The SKU 3122-0004-0001 is a USB Camera with a goBILDA® Case.
It uses a UVC compatible USB Webcam which can record up to 1280 x 800 @ 120fps. It has USB camera controls including brightness, white balance, and exposure.
Its 70° recording angle strikes a great balance between low distortion and wide angle.
This camera uses a global shutter sensor to drastically reduce object distortion when there is relative motion between the camera and the tracked object.

Table of Contents:

Summary of Product Ratings:

Operating Voltage	5V		Dynamic Range	68dB
Input Current	< 200mA		Sensor	OmniVision OV9782
Interface Type	USB 2.0		USB ID	VID: 0xC45, PID: 0x366
Shutter Type	Global shutter		Wire Length	875mm

Supported Resolutions:

Resolution	Max Frame Rate	Format		Resolution	Frame Rate	Format
1280 x 800	120 fps	MJPEG		320 x 240	120 fps	MJPEG
1280 x 720	120 fps	MJPEG		1280 x 800	10 fps	YUY2
800 x 600	120 fps	MJPEG		1280 x 720	10 fps	YUY2
640 x 480	120 fps	MJPEG				

Some resolutions appear in this table twice in two different image formats. This camera supports both MJPEG and YUY2. MJPEG is more compressed, sending each frame of the video as a JPEG. This allows higher frame rates while staying within a reasonable bandwidth for USB 2.0.
YUY2 is a higher quality, lower loss format. It requires more bandwidth to send the same video, and so the frame rate available is much lower than with MJPEG. MJPEG will be a better format option for most robotic vision applications.

Camera Calibration Characteristics:

Resolution	Focal Length (fx, fy)	Principal Point (cx, cy)	Distortion Coefficients
1280 x 800	908.758f, 908.758f	696.345f, 376.979f	0.0374901, -0.0506747, 0, 0, -0.00229137, 0, 0, 0
1280 x 720	908.683f, 908.683f	706.785f, 337.571f	0.0357156, -0.0435922, 0, 0, -0.00930544, 0, 0, 0
800 x 600	684.509f, 684.509f	444.832f, 283.680f	0.034551, -0.0516021, 0, 0, 0.00228923, 0, 0, 0
640 x 480	545.584f, 545.584f	350.588f, 223.150f	0.0290195, -0.0238128, 0, 0, -0.0378927, 0, 0, 0
320 x 240	270.260f, 270.260f	173.207f, 120.382f	-0.0249932, 0.147249, 0, 0, -0.24925, 0, 0, 0

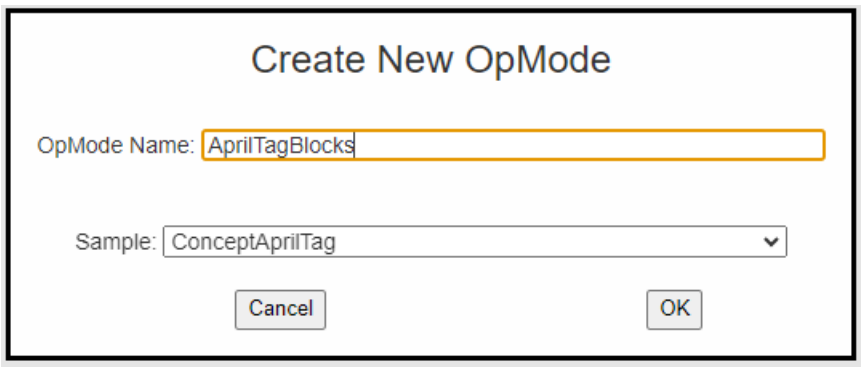
These were calibrated using 3DF Zephyr https://ftc-docs.firstinspires.org/en/latest/programming_resources/vision/camera_calibration/camera-calibration.html
We encourage you to run these tests for yourself, on your camera. These are a great starting point, but manufacturing variances can mean that these characteristics are subtly different camera-to-camera. Accurate calibration characteristics are the most important part of a good vision-based localization system.

Supported USB Camera Parameters:
- Brightness/Exposure
- White Balance

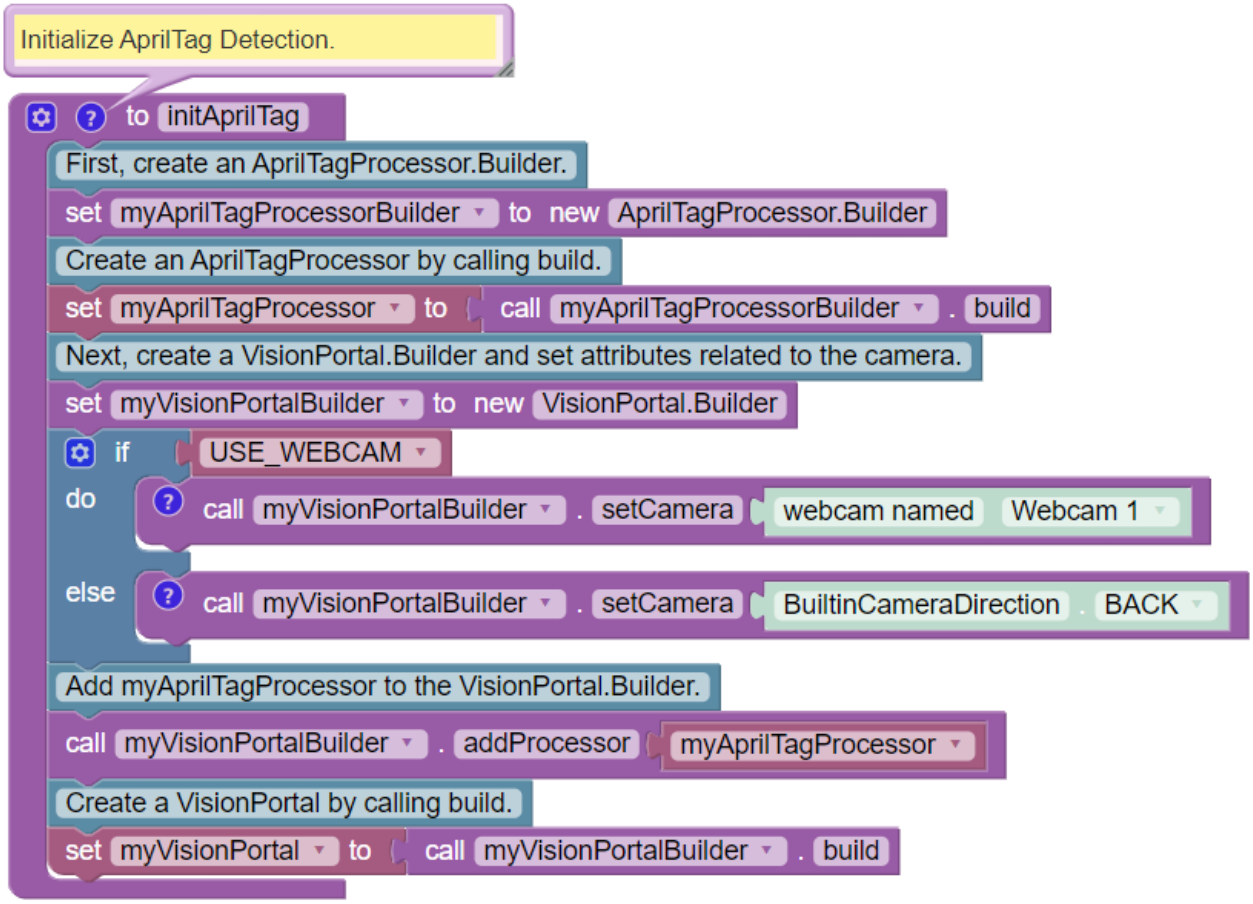
FTC Installation Guide - Blocks:

If you are using FTC Blocks, follow these steps to include these camera calibration characteristics in your code.

Step 1: Create OpMode based on the "ConceptAprilTag" sample



Step 2: Find the "InitAprilTag" function blocks

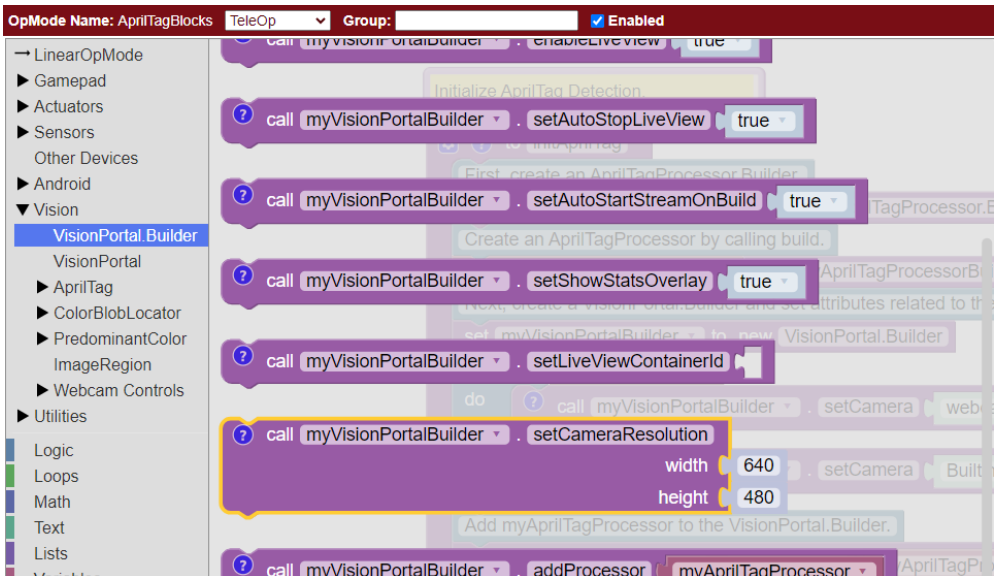


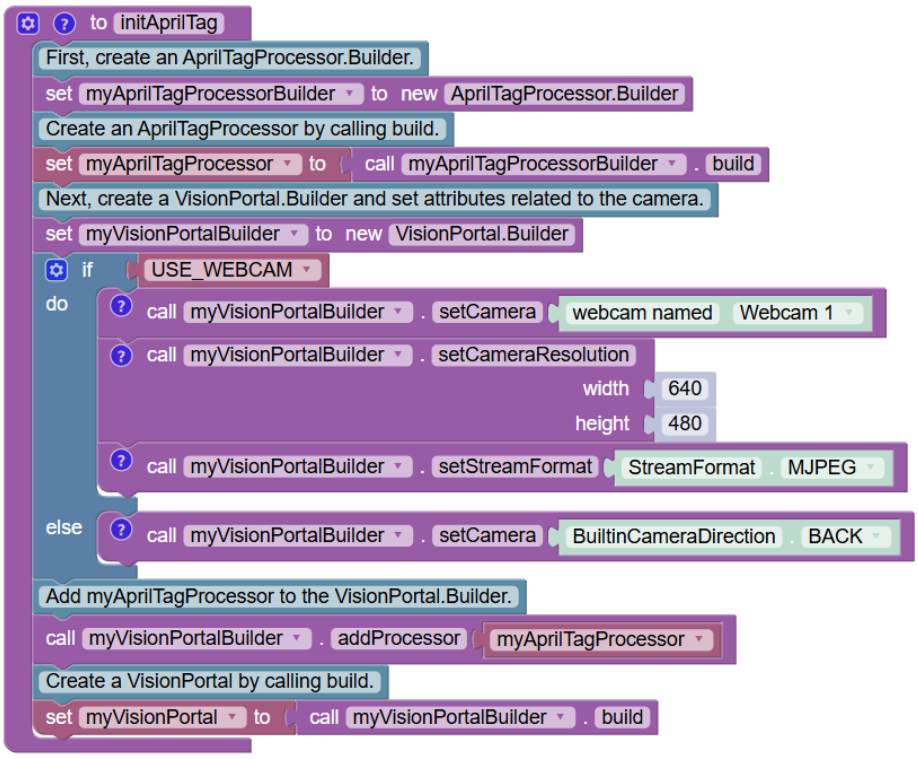
FTC Installation Guide - Blocks Continued:

Step 3: Set Camera Resolution

On the left side of your screen, navigate to Vision, then VisionPortalBuilder, and drag over the "setCameraResolution" Block into the "initAprilTag" function. Put it inside the if "USE_WEBCAM" statement, and set width to 640, and height to 480.

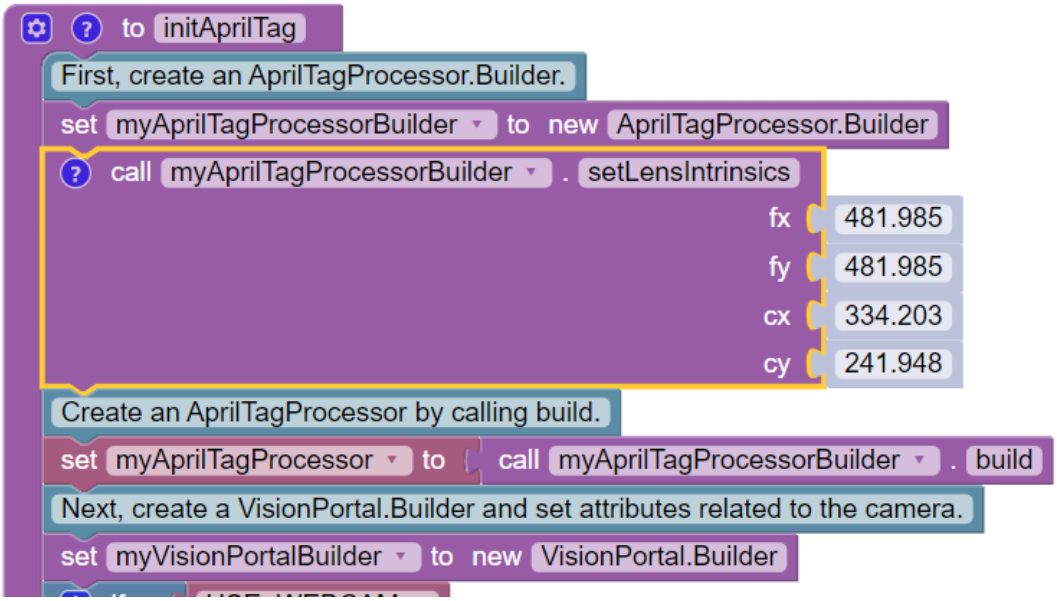
Next, navigate to Vision > AprilTag > AprilTagProcessorBuilder and drag the ".setStreamFormat" block under the ".setCameraResolution" block you just created. Set this block to "MJPEG".





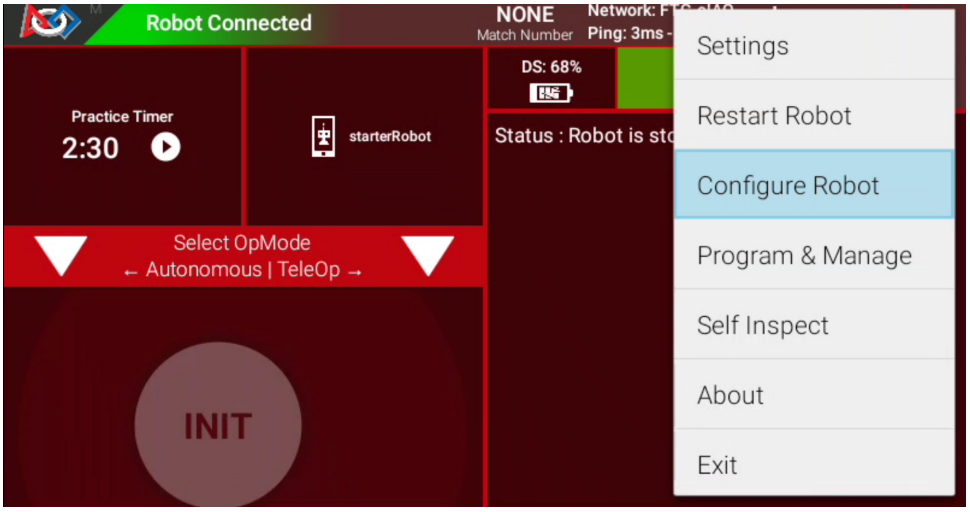
FTC Installation Guide - Blocks Continued:

Step 4: Set Camera Calibration
Navigate to Vision > AprilTag > AprilTagProcessorBuilder and drag over the “setLensIntrinsics” block. Put this under the “myAprilTagProcessorBuilder to new AprilTagProcessor.Builder” Block.
Refer to the Camera Calibration Characteristics table on page 2 to find the values you need to set here, based on your resolution. For 640x480 we need to set fx to 545.584, fy to 545.584, cx to 350.588, and cy to 223.15.



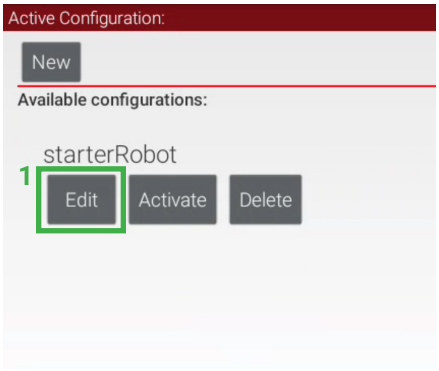
Note: Refer to the table on page 2 for the webcam calibration Characteristics that match the resolution you are using.

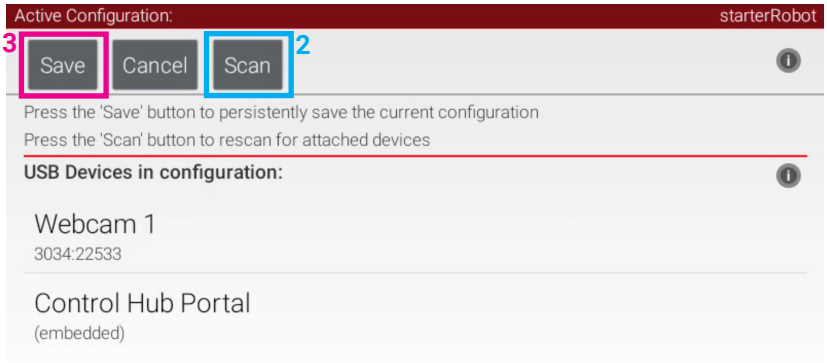
Step 5: Configure Hardware
Plug the Camera into the USB 3.0 port on your Control Hub, click the menu at the top right of your driver’s station, and click “Configure Robot”.



FTC Installation Guide - Blocks Continued:

Step 6: Add Webcam to Robot Configuration
Click “Edit” on the configuration you’d like to use, and then click “Scan”. You should see a device appear called “Webcam 1”. Leave that name as-is, and click the “Save” button at the top left.



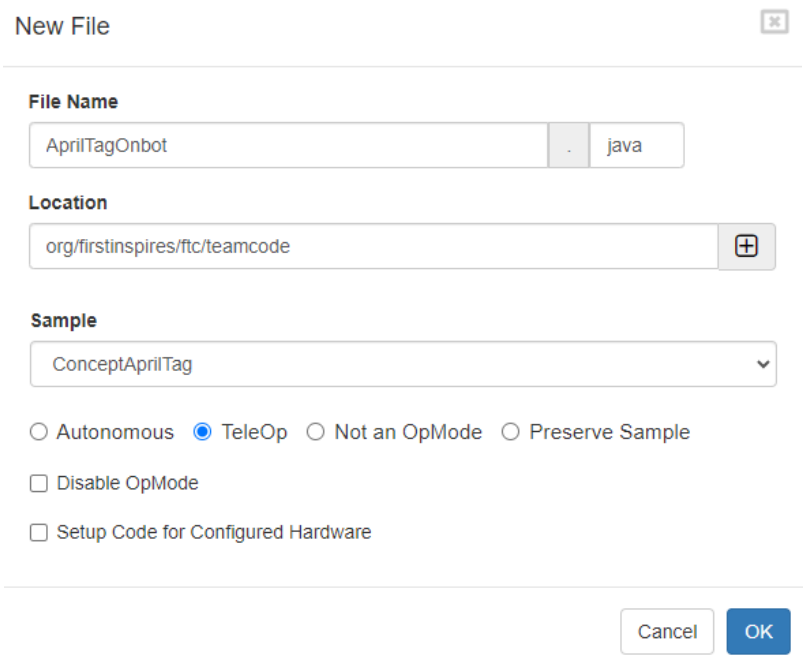


Step 7: Test the OpMode
Now that you’ve created the OpMode and modified it to match your camera, run your freshly-created OpMode and point the camera at an AprilTag. In your Drivers’ Station Telemetry you should see the tag detected, and an estimate of the location of the tag relative to your robot.

FTC Installation Guide - Onbot Java:

In this section, we’ll follow a very similar set of steps to set up this camera for use with Onbot Java.

Step 1: Create OpMode based on the “ConceptAprilTag” sample



FTC Installation Guide - Onbot Java Continued:

Step 2: Set Camera Resolution
On line 123 you will find the function initAprilTag(). This gets called on line 87 to apply all these settings at once. Using a function here cleans up the code a lot.
Inside the initAprilTag() function, you'll find an if statement on line 157. This checks to see if the user has selected that they are using a webcam, or the built-in camera on an Android device. After line 158, add the statement:

builder.setCameraResolution(new Size(640,480));

This sets the webcam resolution to 640x480p.

After that, add another line with the statement:

builder.setStreamFormat(VisionPortal.StreamFormat.MJPEG);

This sets the camera stream to the more efficient MJPEG format.

```
156         // Set the camera (webcam vs. built-in RC phone camera).
157         if (USE_WEBCAM) {
158             builder.setCamera(hardwareMap.get(WebcamName.class, "Webcam 1"));
159             builder.setCameraResolution(new Size(640,480));
160             builder.setStreamFormat(VisionPortal.StreamFormat.MJPEG);
161         } else {
162             builder.setCamera(BuiltinCameraDirection.BACK);
```

Step 3: Set Camera Calibration
On line 139 you'll find a commented-out line of code that sets the "lens intrinsics" for your camera. Uncomment this line. And change the calibration values to match the intrinsics found for your selected resolution on page 2 of this PDF.
.setLensIntrinsics(545.584, 545.584, 350.588, 223.15)

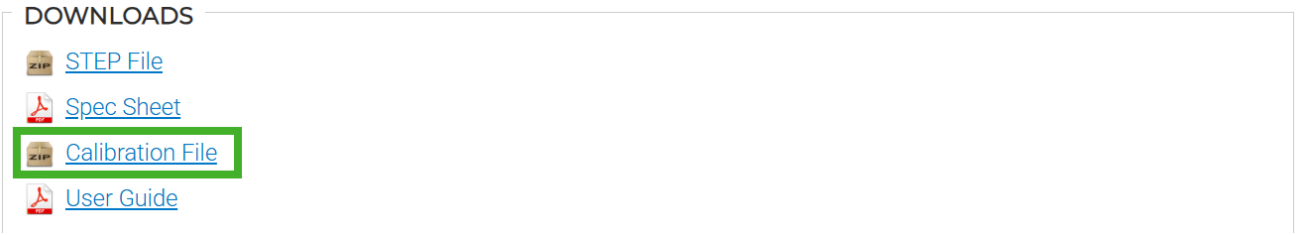
```
124
125         // Create the AprilTag processor.
126         aprilTag = new AprilTagProcessor.Builder()
127
128             // The following default settings are available to un-comment and edit as needed.
129             //.setDrawAxes(false)
130             //.setDrawCubeProjection(false)
131             //.setDrawTagOutline(true)
132             //.setTagFamily(AprilTagProcessor.TagFamily.TAG_36h11)
133             //.setTagLibrary(AprilTagGameDatabase.getCenterStageTagLibrary())
134             //.setOutputUnits(DistanceUnit.INCH, AngleUnit.DEGREES)
135
136             // == CAMERA CALIBRATION ==
137             // If you do not manually specify calibration parameters, the SDK will attempt
138             // to load a predefined calibration for your camera.
139             .setLensIntrinsics(481.985, 481.985, 334.203, 241.948)
140             // ... these parameters are fx, fy, cx, cy.
141
142             .build();
143
```

Step 4: Set Hardware Configuration
Refer to steps 5 and 6 of the Blocks installation guide for a walkthrough of how to add the webcam to your robot's configuration file.

FTC Installation Guide - Android Studio:

The Android Studio installation is very different from either Blocks or Onbot Java. You can still set your camera calibration in your Java code by calling .setLensIntrinsics(). But you can also just upload a webcam config file which will capture our defaults at a number of different resolutions.

Step 1: Download the Calibration File
File is available here: https://www.gobilda.com/global-shutter-usb-camera-with-gobilda-case-100-fps/



Step 2: Upload File to TeamWebCamCalibrations folder
Once you have downloaded the calibration file, unzip it and copy the teamwebcamcalibrations file to your clipboard.Android Studio, navigate to your TeamCode folder, then to res > xml and you'll find a file called "teamwebcamcalibrations.xml. Click this file and click paste. It will ask if you would like to overwrite the existing file, click overwrite. Once this is complete, any code you build using this webcam will automatically find the intrinsics we have calibrated for it.