NanoScout

by Michael Sidorov

October 31, 2021

Contents

Chapter 1

Introduction

1.1 Projects' Goal

The main goal of this project is to facilitate quality assurance of samples in a nano-scale, i.e., to design an easy, cheap and reliable first step detection mechanism. Our system is design to answer a simple question - does the sample contain the target (e.g., virus, microb, chemical etc.) or not, the particulars are omitted, and thus the time saved! As in many cases the particulars are not really that important (e.g., we don't care what kind of contamination is there in the food product, if it has any - we should through it), so the very fact of particulars omitment is our main strength, as it saves time!

Chapter 2

Graphical User Interface (GUI)

The GUI is divided into four sections, which correspond to different aspects of the system, viz. physical, image, smart search and image acquisition configurations, as shown in fig. 2.1. **Physical Configurations** include controls, which are being changed through mechanical manipulations. All the configurations regarding the image (e.g., resolution, image quality etc.) are included in the **Image Configurations** section. The **Smart Search** section includes configurations which relate to the different methods and algorythms which are being employed at the core of the system. Finally, the **Image Acquisition Settings** relate to the specific way we want to receive the images which are being produced (e.g., where to save them, do we want to send them to the cloud etc.)

2.1 Physical Settings

- **Sensor** - controls the sensor which we desire to use, viz. Scanning Electron Microscope (i.e., SEM) or the optical microscope (i.e., NavCam).
- **Detector** - is an option (for SEM only) which switches between Secondary Electron Detector (i.e., SED) and BSD with all its options available in the corresponding drop-down list
- **High Tension and Spot Size** - these are parameters which control the resolution of the acquired image. I.e., High Tension controls the number of electrons fired and Spot Size controls their dispersion, so to achieve the best resolution we should increase the High Tension and decrease the Spot Size
- **Vacuum** - may be 1, 10 and 60 [Pascal], corresponding to High, Medium and Low vacuum in the sample chamber, where for the use of SED the high vacuum (i.e., 1 [Pascal]) should be obtained first, so it's necessary to introduce an artificial delay when moving from NavCam to SEM.

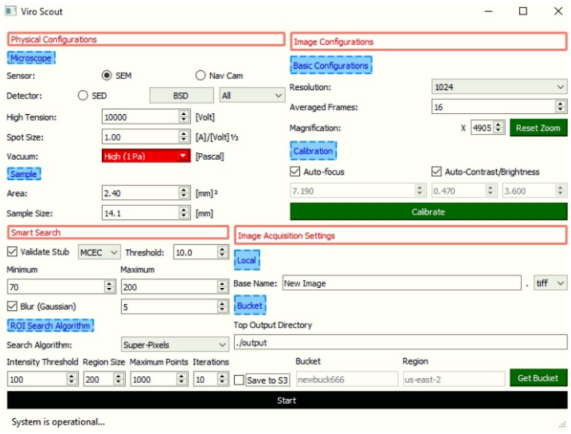

Figure 2.1: New GUI of the application

- **Area** - controls the area of the sample which is to be searched
- **Sample Size** - controls the sample size which is a factor which influences the movement of the beam when translating pixels into [meters], which represent locations on the sample. This setting is inherent to each sample and does not change in the course of the image acquisition process

2.2 Image Configurations

- **Resolution** - which is symmetrical for width and height, e.g., when you choose 1024, the acquired image would be of resolution of 1MP, or 1024X1024
- **Average Frames** - this option controls the number of frames which are taken in a burst and averaged each time an image is acquired (i.e., superresolution). Ranges between 1 to 16, and should enhance the image quality by eliminating artifacts which may appear in a single shot of the image. A clear tradeoff here is on the scale of speed - quality, i.e., the fewer frames we'd average each image acquisition the faster this acquisition will be, but the acquired image may suffer from a degraded image quality
- **Magnification** - this setting controls the working magnification of the system, and should be set to suit the size of the particle which we are looking for, as in the process of the "Smart Search" the system will zoom-in the ROI locations zooming in it by this value
- **Focus/Contrast/Brightness** - may be controlled manually, but pressing the "Calibrate" button instantly adjusts these parameters to the optimal values

2.3 Smart Search

- **Stub Validation** - you can choose not to perform this step to save the time in time-constrained situations
- **ROI Search Algorithm** - many algorithms may be used for this purpose, so this part is interchangeable, and the addition of new methods is made very easy, which constitutes around 4 changes in the code each time

2.4 Image Acquisition Settings

- **Base Name** - as images will be shot in a burst setting, they are appended a running index, so the base name of each image (which will be shared between them) may be provided here
- **File Type** - may be set to either tiff, jpg or bmp
- **Bucket** - the images can be automatically placed in the s3 bucket by connecting to it

Chapter 3

Algo

3.1 Overview

The **Algo** constitutes the core of the system, i.e., this is where all the search techniques are being employed, detections are being performed and decisions are being made. **Algo** bassically consists of two parts, viz. the **SmartSearch©** and the **Detector**

3.2 SmartSearch©

3.2.1 Stub Validation

The purpose of this part is to gauge the quality of the stub, i.e., the potential information which may be extracted from it. If this measure is too low, it may indicate two things, viz. that the resolution of the microscope is suboptimal, or that the sample is flawed.

We use the Mean Canny Edge Criterion (i.e., MCEC) to infere the amount of the information in the initial stub image. Thw procedure is very simple and is based on the assumption that usible information in the stub image should translate to morphological dissimilarities in the image, which in their turn may be represented as edges (i.e., high-gradient conturs) in the image. These contures are easily detected via well known techniques, such as laplacian or Canny edge detection. After marking the conturs found in the image with 1., while zeroing all the other pixels, we sum the pixels in one of the directions (i.e., height or width), and compute the mean and the standard deviation of the intensity sum. As may be seen in 3.2 there is a clear separation between flawed and valida stub images. Sample acquisition is a slow and error prone process, and as such its repetition should be avoided, so first we'd try to improve the image quality by adjusting the resolution, which is controlled by the

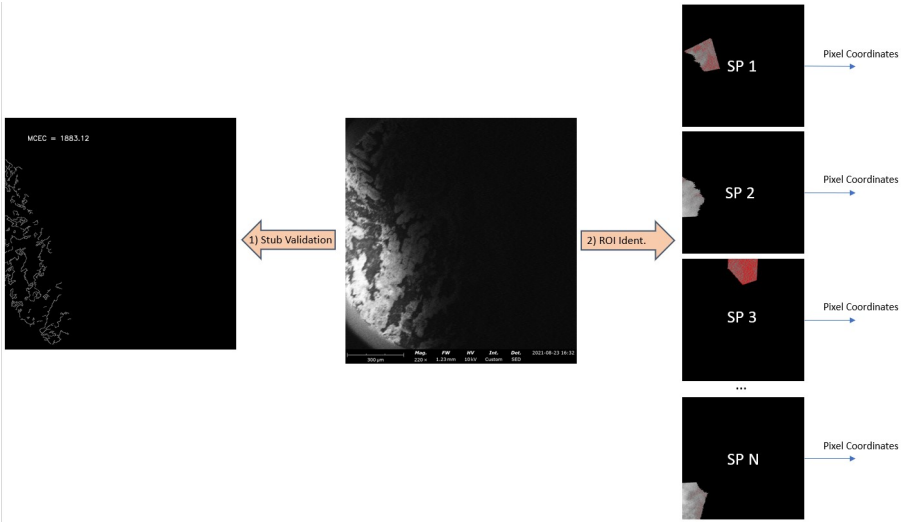

Figure 3.1: Smart search schematic flow

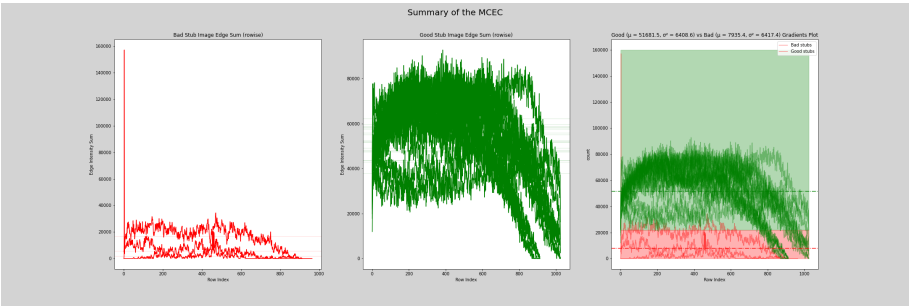

Figure 3.2: The experiments of the MCEC on 5 "good" and 20 "bad" stub images. As we see there is a clear decision boundery which may facilitate separating the stub images.

high-tension and the spot-size parameters. If after the adjustments to the resolution the validation still fails - we should instruct the user to replace the sample.

3.2.2 Region of Interest (ROI) Identification

Here we employ a 2-step technique to facilitate the search of the pathogen. As may be seen in fig. 3.1, after we validate the stub, we divide the stub image into super-pixels, and in each one of them we look for dark regions (based on the "black-hole theory" of Charls), which are controlled by a threshold parameter. After we have received the pixel locations of the darker regions, i.e., "black-holes", we zoom in this region, focus on the image and acquire it.

3.2.3 Auto-Focus

This is a crucial part of the whole system as the goal is the acquisition of the images of pathogens in a high magnification ($\sim 80 - 100k$). There are two built-in options viz. PyPhenom.SemAutoFocus(), PyPhenom.Aplication.FineFocus() and custom made option, which just skips over magnifications. The autofocus algorithm should be further explored.

3.3 Detector

The detector will be trained via an **Auto-Labeling** technique [1]. The technique described in [1] is comprised of two separate stages, viz. *feature extraction* and *label assignment*, where the first stage is meant to extract the most dominant features from the images, and the second to group simmilar images together.

As we teach the network in an *unsipervised* manner, we don't have labels to rely on to know when we are correct and when we are wrong, so we have to let the network figure out what features to search for, and which images should be grouped together. One obstacle which may present itself in this situation is network learning useless features (e.g., separate pixel values etc.). To avoid this obstacle, and make the model more robust, we train it on images together with their augmentations and try to minimize the regular loss of semantic learning techniques, τ, together with the distanse of the original image and its' multiple augmentations. The idea behind this technique is that the network should learn "strong" features, i.e., which doesn't change under simple augmentations.

Bibliography

[1] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans **and** Luc Van Gool. 2020. Scan: learning to classify images without labels. **in***European Conference on Computer Vision* Springer, 268–285.