USB 카메라 프로그램 (Thread)

USB 카메라 프로그램 (Thread)

```
import cv2
import threading
import time
import logging
import os

logger = logging.getLogger(__name__)
archive_path = os.path.join(os.path.dirname(os.path.realpath(__file__)), 'archive')

thread = None
status = True

class UsbWebCamera:

    def __init__(self,fps=20,video_source=0):
        logger.info("Initializing usb camera class with {fps} " + str(fps) + " and video_source={" + str(video_source) +
"}")
        self.fps = fps
        self.video_source = video_source
        self.camera = cv2.VideoCapture(self.video_source)

        self.camera.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
        self.camera.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)

        self.max_frames = 5*self.fps
        self.frames = []
        self.isrunning = False

    def run(self):
        logging.debug("Perparing Usb Camera thread")
        global thread
        if thread is None:
            logging.debug("Creating thread")
            thread = threading.Thread(target=self._capture_loop,daemon=True)
            logger.debug("Starting thread")
            self.isrunning = True
            thread.start()
            logger.info("Thread started")

    def _capture_loop(self):
        global status
        dt = 1/self.fps
        logger.debug("Observation started")
```

```
    while self.isrunning:
      if(status):
        v,im = self.camera.read()
        im = cv2.flip(im, 1)
        #im = cv2.flip(im, 0)
        if v:
          if len(self.frames)==self.max_frames:
            self.frames = self.frames[1:]
          self.frames.append(im)
      time.sleep(dt)
    logger.info("Thread stopped successfully")

  def status(self):
    global status
    logger.debug("Status isrunning" + str(status))
    if(status):
      content = "STOP THREAD !"
      status = False
    else:
      content = "START THREAD !"
      status = True
    return content

  def get_frame(self, _bytes=True):
    if len(self.frames)>0:
      if _bytes:
        img = cv2.imencode('.png',self.frames[-1])[1].tobytes()
      else:
        img = self.frames[-1]
    else:
      logger.info("Change cam source 1")
      self.video_source = 1
      self.camera = cv2.VideoCapture(self.video_source)
      self.camera.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
      self.camera.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
      with open(archive_path+"/not_found.jpeg","rb") as f:
        img = f.read()
    return img
```

Revision #1
★Created 1 October 2024 13:58:13 by Hyeon Su Ryu
Updated 1 October 2024 13:59:25 by Hyeon Su Ryu