Hog classifiers with a far more accurate convolutional neural network based classifier. CNNs can be slow and computationally very expensive to solve this an algorithm call Selective search is used, this reduces the number of windows that are fed to the classifier. It does this by using things texture, colour, intensity.

2.2.5 YOLO (You Only Look Once)

In YOLO object detection is treated as a simple regression problem that takes an input image and learns the class probabilities and bounding box coordinates. It does this by dividing the image into a grid of S x S and each grid then predict N bounding boxes and a confidence. The confidence represents the accuracy of the bounding box and whether it contains an object. The threshold for the confidence can be set which will remove and boundary boxes below the threshold. Another difference with YOLO is that it sees the complete image instead of generated windows, this contextual information can help in avoiding false positives. A limitation of YOLO is that is only predicts one type of class per grid meaning, it has trouble with detecting very small objects.

2.2.6 SSD (Single Shot Detector)

The Single Shot Detector strikes a good balance between accuracy and speed. It runs a CNN on and input image once and calculates a feature map, a 3x3 convolutional kernel is run on the image to predict boundary boxes and classification. Anchor boxes are used at different aspect ratios and it SSD learns the off-set rather than the learning the box. Bounding boxes are predicted after multiple convolutional layers this allow the algorithm to detect objects at various scale since each convolutional layer operates at a different scale.

3. Article Review

Title: Signs and Symptoms of plant disease: IS it fungal, viral or bacterial?

Author: Jim Isleib Date: 19/12/2012

The article "Signs and Symptoms of plant disease: Is it fungal, viral or bacterial?" written by Jim Isleib discusses the signs and symptoms of plant disease and the three main pathogenic microbes: fungus, bacteria and virus.

Almost 85% of most plant disease is caused by fungal or fungal like organism. nematodes also known as roundworm can also cause disease in plants. Some diseases are classified as abiotic, meaning they are non-infectious and include damage from air pollution, nutritional deficiencies or toxicities, and grow under less than optional conditions. When an unknown plant problem appears, abiotic diseases, herbicide injury and nematode problem must be considered as possibilities.

A sign of plant disease is where there is physical evidence of the pathogen. In the case of as powdery mildew on a lilac leaf, that powdery mildew is actually a parasitic fungal disease organism. A symptom of plant disease is a visible effect of disease on the plant. These can include a change of colour or shape. Wilting is also a symptom of disease specifically Verticilum wilt. With a symptom you are not actually seeing the disease pathogen, but rather a symptom that is caused by the pathogen.

Fungal disease signs may include Leaf rust, Stem rust, Sclerotinia and Powdery mildew. Fungal disease symptoms can be Bird-eye spot on berries, Damping off of seedlings, Leaf spot, Chlorosis. Bacterial disease sign can be difficult to observe but can include Bacterial streaming in water from a cut stem. Bacterial disease symptoms can include Leaf spot with yellow halo, Fruit spot, Canker, Crown gall and Shepard's crook stem ends on woody plants. There are no viral disease signs as the viruses themselves cannot be seen. Viral disease symptoms may include Mosaic leaf pattern, Crinkled leaves, Yellowed leaves and plant stunting.

This article covers the main types of pathogens in plants and some of the signs and symptoms that may come along with them. It has given me a good insight into classification and identification of plant disease as well as the difference between a sign and a symptom of plant disease. This information will help greatly when developing the computer vision aspect of the application and also help improve the accuracy of the application.

4. System Overview

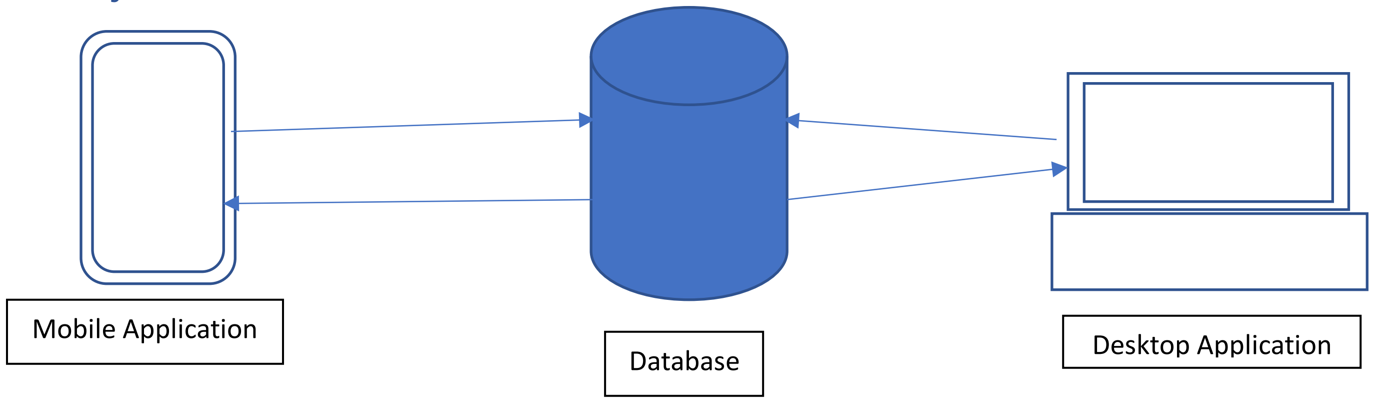


Fig 3.1 – Brief overview of possible system

References

envirocore. (2018). *envirocore.ie*. Retrieved from envirocore.ie: http://www.envirocore.ie/

Kaiser, A. (2017, December 12). *What is Computer Vision?* Retrieved from hayo: https://hayo.io/computer-vision/

Kim, D. (2013). *Sobel Operator and Canny Edge Detector.*

opencv.org. (2018). *opencv.org/about.html*. Retrieved from opencv.org: https://opencv.org/about.html

opencv.org/intel-acquires-itseez.html. (2016, May 27). Retrieved from opencv.org: https://opencv.org/intel-acquires-itseez.html

Sachan, A. (2017). *cv-tricks.com/object-detection*. Retrieved from cv-tricks: https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/

Sfiligoj, E. (2018, April 9). *The Disease/Insect Outlook for 2018: New Challenges at Every Turn*. Retrieved from croplife: www.croplife.com

Shapiro, L. G., & Stockman, G. C. (2001). *Computer Vision.* New Jersey: Prentice Hall.

uk.mathworks.com/discovery/edge-detection.html. (n.d.). Retrieved from mathworks.com: https://uk.mathworks.com/discovery/edge-detection.html