Warehouse stock counting prototype using Raspberry Pi and OpenCV

Edwin Balderas
Embedded System Especialization
ITESO
Tlaquepaque, México
SE729011@iteso.mx

***Abstract*—A precise and up-to-date warehouse inventory is important to increase efficiency of item tracking and to reduce errors and the time invested to complete these tasks by employees. This paper proposes a warehouse stock counting prototype to improve operation efficiency. The prototype is built on the Raspberry Pi 4 development board and the OpenCV library to identify the warehouse item based on its color. The prototype can identify two different items and shows the available stock as the items leave the warehouse. The prototype maintains an up-to-date inventory and shows the stock on a screen in real time.**

Keywords—OpenCV, Raspberry Pi, Raspberry Pi OS, Prototype, Warehouse, Items

I. INTRODUCTION

Nowadays, industries in the warehousing and storage subsector require an efficient management system to perform accurate inventories, optimize operations, and reduce costs. However, the main problems that routinely affect warehouse operations include keeping track of stock in real time by counting how many items leave or enter the warehouse and those items currently stored in the warehouse. At least two employees are required to verify and count the stock in real time, and then confirm the accuracy of the data. This is not only a time-consuming process for employees, but it is also susceptible to human error causing the loss of revenue, and ultimately, a negative outcome for the company [1].

The use of technology such as software to manage warehouse inventories is common today but these types of software need to be fed by employees, making this process still exposed to mistakes. Computer vision is a technology that can analyze and process digital images to obtain precise information, such as content-based video. This technology is currently used in warehouse operations where the product needs to be identified to verify the barcode in real time [2]. OpenCV (Open Source Computer Vision Library) is an essential part of the computer vision tool that can be used to achieve an accurate inventory of the items stored inside a warehouse in real time. In this context, OpenCV uses the digital images as an information source.

The OpenCV library has predefined functions that help identify and analyze the features of an image, such as the color. This feature can be used to identify the color of the warehouse item. The color of a digital image is in the RGB (Red, Green, Blue) model [3]. The predefined functions are able to locate the item on the video according to its color. The function can be configured to identify the item as inside or outside the warehouse. As a result, OpenCV offers a reliable and practical solution for developing a warehouse stock counting prototype system.

The objective of this work is to present a prototype of an embedded system capable of counting the number of items in a warehouse. This prototype consists of a Raspberry Pi 4 development kit (Hardware) used as a computer with an HDMI monitor, mouse and keyboard, and the Raspberry OS (Operative System) based on Debian Linux. It is designed for using hardware resources in the best possible way, with the OpenCV library as part of the software. The focus of the prototype system is to count the items by using the hardware and software previously mentioned, however, the system is also capable of identifying two different items by identifying the respective number of both items on the monitor in real time.

The paper is organized as follows: section II describes the design process of the hardware and software used to develop the prototype. Then, in section III, the results for each case are presented, and the last section IV, describes the conclusion and the future work for this paper.

II. PROTOTYPE DEVELOPMENT

The waterfall methodology is used to develop the proposed prototype system. This process consists of 5 stages: Requirements analysis, system design (hardware and software), implementation, verification, and maintenance (Fig. 1).

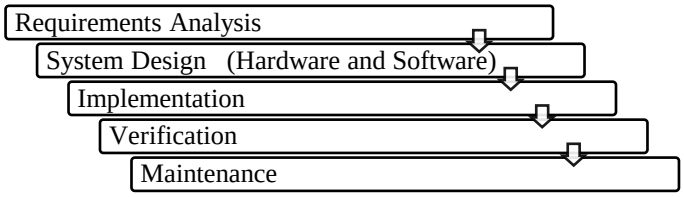

Fig. 1 Process to develop the prototype system.

A. *Requirements Analysis*

The prototype must meet and fulfill the following requirements [4]:

- Portable device (small in size)
- Allow the user to know the number of items in the warehouse in real time by detecting item direction
- Be able to identify between two types of items

B. *Prototype System Design*

To meet the requirements for the design stage, the prototype is developed in two parts: the first part for the hardware and the second, for the software. After analyzing the requirements, the following prototype is proposed (Fig. 2).

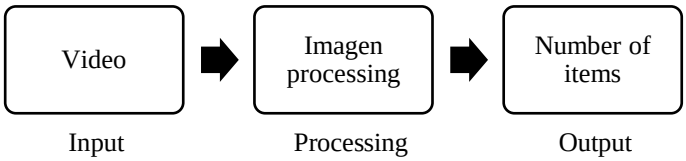

Fig. 2 Diagram of the prototype system.

The architecture proposed for this prototype is developed in layers (Fig. 3).

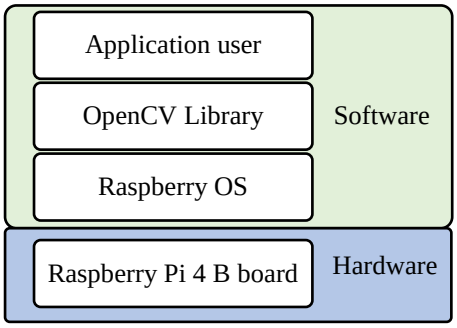

Fig. 3 Structure in layers of the implemented prototype system.

1) *Hardware design*

For the operation of the prototype, external components, such as a keyboard, mouse, microSD card, power supply, and an HDMI monitor are needed. The relationship between the prototype components is shown in Fig 4.

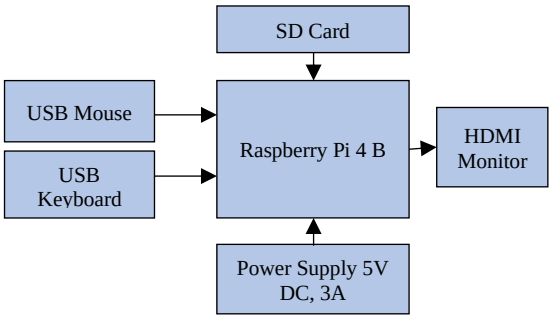

Fig. 4 The relationship between the prototype components.

The size of the prototype is crucial since it should take as little space as possible to allow its use as a portable device once it has been installed for warehouse operations. The Raspberry Pi 4 B was selected due to its processing unit, which is able to perform the execution of the required OpenCV tools, and it can also be connected with peripherals to control the Raspberry Pi and facilitate the installation in the warehouse. Raspberry Pi 4 uses open source software and reduces the upgrade cost if required. The following functional characteristics of Raspberry Pi are shown in Table 1.

TABLE I. PROTOTYPE COMPONENTS

Item	**Hardware**	
	Component	***Description***
1	Mouse	USB Mouse Cooler Master optical
2	Keyboard	USB Keyboard QWERTY type
3	SD Card	Kingston microSD card 32 GB of capacity
4	Power Supply	Power supply 5V DC, 3A
5	HDMI Monitor	HDMI 2K LG 25UM58
6	Raspberry Pi 4 B	•Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz •4GB LPDDR4-3200 SDRAM •2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE •Gigabit Ethernet •2 USB 3.0 ports; 2 USB 2.0 ports. •2 × micro-HDMI ports (up to 4kp60 supported) •H264 (1080p60 decode, 1080p30 encode) •OpenGL ES 3.0 graphics •Micro-SD card slot for loading operating system and data storage •5V DC via GPIO header (minimum 3A*) •Operating temperature: 0 – 50 degrees C ambient

2)*Software Design*

The software proposed to develop the algorithm is mainly made up of the predefined functions that the OpenCV tool offers. The functions are used in the application layer, which interacts directly with the user. The functions performed by each layer are described below.

a. **Raspberry Pi OS**: The second layer is the operating system, it performs the function of managing hardware resources, and at the same time it serves as the basis for the OpenCV library [5].

b. **OpenCV Library**: The third layer is a computer vision tool used to manipulate and extract information from an image by using the set of predefined functions. It is necessary for developing the application user.

Identify applicable funding agency here. If none, delete this text box.

c. Application user: In this layer, the algorithm is developed using the predefined functions from the OpenCV library. After processing the image, the application can detect the number of items in the warehouse, then the information is presented to the user. The flow of the proposed application algorithm is shown in Fig. 7.

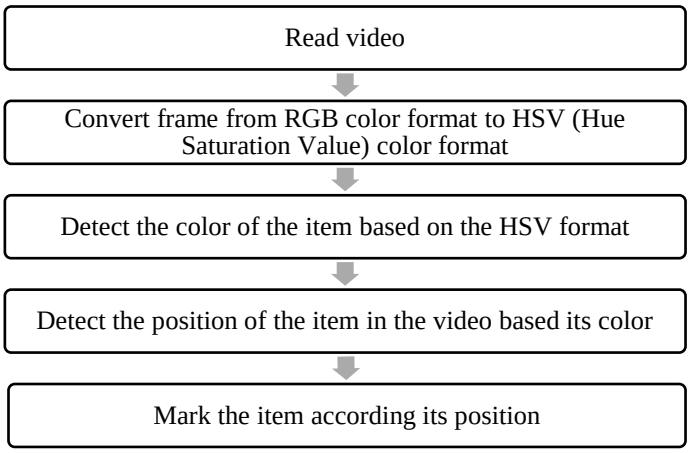

Fig. 7 Flow of the proposed application algorithm.

C. Implementation

The prototype implementation begins with connecting the components as shown in Fig. 8. The main component that performs software processing and communication with the user is the Raspberry Pi 4 B.

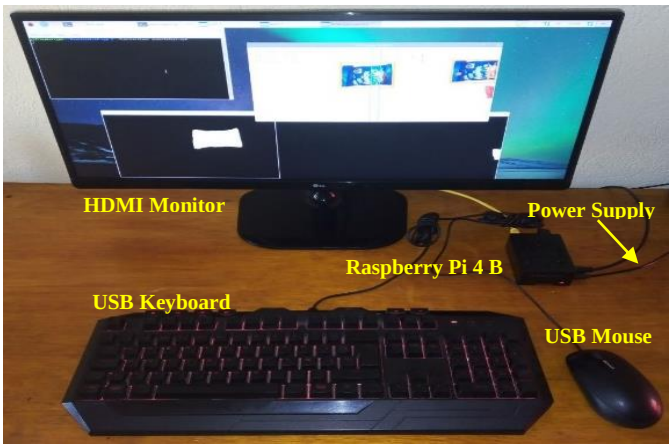

Fig. 8 The connected prototype components.

The code developed in the application layer based on the predefined functions of OpenCV is shown in Fig. 9. The process begins by reading the video, then converts each video frame from RGB to HSV (Hue Saturation Value) color. The image in HSV color is used to limit item color as a filter. The filtered color of the item is used to create a binary image, so that it can be easily analyzed [6]. From the binary image, the contour of each detected item is obtained, which is used to identify the center of each item. The center is used to determine if it is inside or outside of the warehouse depending on the its position of the item in the video.

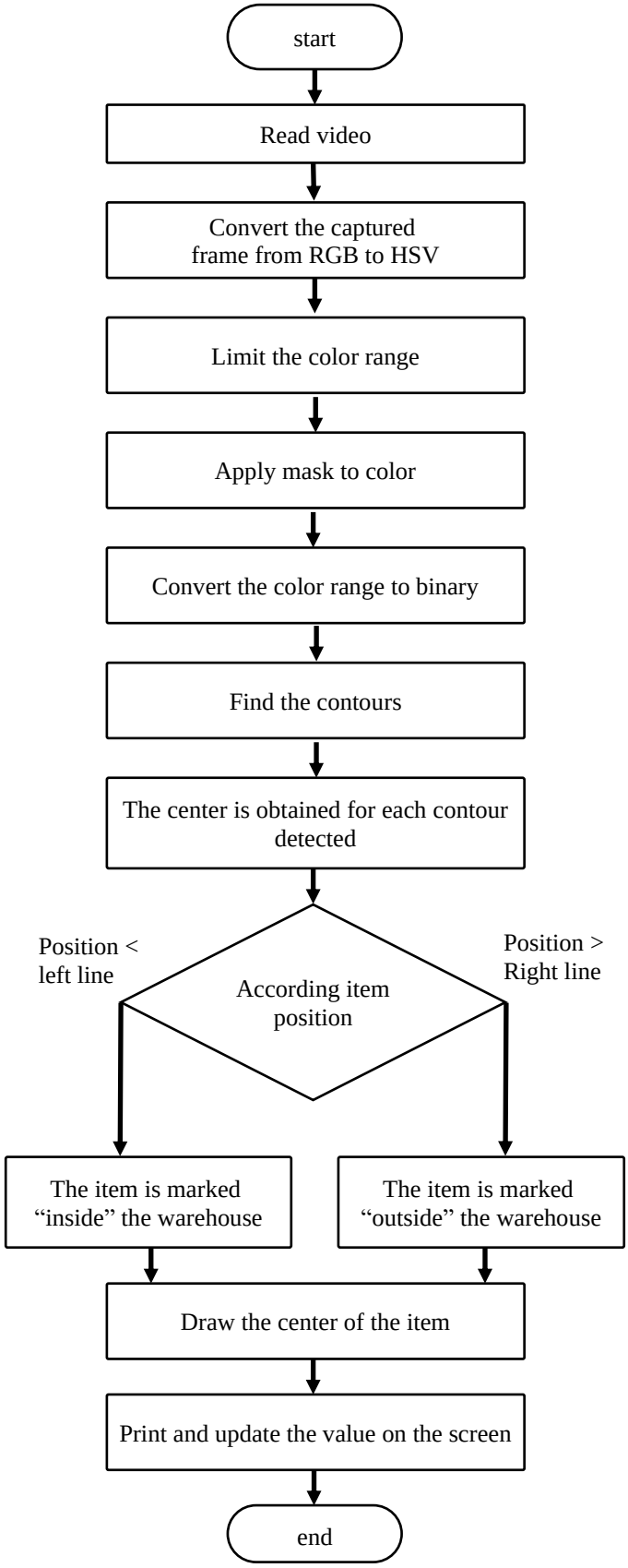

Fig. 9 Code development workflow.

D. *Verification and Maitenance*

The installation of the Raspberry Pi OS and the application code was successful on the hardware. The application code was tested under controlled conditions, such as the size and color of the item to verify correct behavior. As part of the maintenance that the prototype code needs to be updated if a new item is required.

III. Results

The prototype counts the number of items that enter and leave the warehouse. It can count two types of items at the same time and recognizes the item in the video and marks it with a contour of one color as it draws the center of the item with a dot. The center of “Item_1” is drawn in yellow and the contour is pink, and the “Item_2” has a yellow contour and pink center. While items are placed before the light green line and go to left side of the video, if the second cyan line is reached, the prototype adds the item as “inside” the warehouse, and then shows the current stock on screen (Fig. 10-a). The mentioned flow is performed for each item in this prototype.

The stock that leaves the warehouse is shown as “OUT” on the right side screen; if items come from the left side to the right side of video and the cyan light green line is reached, the “Item_1” is added, indicating that items leave the warehouse; then these items are removed from the inside stock by showing the up-to-date stock (Fig. 10-b).

(a) (b)

Fig. 10 Warehouse stock counting prototype running: (a) items entering to warehouse, (b) items leaving the warehouse.

Conclusions and Future Works

The developed prototype identifies the color of an item, making it possible to count different items in real time. The results show that items are counted even if they have letters and pictures, allowing it to be applied to different types of items. The prototype allows to recognize a new item just by adding the item's color in the main code. However, the prototype has the limitation that if an item has the same background color, it causes conflict with the other items.

The prototype can be upgraded by connecting a web cam either a Raspberry Pi camera module or USB cam, and the prototype can be connected to network by using a remote access device.

References

[1] X. Tang and J. Huang, "Research on the problem of warehouse supervisors' arrangement of medicine logistics," *2016 International Conference on Logistics, Informatics and Service Sciences (LISS)*, Sydney, NSW, 2016, pp. 1-5, doi: 10.1109/LISS.2016.7854332.

[2] J. Wells, “Evans deploys machine vision AI to improve warehouse operations,” KMWorld, 17-Apr-2020. [Online]. Available: https://www.kmworld.com/Articles/News/KM-In-Practice/Evans-deploys-machine-vision-AI-to-improve-warehouse-operations-140330.aspx. [Accessed: 02-Dec-2020].

[3] S. D. Gajbhiye and P. P. Gundewar, "A real-time color-based object tracking and occlusion handling using ARM cortex-A7," 2015 Annual IEEE India Conference (INDICON), New Delhi, 2015, pp. 1-6, doi: 10.1109/INDICON.2015.7443641.

[4] ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes -- Requirements engineering," in ISO/IEC/IEEE 29148:2018(E) , vol., no., pp.1-104, 30 Nov. 2018, doi: 10.1109/IEEESTD.2018.8559686.

[5] J. Hughes, “raspberrypi/documentation,” *GitHub*, 2020. [Online]. Available: https://github.com/raspberrypi/documentation/blob/master/linux/README.md. [Accessed: 03-Dec-2020].

[6] R. S. Deepthi and S. Sankaraiah, "Implementation of mobile platform using Qt and OpenCV for image processing applications," 2011 IEEE Conference on Open Systems, Langkawi, 2011, pp. 284-289, doi: 10.1109/ICOS.2011.6079235.