A Crash Course on SimpleCV

Katherine Scott

SightMachine

kat@sightmachine.com anthony@sightmachine.com

February 27, 2013

Overview

Get Started!

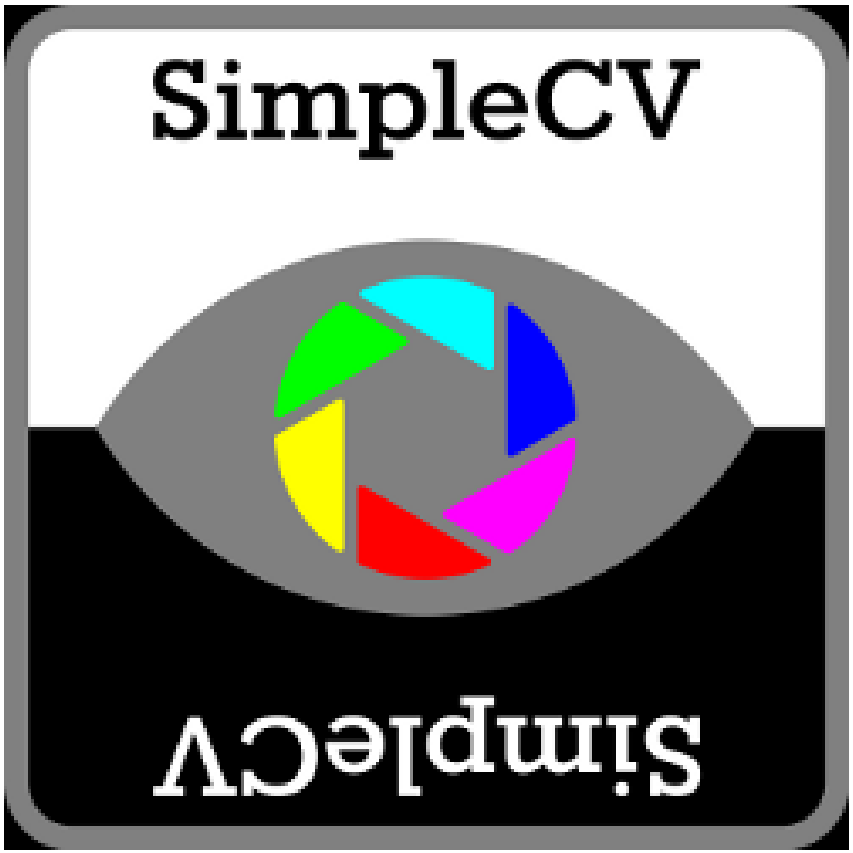
There are a lot of dependencies for SimpleCV and it is a bit tough for beginners. We've brought disks that are ready to go!

- Windows / Linux
  - Boot from USB drive.
  - Alternatively install VirtualBox and the image.
  - `https://www.virtualbox.org/`
- Macs
  - Newer macs are persnikety about booting from a USB drive.
  - Install virtual box and the ISO and go to town.
- When you get home install from SuperPack or preferably source libs.
  - take awhile and is not a perfect science.
  - `https://github.com/ingenuitas/SimpleCV`
  - If you want to contribute this is a great place to start.

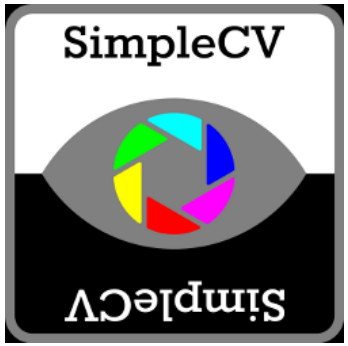
About the tutorial

- It will be a lot of live coding. I'll lead, you follow along.
- If you have a question feel free to interrupt.
- If you are having an issue raise a flag. Anthony will help you.

What Makes Up SimpleCV?



SimpleCV != OpenCV



- OpenCV is really busy, we help by wrapping python.
- We add lots of other fun stuff (OCR, Barcodes, etc.)
- We are not competing, we are complementing.
- Purposes are different. Python is great for prototyping. C++ great for embedded.

Core Dependencies

- OpenCV Python Bindings
- Numpy
- SciPy
- SciKits Learn and Orange
- PyGame (this is going away)
- Python Imaging Library (PIL)
- ipython
- PIL (Python Imaging Library)

Optional Dependencies

- Barcodes- Zebra Crossing ZXIng
- Optical Character Recognition (OCR) - Tesseract
- Beautiful Soup
- Kinect Support - freenect
- Unit Tests - nose
- Web Stuff - flask / CherryPy
- Arduino - pyfirmata
- Many Many Many more.

This is why we put everything in a superpack / virtual box / bootable drive

- Just get to the core library functions.
- We encourage you to install the full library when you get home.
- Help is available if you need it.

Getting Help after the tutorial.

- Primary Source: `http://help.simplecv.org/questions/`
- Documentation `http://www.simplecv.org/docs/`
- Tweet at us: @Simple_CV
- Another Good Resource: `http://www.reddit.com/r/ComputerVision`

On the Printed Page

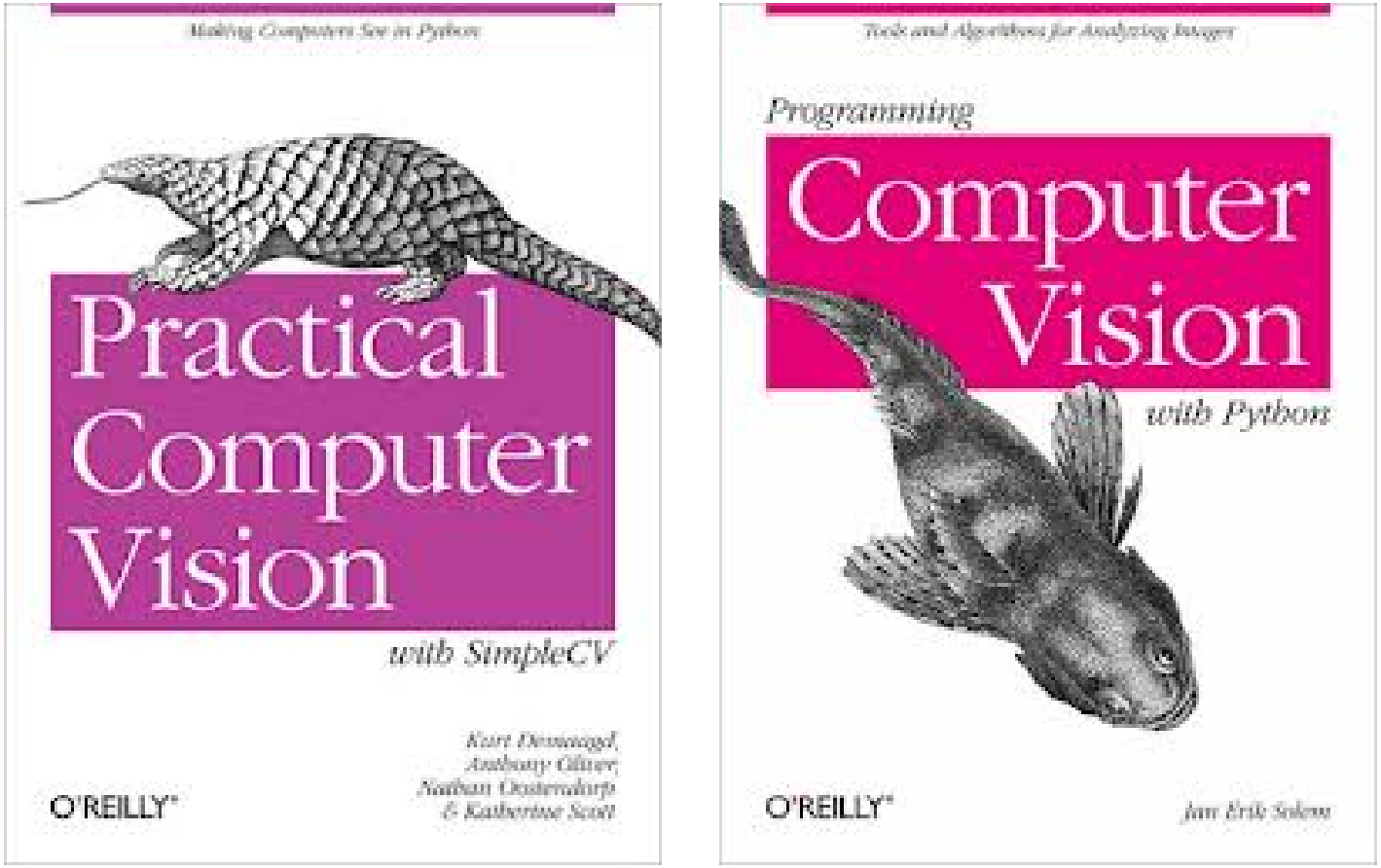


Figure: Two books about using Python for Computer Vision

So why are we doing this?

- We are really nice people who believe in Python and Open Source.
- We are trying to disrupt industrial quality control systems.

Early Prototypes

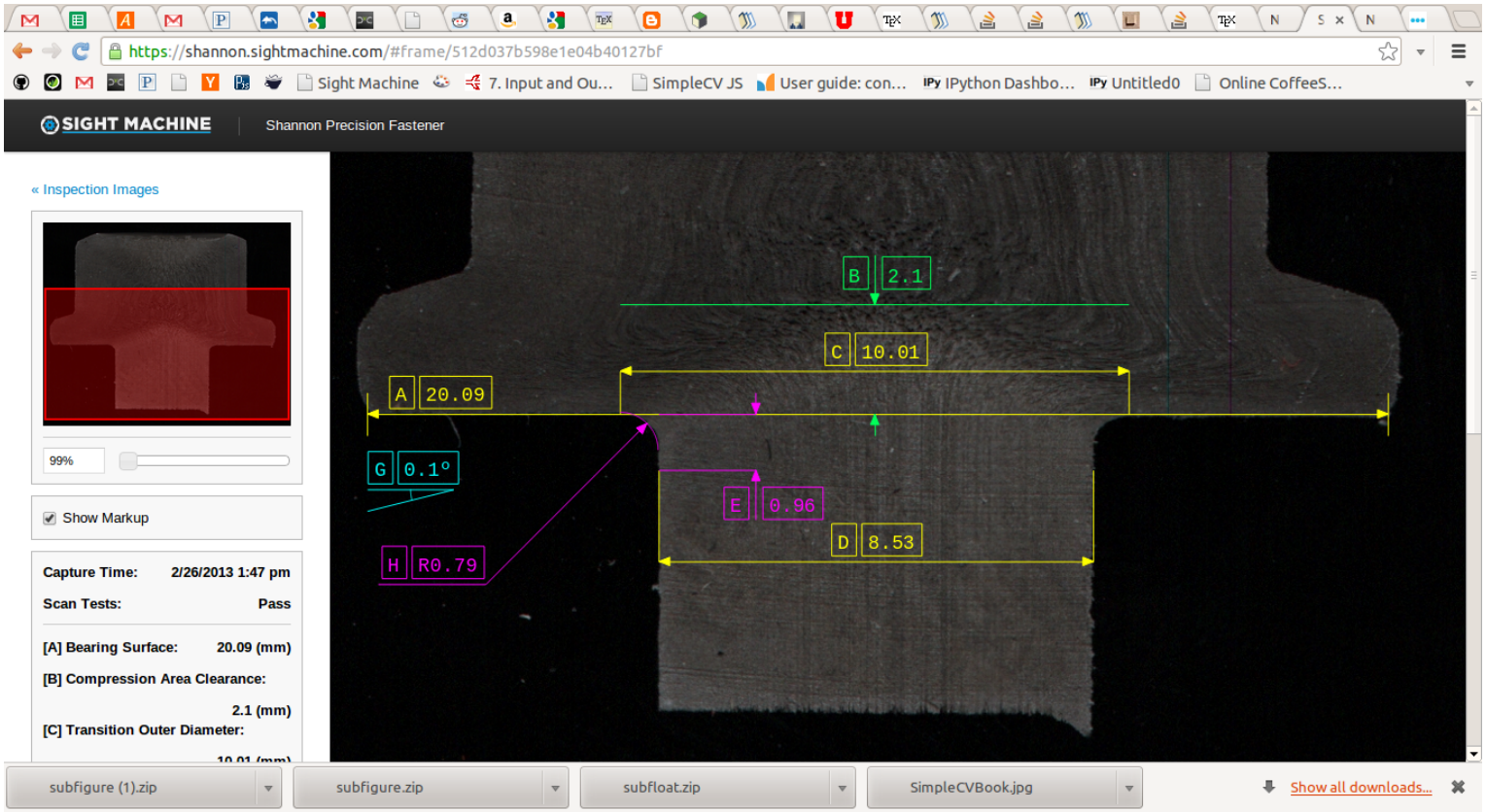


Figure: Early Customer - Industrial Fastener Morphology and Metallurgy

Early Prototypes

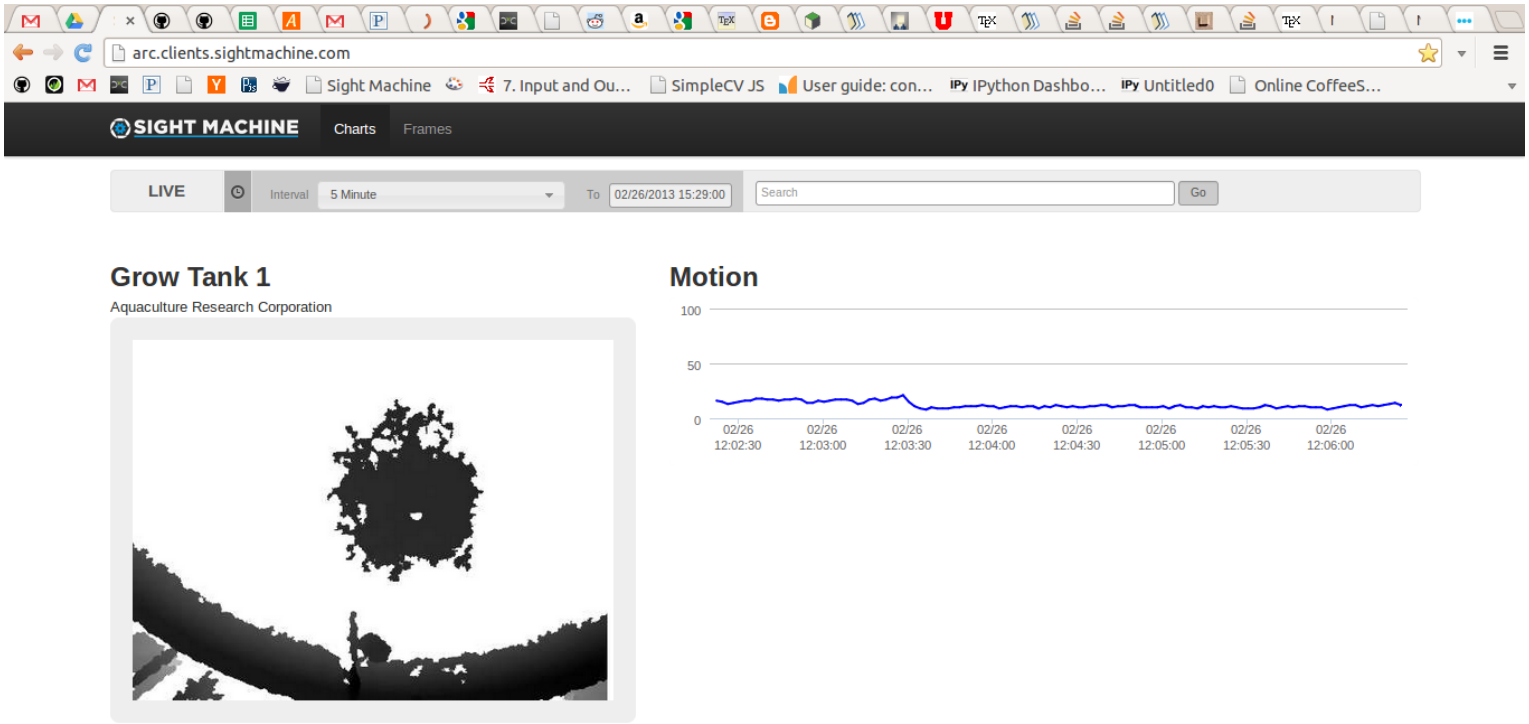
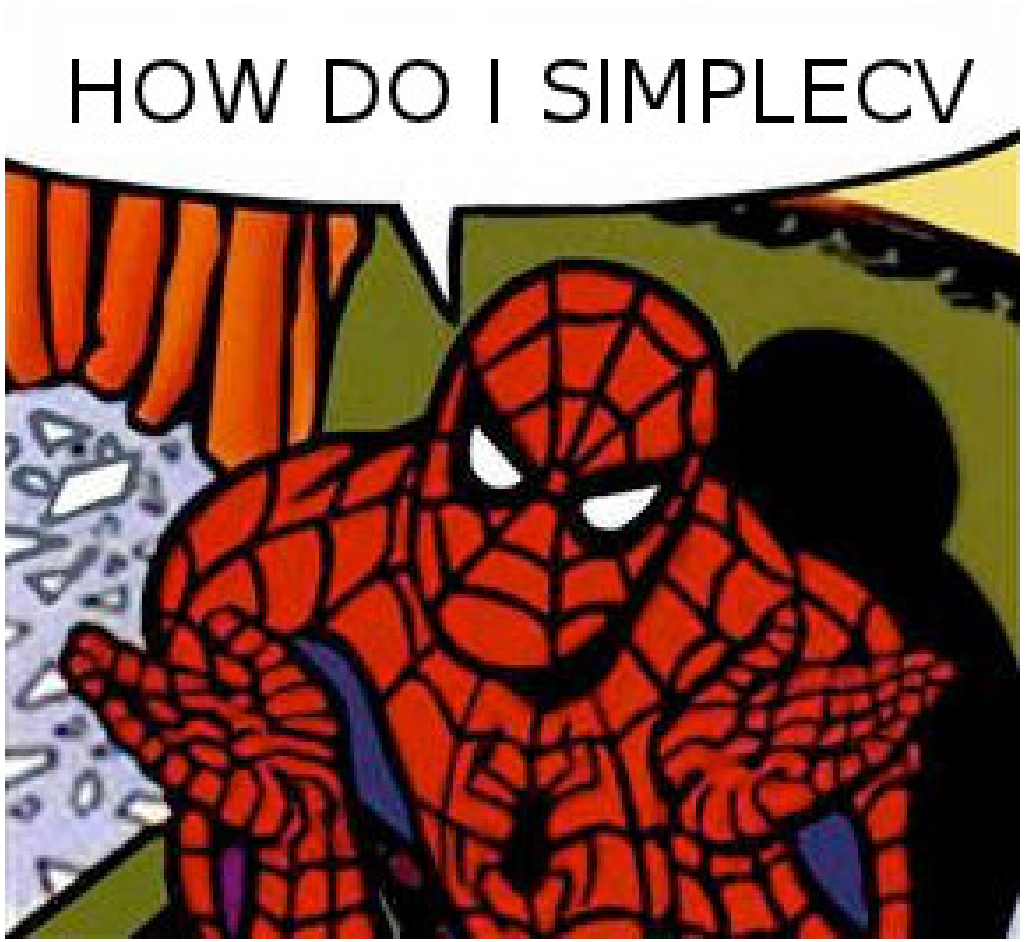


Figure: Early Customer - Aquaponics Research Facility

How do I SimpleCV?



Where do I write my code?

So how do I SimpleCV?

- In a python file, just like any other library.
- In a command line REPL like iPython.
- In the browser using iPython Notebooks (we'll use this today).

We really like iPython. It is kinda like using Matlab without the $ 5000 per seat license cost.

How does fit into a work flow?

At SightMachine we roughly use these three tools for different parts of our workflow.

Tool	**Uses**
iPython REPL	Prototypes, Sanity Checks, Etc
iPython Web Notebook	Testing and Development
Python Files	Deployment Code

Table: SimpleCV Workflow

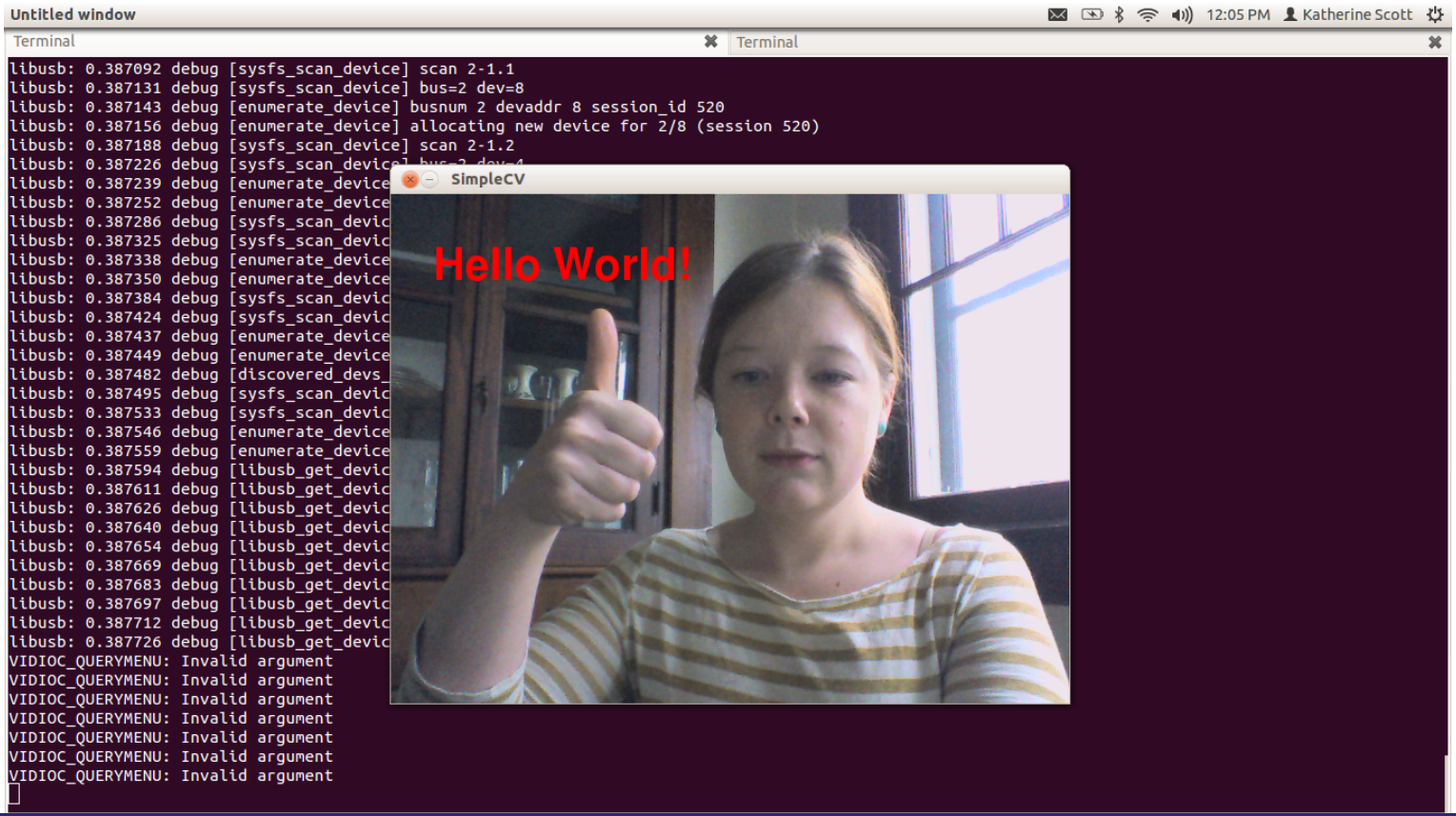
SimpleCV Hello World as a Script

Example (HelloWorld.py)

```
1  from SimpleCV import Image, Display, Color, Camera
2  cam = Camera(0) #Get the first camera
3  disp = Display((640,480)) # Create a 640x480 Display
4  while( disp.isNotDone() ):
5      img = cam.getImage() # get an image
6      # write text at 40,40 font_size 60pts, color is red
7      img.drawText("Hello World!",40,40,
8                   fontsize=60,color=Color.RED )
9      img.save(disp) # show it
10
```

How do I run Hello World?

- Run the py file with *python HelloWorld.py* in the command.
- Close it by pressing *esc* or *ctrl* − *c*



The SimpleCV Shell - Custom iPython REPL

Sometimes you just want to test an idea without writing a full script. For this reason we created the SimpleCV shell, which is a custom ipython instance. The SimpleCV shell will allow you to:

- Test your ideas in a REPL similar to Matlab.
- Access the SimpleCV documentation.
- Import modules that you are working with to test.
- Run through an interactive tutorial.

Starting the SimpleCV Shell

In OSX and Linux just type *simplecv* at the command line. On Windows you just click on the SimpleCV icon.

Example (Shell Basics)

```
+-------------------------------------------------------------+
 SimpleCV 1.3.0 [interactive shell] - http://simplecv.org
+-------------------------------------------------------------+
Commands:
"exit()" or press "Ctrl+ D" to exit the shell
"clear" to clear the shell screen
"tutorial" to begin the SimpleCV interactive tutorial
"example" gives a list of examples you can run
"forums" will launch a web browser for the help forums
"walkthrough" will launch a web browser with a walkthrough
Usage:
dot complete works to show library
for example: Image().save("/tmp/test.jpg") will dot complete
just by touching TAB after typing Image().
Documentation:
help(Image), ? Image, Image ? , or Image() ? all do the same
"docs" will launch webbrowser showing documentation
```

SimpleCV Shell Like a Boss

- Putting a ? in front of a class or method will give you documentation. The "/" key will let you search.
- iPython has tab completion for methods.
- Up arrow will give you previous commands.
- %paste will let you paste formatted code.
- Other cool stuff can be found by googling iPython magic

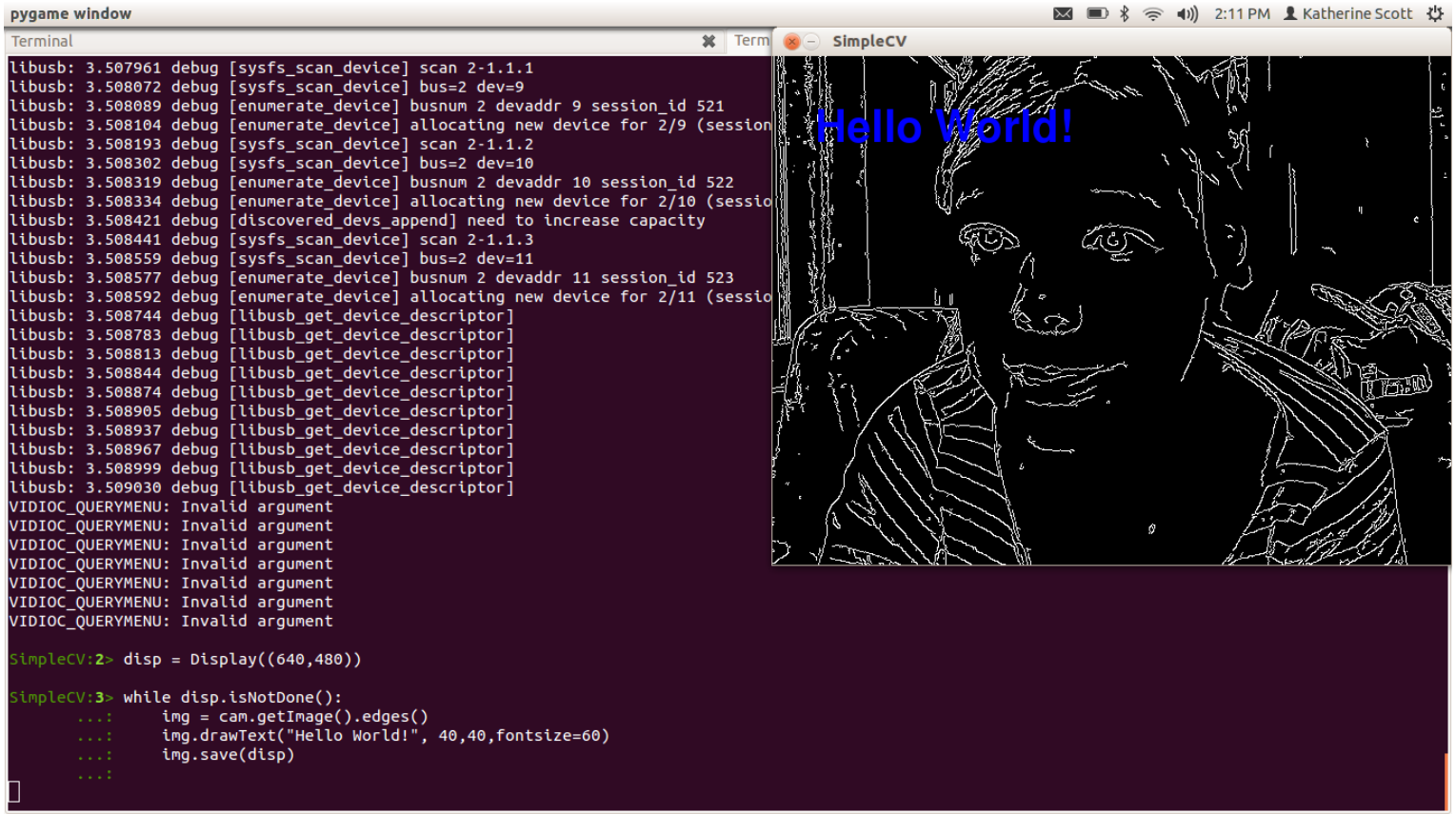
Let's repeat Hello World in SimpleCV Shell

Example (In the SimpleCV shell)

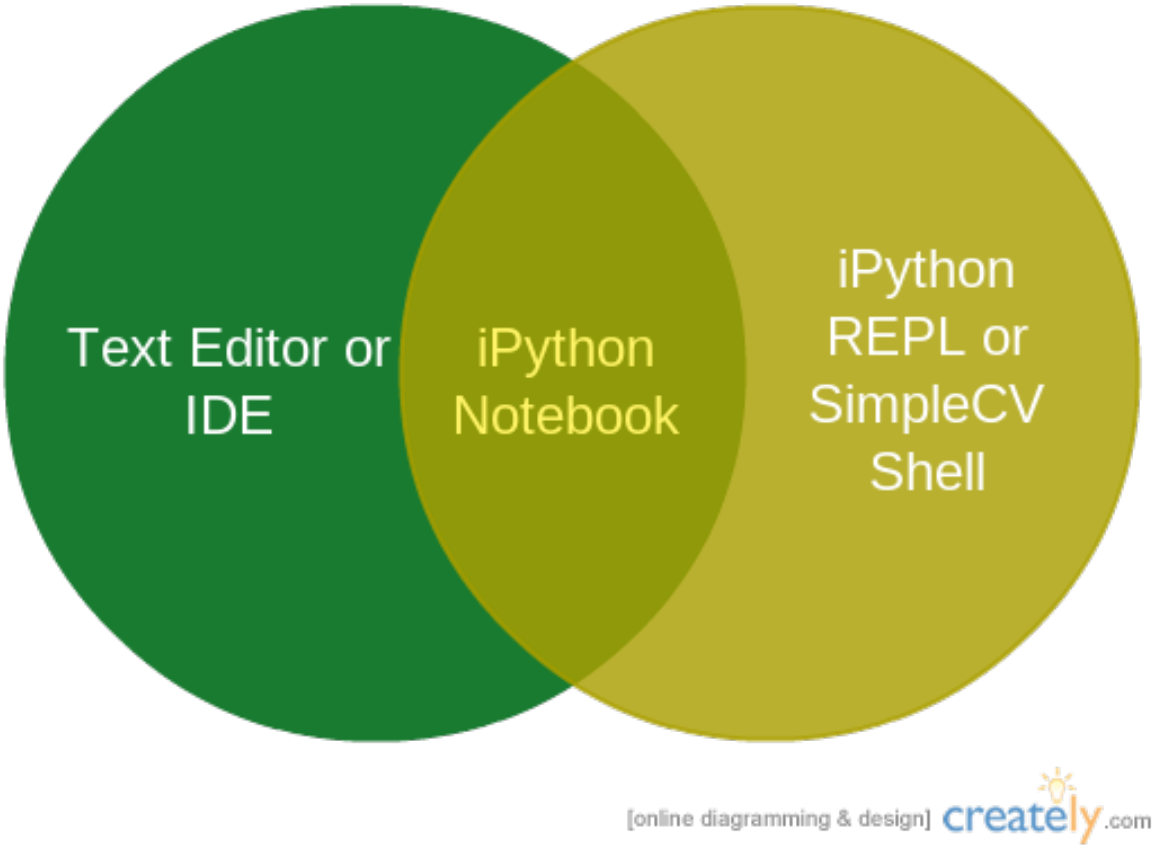
```
SimpleCV:1> cam = Camera()
SimpleCV:2> disp = Display((640,480))
SimpleCV:3> while disp.isNotDone():
       ...:     img = cam.getImage().edges()
       ...:     img.drawText("Hello World!",40,40,fontsize=60)
       ...:     img.save(disp)
       ...:
SimpleCV:4> exit
```

- Just push return after each line.
- iPython will do tabbing in the while loop.
- *esc* to quit or $ctrl - c$.
- type "exit" to quit.

Yes, it really is that simple.

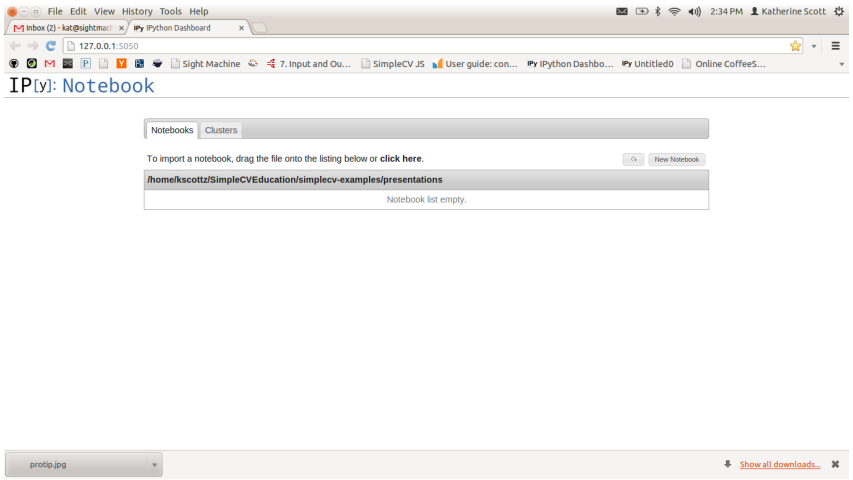


Why use iPython Web Notebooks



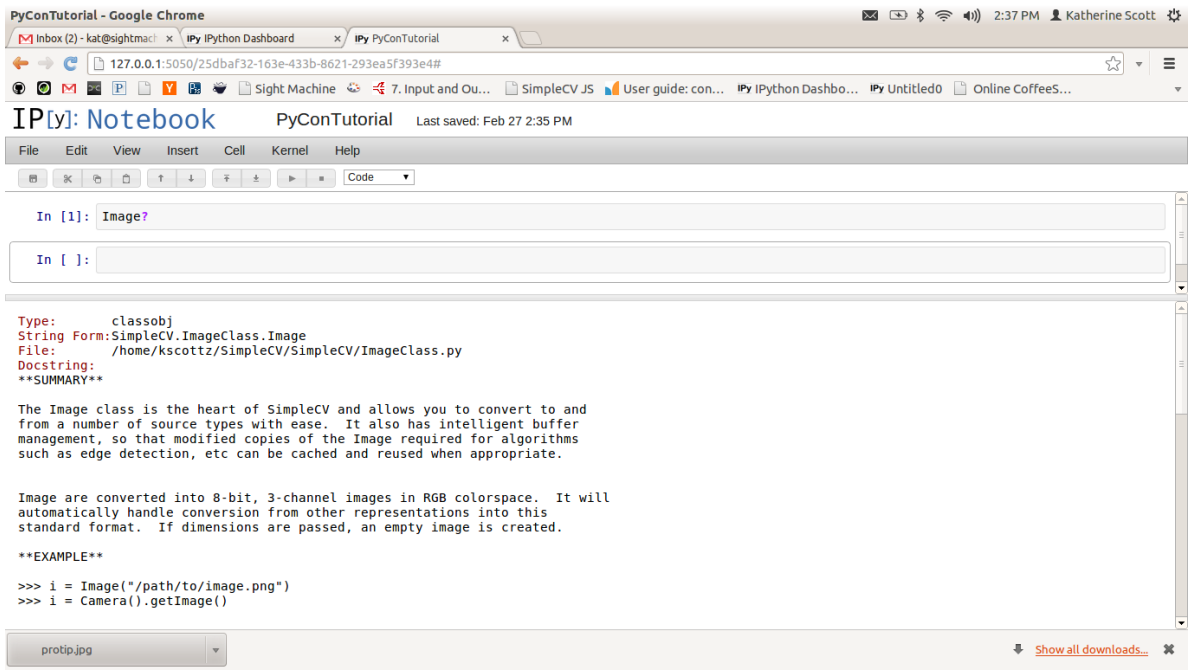
- Web notebooks give you the best features of an IDE and a REPL

How do I use the notebook?



- From the shell just type *simplecv notebook*.
- You will get to a dashboard to create a new notebook.
- By default notebooks are in the path where you start ipython.

How do I use the notebook?



- Everything we mentioned about the SimpleCV shell still holds.
- Magic commands, inline documentation, etc. still work.
- *enter* starts a new line.
- *ctrl* – *enter* executes a line.

Caveats about iPython Web Notebooks

- iPython Web Notebooks are still version 0.1.4
- **There is no auto-save. Get in the *ctrl* − *s* save habit.**
- If you edit a module you import you must restart the core.
- Minimal editing support. No find/replace.
- The core can sometimes crash on large images.
- The notebooks hold on to data by default. This can fill up your version control system fast. Try the download as python command from the gui.

Image Loading Basics II

- You can get the image file name using *img.filename*
- Images can also come from appropriately shaped numpy arrays.
- PIL and OpenCV images can also be passed into the image.
- Can also take a URL to an image.
- The img.getEXIFData() command can show jpg EXIF data.

Saving an Image

- The img.save() command is used to save images.
- You can save as just about any format, PNG, JPG, WebP, etc.
- Calling save with no parameters saves it to temp directory.
- Using the params flag you can set compression, e.g. set compression quality.

Using a USB Camera

- Most usb cameras use the Camera class.
- The camera class takes a camera index, *usually* this is the order cameras are plugged into the computer.
- The camera also has properties that you can get and set, or use a propmap dictionary to set.
- *Support for camera properties is vendor specific and spotty at best.*
- Cameras also have a *threaded* parameter. Set this to false to run multiple cameras.

- Camera.getImage() will return the current image.
- Camera.getAllProperties() will return the cameras properties.
- Camera.getProperty() and Camera.setProperty() may let you set properties. *This is highly vendor dependent and usually poorly documented.*

- Camera.getImage() will return the current image.
- Camera.getAllProperties() will return the cameras properties.
- Camera.getProperty() and Camera.setProperty() may let you set properties. *This is highly vendor dependent and usually poorly documented.*

- Camera.getImage() will return the current image.
- Camera.getAllProperties() will return the cameras properties.
- Camera.getProperty() and Camera.setProperty() may let you set properties. *This is highly vendor dependent and usually poorly documented.*

Briefly: Other Cameras

- Kinect - Depth Camera
  - Uses freenect drivers, not the OpneNI drivers.
  - Kinect.getImage and Kinect.getDepth
  - Note that these aren't well calibrated together.
- JpegStreamReader - IP Cameras
  - Give it a url to camera's web feed, and scrape images.
  - Getting the URL straight can be tricky.
- Virtual Camera
  - A virtual camera that pulls from a directory full of images, or video.
  - Interface to video files for processing.

Briefly: Other Cameras

- Document Scanners
  - SANE compatible devices.
  - Allow you to set resolution and ROI.
- Digital Camera
  - Uses Piggy Photo Library
  - Works with most DSLRs and point and shoots.
- AVT Camera
  - Professional digital imaging cameras with interchangeable optics.
  - Fine grain control of camera parameters.

Image Sets

ImageSets are lists of images. They are great for aggregating datasets.

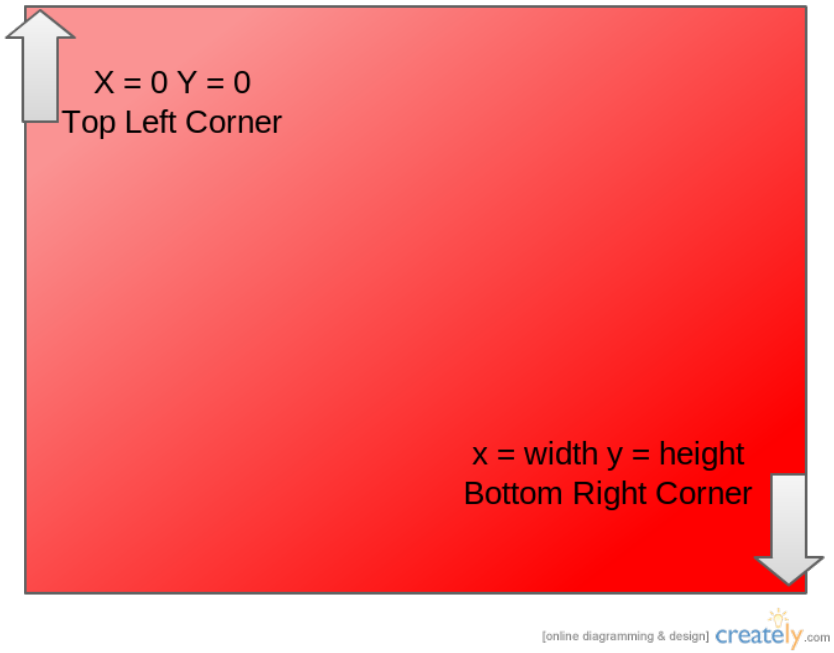
- By default load all image files in a directory.
- Can iterate over the list using list comps or for loops.
- Using the BeautifulSoup library can download sets from google.
- Can save image sets to directories, or animated gifs!
- The show command works just like on image class.
- Can apply averages to images.
- More coming soon.

ImageSet Example

Example (Image Sets)

```
mySet = ImageSet()
mySet.download('cats') # download cats
mySet.show() # show them to us
avgCat = mySet.average() # get the avg cat
avgCat.show() # show the avg cat
mySet[3].show() # show the third kitty
resized = mySet.standardize(128,64)
resized.save('cat.gif')
mySet.save('cat.gif') # save the cats as a gif
for cat in mySet: # iterate over the set of cats
    cat.binarize().show()
```

Getting at a pixel



- SimpleCV treats images as two dimensional arrays of color value tuples.
- Each tuple holds three values (Red,Green,Blue).
- Pixels start in the top left corner at zero.

Pixel Manipulation Example

Example (Getting at those pixels)

```
img = Image('helloworld.jpg')
c = img[0,0] # get a pixel
print c
print img.getPixel(0,0) # get another way
test = img[200:300,200:300]
test.show() # the result is an image
test = img[50:,:]
test.show() # again using slice
test = img[0:5,0:5]
print test.getNumpy() # get the raw values
img[0:105,0:105] = Color.RED # WRONG!
img.show() # does not work
test = img.getNumpy() #RIGHT!
test[0:105,0:105] = Color.RED
img2 = Image(test)
img2.show()
```

Getting at a pixel

- Images are read only. To write a pixel directly you need create a new image. Usually this happens in numpy
- Images support the list slice notation but return images.
- To get at the raw pixel values use **getNumpy()** and **getGrayNumpy()**
- Numpy values can be accessed using slices the first parameter is x, the second is y, and the third is the channel in RGB order. For example $npimg[x][y][0]$.
- The **Image.width** and **Image.height** member variables can help you find your way.

Another Pixel Manipulation Example

Example (Fancy Manipulations)

```
img = Image('helloworld.jpg')
gray = img.getGrayNumpy() # get the gray scale np image
colored = img.getNumpy()  # and the colored one
print (img.width,img.height) #tell us the image size
colored[0:20,:] = Color.BLUE # set the left side to blue
colored[:,0:20] = Color.GREEN # set the top row to green
colored[40:80,40:80]       \quad
     \includegraphics[width=0.4\linewidth]{JanEricBook.jpg}
= Color.YELLOW # make a yellow square
x,y = np.where(gray>230) # find bright pixels > 230
for xf,yf in zip(x,y): # for each of those
    colored[xf][yf] = Color.RED #make them red
img2 = Image(colored) # create an image
img2.show() # and show it
# now set the whole blue channel to 255
colored[:,:,2] = 255
# and show us that
img3 = Image(colored)
img3.show()
```

Getting at a pixel

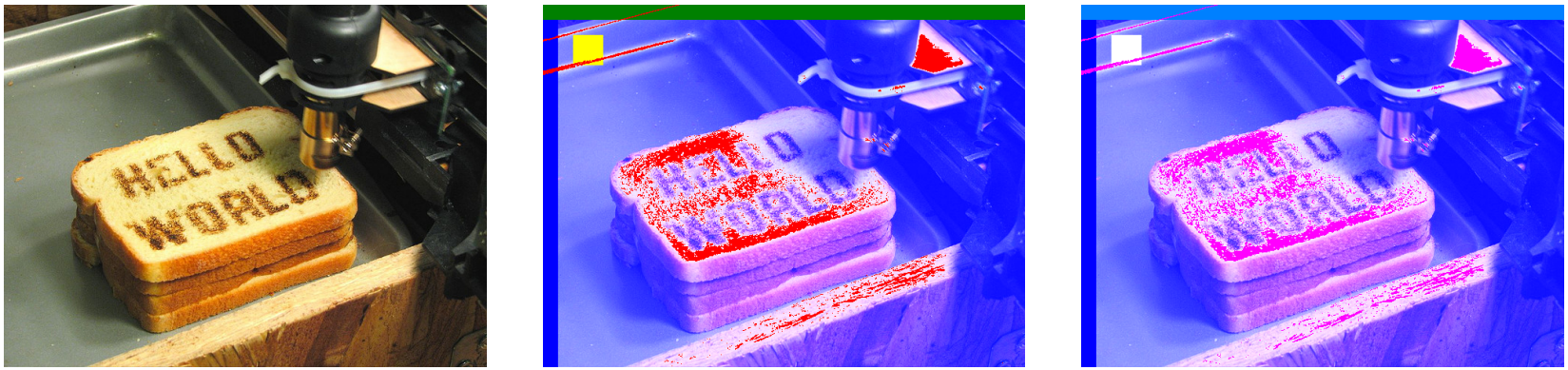


Figure: Not bad for less than 20 lines of code.

Paragraphs of Text

Sed iaculis dapibus gravida. Morbi sed tortor erat, nec interdum arcu. Sed id lorem lectus. Quisque viverra augue id sem ornare non aliquam nibh tristique. Aenean in ligula nisl. Nulla sed tellus ipsum. Donec vestibulum ligula non lorem vulputate fermentum accumsan neque mollis.

Sed diam enim, sagittis nec condimentum sit amet, ullamcorper sit amet libero. Aliquam vel dui orci, a porta odio. Nullam id suscipit ipsum. Aenean lobortis commodo seDerkt commodo leo gravida vitae. Pellentesque vehicula ante iaculis arcu pretium rutrum eget sit amet purus. Integer ornare nulla quis neque ultrices lobortis. Vestibulum ultrices tincidunt libero, quis commodo erat ullamcorper id.

Bullet Points

- Lorem ipsum dolor sit amet, consectetur adipiscing elit
- Aliquam blandit faucibus nisi, sit amet dapibus enim tempus eu
- Nulla commodo, erat quis gravida posuere, elit lacus lobortis est, quis porttitor odio mauris at libero
- Nam cursus est eget velit posuere pellentesque
- Vestibulum faucibus velit a augue condimentum quis convallis nulla gravida

Multiple Columns

Heading

1. Statement
2. Explanation
3. Example

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lectus nisl, ultricies in feugiat rutrum, porttitor sit amet augue. Aliquam ut tortor mauris. Sed volutpat ante purus, quis accumsan dolor.

Table

Treatments	Response 1	Response 2
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Table: Table caption

Theorem

Theorem (Mass–energy equivalence)

$E = mc^2$

Verbatim

Example (Theorem Slide Code)

```
def doStuff(a,b,c=[1,2,3]):
    a = 5
    b = a
    c.reverse()

derp = [1,2,3,4]
for i in derp:
   doStuff()
   pass
print deep
```

Verbatim

Example (Theorem Slide Code)

```
$E = mc^2$
```

Figure

Uncomment the code on this slide to include your own image from the same directory as the template .TeX file.

Citation

An example of the `\cite` command to cite within the presentation:

This statement requires citation [Smith, 2012].

References

John Smith (2012)

Title of the publication

Journal Name 12(3), 45 – 678.

The End