# DAILY ASSESSMENT

|  |  |  |  |
| --- | --- | --- | --- |
| Date: | 14/07/2020 | Name: | Chesmi B R |
| Course: | **Computer vision basics** | USN: | 4AL16EC100 |
| Topic: | **Week 2:light, colour and image formation in matlab** | Semester & Section: | 8TH SEM & A Section |
| Github Repository: | chesmibr |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |

# Color Detection

### Introduction

A color detection algorithm identifies pixels in an image that match a specified color or color range. The color of detected pixels can then be changed to distinguish them from the rest of the image.

This example introduces a workflow to take Simulink model of a color detection algorithm from simulation to deployment onto device:

* Simulate the color detection model
* Deploy the color detection model and use the **Video Display** block to show simulated data on the Android device
* Modify the color detection model to use real-time images from the **Camera** Block

### Prerequisites

We recommend completing [Getting Started with Android™ Devices](https://in.mathworks.com/help/supportpkg/android/ref/getting-started-with-android-devices.html) example.

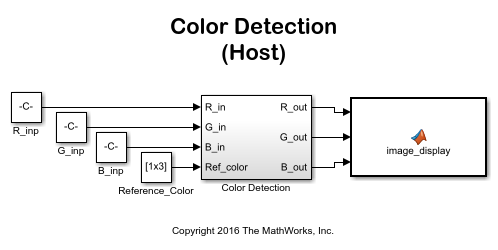
### Required Hardware

* Android device, configured using [androidhwsetup](matlab:androidhwsetup" \t "_blank)
* USB cable to connect the device to host computer

### Task 1 - Simulate Model on Host Computer

In this task, you will simulate color detection model and observe results for the images from workspace.

**1.** Open the [Color Detection on Images](matlab:open_system('androidColorDetectionImage.slx')" \t "_blank) Model



**2.** Load and display a sample color image into the MATLAB® workspace by executing the following command in the MATLAB command window:

RGB\_image = imread('peppers.png');

imshow(RGB\_image)

**3.** In the model, double-click on the **Reference Color** block to open the block mask.

**4.** In the block mask, set the **Constant value** field to the vector [ 58, 68, 117 ] to detect the color purple in the image. The vector value represents the red, green, and blue components of the color for detection (e.g. [R, G, B] ).

**5.** In the model, click the Run button on the toolbar to run the simulation. When the simulation completes, you can see the resulting image where the originally purple regions are now white.

**6.** (Optionally) You can modify the value of the **Reference Color** block and run the simulation again to isolate different color regions from the image.

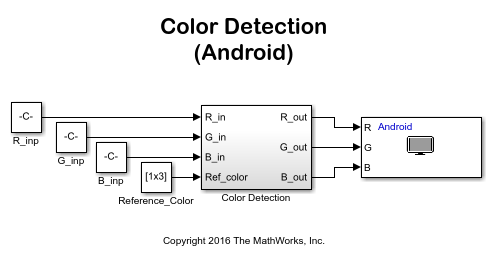
### Task 2 - Run Model on Device

In this task, you will create a new model to use the **Video Display** block and deploy the model algorithm to the device

**1.** Open a new Simulink model and copy the contents of the first model into the new model.

**2.** In the new model, remove the MATLAB Function image\_display block.

**3.** From the Simulink Library Browser, add the Android **Video Display** block to the model. A [pre-configured model](matlab:androidcolordetection) is included for your convenience.



**4.** Connect your configured Android device to your host computer. If you have not previously configured your device, run [androidhwsetup](matlab:androidhwsetup" \t "_blank) from the MATLAB command prompt to configure your Android device first.

**5.** In your Simulink model, click **Simulation > Model Configuration Parameters** to open **Configuration Parameters** dialog.

**6.** Select the **Hardware Implementation** pane and set the **Hardware board** to the **Android Device** you are using.

**7.** In your Simulink model, click the **Deploy to Hardware** button on the toolbar.

**8.** The image with the masked colors will appear on your device screen

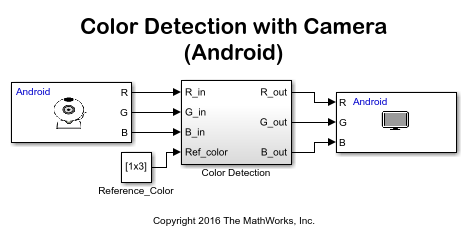
### Task 3 - Run Model on Device with Real-time Video

In this task, you will update the previous model to use frames captured from your Android device camera allowing for color detection in real-time images.

**1.** Using the model created in Task 2, remove the Constant blocks labelled R\_inp, G\_inp, and B\_inp.

**2.** From the Simulink Library Browser, add the Android **Camera** block to the model.

**3.** Connect the R, G, and B ports of the Camera block to the R\_in, G\_in, and B\_in ports of the Color Detection block, respectively. A [pre-configured model](matlab:androidColorDetectionCamera) is included for your convenience.



**4.** In your Simulink model, click the **Deploy to Hardware** button on the toolbar.

**5.** The camera images with the masked colors appear on your device screen

**DAILY ASSESSMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **14/07/2020** | **Name:** | **Chesmi B R** |
| **Course:** | **Trailhead salesforce developer** | **USN:** | **4AL16EC100** |
| **Topic:** | **Platform development basics** | **Semester & Section:** | **8TH SEM & A Section** |
| **Github Repository:** | **chesmibr** |  |  |

|  |
| --- |
| **AFTERNOON SESSION DETAILS** |
|  |

|  |
| --- |
| **Report**-  At Salesforce, we group our services by clouds. There’s Sales Cloud for CRM, Service Cloud for customer support, and a handful of other clouds that help companies support their business functions. And while each of these clouds serves a unique purpose, there’s one thing they all have in common: the power of the Salesforce platform.  What is the Salesforce platform, exactly?  Like any platform, the Salesforce platform is a group of technologies that supports the development of other technologies on top of it. What makes it unique is that the platform supports not only all the Salesforce clouds, but it also supports custom functionality built by our customers and partners. This functionality ranges from simple page layouts to full-scale applications.  If you’re here today, we’re assuming you know a bit about software development. Throughout this module, we’re going to give you an overview of development on the Salesforce platform. We talk about some of the pillars of Salesforce development and how they work together to create a robust system. We even touch on some common questions that developers new to the platform run into as they get started.  Before we continue, let’s make sure we’re on the same page. If you’re brand new to Salesforce and you haven’t completed the [Salesforce Platform Basics module](https://trailhead.salesforce.com/modules/starting_force_com), we suggest you do that before you keep reading.  Once you’re done with that, you’re ready to get started! Platform Building Blocks As we mentioned, the platform not only forms the foundation of core Salesforce products like Sales Cloud and Service Cloud, but it also lets you build your own functionality. Building your own functionality can mean customizing existing Salesforce offerings or it can mean building something from scratch.  Let’s focus on that latter part and talk about what the Salesforce platform offers developers.  Our core platform lets you develop custom data models and applications for desktop and mobile. And with the platform behind your development, you can build robust systems at a rapid pace.  And then there’s the Heroku platform. Heroku gives developers the power to build highly scalable web apps and back-end services using Python, Ruby, Go, and more. It also provides database tools to sync seamlessly with data from Salesforce.  And then there’s the host of Salesforce APIs. These let developers integrate and connect all their enterprise data, networks, and identity information.  And then there’s the Mobile SDK. The Mobile SDK is a suite of technologies that lets you build native, HTML5, and hybrid apps that have the same reliability and security as the Salesforce app.  And then... wait. Let’s stop for a second.  The problem with the platform and all its parts is that listing them out takes a really long time. And just talking about them doesn’t help you understand everything they do. Let’s take a different approach and talk about what we can do with the platform. Or, more precisely, what we can build with it.  The DreamHouse App  Let’s float a scenario. Throughout the rest of this module, we use this scenario to explore the many exciting tools and technologies that the Salesforce platform provides.  You’re a developer for DreamHouse Realty, a company that aggregates real estate listings to better connect homebuyers and real estate agents. Your boss asks you to build a new system to track real estate listings. Your internal employees will use it to track and communicate about properties. Your partner real estate brokers will use it to access information about customers. And your customers will view properties and contact brokers for viewings.  Building an app like this one from scratch isn’t an easy thing to do. Taking on this project in real life can involve a long, complicated list of functional requirements and the implementation of special integrations for your company’s business data. Working by yourself, it can take you months to get something out the door.  But before your stress builds and you melt into a puddle of existential dread, remember: You’ve got the platform. And building complex business applications at a breakneck pace is what the platform’s all about.  We’re going to show you a fully functional version of the DreamHouse app so you can get a feel for how it was built. As we move through, we discuss important Salesforce development concepts using the app to guide us. Install the DreamHouse App To follow along and practice the steps in this module, you need to install the DreamHouse package in your Trailhead Playground. Follow the instructions here to launch a playground and install the package. You also use this package and playground when it’s time to complete the hands-on challenge.  Launch your Trailhead Playground by scrolling to the bottom of this page and clicking **Launch**. If you see a tab in your org labeled Install a Package, great. Follow the steps below.  If not, from the App Launcher (App Launcher icon), find and select **Playground Starter** and follow the steps. If you don’t see the Playground Starter app, copy [this package installation link](https://login.salesforce.com/packaging/installPackage.apexp?p0=04tB00000009UeX) and check out [Install a Package or App to Complete a Trailhead Challenge](https://trailhead.salesforce.com/help?article=Installing-a-package-or-app-to-complete-a-Trailhead-challenge) on Trailhead Help.   1. Click the Install a Package tab. 2. Paste 04tB00000009UeX into the field. 3. Click **Install**. 4. Select **Install for All Users**, then click **Install**. |