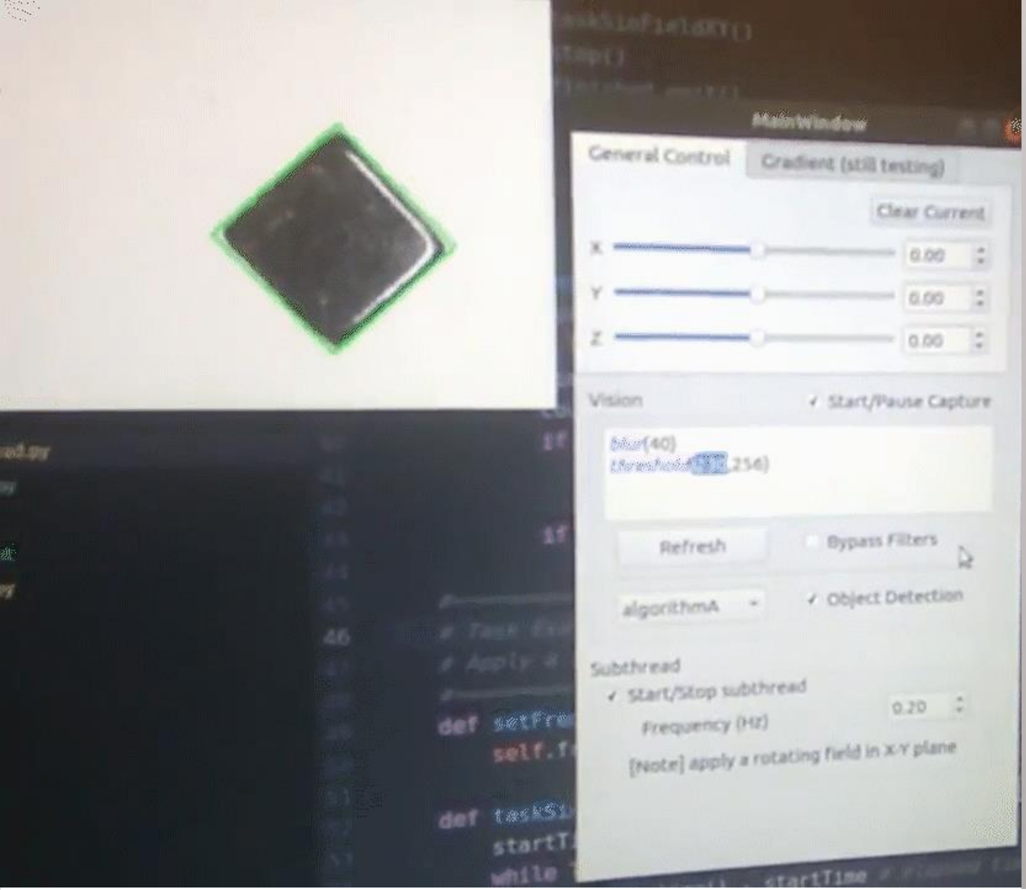


X 0.00

Y 0.00

Z 0.00

✓ Start/Pause Capture

```
blur(40)
thresholdA([illegible],256)
```

□ Bypass Filters

algorithmA

✓ Object Detection

✓ Start/Stop subthread

Frequency (Hz) 0.20

[Note] apply a rotating field in X-Y plane

Guide to the Python-based Coil System

https://github.com/atelier-ritz/CoilSystemPython

Table of contents

- Dependencies
- In a nutshell
- Program Structure
- Modify GUI
- About camera
- Image filters
- Object detection
- Multithreading

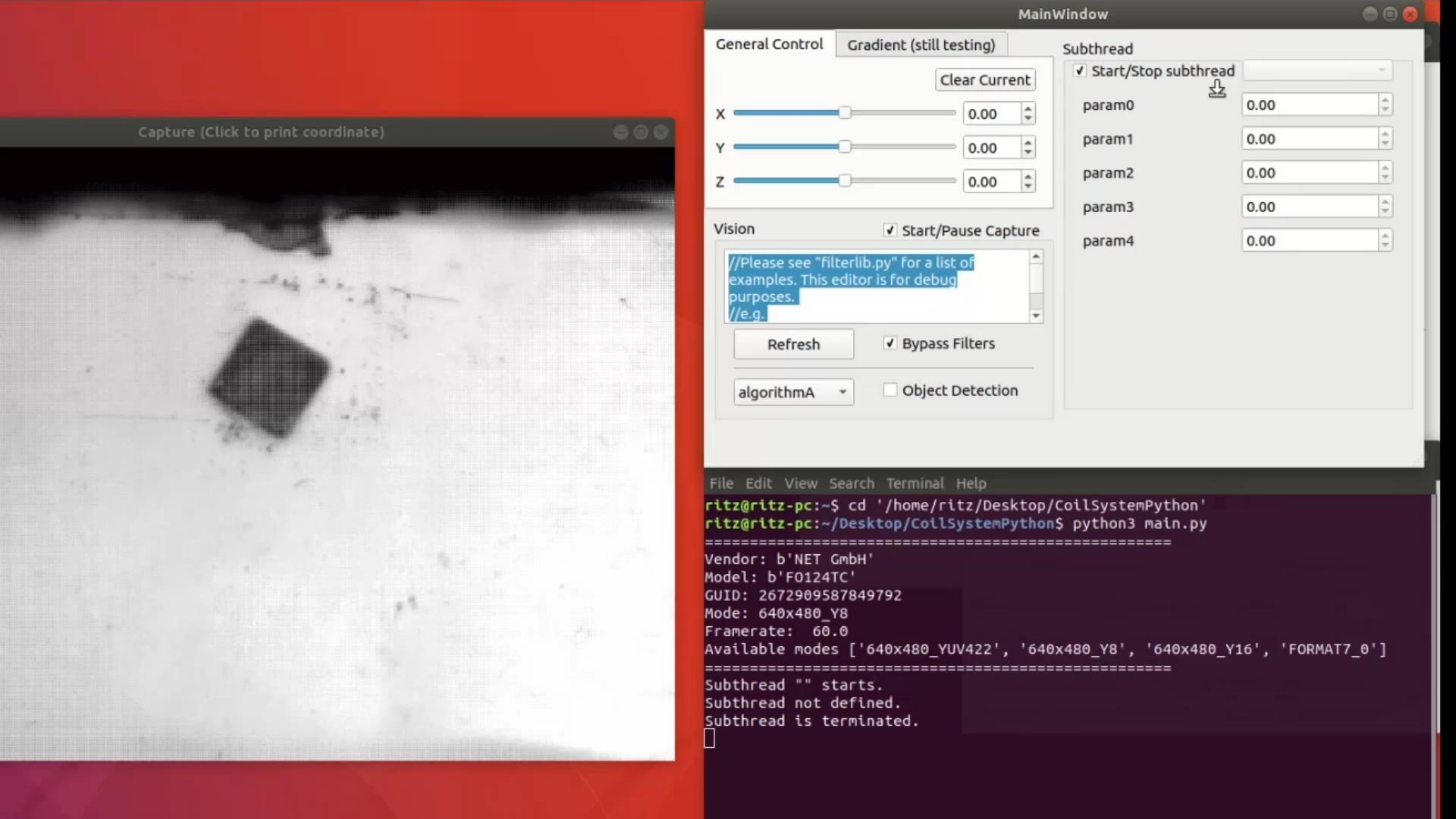
Dependencies

Recommend Ubuntu16

====== Tested on Ubuntu 17.10 ======

- Python 3.6 pre-installed in Ubuntu 17.10
- PyQt5 pip3 install pyqt5
  - *What is PyQt https://riverbankcomputing.com/software/pyqt/intro*
- Opencv pip3 install opencv-python, pip3 install opencv-contrib-python
- Pydc1394
  - *Firewire camera module https://github.com/jordens/pydc1394*
- Qt-designer sudo apt-get install qt4-designer
  - *GUI designer*

https://github.com/atelier-ritz/CoilSystemPython



Subthread

☑ Start/Stop subthread

param0 0.00

param1 0.00

param2 0.00

param3 0.00

param4 0.00

X 0.00

Y 0.00

Z 0.00

Vision ☑ Start/Pause Capture

```
//Please see "filterlib.py" for a list of
examples. This editor is for debug
purposes.
//e.g.
```

☑ Bypass Filters

algorithmA

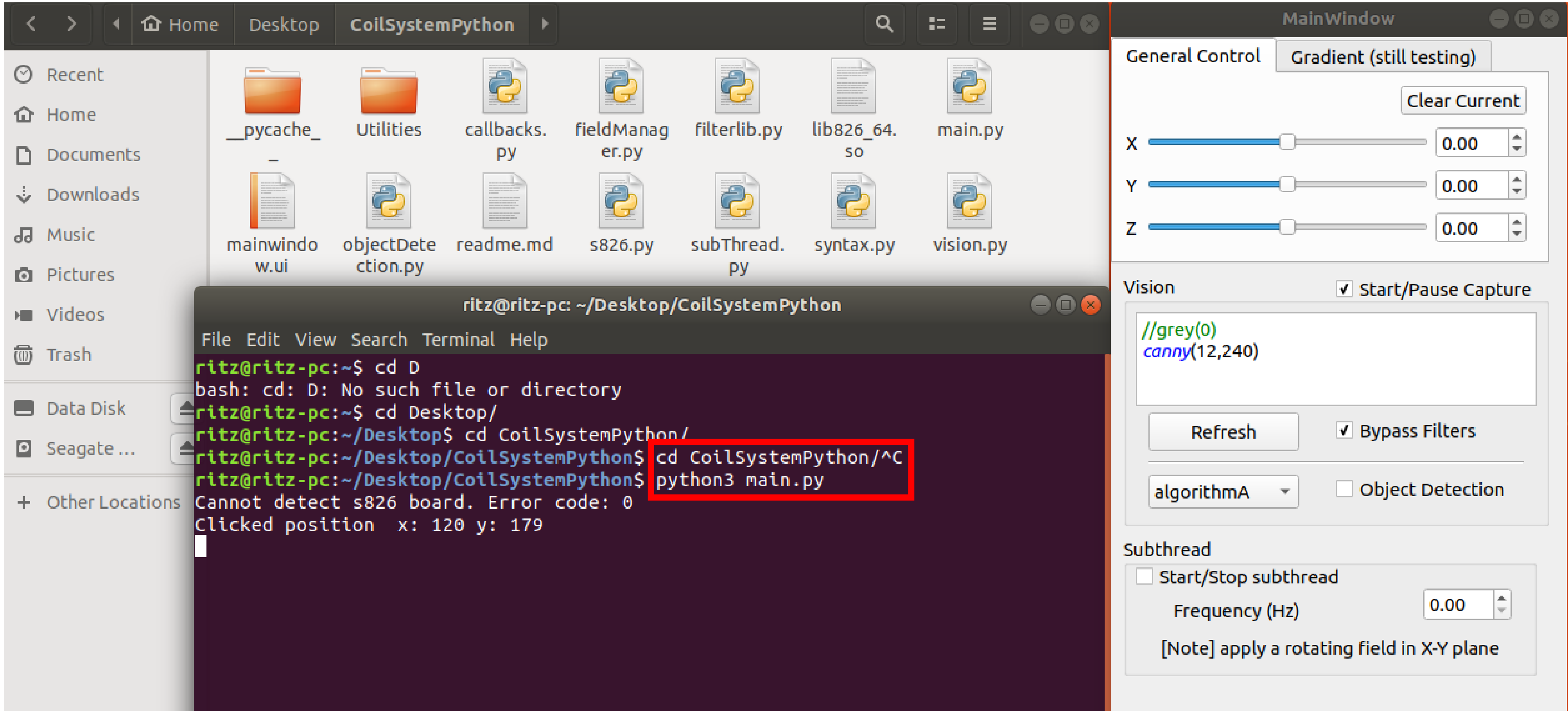
□ Object Detection

```
ritz@ritz-pc:~$ cd '/home/ritz/Desktop/CoilSystemPython'
ritz@ritz-pc:~/Desktop/CoilSystemPython$ python3 main.py
====================================================
Vendor: b'NET GmbH'
Model: b'FO124TC'
GUID: 2672909587849792
Mode: 640x480_Y8
Framerate:  60.0
Available modes ['640x480_YUV422', '640x480_Y8', '640x480_Y16', 'FORMAT7_0']
====================================================
Subthread "" starts.
Subthread not defined.
Subthread is terminated.
```

In a nutshell

Go to the working directory.

Run “python3 main.py”



```
ritz@ritz-pc:~$ cd D
bash: cd: D: No such file or directory
ritz@ritz-pc:~$ cd Desktop/
ritz@ritz-pc:~/Desktop$ cd CoilSystemPython/
ritz@ritz-pc:~/Desktop/CoilSystemPython$ cd CoilSystemPython/^C
ritz@ritz-pc:~/Desktop/CoilSystemPython$ python3 main.py
Cannot detect s826 board. Error code: 0
Clicked position  x: 120 y: 179
```

X 0.00

Y 0.00

Z 0.00

☑ Start/Pause Capture

```
//grey(0)
canny(12,240)
```

☑ Bypass Filters

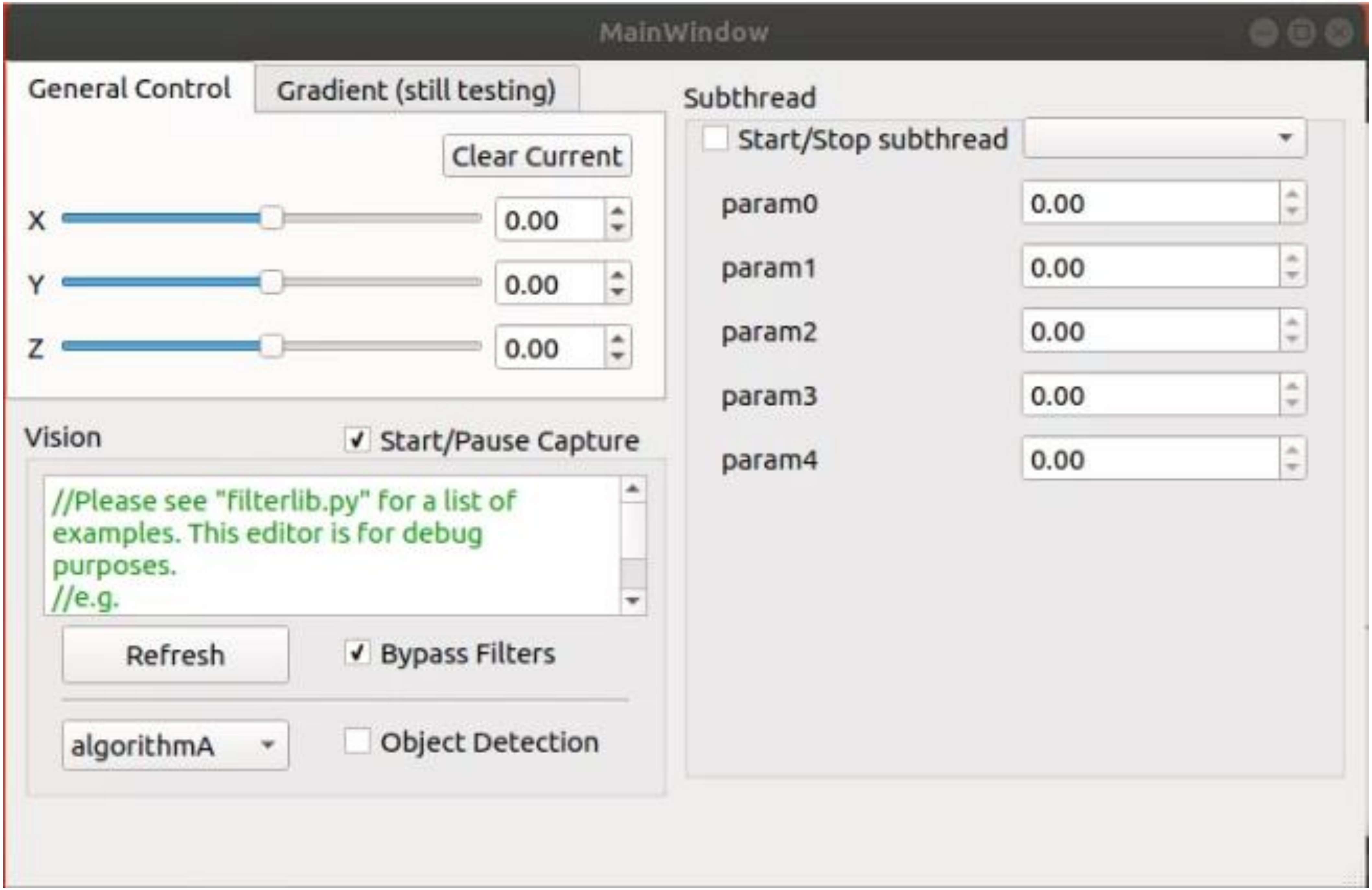
algorithmA

□ Object Detection

□ Start/Stop subthread

Frequency (Hz) 0.00

[Note] apply a rotating field in X-Y plane



X 0.00

Y 0.00

Z 0.00

☑ Start/Pause Capture

```
//Please see "filterlib.py" for a list of
examples. This editor is for debug
purposes.
//e.g.
```

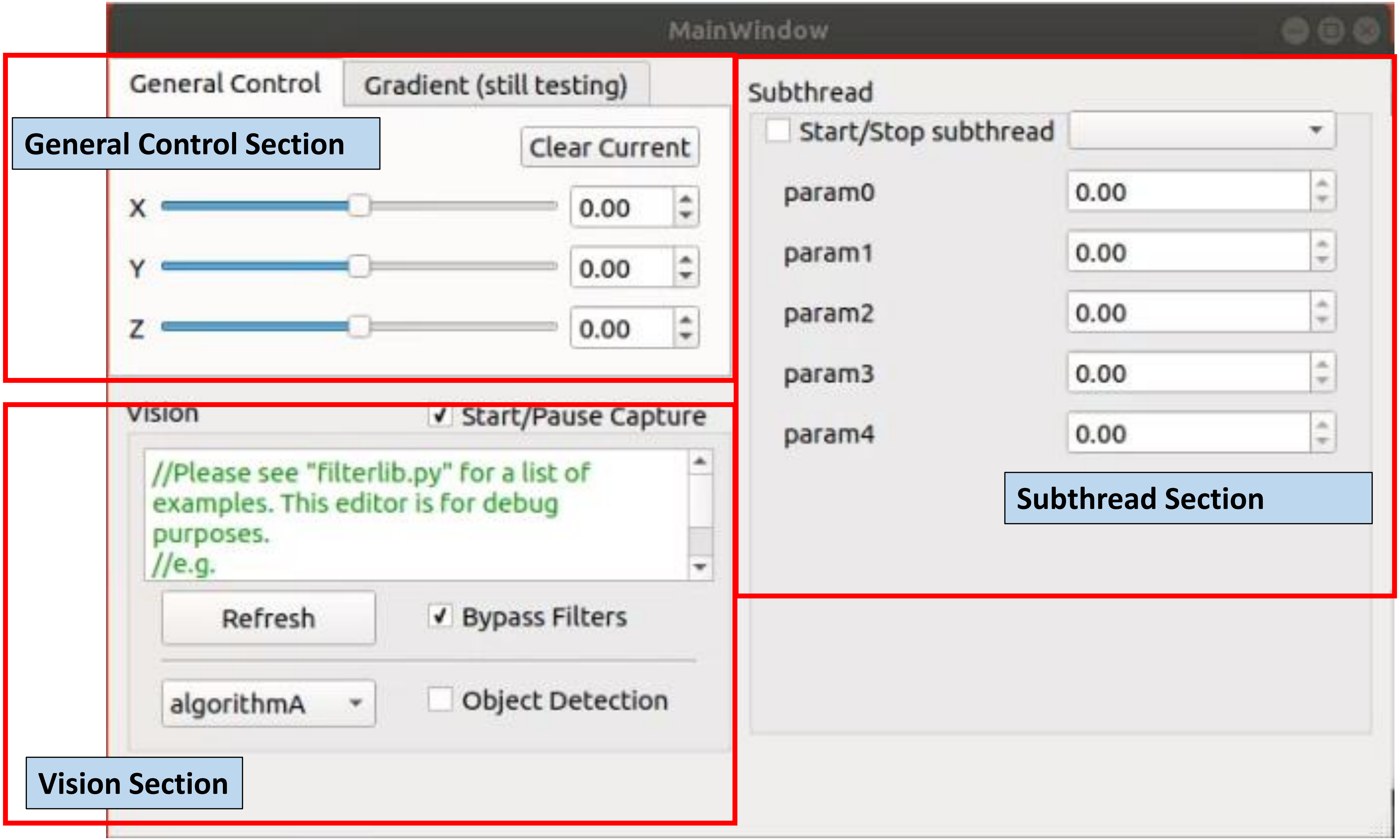
☑ Bypass Filters

algorithmA

□ Object Detection

□ Start/Stop subthread

param0	0.00
param1	0.00
param2	0.00
param3	0.00
param4	0.00



General Control Section

X 0.00

Y 0.00

Z 0.00

Vision Section

☑ Start/Pause Capture

```
//Please see "filterlib.py" for a list of
examples. This editor is for debug
purposes.
//e.g.
```

☑ Bypass Filters

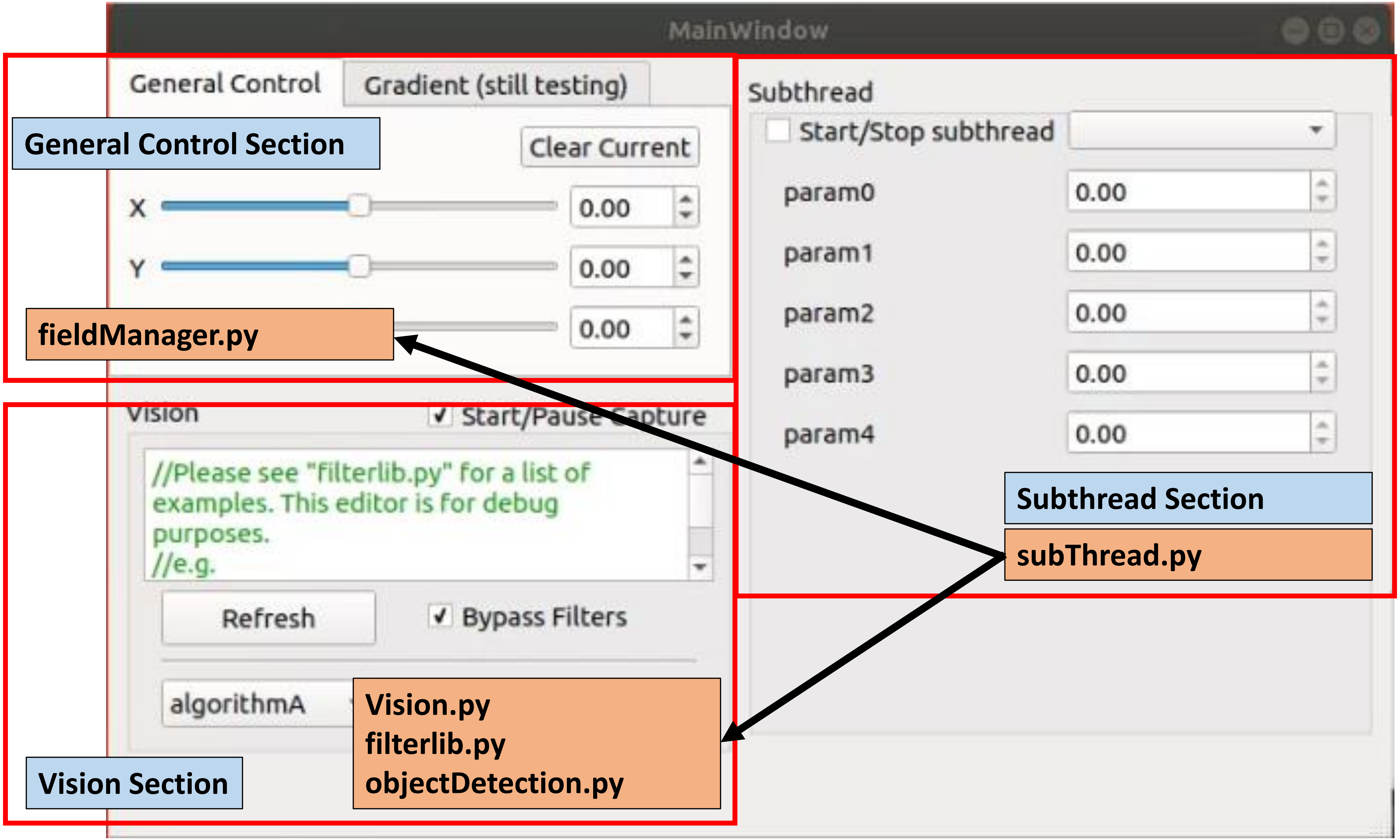
algorithmA

□ Object Detection

Subthread Section

□ Start/Stop subthread

param0	0.00
param1	0.00
param2	0.00
param3	0.00
param4	0.00



General Control Section

X 0.00

Y 0.00

0.00

fieldManager.py

☐ Start/Stop subthread

param0 0.00

param1 0.00

param2 0.00

param3 0.00

param4 0.00

Subthread Section

subThread.py

☑ Start/Pause Capture

//Please see "filterlib.py" for a list of examples. This editor is for debug purposes.
//e.g.

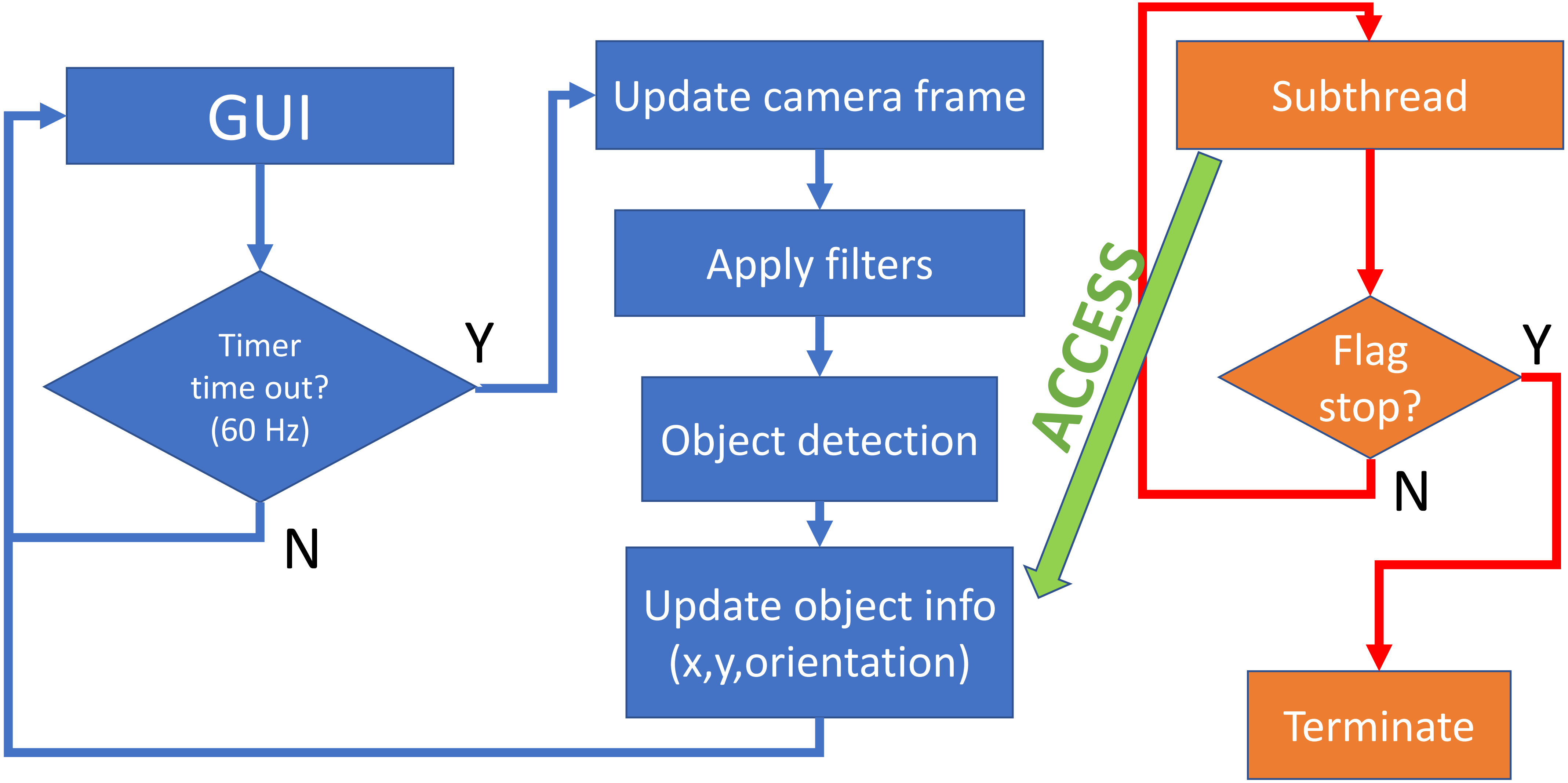
☑ Bypass Filters

algorithmA

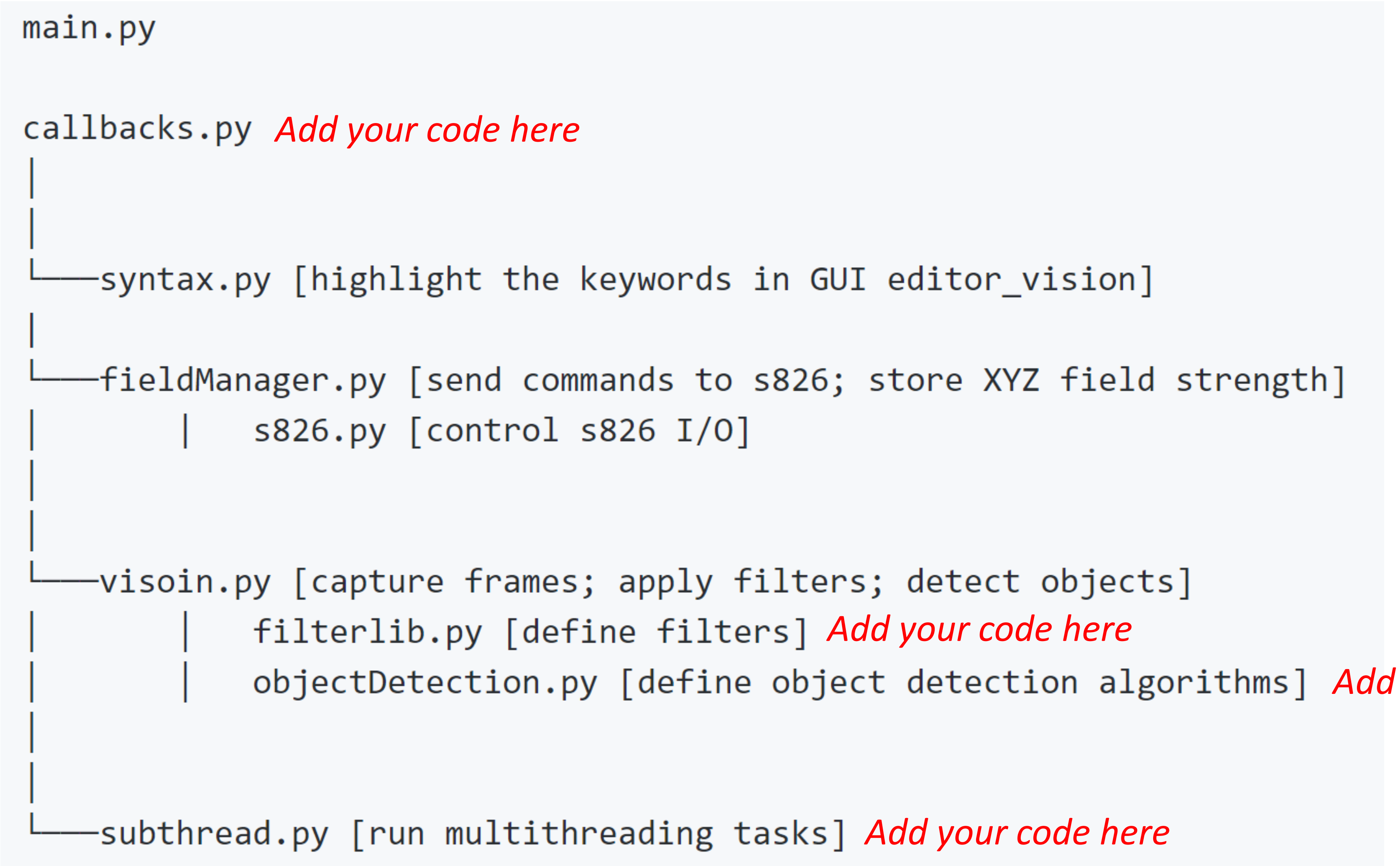
Vision.py
filterlib.py
objectDetection.py

Vision Section

Program structure



Program structure

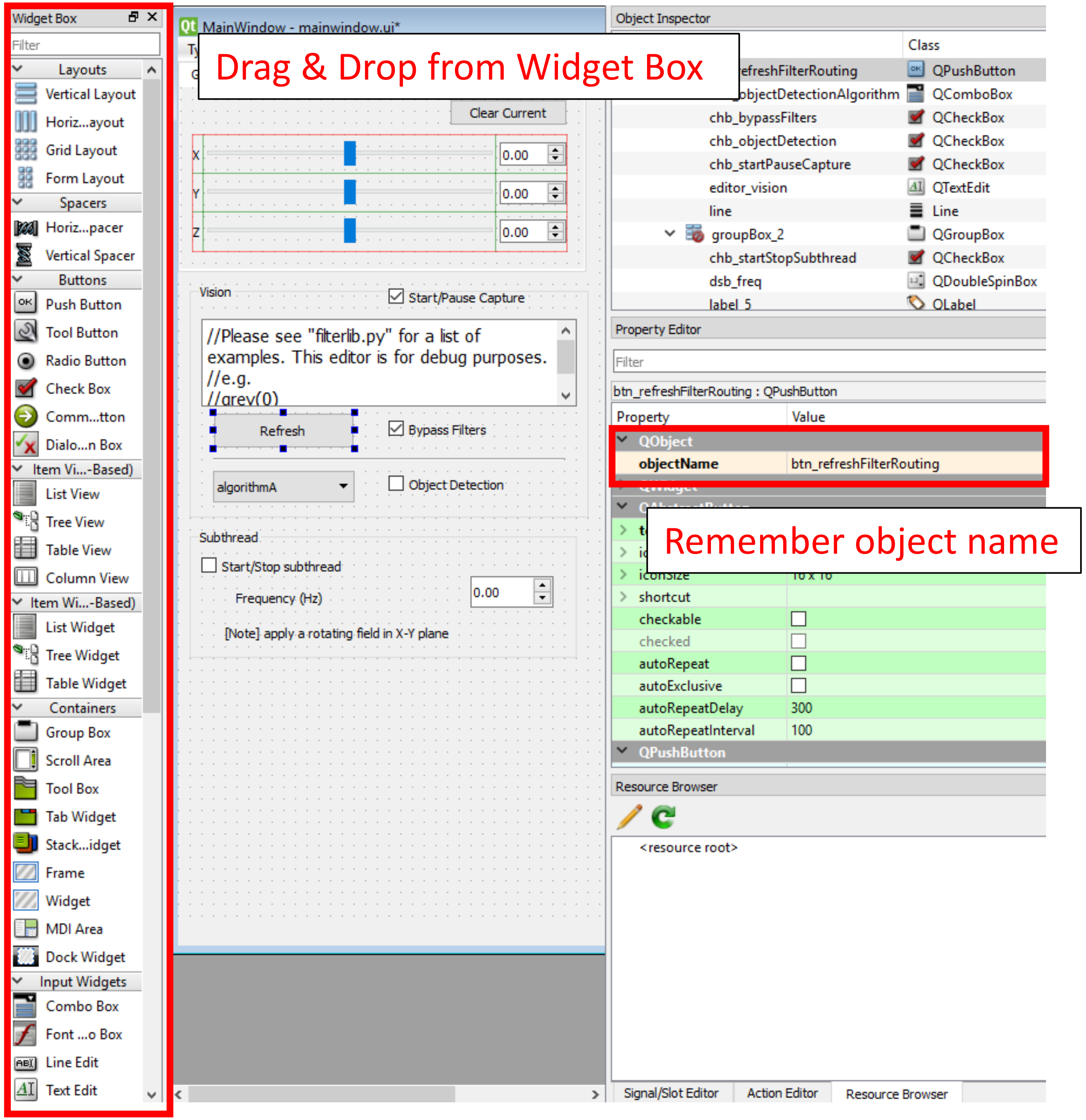


```
main.py

callbacks.py  Add your code here
|
|
└──syntax.py [highlight the keywords in GUI editor_vision]
|
└──fieldManager.py [send commands to s826; store XYZ field strength]
|       |   s826.py [control s826 I/O]
|
|
└──visoin.py [capture frames; apply filters; detect objects]
|       |   filterlib.py [define filters] Add your code here
|       |   objectDetection.py [define object detection algorithms]  Add your code here
|
|
└──subthread.py [run multithreading tasks] Add your code here
```

Modify GUI

1. Open “Mainwindow.ui” with qt-designer.



Drag & Drop from Widget Box

Remember object name

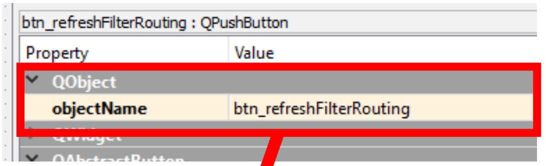
```
//Please see "filterlib.py" for a list of
examples. This editor is for debug purposes.
//e.g.
//grey(0)
```

Object	Class
...efreshFilterRouting	QPushButton
...objectDetectionAlgorithm	QComboBox
chb_bypassFilters	QCheckBox
chb_objectDetection	QCheckBox
chb_startPauseCapture	QCheckBox
editor_vision	QTextEdit
line	Line
groupBox_2	QGroupBox
chb_startStopSubthread	QCheckBox
dsb_freq	QDoubleSpinBox
label_5	QLabel

btn_refreshFilterRouting : QPushButton

Property	Value
QObject	
objectName	btn_refreshFilterRouting
QWidget	
QAbstractButton	
t...	
ic...	
iconSize	16 x 16
shortcut	
checkable	☐
checked	☐
autoRepeat	☐
autoExclusive	☐
autoRepeatDelay	300
autoRepeatInterval	100
QPushButton	

Modify GUI



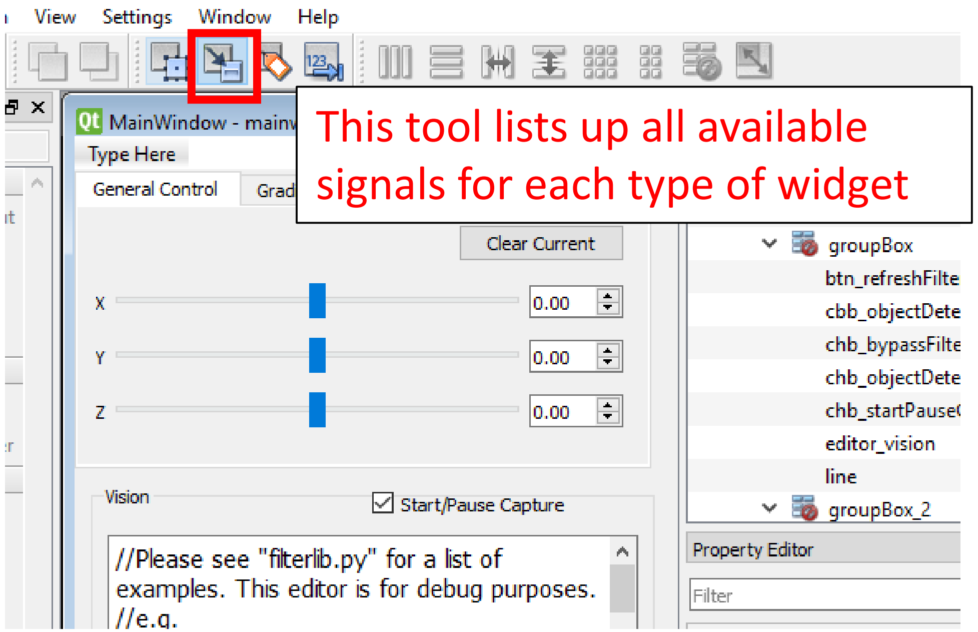
btn_refreshFilterRouting : QPushButton

Property	Value
QObject	
objectName	btn_refreshFilterRouting

<objectName>

1. Open “Mainwindow.ui” with qt-designer.
2. Oepn “callbacks.py” and edit connectSignals()

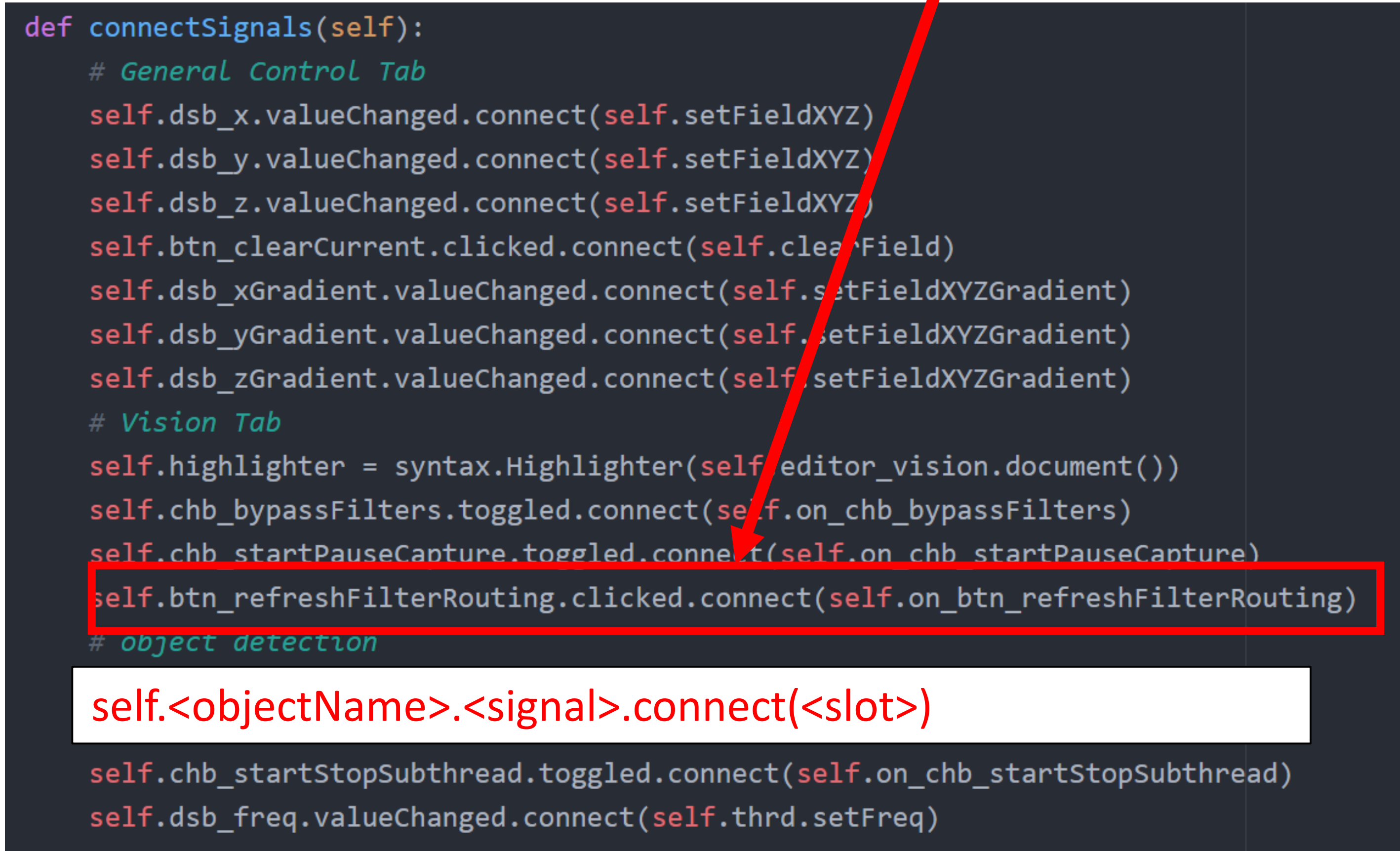
More about <signal> and <slot> http://pyqt.sourceforge.net/Docs/PyQt4/new_style_signals_slots.html



☑ Start/Pause Capture

//Please see "filterlib.py" for a list of examples. This editor is for debug purposes.
//e.g.

This tool lists up all available signals for each type of widget



```
def connectSignals(self):
    # General Control Tab
    self.dsb_x.valueChanged.connect(self.setFieldXYZ)
    self.dsb_y.valueChanged.connect(self.setFieldXYZ)
    self.dsb_z.valueChanged.connect(self.setFieldXYZ)
    self.btn_clearCurrent.clicked.connect(self.clearField)
    self.dsb_xGradient.valueChanged.connect(self.setFieldXYZGradient)
    self.dsb_yGradient.valueChanged.connect(self.setFieldXYZGradient)
    self.dsb_zGradient.valueChanged.connect(self.setFieldXYZGradient)
    # Vision Tab
    self.highlighter = syntax.Highlighter(self.editor_vision.document())
    self.chb_bypassFilters.toggled.connect(self.on_chb_bypassFilters)
    self.chb_startPauseCapture.toggled.connect(self.on_chb_startPauseCapture)
    self.btn_refreshFilterRouting.clicked.connect(self.on_btn_refreshFilterRouting)
    # object detection

    self.chb_startStopSubthread.toggled.connect(self.on_chb_startStopSubthread)
    self.dsb_freq.valueChanged.connect(self.thrd.setFreq)
```

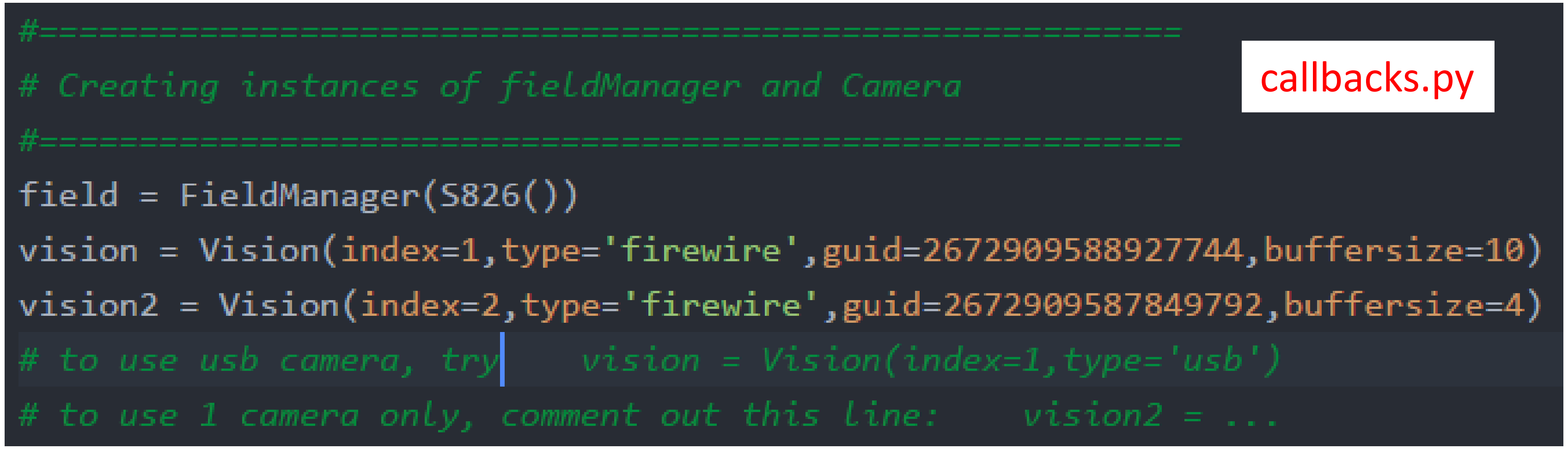
self.<objectName>.<signal>.connect(<slot>)

About cameras

- Support up to 2 cameras.
- Support USB camera or firewire camera (default).
- In this program, all the image filters and object detection in the code applies only to camera 1.
- For Firewire cameras, you must specify the buffer size. (Higher frames requires larger buffer size. Current parameters are meant for 640*480_Y8 greyscale images at 30 Hz)

About cameras

- Q: How do I check (select) the mode of the camera?
- A: Type “coriander” in terminal.
- Q: How do I know the guid of the camera?
- A: Refer to “Utilities” folder in the repository. (For USB cameras just disregard the guid)



callbacks.py

```
#=========================================================
# Creating instances of fieldManager and Camera
#=========================================================
field = FieldManager(S826())
vision = Vision(index=1,type='firewire',guid=2672909588927744,buffersize=10)
vision2 = Vision(index=2,type='firewire',guid=2672909587849792,buffersize=4)
# to use usb camera, try    vision = Vision(index=1,type='usb')
# to use 1 camera only, comment out this line:    vision2 = ...
```

Image filters

Note:
ONLY include alphabets,numbers, and underbars in your filter name.

1. Open “filterlib.py” and add your custom filter.

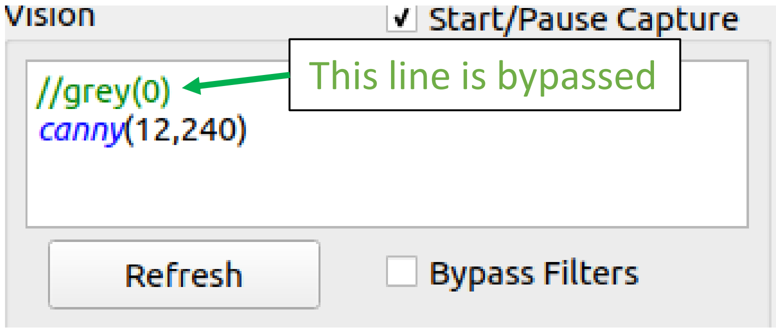
   Attention: need to handle variable conversion by yourself

   E.g. str -> int/float, define upper/lower bounds

2. Use it directly in the GUI.

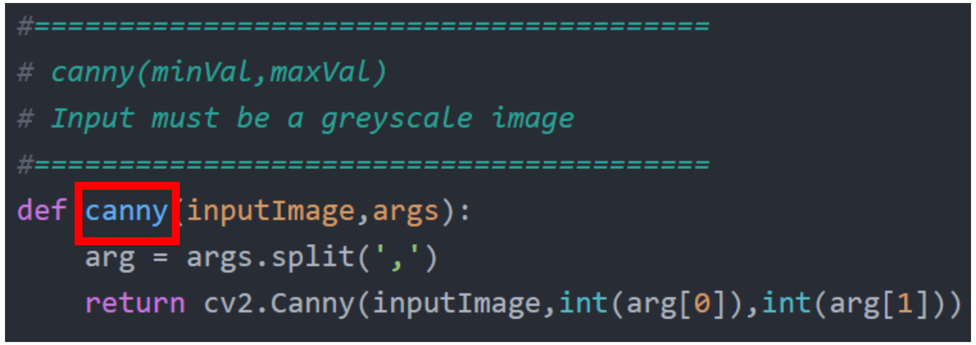
   Double slash to comment it out

Filters are connected in series and applied in order.

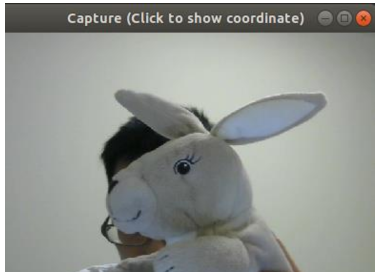


```
//grey(0)
canny(12,240)
```

This line is bypassed



```
#=========================================
# canny(minVal,maxVal)
# Input must be a greyscale image
#=========================================
def canny(inputImage,args):
    arg = args.split(',')
    return cv2.Canny(inputImage,int(arg[0]),int(arg[1]))
```



Original image

Filtered image

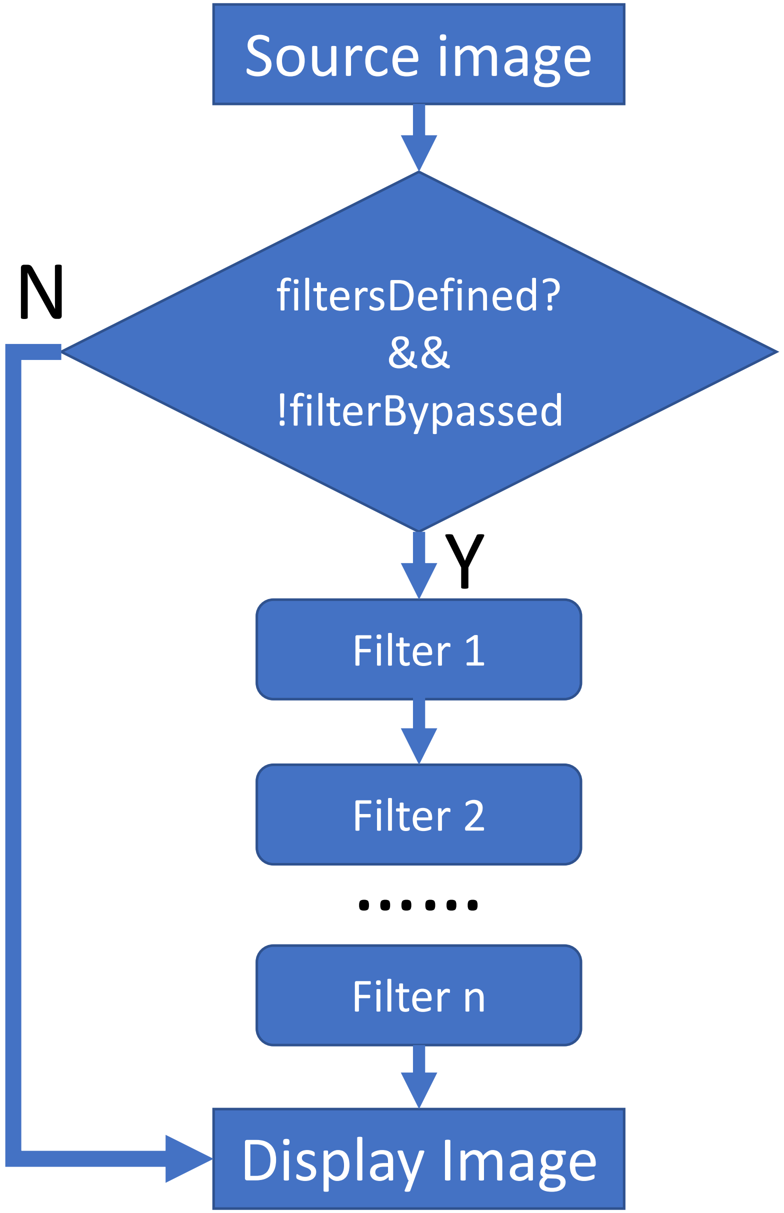


Image filters

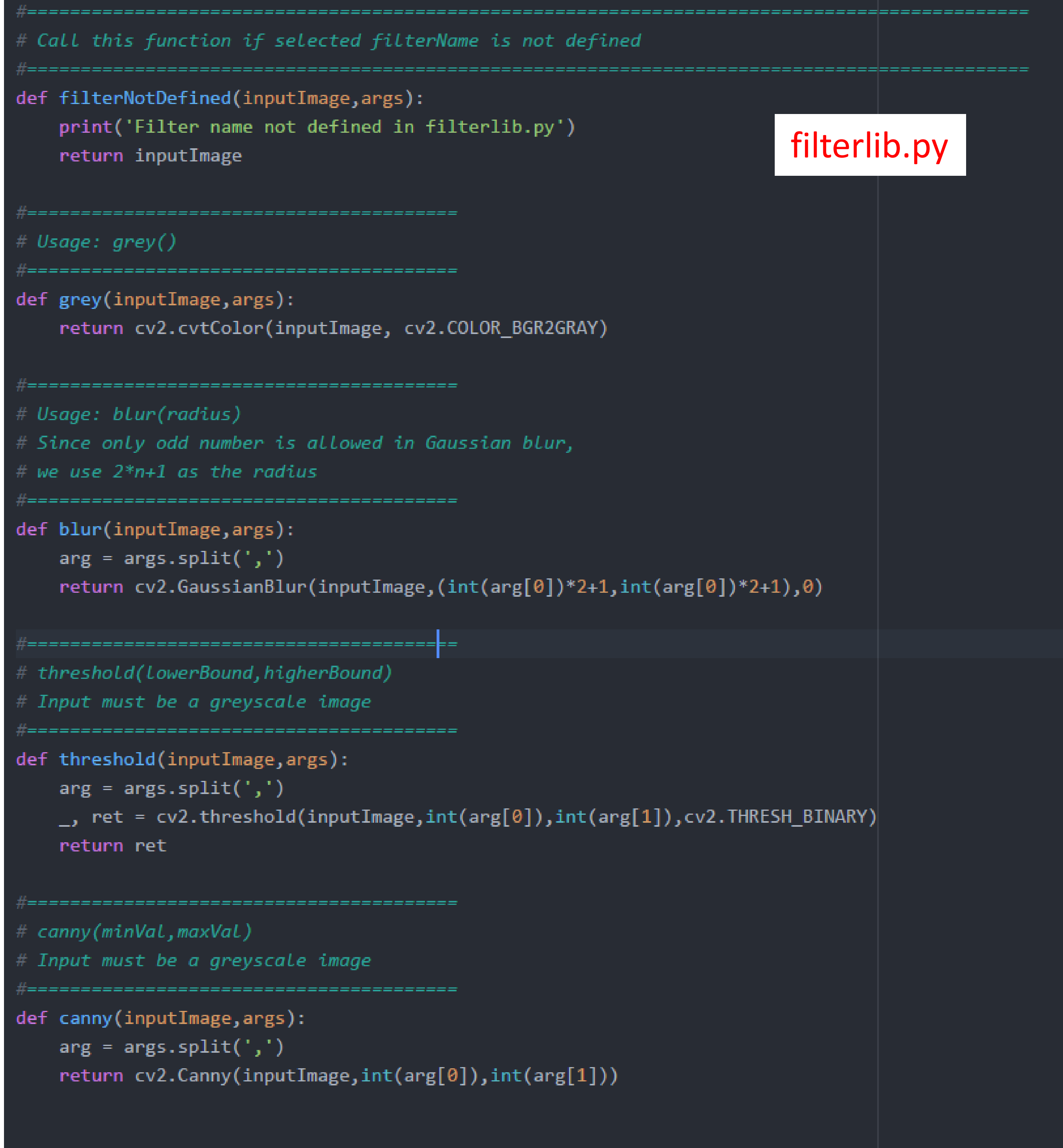
You need to specify the type of the input image.

e.g.

grey() -> makes a color image to greyscale image

You cannot apply a grey() filter to a greyscale image.

You need to be aware of the image data type because the function doesn’t handle it for you.



filterlib.py

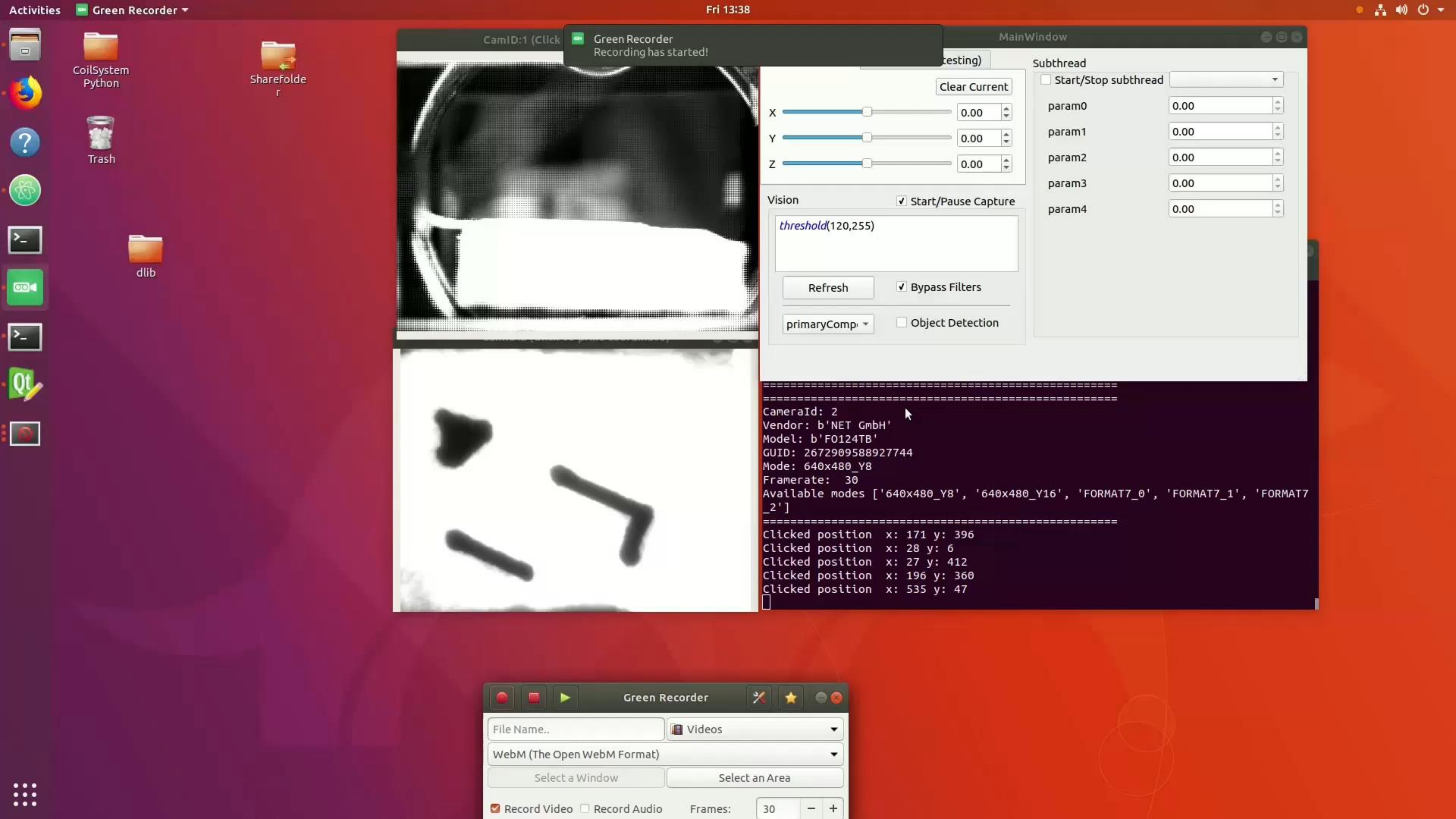
```
#=============================================================================================
# Call this function if selected filterName is not defined
#=============================================================================================
def filterNotDefined(inputImage,args):
    print('Filter name not defined in filterlib.py')
    return inputImage

#========================================
# Usage: grey()
#========================================
def grey(inputImage,args):
    return cv2.cvtColor(inputImage, cv2.COLOR_BGR2GRAY)

#========================================
# Usage: blur(radius)
# Since only odd number is allowed in Gaussian blur,
# we use 2*n+1 as the radius
#========================================
def blur(inputImage,args):
    arg = args.split(',')
    return cv2.GaussianBlur(inputImage,(int(arg[0])*2+1,int(arg[0])*2+1),0)

#========================================
# threshold(lowerBound,higherBound)
# Input must be a greyscale image
#========================================
def threshold(inputImage,args):
    arg = args.split(',')
    _, ret = cv2.threshold(inputImage,int(arg[0]),int(arg[1]),cv2.THRESH_BINARY)
    return ret

#========================================
# canny(minVal,maxVal)
# Input must be a greyscale image
#========================================
def canny(inputImage,args):
    arg = args.split(',')
    return cv2.Canny(inputImage,int(arg[0]),int(arg[1]))
```



Green Recorder
Recording has started!

☐ Start/Stop subthread

param0	0.00
param1	0.00
param2	0.00
param3	0.00
param4	0.00

X 0.00
Y 0.00
Z 0.00

☑ Start/Pause Capture

threshold(120,255)

☑ Bypass Filters

primaryComp

☐ Object Detection

```
==================================================
==================================================
CameraId: 2
Vendor: b'NET GmbH'
Model: b'FO124TB'
GUID: 2672909588927744
Mode: 640x480_Y8
Framerate:  30
Available modes ['640x480_Y8', '640x480_Y16', 'FORMAT7_0', 'FORMAT7_1', 'FORMAT7
_2']
==================================================
Clicked position  x: 171 y: 396
Clicked position  x: 28 y: 6
Clicked position  x: 27 y: 412
Clicked position  x: 196 y: 360
Clicked position  x: 535 y: 47
```

Green Recorder

File Name..

Videos

WebM (The Open WebM Format)

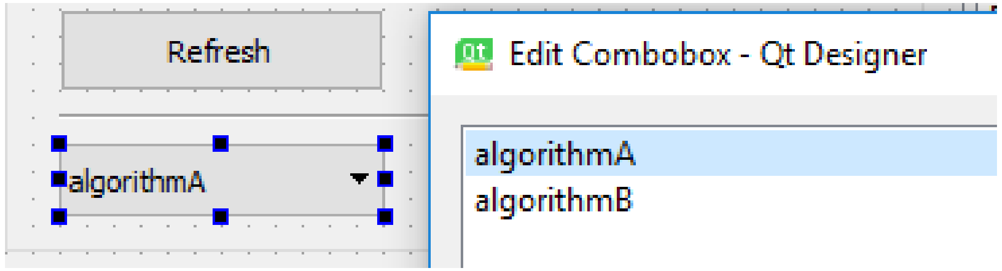
☑ Record Video ☐ Record Audio Frames: 30

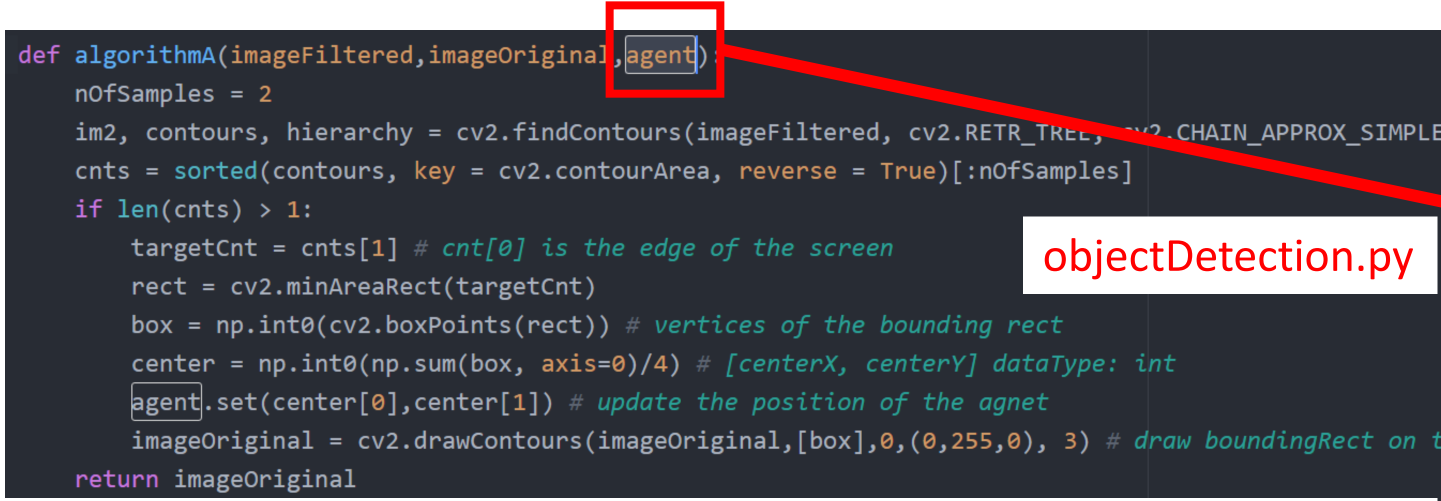
Object detection

Note:
When object detection is enabled, the original image overlayed with the detected object will be shown instead of the filtered image.

1. Add the name of your object detection algorithm to the GUI.
2. Define your algorithm in “objectDetection.py”.

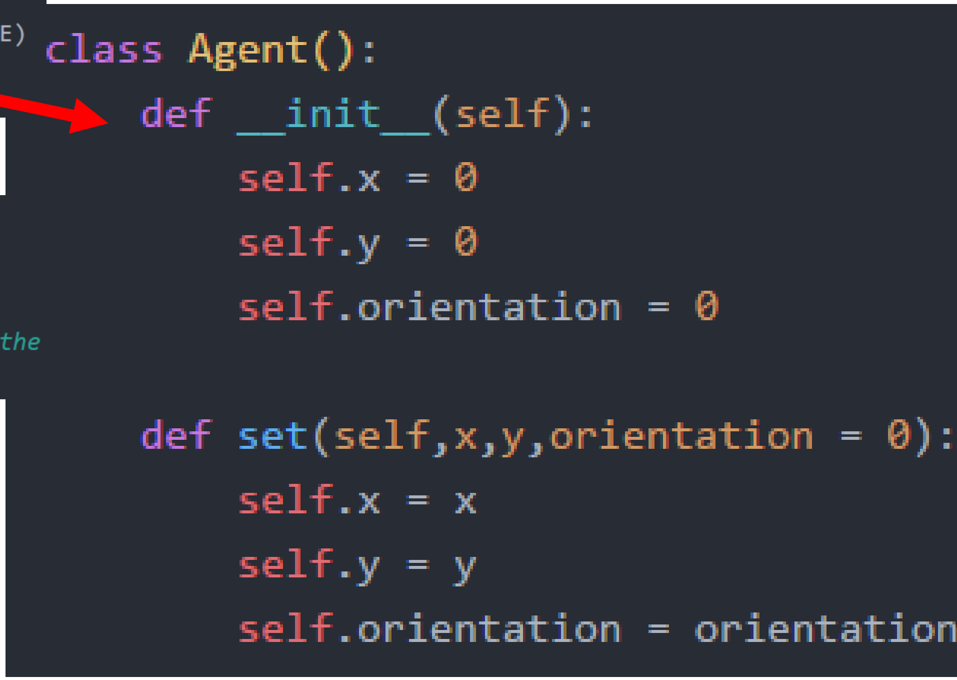
   See sample algorithmA() or google “opencv object detection python”





```
def algorithmA(imageFiltered,imageOriginal,agent):
    nOfSamples = 2
    im2, contours, hierarchy = cv2.findContours(imageFiltered, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    cnts = sorted(contours, key = cv2.contourArea, reverse = True)[:nOfSamples]
    if len(cnts) > 1:
        targetCnt = cnts[1] # cnt[0] is the edge of the screen
        rect = cv2.minAreaRect(targetCnt)
        box = np.int0(cv2.boxPoints(rect)) # vertices of the bounding rect
        center = np.int0(np.sum(box, axis=0)/4) # [centerX, centerY] dataType: int
        agent.set(center[0],center[1]) # update the position of the agnet
        imageOriginal = cv2.drawContours(imageOriginal,[box],0,(0,255,0), 3) # draw boundingRect on the
    return imageOriginal
```

objectDetection.py



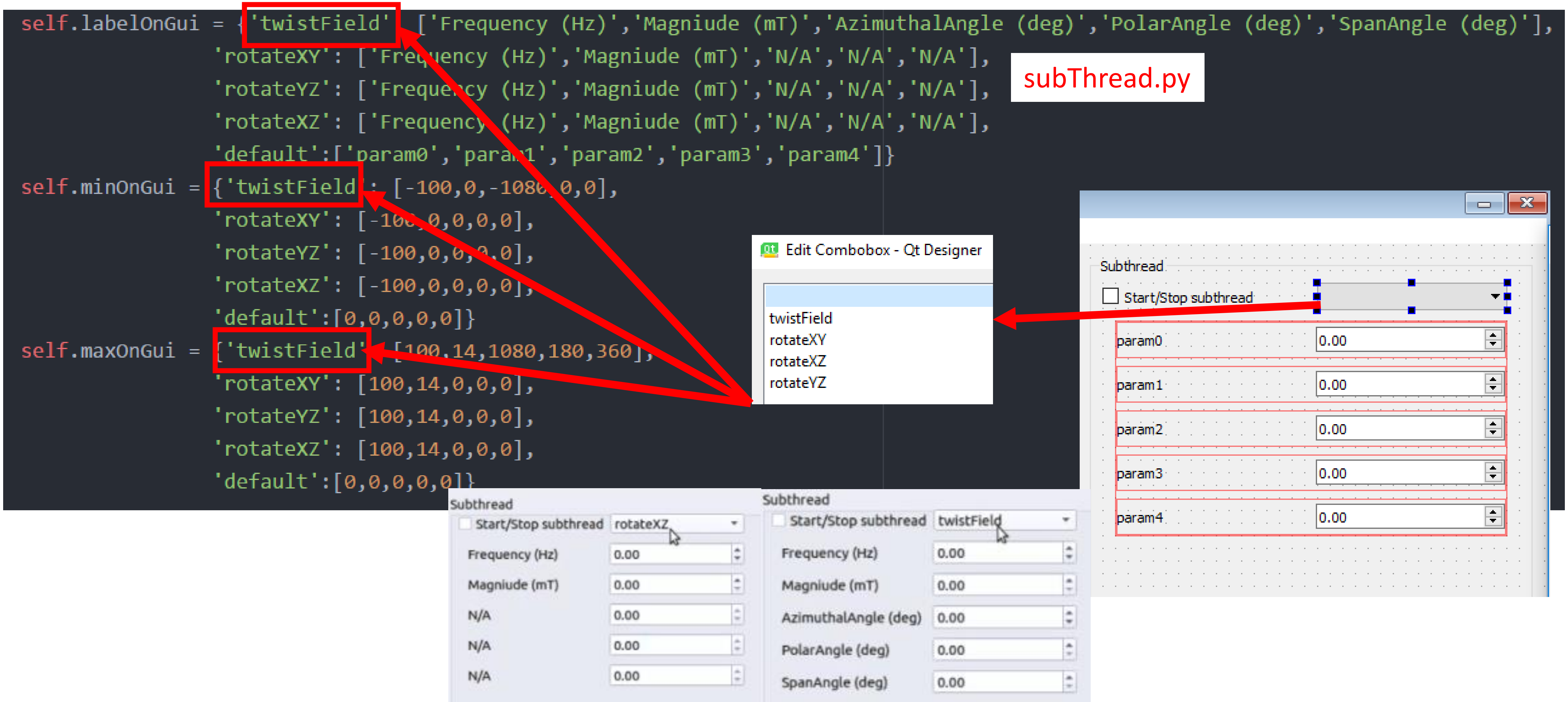
```
class Agent():
    def __init__(self):
        self.x = 0
        self.y = 0
        self.orientation = 0

    def set(self,x,y,orientation = 0):
        self.x = x
        self.y = y
        self.orientation = orientation
```

3. Instances of **Agent** class can be accessed via “vision.<agentName>”. Information about the agents are often used in a subthread.

Subthread

Use a subthread when you want to apply a time-varying magnetic field with respect to the position/orientation of the agents.



```
self.labelOnGui = {'twistField': ['Frequency (Hz)','Magniude (mT)','AzimuthalAngle (deg)','PolarAngle (deg)','SpanAngle (deg)'],
                'rotateXY': ['Frequency (Hz)','Magniude (mT)','N/A','N/A','N/A'],
                'rotateYZ': ['Frequency (Hz)','Magniude (mT)','N/A','N/A','N/A'],
                'rotateXZ': ['Frequency (Hz)','Magniude (mT)','N/A','N/A','N/A'],
                'default':['param0','param1','param2','param3','param4']}
self.minOnGui = {'twistField': [-100,0,-1080,0,0],
                'rotateXY': [-100,0,0,0,0],
                'rotateYZ': [-100,0,0,0,0],
                'rotateXZ': [-100,0,0,0,0],
                'default':[0,0,0,0,0]}
self.maxOnGui = {'twistField': [100,14,1080,180,360],
                'rotateXY': [100,14,0,0,0],
                'rotateYZ': [100,14,0,0,0],
                'rotateXZ': [100,14,0,0,0],
                'default':[0,0,0,0,0]}
```

subThread.py

Edit Combobox - Qt Designer

twistField
rotateXY
rotateXZ
rotateYZ

Subthread

□ Start/Stop subthread

param0	0.00
param1	0.00
param2	0.00
param3	0.00
param4	0.00

Subthread

□ Start/Stop subthread rotateXZ

Frequency (Hz)	0.00
Magniude (mT)	0.00
N/A	0.00
N/A	0.00
N/A	0.00

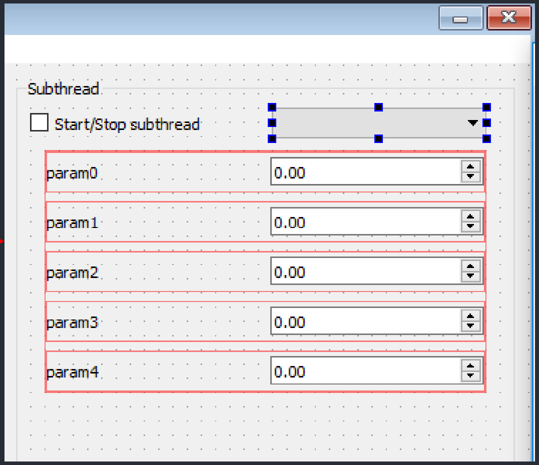
Subthread

□ Start/Stop subthread twistField

Frequency (Hz)	0.00
Magniude (mT)	0.00
AzimuthalAngle (deg)	0.00
PolarAngle (deg)	0.00
SpanAngle (deg)	0.00

```
def twistField(self):
    #====================d=======
    # reference params
    # 0 'Frequency (Hz)'
    # 1 'Magniude (mT)'
    # 2 'AzimuthalAngle (deg)'
    # 3 'PolarAngle (deg)'
    # 4 'SpanAngle (deg)'
    #============================
    startTime = time.time()
    record = 'Time(s), FieldX(mT), FiledY(mT), FieldZ(mT), X(pixel), Y(pixel) \n' # output to a txt file
    counter = 0
    while True:
        t = time.time() - startTime # elapsed time (sec)
        fieldX = self.params[1]* ( cosd(self.params[2])*cosd(self.params[3])*cosd(90-self.params[4]*0.5)*cos(2*pi*self.params[0]*t) - s
        fieldY = self.params[1]* ( sind(self.params[2])*cosd(self.params[3])*cosd(90-self.params[4]*0.5)*cos(2*pi*self.params[0]*t) + c
        fieldZ = self.params[1]* (-sind(self.params[3])*cosd(90-self.params[4]*0.5)*cos(2*pi*self.params[0]*t) + cosd(self.params[3])*c
        self.field.setX(fieldX)
        self.field.setY(fieldY)
        self.field.setZ(fieldZ)
        # save to txt
        counter += 1
        if counter > 300:
            counter = 0
            record = record + '{:.5f}, {:.2f}, {:.2f}, {:.2f}, {}, {}\n'.format(t,self.field.x,self.field.y,self.field.z,self.vision.ag
        if self.stopped:
            text_file = open("Output.txt", "w")
            text_file.write(record)
```

params[0], params[1], ...



Subthread

□ Start/Stop subthread

param0 0.00

param1 0.00

param2 0.00

param3 0.00

param4 0.00

Obtain elapsed time

setField

Obtain XYZ field strength

Stop flag

Obtain object position

```
self.vision.agent1.x,self.vision.agent1.y)
```

Tips

- Left click on the camera image returns xy coordinate in the terminal.
- Add machine learning packages (e.g. dlib) for object tracking.
  - *https://www.youtube.com/watch?v=ORgMddcNHvU*
- If you are using Ubuntu 17, use green-recorder for screen recording.
  - *Ubuntu 17.10 rolls back to GNOME shell (Ubuntu has been using Unity shell since Ubuntu 11), so some screen recording software is not working properly.*
  - *https://github.com/foss-project/green-recorder*
  - *You might have trouble converting WEBM format. One of the reasons I would recommend Ubuntu 16.*
- We need you to improve it!