

# User Manual ArchiPhen Software

## INSTALLATION

### ***Python***

ArchiPhen is program with the Python programming language. To run the program, please download python from:

<https://www.python.org/> .

### ***Python Dependencies***

The following python libraries are required for ArchiPhen to function properly

*wxpython* for GUI building <https://wxpython.org/> , *PIL* for image handling <https://python-pillow.org/>, *zbar* for QR encoding and decoding <https://pypi.python.org/pypi/zbar> , Matplotlib for producing graphs and plots <http://matplotlib.org/>, *numpy* for scientific computing <http://www.numpy.org/>

### ***ImageJ / Fiji***

There are various image processing tools included in ArchiPhen. These task are carried out using ImageJ. To install ImageJ, download the binaries from:

<https://imagej.nih.gov/ij/>

<https://fiji.sc/>

For ArchiPhen to be able to call ImageJ plugins externally, the path to the ImageJ executable must be specified in the environment variables. Set the path to the ImageJ.exe executable in the Environment Variable

### ***ImageJ Plugins***

Currently, ArchiPhen uses three custom-made ImageJ plugins to perform task such as identification of QR codes in the image, identification of the region of interest for the roots, and the tracing of the roots. The plugins are installed into image by copying required files in the Plugin folder of ImageJ:

root\_analysis\_QR.class : The file contains algorithms for identification of QR codes from the image. Note that the algorithm was designed to function for the images that were produced in our labs. QR codes may not be detected for other setup. The file is placed in the ArchiPhen\ImageJ\ folder

root\_analysis\_bounds.class : The file contains algorithms for identification of the region where roots are growing. Note that the algorithm was designed to function for the images that were produced in our labs, using blue germination paper. The algorithm may not perform optimally for other setup. The file is placed in the ArchiPhen\ImageJ\ folder

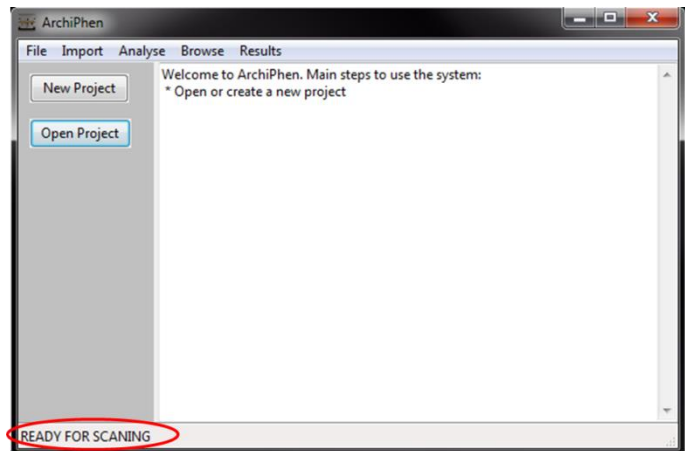
RootTracing folder: the folder contains the source code and binary files for the tracing of roots using Dijkstra's optimal path search algorithm. To allow ArchiPhen to run the tracing algorithm, you must unzip the root\_tracing.zip. The file is placed in the ArchiPhen\ImageJ\ folder.

***The program has been tested with***

Windows 7, Python 2.7.9, wxpython 3.0.2.0, PIL1.1.7, Matplotlib1.4.3, numpy1.9.1

## FILE MENU / LEFT PAN buttons

Creating or opening existing projects is carried out using the File menu, or alternatively the buttons on the left pan for fast access to these functionalities



*Figure 1: screenshot of ArchiPhen GUI when ready*

Once a valid project has been created or opened, the task bar indicate the program is ready to operate

## IMPORT MENU

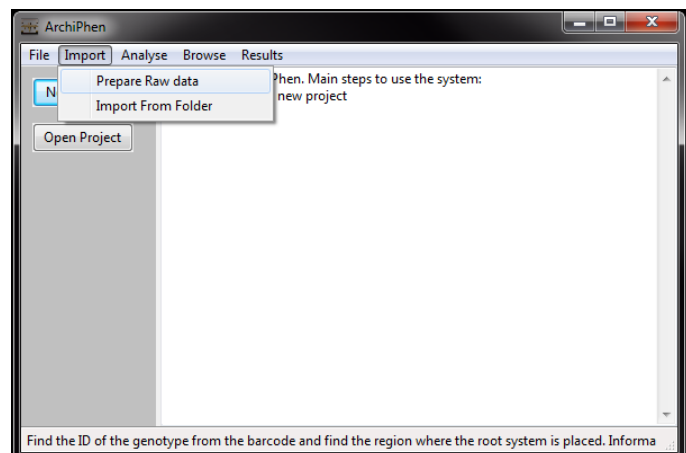


Figure 2: screenshot of ArchiPhen Import menu

### Read QR codes from raw image data

The next stage is to import images into the database. Before this can be achieved, an ImageJ plugin must be run to identify the region of the images where there is a QR code and to read the QR code to identify the genotype and replicate number of the experiment. This is achieved from the ArchiPhen graphical user interface by selecting Import\Prepare Raw data menu.

This task uses ImageJ plugins:

- Make sure relevant .class files and plugin folder and have been copied in ImageJ's plugin folders
- Make sure you have set the path to ImageJ's executable in Windows environment variables

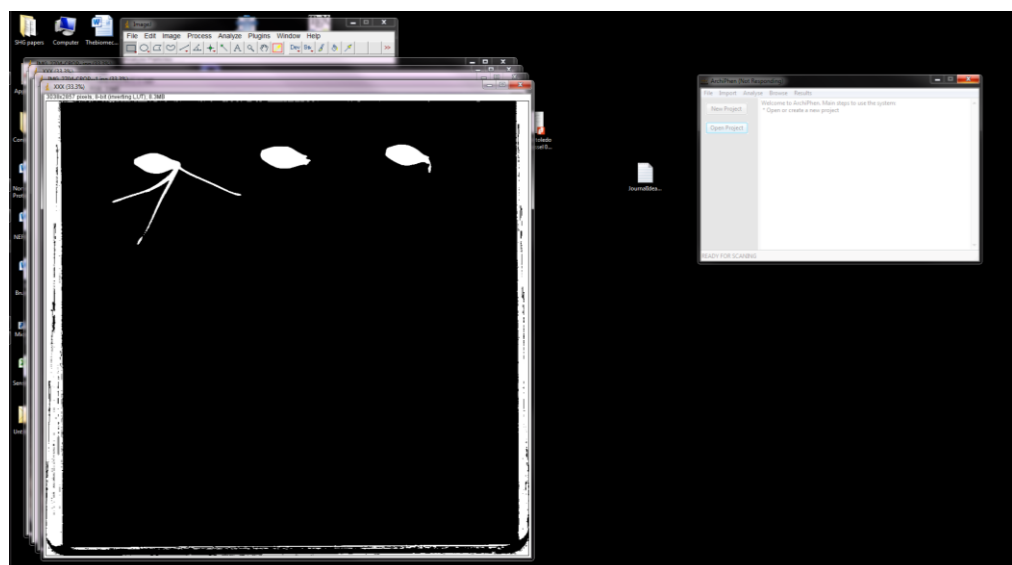


Figure 3: screenshot of ArchiPhen when Preparing the raw data

### *Import data from files*

For each raw image data, the `root_analysis_QR.class` and `root_analysis_bounds.class` plugins produce two additional images. One that contains only the QR code, and another that is the region of the image that contains the blue germination paper where the root is growing. Once these files have been produced, they can be imported in the project data tree by going to the import menu and selecting Import From Folder .

The algorithm will run through all images in the folder and try to decode the QR code in them. For each QR code read, the image will be copied in the database. Occasionally however, a QR code will be found but the decoder will fail to read genotype, replicate or species. In which case, an interface will be created with the image of the label. In this case the user is requested to input the details of the genotype manually so that the image data can be introduced in the database manually.



*Figure 4: screenshot of ArchiPhen when QR code not read*

## ANALYSE MENU

Analyse menu is used to process the data before the roots have been traced.

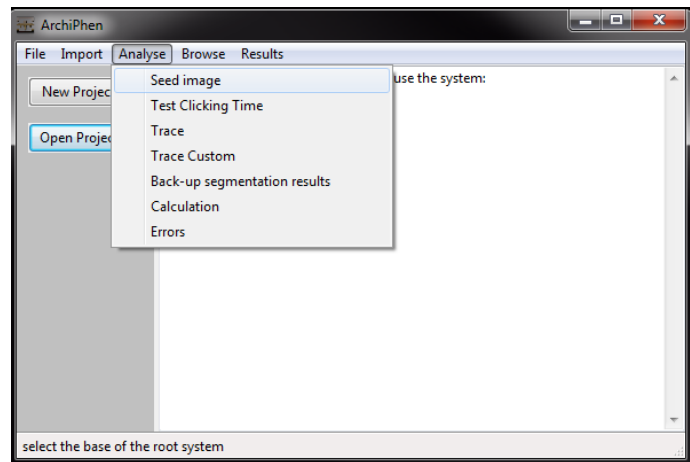


Figure 5: screenshot of ArchiPhen Analyse menu

## Placement of markers

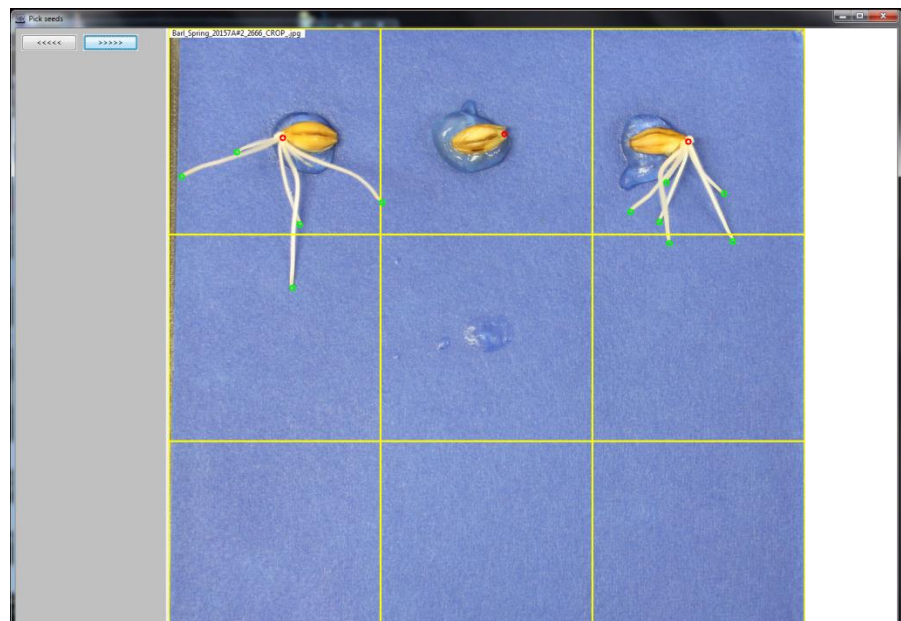


Figure 5: screenshot of ArchiPhen's GUI for placing markers

Placement of markers on images of the database is carried out through an interface called from the Analyse\Seed image menu.

There are two types of markers on the image. The first type of marker is used to indicate the position of the seed. It is indicated by a **red circle**. A seed marker is place with a **left click** when the mouse pointer is placed on the desired position. Alternatively, a touch on a touch sensitive screen

can replace the left click. The second type of marker is used to indicate the tip of the root. It is indicated by a **green circle**. A root tip marker is placed using **Shift + left click** on the desired position. Alternatively, a touch on a touch sensitive screen can replace the left click. Removal of a marker is achieved by **Alt + left click**. When the mouse pointer is closed to an existing marker, Alt + left click will remove the closest marker from the pointer.

### *Test scripts*

Other elements in the Analyse menu are for testing purposes (determining clicking times and recording errors). These should not be used in normal operations

## BROWSE MENU

The browse menu is used to explore the data in the database and run scripts to process the data

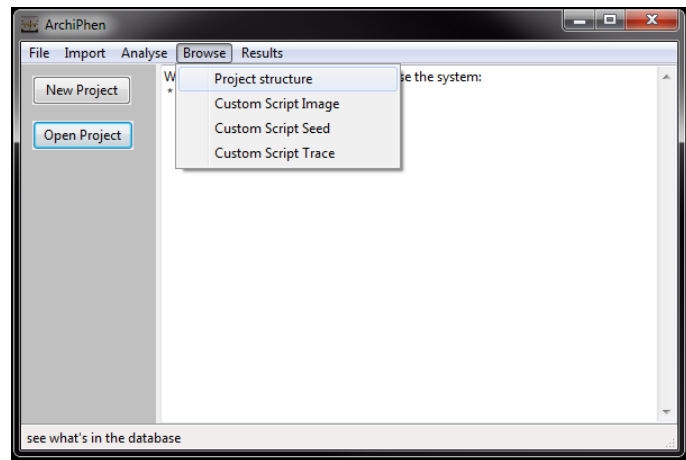


Figure 6: screenshot of ArchiPhen's Browse menu

### Project Structure

This command scans for all the data stored in the database and output a list of genotypes, species.

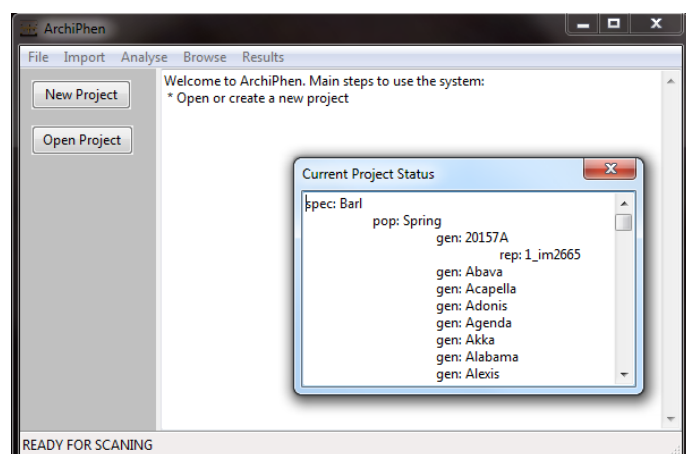


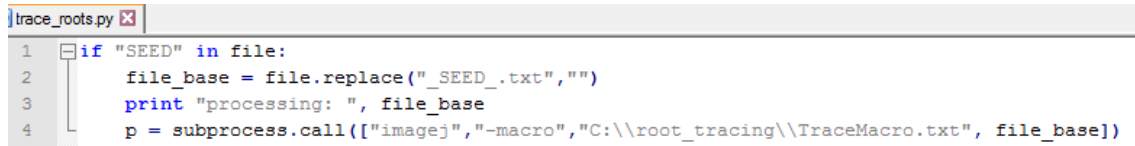
Figure 7: screenshot of ArchiPhen's Project structure output

### Run Scripts

Performing tasks on the database is carried out using scripts that are run from the graphical interface of ArchiPhen. The advantage of that functionality is that powerful analyses can be run without the requirement for a deep understanding of the architecture of the ArchiPhen software. Scripts are programmed using the Python programming language. There are three types of scripts. Scripts that browse the database for image files (Image scripts) and allow executing custom commands on such images, scripts that browse the database for markers position data (Seed scripts) and allow executing custom script on such data. Finally, scripts that are dedicate to run root tracing algorithms on image data. Example of scrips used for tracing root from marker's data is shown in



Figure 7. In this case, the scripts have access to each file in the database and an ImageJ plugin is called to perform the tracing using the path to the file in the database. Because the script is not included in the ImageJ of Fiji distribution, running the plugin requires copying the relevant files in Plugin folder of ImageJ of Fiji.



```
1 if "SEED" in file:
2     file_base = file.replace("_SEED_.txt", "")
3     print "processing: ", file_base
4     p = subprocess.call(["imagej", "-macro", "C:\\root_tracing\\TraceMacro.txt", file_base])
```

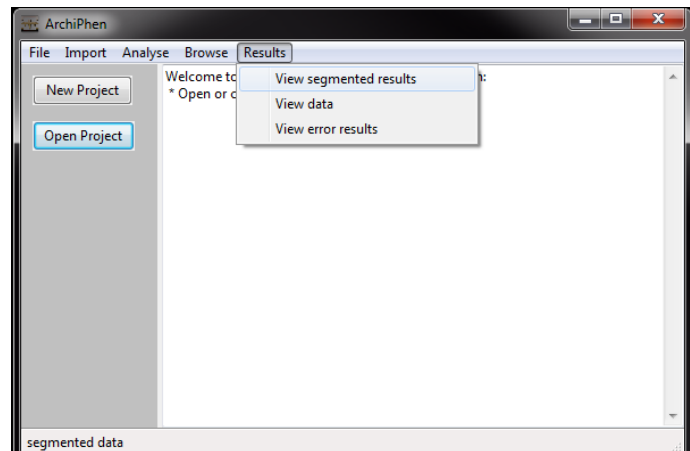
*Figure 8: Example of custom script used to trace root in ArchiPhen*

This task uses ImageJ plugins:

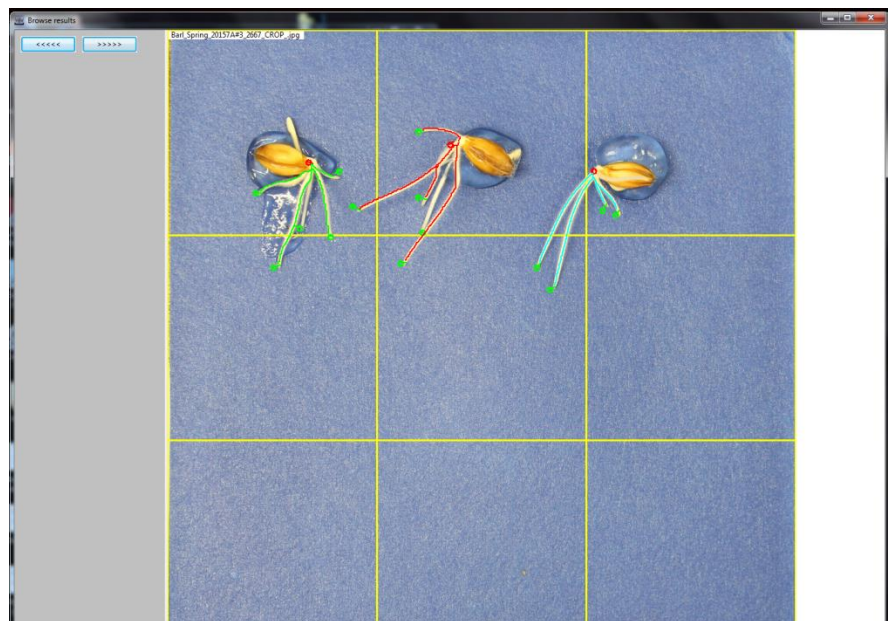
- Make sure relevant .class files and plugin folder and have been copied in ImageJ's plugin folders
- Make sure you have set the path to ImageJ's executable in Windows environment variables

## RESULTS MENU

The results menu is used to visualise the results of the various analyses that have been run on the data.



*View segmented results*



*View data*

TODO