

PC-Based Classification of Fermented or Defective Cacao Bean with Bean Count

User's Manual

Table of Contents

1.0	General Information	1
1.1	System Overview	2
1.2	Organization of the Manual	2
2.0	System Summary	3
2.1	System Configuration	3
	2.1.1 Software Requirements	4
	2.1.2 Hardware Requirements	4
2.2	User Information	4
2.3	Physical Setup and Peripherals	5
2.4	Camera Settings	6
3.0	Using the Prototype	7
3.1	Step by step operation	8
4.0	Fixing Issues	12
4.1	The system has error classifying	13
4.2	Improving the classifier	13

General Information

1.0 General Information

General Information section explain in general terms the prototype and the purpose for which it is intended.

1.1 System Overview

PC – Based Classification of Fermented or Defective Cacao Bean with Bean Count is a prototype, which allows cacao processors to evaluate a sample of cacao faster and more reliable than the manual inspection. It is only a part of the whole processor of grading the cacao bean. It is operated with the aid of image processing, SVM classifier together with Gizduino. The porotype's operation is still under development.

1.2 The user's manual consists of five sections: General Information, System Summary, Using the System, and Fixing Issues.

General Information section explain in general terms the prototype and the purpose for which it is intended.

System Summary provides a general overview of the prototype. The summary outlines the uses of the system's hardware and software requirements and system's configurations.

Using The System section provides a detailed description of system functions.

System Summary

2.0 System Summary

System Summary provides a general overview of the prototype. The summary outlines the uses of the system's hardware and software requirements and system's configurations

2.1 System Configuration

2.1.1 Software Requirements

The prototype operates on computers provided that it has the following software installed:

1. Operating system that supports Python 2.7
2. EOS Utility
3. Python 2.7
 - a. Numpy
 - b. Scipy
 - c. OpenCV Library
 - d. Scikit – learn
 - e. matplotlib

2.1.2 Hardware Requirements

The prototype operates with the aid of the following hardware:

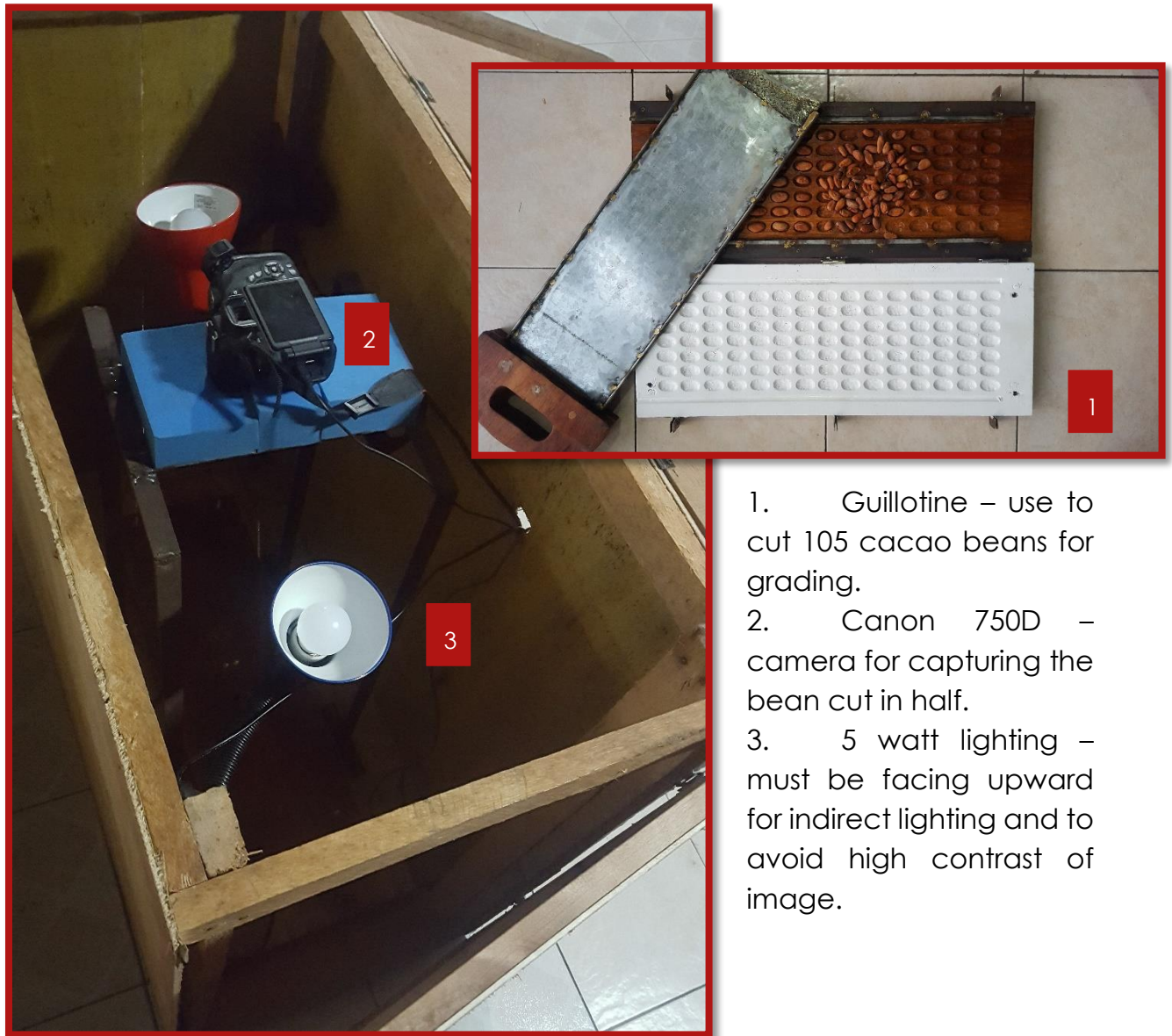
1. Canon 750D
2. Gizduino
3. Analog to Digital Converter
4. Load Cell
5. Two 5 watt bulbs
6. Guillotine Cutter

The system must be enclosed for controlled lighting.

2.2 User Information

Everyone can use the prototype, but it is highly recommended that the prototype will be operated by someone who has knowledge in python programming and machine learning. However the prototype is still underdeveloped and can be further improved to be more user friendly and portable.

2.3 Physical Setup and Peripherals



1. Guillotine – use to cut 105 cacao beans for grading.
2. Canon 750D – camera for capturing the bean cut in half.
3. 5 watt lighting – must be facing upward for indirect lighting and to avoid high contrast of image.

2.4 Canon 750 D camera settings

ISO	100
Aperture	5.0
Shutter Speed	3"2
White Balance	Automatic
Focus	Manual
Metering Mode	Center Weighted Average
Height of Camera from the Base	55cm

Using the Prototype

3.0 Using the Prototype

This section provides the detailed description of the prototype's function.

3.1 Step by step operation

3.1.1 Get a sample of 105 of cacao beans.



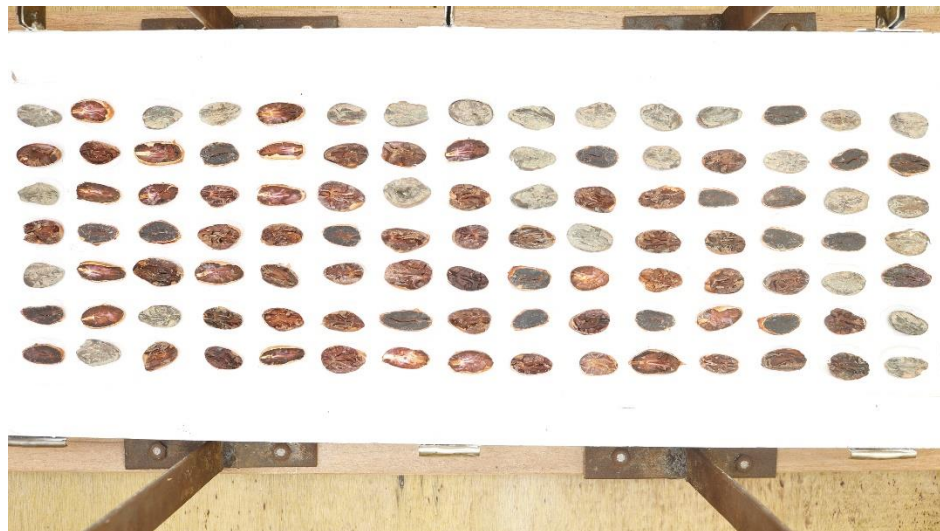
3.1.2 Place the beans in the slots in the guillotine.

3.1.3 Close the guillotine and make sure it is locked.

3.1.4 Insert the blade and put pressure until the handle reach the side of the guillotine.

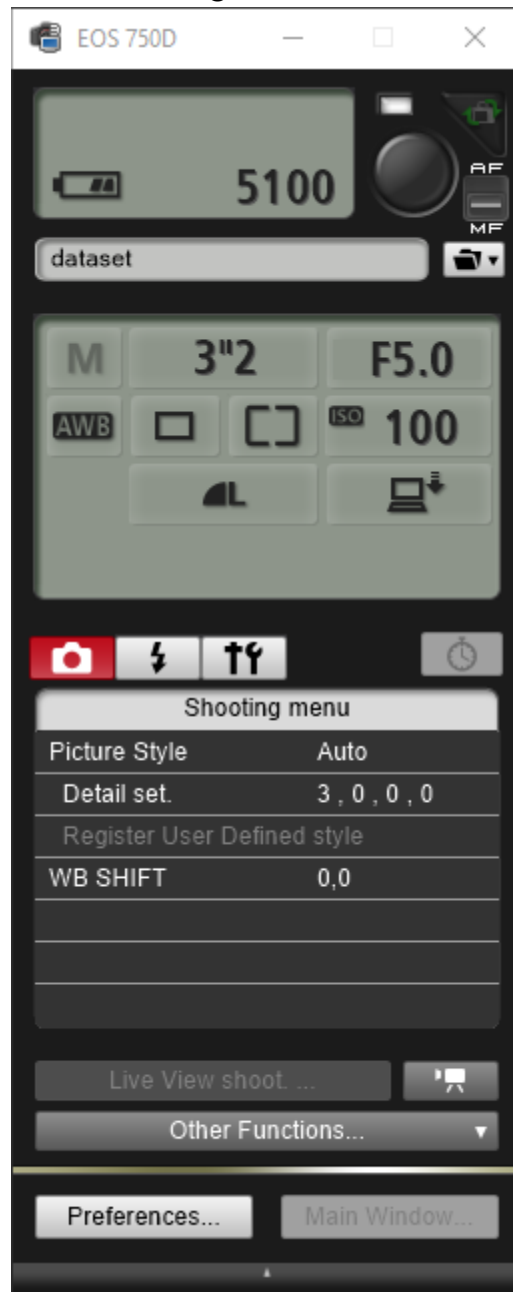
3.1.5 Remove the half of the guillotine where the blade is attached leaving the white half plate of the guillotine.

3.1.6 Place the guillotine inside the setup. Make sure the slots are all occupied



3.1.7 Open the camera and make sure the settings comply with the above description (2.4)

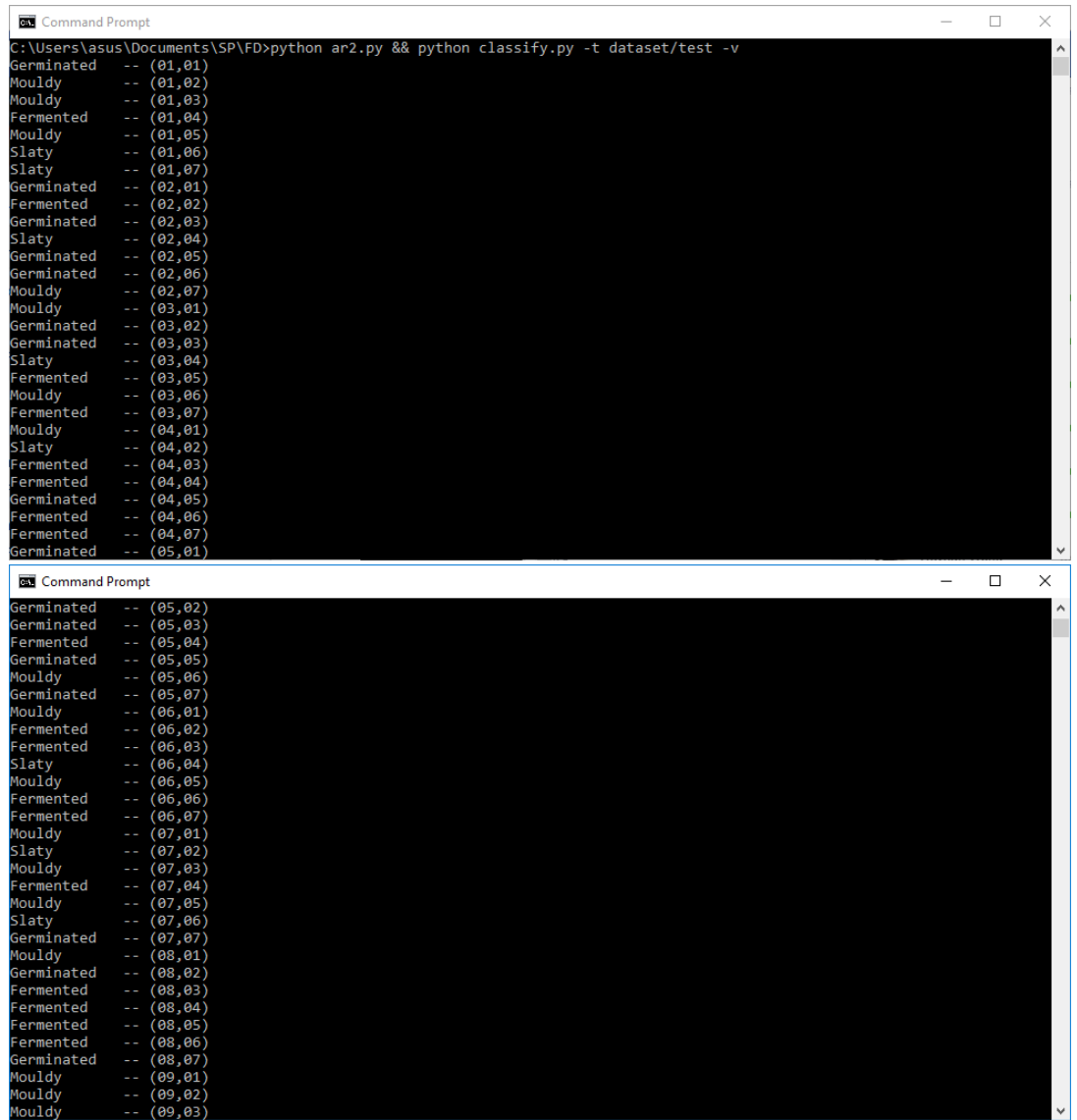
3.1.8 Capture the image using EOS Utility software. Make sure that the image is within the directory of the dataset: e.g. C:\Users\Program\Documents\testing\dataset



3.1.9 Go to the directory of the algorithm and run CMD

3.1.10 Input: ***\$ python array.py && python classify.py -t dataset/test/ -v***

3.1.11 The output should be look like this:



```
Command Prompt
C:\Users\asus\Documents\SP\FD>python ar2.py && python classify.py -t dataset/test -v
Germinated -- (01,01)
Mouldy -- (01,02)
Mouldy -- (01,03)
Fermented -- (01,04)
Mouldy -- (01,05)
Slaty -- (01,06)
Slaty -- (01,07)
Germinated -- (02,01)
Fermented -- (02,02)
Germinated -- (02,03)
Slaty -- (02,04)
Germinated -- (02,05)
Germinated -- (02,06)
Mouldy -- (02,07)
Mouldy -- (03,01)
Germinated -- (03,02)
Germinated -- (03,03)
Slaty -- (03,04)
Fermented -- (03,05)
Mouldy -- (03,06)
Fermented -- (03,07)
Mouldy -- (04,01)
Slaty -- (04,02)
Fermented -- (04,03)
Fermented -- (04,04)
Germinated -- (04,05)
Fermented -- (04,06)
Fermented -- (04,07)
Germinated -- (05,01)
Germinated -- (05,02)
Germinated -- (05,03)
Fermented -- (05,04)
Germinated -- (05,05)
Mouldy -- (05,06)
Germinated -- (05,07)
Mouldy -- (06,01)
Fermented -- (06,02)
Fermented -- (06,03)
Slaty -- (06,04)
Mouldy -- (06,05)
Fermented -- (06,06)
Fermented -- (06,07)
Mouldy -- (07,01)
Slaty -- (07,02)
Mouldy -- (07,03)
Fermented -- (07,04)
Mouldy -- (07,05)
Slaty -- (07,06)
Germinated -- (07,07)
Mouldy -- (08,01)
Germinated -- (08,02)
Fermented -- (08,03)
Fermented -- (08,04)
Fermented -- (08,05)
Fermented -- (08,06)
Germinated -- (08,07)
Mouldy -- (09,01)
Mouldy -- (09,02)
Mouldy -- (09,03)
```

```
Command Prompt
Fermented -- (09,04)
Mouldy -- (09,05)
Slaty -- (09,06)
Germinated -- (09,07)
Mouldy -- (10,01)
Slaty -- (10,02)
Fermented -- (10,03)
Mouldy -- (10,04)
Germinated -- (10,05)
Fermented -- (10,06)
Fermented -- (10,07)
Mouldy -- (11,01)
Mouldy -- (11,02)
Slaty -- (11,03)
Fermented -- (11,04)
Fermented -- (11,05)
Slaty -- (11,06)
Germinated -- (11,07)
Mouldy -- (12,01)
Fermented -- (12,02)
Slaty -- (12,03)
Fermented -- (12,04)
Fermented -- (12,05)
Germinated -- (12,06)
Fermented -- (12,07)
Slaty -- (13,01)
Mouldy -- (13,02)
Fermented -- (13,03)
Slaty -- (13,04)
Fermented -- (13,05)

Command Prompt
Mouldy -- (12,01)
Fermented -- (12,02)
Slaty -- (12,03)
Fermented -- (12,04)
Fermented -- (12,05)
Germinated -- (12,06)
Fermented -- (12,07)
Slaty -- (13,01)
Mouldy -- (13,02)
Fermented -- (13,03)
Slaty -- (13,04)
Fermented -- (13,05)
Slaty -- (13,06)
Fermented -- (13,07)
Mouldy -- (14,01)
Slaty -- (14,02)
Mouldy -- (14,03)
Slaty -- (14,04)
Mouldy -- (14,05)
Fermented -- (14,06)
Fermented -- (14,07)
Mouldy -- (15,01)
Slaty -- (15,02)
Mouldy -- (15,03)
Mouldy -- (15,04)
Slaty -- (15,05)
Mouldy -- (15,06)
Mouldy -- (15,07)
Counter({'Fermented': 33, 'Mouldy': 32, 'Germinated': 20, 'Slaty': 20})

Command Prompt - python classify.py -t dataset/test
C:\Users\asus\Desktop\bow\bag-of-words-master>python classify.py -t dataset/test
Counter({'Fermented': 38, 'Mouldy': 27, 'Germinated': 20, 'Slaty': 20})
Waiting for device
COM5
Weight: 56.06 grams Bean Count: 93
```

Fixing Issues

4.0 Fixing Issues

4.1 The system has error classifying

- 4.1.1 Check if the slots are all occupied by cacao beans.
- 4.1.2 Make sure all the python libraries are updated and met the requirements.
- 4.1.3 Make sure the camera settings met the requirements.
- 4.1.4 Check the lighting if the requirements are met.
- 4.1.5 Retrain the classifier.

4.2 Improving the classifier

- 4.2.1 Retrain the classifier with more samples of cacao per class.
Make sure that each class have the same number of samples.
Run the command: **`$ python train.py -t dataset/train`**
- 4.2.2 Increase the cluster for training.
- 4.2.3 Use a different camera with higher specification.