PI 카메라 프로그램 (Thread)

PI 카메라 프로그램 (Thread)

```python
# import the necessary packages
from picamera.array import PiRGBArray
from picamera import PiCamera
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 = False

# Set up camera constants
#IM_WIDTH = 1024
#IM_HEIGHT = 768
IM_WIDTH = 640
IM_HEIGHT = 480

class PiWebCamera:
    def __init__(self,fps=20):
        logger.info(f"Initializing pi camera class with {fps} fps ")
        self.fps = fps
        self.camera = PiCamera(framerate=24)
        self.camera.resolution = (IM_WIDTH,IM_HEIGHT)
        # self.camera.rotation = 180
        self.rawCapture = PiRGBArray(self.camera, size=(IM_WIDTH,IM_HEIGHT))
        self.rawCapture.truncate(0)
        self.max_frames = 5*self.fps
        self.frames = []
        self.isrunning = False

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

    def _capture_loop(self):
        global status
        dt = 1/self.fps
        logger.debug("Observation started")
        while self.isrunning:
            if(status):
                self.camera.capture(self.rawCapture, format="bgr",
use_video_port=True)
                self.rawCapture.truncate(0)
                image = self.rawCapture.array
                if len(self.frames)==self.max_frames:
                    self.frames = self.frames[1:]
                self.frames.append(image)
            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:
            with open(archive_path+"/not_found.jpeg","rb") as f:
                img = f.read()
        return img
```

Revision #1
Created 1 October 2024 13:59:48 by Hyeon Su Ryu
Updated 1 October 2024 14:00:20 by Hyeon Su Ryu