```
#!/usr/bin/env python
# capture_sat.py
# call this routine only when you have the telescope pointing in the correct location
# take a series of images around the time of the satellite transition if there is discernable movement in
# the sky; worry about cleaning up the images later...
# save to disk potential shots of the satellite
# mb, (07/2007), 02/2008
#-----------------------------------------------
import os, sys, time
from opencv.cv import *
from opencv.highgui import *
import datetime
#-----------------------------------------------

# create a window to display the image
cvNamedWindow ('Camera', CV_WINDOW_AUTOSIZE)
device = 0
#get this information from your camera
img_width = 640
img_height = 480

#create a few placeholder images
oldbw= cvCreateImage (cvSize (img_width, img_height), 8, 1)     #8bit, single channel
cvSetZero(oldbw)
bw= cvCreateImage (cvSize (img_width, img_height), 8, 1)        #8bit, single channel
cvSetZero(bw)
diff= cvCreateImage (cvSize (img_width, img_height), 8, 1)      #8bit, single channel
cvSetZero(diff)
testimages = 25
offset = 10
sum_max = 0

#-----------------------------------------------
if len (sys.argv) == 1:
    # no argument on the command line, try to use the camera's first device
    capture = cvCreateCameraCapture (device)
    # set the wanted image size from the camera
    cvSetCaptureProperty (capture, CV_CAP_PROP_FRAME_WIDTH, img_width)
    cvSetCaptureProperty (capture, CV_CAP_PROP_FRAME_HEIGHT, img_height)
else:
    # we have an argument on the command line, we can assume this is a file name, so open it
    capture = cvCreateFileCapture (sys.argv [1])

# check that capture device is OK
if not capture:
    print "Error opening capture device"
    sys.exit (1)
#-----------------------------------------
```

```
# set the intervals, given an expected transit time
now = datetime.datetime.now()
# start looking offset minutes before the expected transition until offset minute after
offset = 2
#change according to the particular time
transit_time = "23:07:35"
tr_hr = int(transit_time.split(":")[0])
tr_min = int(transit_time.split(":")[1])
tr_sec = int(transit_time.split(":")[2])

#use this one for the know transit time
transit = datetime.datetime(now.year, now.month, now.day, tr_hr, tr_min, tr_sec)

#for testing using current time
#transit = datetime.datetime(now.year, now.month, now.day, now.hour, now.minute, now.second)

intervalA = datetime.timedelta(minutes=-offset)
intervalB = datetime.timedelta(minutes=offset)
watchstart = transit + intervalA
watchstop = transit + intervalB

print "starting at: ", watchstart
print "stoping at: ", watchstop
#------------------------------------------------

#find the threshold for the current viewing situation
print "creating the threshold - please wait"
for i in range(0, testimages):
        frame = cvQueryFrame (capture)
        if frame is None:
                break
           cvConvertImage(frame, bw, 0)
        cvSub(bw, oldbw, diff)
           min, max = cvMinMaxLoc(diff)
           sum_max = sum_max + max
           cvCopy(bw, oldbw)

threshold = int(sum_max / testimages) + offset
print "using this threshold: ", threshold

#------------------------------------------------------




#check the time; if in interval and the sky changes, keep the image
while ((now > watchstart) & (now < watchstop)):
        # 1. capture the current image
        frame = cvQueryFrame (capture)
```

```
        if frame is None:
                # no image captured... end the processing
                break
        # mirror the captured image if it is mirrored and show it
        #cvFlip (frame, None, 1)
        cvConvertImage(frame, bw, 0)

        #if a change occured (satellite moves across a portion of the screen), save the image, else discard
        cvSub(bw, oldbw, diff)
           min, max = cvMinMaxLoc(diff)
           if(max > threshold):
                        print "there is something out there!", max
                        filename = "images/" +"im_" + str(now )+ ".jpg"
                        cvSaveImage(filename, frame)

           cvShowImage ('Camera', diff)
          cvCopy(bw, oldbw)

           #update time
           now = datetime.datetime.now()

        # handle events
        k = cvWaitKey (10)
        if k == '\x1b':
                # user has press the ESC key, so exit
                break

print "finished watching..."
#----------------------------------------------------------
```