

How to build your own Roboscope – Roy Wollman

Disclaimer – the Microscopy Toolbox is in its EARLY (!!!) stages of development, the text below explains how our microscopy system is / will be set up and what the motivation behind it was. For more updated details see:

<http://www.mcb.ucdavis.edu/faculty-labs/scholey/Roy/Roboscope.html>

1. Introduction (Propaganda)

What is a Roboscope, and why would you want one?

Biology is complex. This is rather unfortunate; it would make life much easier for all of us if cells were simple and composed of only a small number of molecules. However, this is not the case. To be able to understand complex cell biological phenomena we need to be able to look at an increasing number of perturbations and readouts of the systems. The days of "one gene, one PhD" are over. How can we deal with these complicated cells? There are many answers for this question at many different levels, some institutes have even set up new "systems biology" departments to try and answer this question. I would argue that one approach to try and find answers to these questions is encompassed in the buzzwords "scalable research" or even better "scalability".

It's a nice buzzword, but what does it really mean? Let's say that you just spent 2 months to set up your assay to test the effect of some experimental perturbation on your favorite cell type. If in order to test another perturbation you will need another 2 months then your work is not scalable. If however, after setting up the system, each additional perturbation takes only one more day, e.g. it is "cheap" in terms of the additional resources it takes; one can argue that it is a scalable research. Scalable research has already revolutionized modern biology in many aspects (think microarray and shotgun sequencing) however, some areas of biology are still far from scalability. Microscopy is arguably the most important tool in cell biology. At its current state, microscopy in general is non-scalable. There are exceptions to this rule. Few expansive 'high-throughput microscopes' exist out there. However, at large, microscopes still need someone (an underpaid graduate student, that is) to make them work.

Automated microscopy, or robotic microscopy, or even better *Roboscopes* are an attempt to make microscopy more scalable. Imagine that you are interested in knowing what effects cell shape; since there are many possible genes that may affect this cell's morphology, and let's for the sake of argument say that you got a small library of 500 dsRNA to test their effect on the cell's morphology. Now you have the task of looking at all 500 perturbations one by one using your lab's microscope. This could be a daunting task that could take a huge amount of time and might get very boring very fast. Now imagine that you have a Roboscope that can take the images for you, wouldn't that be nice? You'll still need to treat cells and fix them in some multi-well format but fixing 10 wells or 96 wells in a 96 well plate is not such a big difference. This is just an example of why automation can not only make life easier, but can create a qualitative change in the type of questions we can ask with ease.

2. Building a Roboscope – a personal experience

After spending too much time listening to my own propaganda (see section 1), I decided as part of my PhD work to set up an automated microscope system. With the support of my Advisor (Prof. Jonathan Scholey) we were able to raise funds for an automated high throughput microscope, (codename *Throopi the Roboscope*) and together with the help of Michael Paddy from the imaging facility in the Molecular and Cellular Biology section we got *Throopi* to work. The original idea was to write all the drivers and acquisition control from scratch, luckily I crossed paths with Nenad and Nico and the μ Manager project. We're proud to be the first working system outside of the Vale lab and the first system to use μ Manager only as a programming interface. Besides saving a lot of programming hours, the use of μ Manager for acquisition control makes Roboscope a more abstract and generally applicable concept. It is now possible to set up a new Roboscope based on different microscope vendors, change filter wheels etc. as long as they will be supported by μ Manager. This encouraged me to set up the software side of things in a more general fashion and is what led to the birth of the Microscopy Toolbox for Matlab. The microscopy toolbox is a set of functions that allows very easy integration of μ Manager within the Matlab environment and will allow its users to focus on what is really important, making roboscopes more 'intelligent' and unleash the power of automated scalable microscopy.

3. Roboscope design overview

What are the critical components of a Roboscope? To answer this question an interesting exercise is to think of what you, a human microscope operator, do when you work with a microscope. Basically, you make decisions. You decide what to image. you basically make two types of decisions:

1. Where to image in the XY plane, e.g. what objects (cells) are of interest
2. Where to image in the Z axis, e.g. make sure the image is in focus.

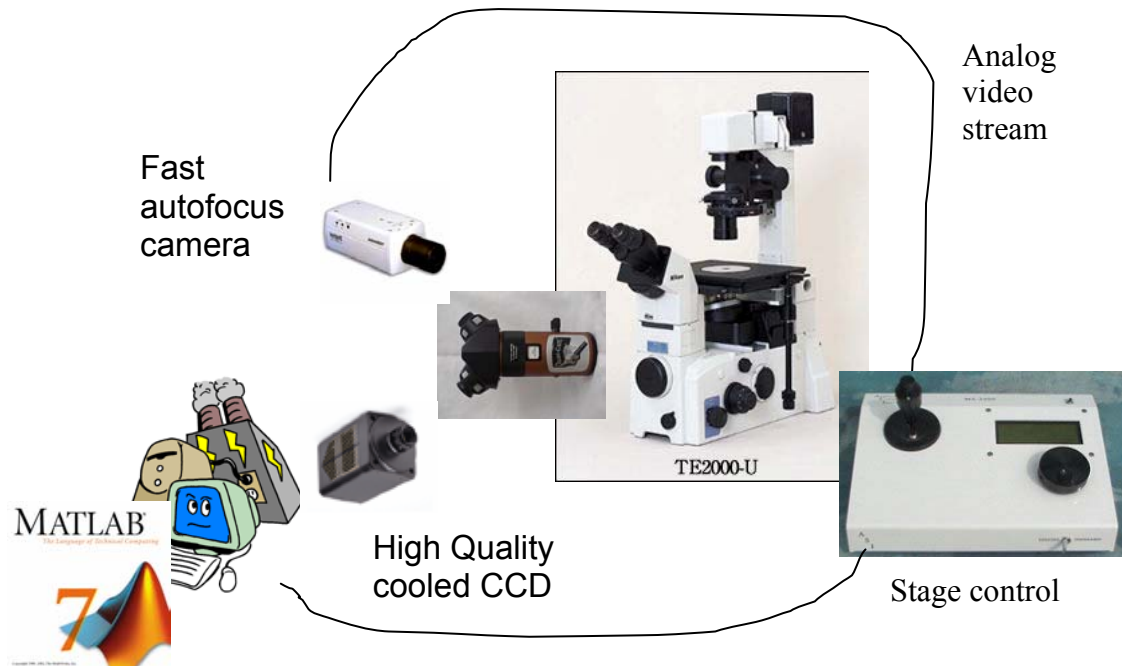
If you are going to build your own Roboscope, it basically means that you are planning to let a computer make these two decisions for you. XY decisions are easier. In many applications, you can simply image blindly 100 random sites on the coverslip and you should be ok. Of course this is not the idea behind a Roboscope. You would like to see your Roboscope making smart decisions on what cells are more interesting or 'beautiful' and are worth imaging. Even imaging random sites blindly can be a big step toward "scalability" and random sites are very easy to implement. However, imaging randomly in the Z-axis will be disastrous. You will get a lot of blank images or at best extremely out-of-focus cells. Therefore the decision of where to image in the Z-axis is the most crucial one and must be solved in a satisfactory manner before any other automation can be considered.

In general there are two strategies for automatic focus systems. The first uses some external readout of the position of the coverslip, there are several commercial fancy microscopes that do this (IxMicro from molecular devices for example, and many others). Those systems use some additional optics to deduce from the reflection of the coverslip its position and you are good to go. A disadvantage of this is that the cells are not always at the bottom of the coverslip! The workaround is to shift the focus a predetermined number of microns from the bottom of the coverslip. This works pretty well for low NA

objectives (larger depth of field) but has some problems with higher end objectives or cells with varying width.

The alternative autofocus method is to determine where the cells are by actually looking at the image. This is achieved by using a scoring function in which a computer can estimate how 'sharp' objects are by conducting a Z-sweep and measuring the focus score at each point on the slide and then picking only the Z position that maximizes the focus score (this was a very long and hard to understand sentence...). This scoring function has an obvious disadvantage; it requires exposing every site you want to image multiple times to determine where the best focus is. All those extra exposures can take forever and even worse, bleach the sample. These two problems (time and bleaching) have multiple types of solutions. For bleaching one can either look at a dye in which bleaching is not a big problem (like DAPI) or image in phase / white light. For speed, one can either get a faster camera or alternatively have a dedicated piece of hardware to calculate the focus score which will speed things up. When designing *Throopi* we decided to have a dedicated hardware to take care of the focus score for us. I don't think it's absolutely necessary and probably a fast camera might do the trick as well, I never tried it.

What is a dedicated hardware autofocus system? Basically it is a piece of hardware that gets a stream of images as an analog video signal and determines where the best focus is. All three big motorized stage manufacturer: ASI, Ludl and Prior will happily sell you an upgrade to their stage to which you can plug in an analog video, and it will return a focus score and determine the right Z-position (another long one...). The advantage is clear; it's fast and relatively painless to autofocus. However, it requires an analog video camera and here is where the next gadget kicks in. "Optical Insights" sell this gizmo called dual-cam, they allow you to add a dichroic in the light path which splits the light at the wavelength of your choice in two cameras: A fast analog video camera and a higher quality slower CCD camera for the 'real' work. The scheme below should give you a rough idea of how it looks (see next page).



At the center is the inverted microscope. Following the light-path to the left, the next thing is the dual-cam from OI. It splits the light into the two cameras. The upper (right) camera is the analog camera that streams the analog video signal into the stage controller at the right. The down (left) camera is your regular CCD camera and the computer that controls it all.

4. Roboscope hardware specs

Here are the specs of our system, most of the reasoning behind the design was presented above. This is mostly the 'shopping' list of what we ended up getting.

a. Hardware

i. Microscope parts

1. Nikon inverted frame – Nikon TS2000S. Out of the three TS2000 series of inverted Nikon microscopes this is the simplest and cheapest one. The TS2000E is the fully loaded system and the TS2000U had a few extra ports that we didn't really need.
2. Objectives – All dry objectives, a 40x 0.95 NA will be the workhorse, we also got a 10x 0.3 N.A. and 20x 0.45 N.A. . All phase.
3. Cameras - the CCD camera, ImageMax we got from the imaging facility, it's a bit old but performs great. The analog video camera was basically sitting on a shelf in the microscope room and it seems that no one even cared it existed...
4. Filter sets – to maximize the potential output and make the system as versatile as we could we are using the quad-filter set (86012V2) from Chroma. Notice that this will require a custom

dichroic on the dual-cam since the Dapi range is at ~470 and not 430.

ii. Automation

1. Stage – we used ASI xyz stage (MS-2000 model) and autofocus system, it works great.
2. Filter wheel – Standard Lambda filter (10-2 model) wheels with the controller.
3. Autofocus – an upgrade (\$1500) to the ASI ms-2000 controller.

iii. Optical Insight dual Cam – this is what makes it work, we used a custom dichroic to split the light 50%-50% at 475 nm

b. Software

- i. µManager
- ii. Matlab
- iii. Microscopy toolbox (work in progress...)

5. Getting robust autofocus – making Roboscope reliable

As mentioned above getting good autofocus is the key issue. However, autofocus may fail, and sometimes the roboscope will decide that a dust particle somewhere is really interesting and not the actual cells. The advantage of using the Microscopy toolbox is to have a programmatic interface to the system. This allows few simple 'hacks' that we found to dramatically improve the robustness of the system. The most basic is to always compare the chosen Z positions with the previous site imaged. In cases that the autofocus is really off the target it will choose a Z-position that is drastic change from previous ones, a change that is much bigger than expected due to coverslip curvature. A small “if” statement can do an amazing job at improving the reliability of the overall auto-focus performance. A similar idea can also be implemented in time-lapse movies. There one can compare not only the Z position from a previous site but also at the same site from a previous time points.

6. Combining analysis with acquisition – making Roboscope "intelligent"

This is the 'holy grail' and the end goal for this project. We're not there yet... but stay tuned. Just to give a taste of why we want it, a few words on what we are planning to do. The goal is to maximize the number of concurrent time lapse movies that are taken at the same time. It will be implemented by using timer objects in Matlab. Matlab has an internal event queue, and you can add future and recurring functions\ calls to that queue via timer objects. The main program will constantly scan the coverslip and try to identify mitotic cells, once a mitotic cell is identified, the main program will create a new timer object that will remember the cell XY position and will go back to image it a given number of times. In between the imaging (at around 1 min intervals) the microscope will return to the main program that will continue to look for mitotic cells. An estimated time to take a single image is ~6 sec, including stage movement etc. We hope to be able to image simultaneously 10 dividing cells so overnight we'll get ~100 mitosis movies, now that's scalable ;-)