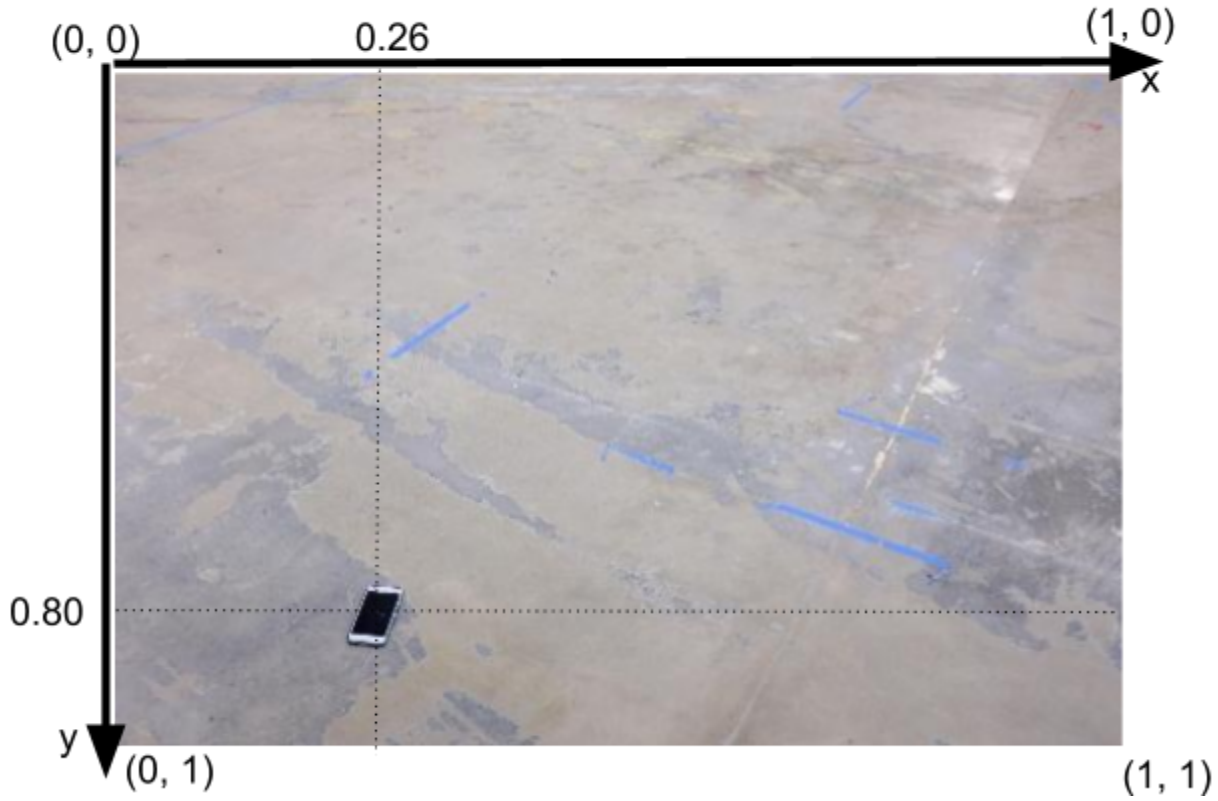
Task description:

You've been tasked to implement a prototype of a visual object detection system for a customer. The task is to find a location of a phone dropped on the floor from a single RGB camera image. The customer has only one type of phone he is interested in detecting. Here is an example of a image with a phone on it:



Consider a normalized XY-coordinate system for an image. Left-top corner of the image is defined as (x, y) = (0, 0), left-bottom as (x, y) = (0, 1), right-top as (x, y) = (1, 0), and finally right-bottom corner as (x, y) = (1, 1). Your "phone detector" has to find normalized coordinates of the center of the phone. In the example above, the coordinates of the phone are approximately (x, y) = (0.26, 0.80). Every image contains a phone.

The customer has prepared a small labeled dataset for you. A dataset consists of approximately 100 jpeg images of the floor from the factory building with a phone on it. There is a file named `labels.txt` that contains normalized coordinates of a phone for each picture. Each line of the `labels.txt` is composed of `img_path`, `x`, `y` separated by spaces:

`img_path`, `x` (coordinate of the phone), `y` (coordinate of the phone)

Here is an example of the first 3 lines from labels.txt:

```
51.jpg 0.2388 0.6012
95.jpg 0.2551 0.3129
84.jpg 0.7122 0.7117
```

The images and `labels.txt` are in the '`find_phone`' folder in the archive attached to this description.

Submission form and evaluation criteria:

The customer held out 8 images with similar statistics to the training set and is going to test your phone detector on them. The customer has an automated script for testing your phone detector. Please submit 2 executable python3 scripts:

1) *train_phone_finder.py* takes a single command line argument which is a path to a folder with labeled images and `labels.txt` that has been attached to this description. This script may generate any artifacts you want in the current folder. Here is what a terminal command will look like:
   ```
   > python train_phone_finder.py ~/find_phone
   ```
2) *find_phone.py* takes a single command line argument which is a path to the jpeg image to be tested. This script may use data in the local folder previously generated by *train_phone_finder.py*. This script has to print the normalized coordinates of the phone detected on this test image in the format shown below. Here is what a terminal command will look like. Please, notice space separated float numbers on a single line without parentheses (!):
   ```
   > python find_phone.py ~/find_phone_test_images/51.jpg
   0.2551 0.3129
   ```

You may submit any other data or library code in the same folder with these two scripts.

We understand that the labeled dataset is quite small and that certain machine learning methods will not work reliably with such a small dataset. Therefore, you are allowed to use absolutely any(!) method for detecting phone positions that you feel is suited to the problem. You are allowed to use any web resources or other materials like publicly available code samples. For example, you can use any state-of-art method that you are aware about. You may use any python3 libraries provided they can be installed via *pip* package manager (including *opencv)*.

A phone is considered to be detected correctly on a test image if your output is within a radius of `0.05` (normalized distance) centered on the phone. Perfect detection performance is not the main goal of this test. For this prototype, your algorithm is expected to detect a phone correctly on 4 out of the 8 test images and to detect at least 70% correctly on the provided labeled dataset. If you do not have enough time, please focus on a submission with clean, well structured code, rather than on the perfect performance.

Additionally, you are welcome to attach your notes regarding possible next steps for your detector and ways to improve the data collection of the customer.