AGILEHAND

User Manual: WP4 – BUILD: AGILEHAND Smart Sensing SUITE

AGILEHAND has received the funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101092043.

Document Information

GRANT AGREEMENT NUMBER 101092043	ACRONYM	AGILEHAND	
FULL TITLE	Smart grading, handling, and packaging solutions for soft and deformable products in agile and reconfigurable lines		
START DATE	01-01-2023	DURATION	36 months
PROJECT URL	https://agilehand.eu/		
WORK PACKAGE	WP4 – BUILD: AGILEHAND Smart Sensing SUITE		
LEAD BENEFICIARY	FBK		
RESPONSIBLE AUTHOR	Dr. Mohamed Lamine Mekhalfi,		
CONTRIBUTIONS FROM	FBK, UNI		
TARGET AUDIENCE	Solution user(s)		
CONTENT	User manual		
ABSTRACT	The scope of this document is to guide the user(s) of the solutions developed within WP4 – BUILD: AGILEHAND Smart Sensing SUITE of the AGILEHAND project.		

Disclaimer

TABLE OF CONTENTS

1. Introduction

1.1. Purpose and Scope

The WP4 of **AGILEHAND** is designed to provide automated, AI-powered grading of soft and deformable food products to ensure consistent quality control in industrial processing lines. This User Manual describes how to operate the system across its four supported use cases, including oranges, raspberries, fish, and chicken fillets. The manual covers the day-to-day use of the system, including operation workflows, and interpretation of grading results.

1.2. Target Audience

This User Manual is intended for operators, quality control personnel, and supervisors who will interact with the **AGILEHAND** grading system during routine production activities. It assumes no prior knowledge of AI or computer vision and focuses on practical instructions for using the system effectively. Technical staff responsible for installation, maintenance, or AI model management should refer to the Installation Manual, which is provided in a separate document

2. System Overview

The **AGILEHAND** grading system combines hardware and software components to automatically analyse and grade soft and deformable food products on industrial conveyor lines. It utilizes advanced computer vision and AI algorithms to ensure consistent quality control.

2.1. Hardware Components

The system uses high-resolution 3D cameras mounted in a fixed, top-down position above a conveyor belt to capture images of food items as they pass below. Depending on the use case, different cameras are employed. The required hardware is listed below:

- **Intel RealSense D456:** Used for grading oranges, fish, and chicken fillets, envisioned in use-cases 1, 3 and 4, respectively.
- **Photoneo MotionCam 3D-S:** Used for raspberry grading, addressed in use-case 2.
- **Tripod**: Each camera is mounted in a fixed, top-down position using a tripod, positioned above a conveyor belt.
- **USB Cable**: A high-speed USB cable connects the camera to the processing computer.
- **Processing Computer with GPU**: A workstation or industrial PC equipped with a compatible GPU (e.g., NVIDIA) is used for real-time AI-based image analysis.
- **Mini Illumination Source:** For the raspberry grading use case, a dedicated mini-illumination source is installed near the MotionCam 3D-S camera to enhance image quality by providing consistent lighting conditions.

These components work together to capture high-quality images of food products as they move along the conveyor and to process them locally in real time.

Note that the Intel RealSense D456 camera is smaller in size and lighter in weight than the MotionCam 3D-S. Therefore, the former sensor is mounted on a single tripod, while the latter

is mounted on a bridge-like setup supported by two tripods. The acquisition setups of both sensors are illustrated in**Figure 1** and **Figure 2**, respectively.

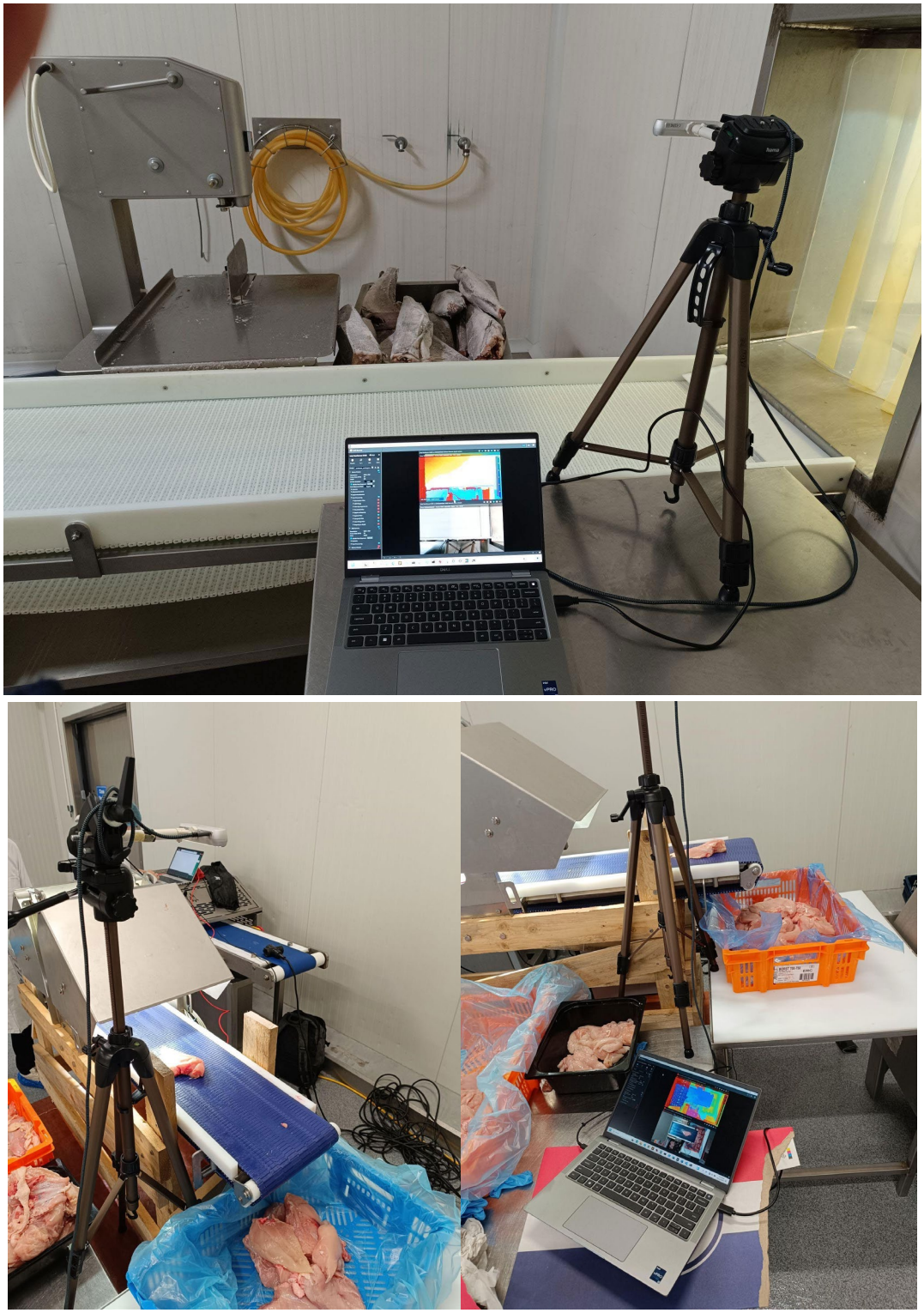

Figure 1. Grading system acquisition setup for fish, chicken and orange grading.

Figure 2. Grading system acquisition setup for raspberry grading.

2.2. Software and AI Processing

The **AGILEHAND** system processes the images captured by the cameras using advanced AI algorithms to perform automated grading of food products in real time.

- The core AI technology is based on the YOLOv8 instance segmentation model, trained on annotated datasets to recognize relevant features such as size, ripeness, cutting lines, and external quality.
- Images are transferred from the camera to the processing computer via USB, where the AI model analyses them and produces grading results.
- AI processing runs locally on a workstation equipped with a GPU to ensure efficient and fast inference.
- The system's software integrates image acquisition, AI inference, and results display/printing in a seamless workflow for operators.

3. Use cases

3.1. Orange Grading Based on Size

This use case focuses on grading oranges based on their size. The system analyzes oranges located in the top layer of a crate moving along the conveyor belt. The workflow is the following:

- The user launches the grading algorithm via the command line (**Figure 3**).
- The algorithm prompts the user to press Enter to capture several depth images, which are averaged to create a baseline depth image (**Figure 4**).
- The user is then prompted again to press Enter to capture the orange crate (**Figure 5**).
- The camera captures 3D images of the entire crate.
- The YOLOv8 instance segmentation model segments all oranges in the crate.
- Based on shape analysis (roundness), the system selects oranges in the top layer.
- For each top-layer orange, the system calculates the metric diameter using the depth map data from the RealSense camera.
- A histogram is generated to summarize the size distribution of the graded oranges.
- The operator monitors the system display to verify grading results and size distribution histograms (**Figure 6**).
- No manual input is required during the grading process.
- In case of system alerts or errors, the operator follows troubleshooting guidelines provided in the Installation Manual.

Figure 3. Orange grading execution.

Figure 4. Screenshot of the first prompt.

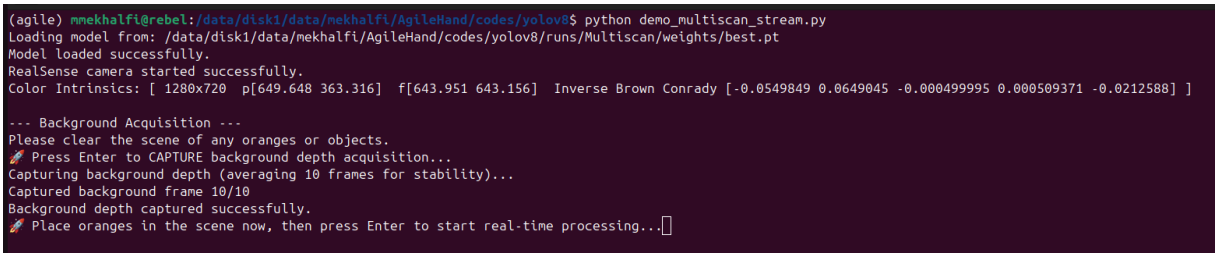

Figure 5. Screenshot of the second prompt.

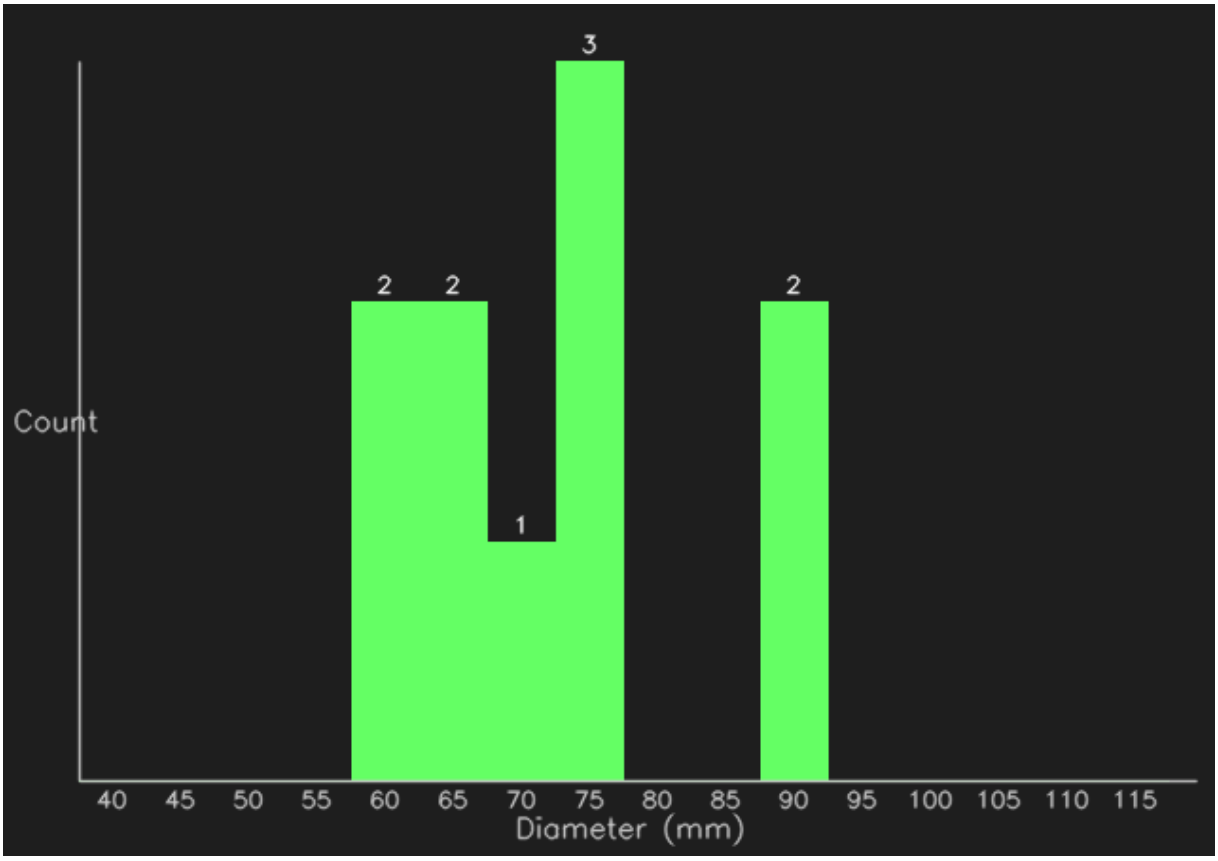

Figure 6. Results monitoring example.

3.2. Raspberry Grading by Ripeness Level

This use case focuses on classifying raspberries into five classes based on their ripeness level. The fruits are graded inside a transparent punnet placed on the conveyor belt. The workflow is provided below:

- The user launches the grading algorithm via the command line (**Figure 7**).
- The Photoneo MotionCam 3D-S camera, mounted on a bridge-like setup supported by two tripods, captures images of the raspberries.
- A mini illumination source provides consistent lighting to improve image quality.
- The YOLOv8 instance segmentation model segments raspberries in the punnet.
- The system analyzes color and texture features from the images to classify each raspberry into one of five ripeness classes.
- Grading results are displayed on the system interface for operator review.
- The user launches the grading application and monitors the grading results on the display. The results should look like the example illustrated in **Figure 8**, where the fruit segments determine the contours of individual raspberries, and the class number

inside each segment determines the ripeness grade (from 1 to 5), the class label 0 stands for the punnet.

- No manual input is required during normal operation. For any alerts or errors, operators should refer to the troubleshooting section of the Installation Manual.

```
(agile) mmekhalfi@rebel:/data/disk1/data/mekhalfi/AgileHand/codes/yolov8$ python santorsola_grading.py
```

Figure 7. Raspberry grading execution.

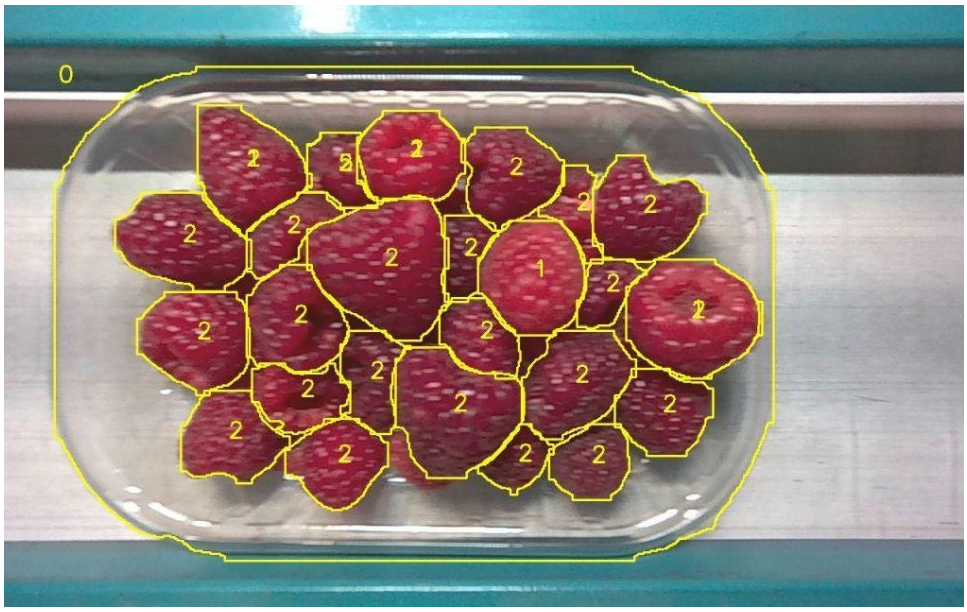

Figure 8. Raspberry grading monitoring example.

3.3. Fish Processing

This use case consists of two subtasks related to fish products:

A. **Cutting Line Determination:**

- The user launches the cutting line determination algorithm via the command line (**Figure 9**).
- The Intel RealSense D456 camera, mounted on a tripod in a fixed top-down position, captures RGB images of whole fish.
- The YOLOv8 instance segmentation model segments the fish and predicts two key points that define the cutting line on the head side.
- The system displays the predicted cutting line on the interface for operator review. The results should look like the example in **Figure 10**.

B. **Fish Steak Grading:**

- The user launches the fish steak grading algorithm via the command line (**Figure 11**).
- The RealSense camera captures images of fish steaks placed on the conveyor belt.
- The YOLOv8 model segments the fish steaks.
- The system measures the metric area of each fish steak and grades it into one of two size classes based on a threshold provided by the operator.
- Grading results are displayed on the system interface for operator review. The results should look like the example in **Figure 12**.
- No manual input is required during normal operation. For any alerts or errors, operators should refer to the troubleshooting section of the Installation Manual.

```
(agile) mmekhalfi@rebel:/data/disk1/data/mekhalfi/AgileHand/codes/yolov8$ python produmar_cuttingline.py
```

Figure 9. Fish cutting line prediction execution.

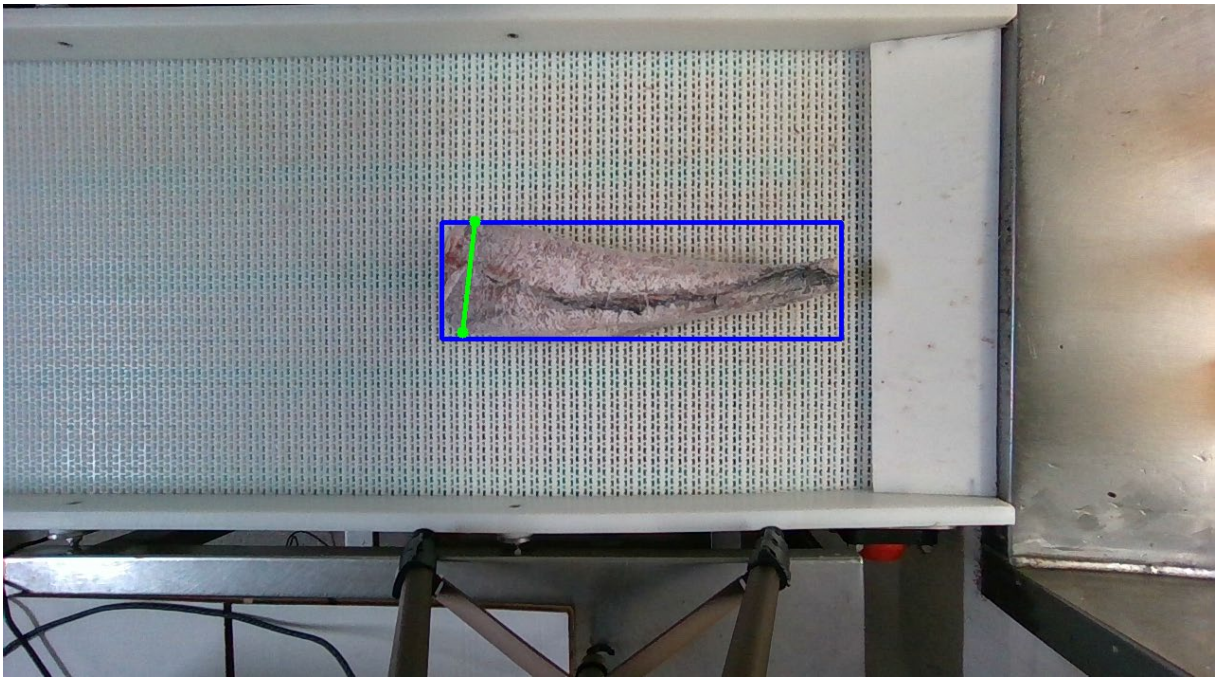

Figure 10. Fish cutting line result visualization. The bounding box determines the detected fish, and the green line represents the predicted cutting line.

```
(agile) mmekhalfi@rebel:/data/disk1/data/mekhalfi/AgileHand/codes/yolov8$ python produmar_steakgrading.py
```

Figure 11. Steak grading execution.

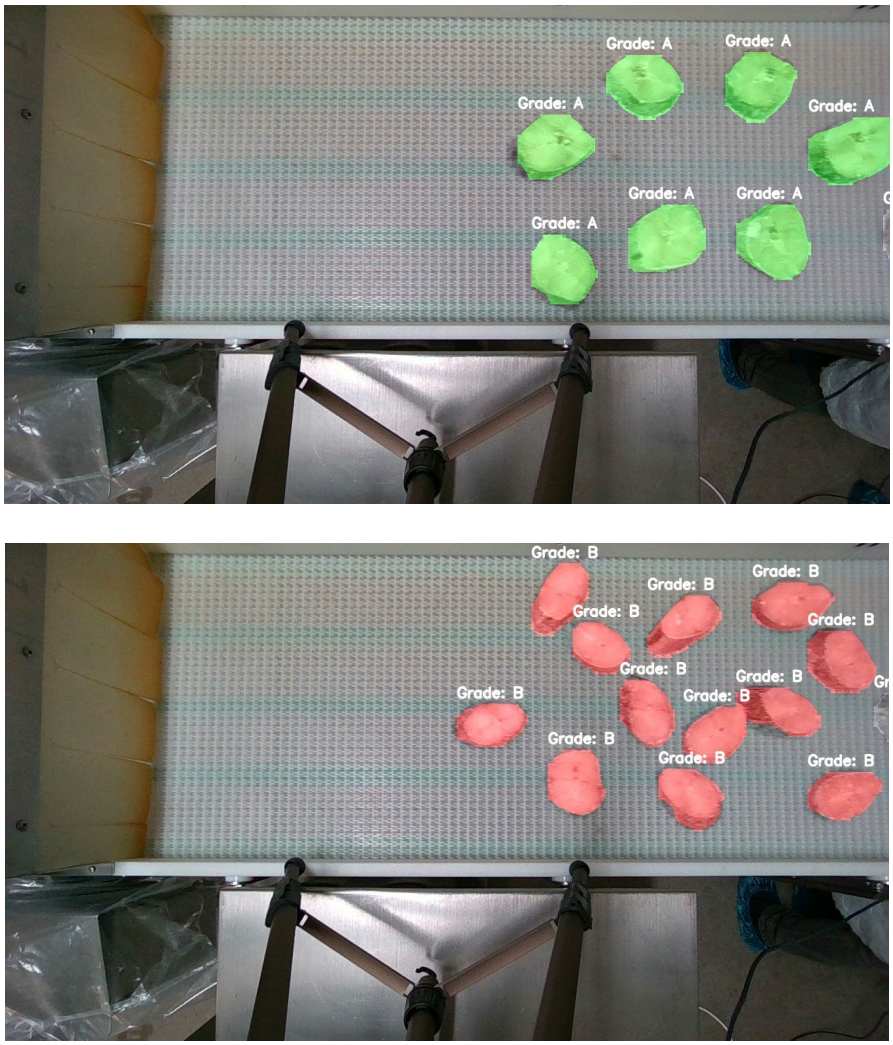

Figure 12. Steak grading result visualization.

3.4. Chicken Fillet Grading

This use case focuses on grading chicken fillets into two classes based on their external quality and appearance. The step-wise workflow is provided here:

- The user launches the chicken grading via the command line (**Figure 13**).
- The Intel RealSense D456 camera, mounted on a tripod in a fixed top-down position, captures 3D images of chicken fillets moving along the conveyor belt.
- The YOLOv8 instance segmentation model segments the chicken fillets in the images and grades each fillet into one of two quality classes.
- Grading results are displayed on the system interface for operator review. The results should look like the example in **Figure 14**.
- No manual input is required during normal operation. Operators should refer to the troubleshooting section in the Installation Manual if any errors or alerts occur.

```
(agile) mmekhalfi@rebel:/data/disk1/data/mekhalfi/AgileHand/codes/yolov8$ python marelec_grading.py
```

Figure 13. Chicken fillet grading execution.

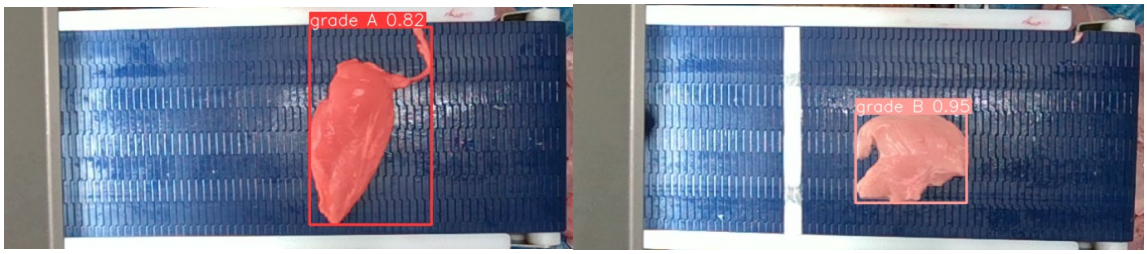

Figure 14. Chicken fillet grading result monitoring. The Fillet grade (A or B) as well as the grading confidence from 0 to 1 are depicted on the top of the fillet.

4. System Usage Guidelines

Proper placement of items ensures reliable image acquisition and grading. Below are the guidelines for each use case:

- **Oranges:** Place the crate on a stable surface (e.g., conveyor belt, platform, or ground). Ensure the camera has an unobstructed top-down view of the crate. The system automatically selects oranges from the visible top layer; no special arrangement of fruits is required.
- **Raspberries:** A clean punnet and moderate spacing help improve results, uniform filling and precise arrangement are not required.
- **Fish cutting line:** Place each fish individually with consistent orientation (e.g., preferably head facing left) to help the system identify the cutting line accurately. Avoid overlaps.
- **Fish steaks grading:** Lay steaks flat and spaced apart enough so that each can be segmented as a separate object. Overlapping should be avoided.
- **Chicken Fillets:** Place fillets flat with the external surface facing upward (preferred). Ensure they are not overlapping to allow accurate segmentation and assessment.

General Notes:

- Ensure the imaging area is clean, stable, and free of reflective or shiny materials.
- Do not obstruct or cover the items during acquisition — avoid placing hands, tools, or any foreign objects in the camera's field of view.
- Make sure items are positioned directly under the camera within the acquisition zone.
- For the raspberry use case, switch on the mini illumination source and verify proper lighting before capture.
- Check that the camera lens is clean and the USB connection to the processing computer is secure.

5. Troubleshooting

5.1. Common Issues and Solutions

Issue	PossibleCause	Suggested Action
System fails to start	Incorrect command or missing dependencies	Verify command syntax; check software installation and dependencies
Camera not detected or no image	USB cable disconnected or faulty camera	Check USB connection; restart camera; verify camera power and status
Poor image quality or dark images	Lens dirty, insufficient lighting	Clean camera lens; ensure illumination source is on (especially for raspberries)
Segmentation fails or incorrect	Model file corrupted or incorrect input	Verify model files; re-run algorithm with proper input
No grading results displayed	Processing error or software crash	Restart the system; check logs for errors;
Unexpected system alerts	Hardware or software malfunction	Follow alert instructions; restart system; if issue continues

5.2. When to Contact Technical Support

Contact technical support if:

- The system does not respond to restarts or commands.
- Hardware malfunctions persist after basic troubleshooting.
- Software errors or crashes occur repeatedly.
- Model performance appears degraded.
- You require assistance with advanced configuration or maintenance.

Include details such as system logs, error messages, and a description of steps taken when reporting issues.

6. Frequently Asked Questions (FAQs)

Q1: Do I need to train or retrain the AI model before using the system?

A: No. The system is delivered with pre-trained models for each use case. No user-side training or annotation is required.

Q2: Can the system work with products other than oranges, raspberries, fish, or chicken fillets?

A: No. The current release only supports the four specified use cases. Extending the system to other products would require AI model development and training.

Q3: Is internet access required for using the system?

A: No. Once installed, the system runs locally and does not require an internet connection.

Q4: How often should the camera lens or setup be cleaned?

A: It's good practice to clean the lens and check setup alignment at the start of each workday or shift to maintain accuracy.

Q5: Can I reposition the camera or move the acquisition setup?
A: Ideally, the camera setup should remain fixed once placed. Moving the camera may affect image quality and grading accuracy. If repositioning is necessary, consult technical staff.