University of Applied Sciences
HOCHSCHULE EMDEN·LEER

Machine Vision with Arduino Portenta H7

Getting Started With the Portenta Vision Shield Camera - Save Images

Vijay Singh

August 1, 2024

Table of Content I

1 Overview

2 Components

3 Conclusion

4 Future Work

Introduction

This tutorial shows you how to capture a frame from the **Portenta Vision Shield** Camera module and save the output as a bitmap image. It will allow you to see the output directly on your computer without using any third party tool. **??**

Objectives

- Capturing the frames from the camera
- Make the bitmap binary file with the correct settings.
- Save the bitmap on a SD Card and PC.
- Visualize the captured image on your computer.

Importance

Real-time object detection has applications in surveillance, industrial automation, robotics, and IoT, among others.

Required Hardware and Software

Description

The project involves the following components:

- **Arduino Portenta H7:** The core microcontroller unit providing processing power and resources for boarding different shields.
- **Portenta Vision Shield:** An accessory for the Portenta H7, equipped with a camera module and display for image capture and processing.
- **USB-C cable:** A pre-trained model deployed on the Portenta H7 for object detection tasks.
- **Micro SD card:** A arduino software to capture the image data by the onboard camera.
- **Arduino IDE:** A processing software helps to visualize the data

Figures

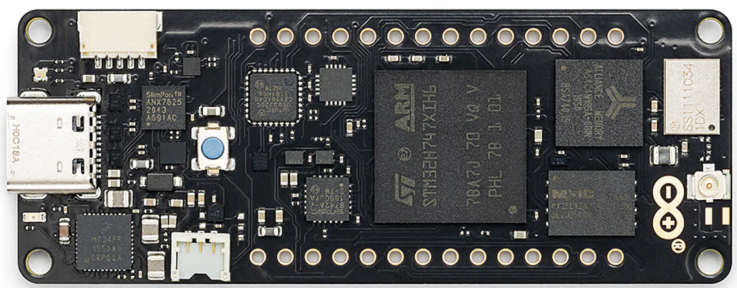


Figure 1: Arduino PortentaH7

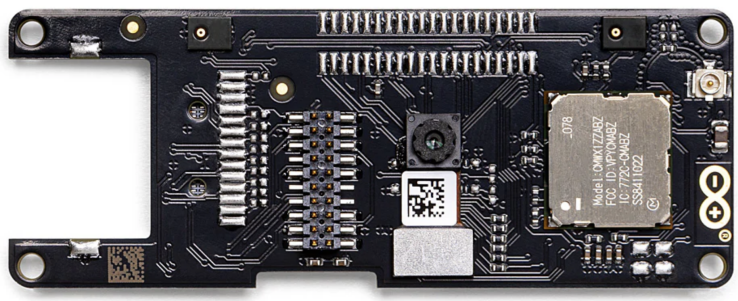


Figure 2: PortentaH7 Vision Shield

1. The Setup:

- Connect the Portenta Vision Shield to your Portenta H7 as shown in the figure. The top and bottom high density connectors are connected to the corresponding ones on the underside of the H7 board. Plug in the H7 to your computer using the USB-C cable.

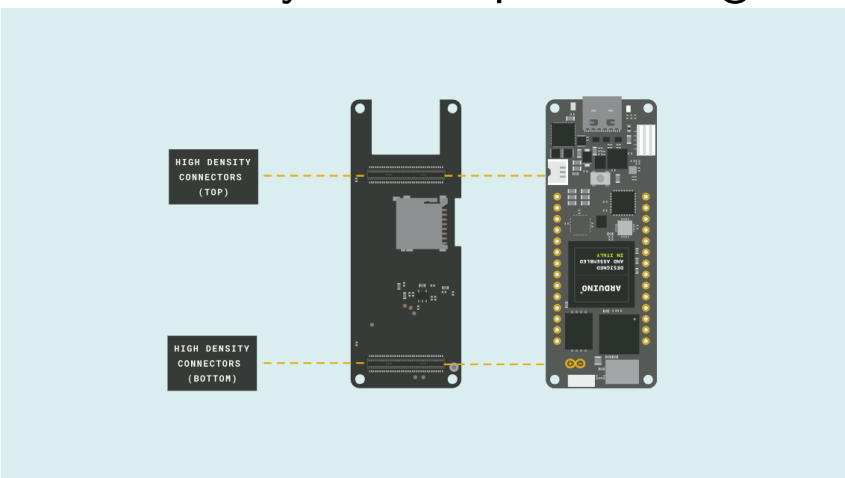


Figure 3: Connection VS

- **The Camera:** You will be using the Himax HM-01B0 camera module which has a resolution of 320 by 240 and the output data its in grayscale with 8 bits per pixel (bpp). It is important to have this in mind as the .bmp (bitmap) format has some needed configuration depending on the data being used.
  Inside the sketch, you can use these libraries to access the camera APIs

```
#include "camera.h" // Multi Media Card APIs
#include "himax.h"  // API to read from the Himax camera fou
```

- **Bitmap File Format:** The bitmap binary file needs to contain some information in order to tell the computer for example the resolution of the picture and the bit-depth (bpp). Bit depth refers to the color information stored in the image. The higher the bit depth of an image, the more colors it can store. As the bit depth increases, the file size of the image also increases, because more color information has to be stored for each pixel in the image.
  The following table shows all the headers, the size of its buffer, offsets, the settings that are being used with their details:

Name	Size	Details
Device Independent Bitmap(DIB)	14 Bytes	Bitmap information, s
File Header	40 Bytes	This header requires t
Palette (Color Map)	1025 Bytes	This header is mandat
Image data	76800 Bytes	The raw image data, i

Table: Bitmap File Format Details

2. The Sketch

You can find the sketch on the latest version of the Arduino-Pro-Tutorials at examples ¿ Vision Shield to SD Card bmp ¿ visionShieldBitmap.ino

Create a new Arduino sketch called CameraCapture.ino.

To capture the frames you will need to use the functions contained in **camera.h** which comes with the Portenta core. This library contains all APIs related to frame capturing, motion detection and pattern recognition. Include the header file in your sketch. content...

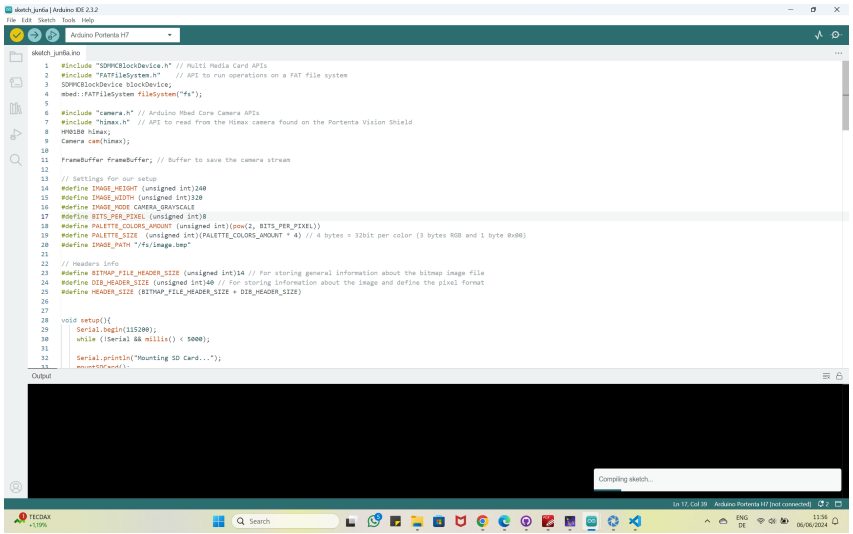


Figure 4: Camera-save

Detailed Explanation of header files

camera.h

- **Functionality:**Provides an interface to work with camera modules in the Arduino environment. It abstracts the camera hardware specifics and offers functions to initialize the camera, capture images, and configure camera settings.
- **Key Functions:** 1. begin(resolution, mode, fps): Initializes the camera with specified resolution, image mode (e.g., grayscale), and frames per second.
  2. grabFrame(buffer, timeout): Captures an image frame and stores it in the provided buffer.
  3. getResolution(), setResolution(), etc.: Functions to get and set camera parameters.

himax.h

- **Functionality:**Contains specific functions and definitions for working with the Himax HM01B0 camera. This file includes lower-level commands tailored to the Himax camera module, managing settings like exposure, gain, and capturing images.
- **Key Functions:** 1. init(): Initializes the Himax camera module.
  2. readFrame(buffer): Reads an image frame from the Himax camera and stores it in the provided buffer.
  3. setExposure(time), setGain(value), etc.: Functions to adjust camera settings specific to the Himax HM01B0 module.

Capture-save.ino

```
#include "SDMMCBlockDevice.h" // Multi Media
    Card APIs
#include "FATFileSystem.h"    // API to run
    operations on a FAT file system
SDMMCBlockDevice blockDevice;
mbed::FATFileSystem fileSystem("fs");

#include "camera.h" // Arduino Mbed Core Camera
    APIs
#include "himax.h"  // API to read from the
    Himax camera found on the Portenta Vision
    Shield
HM01B0 himax;
Camera cam(himax);

FrameBuffer frameBuffer; // Buffer to save the
    camera stream

// Settings for our setup
```

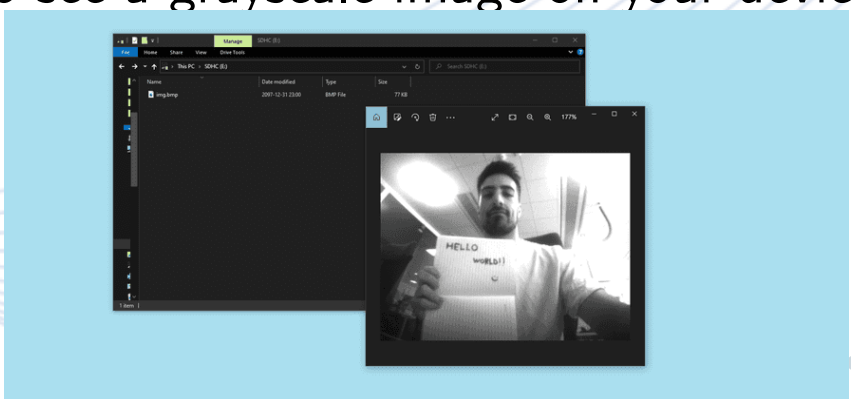
Continue...

```
// Write the bitmap file
fwrite(bitmapFileHeader, 1,
    BITMAP_FILE_HEADER_SIZE, file);
fwrite(bitmapDIBHeader, 1, DIB_HEADER_SIZE, file
    );
fwrite(colorMap, 1, PALETTE_SIZE, file);
fwrite(imageData, 1, IMAGE_HEIGHT * IMAGE_WIDTH,
     file);

// Close the file stream
fclose(file);
}

void countDownBlink(){
for (int i = 0; i < 6; i++){
  digitalWrite(LEDG, i % 2);
  delay(500);
}
digitalWrite(LEDG, HIGH);
```

Upload the Sketch

Saving on SD Card:

- Select the right serial port on your IDE and upload the Arduino sketch to your Portenta H7.
- Insert a micro SD Card into the Portenta Vision Shield.
- Connect the Portenta Vision Shield to the Portenta H7.
- Once the sketch is uploaded, open the Serial Monitor or wait 5 seconds: you should see that everything is fine and the capture has been taken.
- Once the capture is saved, remove the SD Card and plug it into a computer/phone with an SD Card reader, open the storage unit, look for a bitmap called image.bmp and open it to check the result. You will be able to see a grayscale image on your device's image viewer.



Saving on PC:

Steps to Save the Image on Your PC:

- 1. Modify Your Arduino Code:
- 2. Set Up Serial Communication on Your PC:
  a. Use a serial terminal application like PuTTY, CoolTerm, or the built-in Serial Monitor in the Arduino IDE.
  b .Ensure the serial terminal is set to the correct baud rate (115200 in this case) and connected to the correct serial port.
- 3. Receive and Save the Data on Your PC:
  a. Open the serial terminal and start the Arduino sketch. The image data will be sent over the serial connection.
  b. You will need a script or a tool on your PC to capture the incoming serial data and save it to a BMP file.
- Once the sketch is uploaded, open the Serial Monitor or wait 5 seconds: you should see that everything is fine and the capture has been taken.
- Once the capture is saved, remove the SD Card and plug it into a computer/phone with an SD Card reader, open the storage unit, look for a bitmap called image.bmp and open it to check the result. You

Python Script to Receive and Save Image Data on PC:

```
import serial

# Constants from Arduino code
BITMAP_FILE_HEADER_SIZE = 14
DIB_HEADER_SIZE = 40
IMAGE_WIDTH = 320
IMAGE_HEIGHT = 240
PALETTE_COLORS_AMOUNT = 256
PALETTE_SIZE = PALETTE_COLORS_AMOUNT * 4

def receive_image(serial_port, file_path):
ser = serial.Serial(serial_port, 115200, timeout
    =10)
file_size = BITMAP_FILE_HEADER_SIZE +
    DIB_HEADER_SIZE + IMAGE_WIDTH * IMAGE_HEIGHT
     + PALETTE_SIZE

# Read the entire file
image_data = ser.read(file_size)
```

Conclusion

In this presentation you learned how to capture frames with your Portenta Vision Shield's Camera in the Arduino IDE, encode it with the bitmap standards and save it to an SD Card and PC.

Future Work

- Creating a Basic Face Filter With OpenMV
- Connecting the Portenta Vision Shield to The Things Network(TTN) Using LoRa
- Send the camera data through it LoRa.

Thank you

for your attention