Assignment#3

25.02.12 (1day delay)

0. Code refactoring

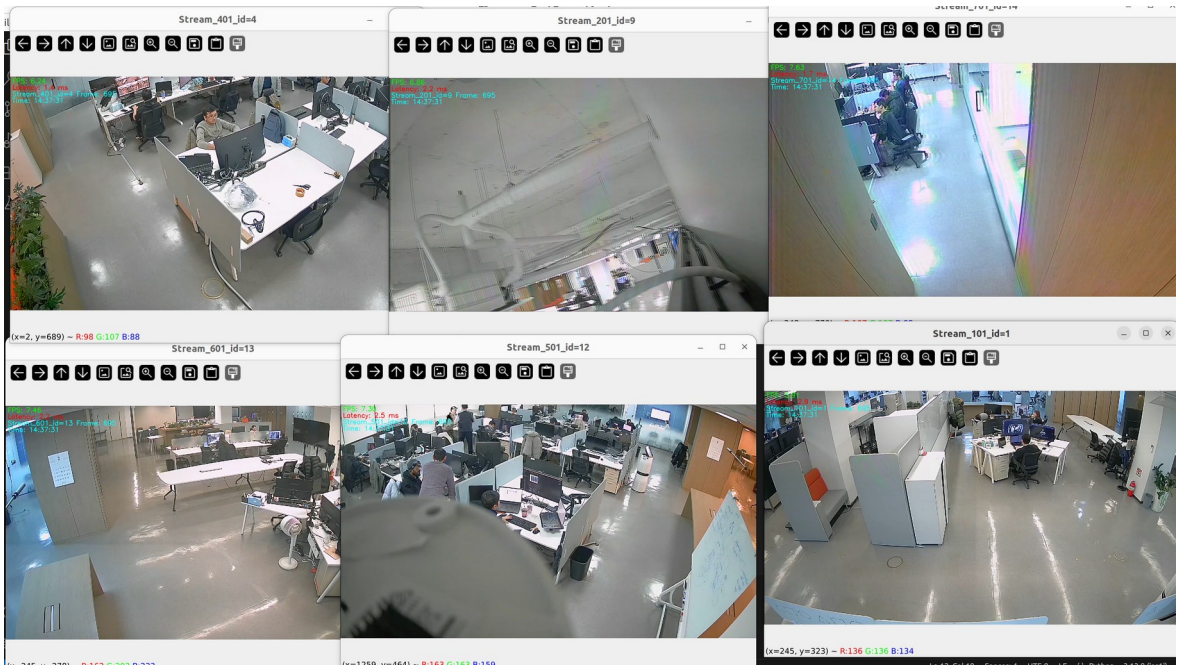
1. Class 내부/외부 사용되는 변수, 함수 구분 및 정의 방식 결정(private or not, class member or not)
2. Try except 구문 활용하여 error 로그 찍기
3. 함수-> 동사_명사

   변수-> 명사

1. FFmpeg-based multi-camera streaming

```
[h264 @ 0x6359d0372540] error while decoding MB 47 32, bytestream -15
지정된 시간(5분)이 경과하여 스트리밍을 종료합니다.

--- FPS 측정 결과 ---
Stream_101_id=1 - 평균 FPS: 7.18 (총 프레임: 2344)
Stream_201_id=2 - 평균 FPS: 7.23 (총 프레임: 2344)
Stream_301_id=3 - 평균 FPS: 7.27 (총 프레임: 2344)
Stream_401_id=4 - 평균 FPS: 7.32 (총 프레임: 2344)
Stream_501_id=5 - 평균 FPS: 7.36 (총 프레임: 2344)
Stream_601_id=6 - 평균 FPS: 7.41 (총 프레임: 2344)
Stream_701_id=7 - 평균 FPS: 7.46 (총 프레임: 2344)
Stream_101_id=8 - 평균 FPS: 7.51 (총 프레임: 2344)
Stream_201_id=9 - 평균 FPS: 7.56 (총 프레임: 2344)
Stream_301_id=10 - 평균 FPS: 7.61 (총 프레임: 2344)
Stream_401_id=11 - 평균 FPS: 7.65 (총 프레임: 2344)
Stream_501_id=12 - 평균 FPS: 7.71 (총 프레임: 2344)
Stream_601_id=13 - 평균 FPS: 7.76 (총 프레임: 2344)
Stream_701_id=14 - 평균 FPS: 7.81 (총 프레임: 2344)
(as1) yt@yt:~/dx/ipcam$
```



Conditions

- 버퍼링이 활성화된 상태
- 약 0.5 초의 최대 지연이 기본값
- 14 개의 rtsp 커넥션

Memory 5gb-> 7.5gb (2.5gb 증가)

FPS value

	case1	case2	case3
cam1	7.18	7.06	7.01
cam2	7.23	7.11	7.06
cam3	7.27	7.15	7.10
cam4	7.32	7.20	7.15
cam5	7.36	7.24	7.19
cam6	7.41	7.29	7.24
cam7	7.46	7.34	7.28
cam8	7.51	7.38	7.33
cam9	7.56	7.43	7.38
cam10	7.61	7.48	7.43
cam11	7.65	7.53	7.48
cam12	7.71	7.58	7.53
cam13	7.76	7.63	7.58
cam14	7.81	7.68	7.63

2. GStreamer-based multi-camera streaming

```python
def __link_elements(self, *elements):
    """
    Helper function to sequentially link multiple GStreamer elements.

    :param elements: list, GStreamer elements to be linked
    :return: bool, True if linking is successful, False otherwise
    """
    for i in range(len(elements) - 1):
        if not elements[i].link(elements[i + 1]):
            print(f"링크 실패: {elements[i].get_name()} -> {elements[i+1].get_name()}")
            return False
    return True
```

About “__link_elements” func :

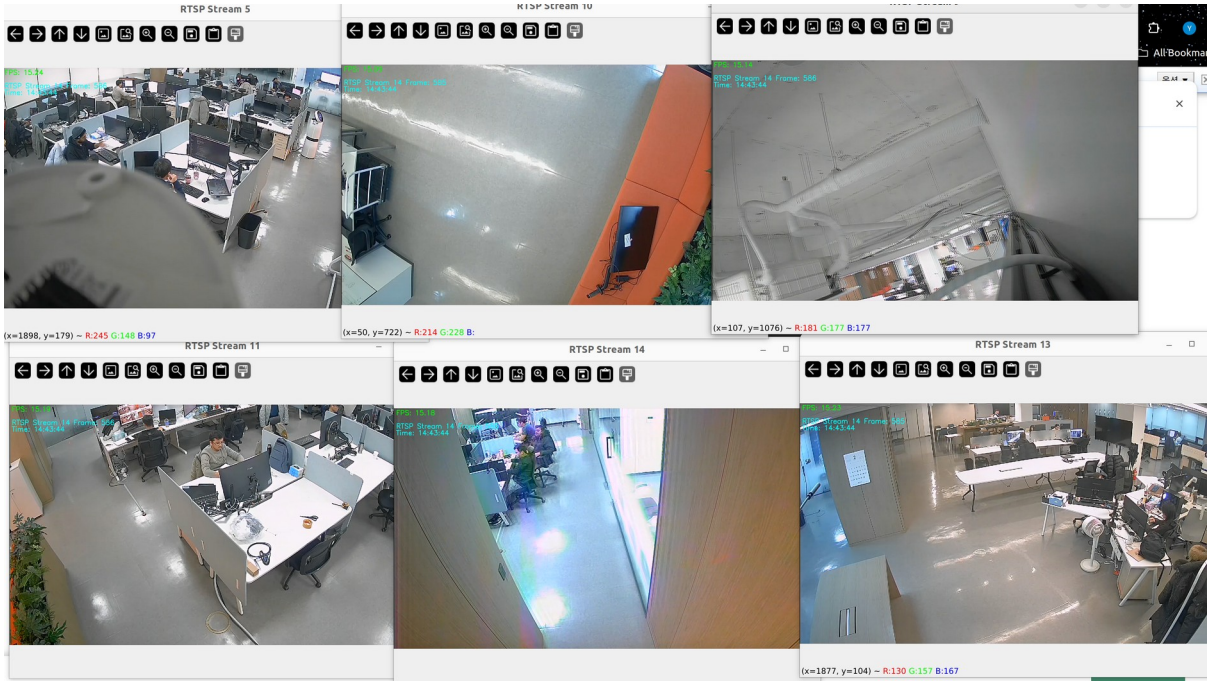
Input : Gstreamer elements 들이 입력 순서에 따라 순차적으로 들어옴 (순서가 중요함)

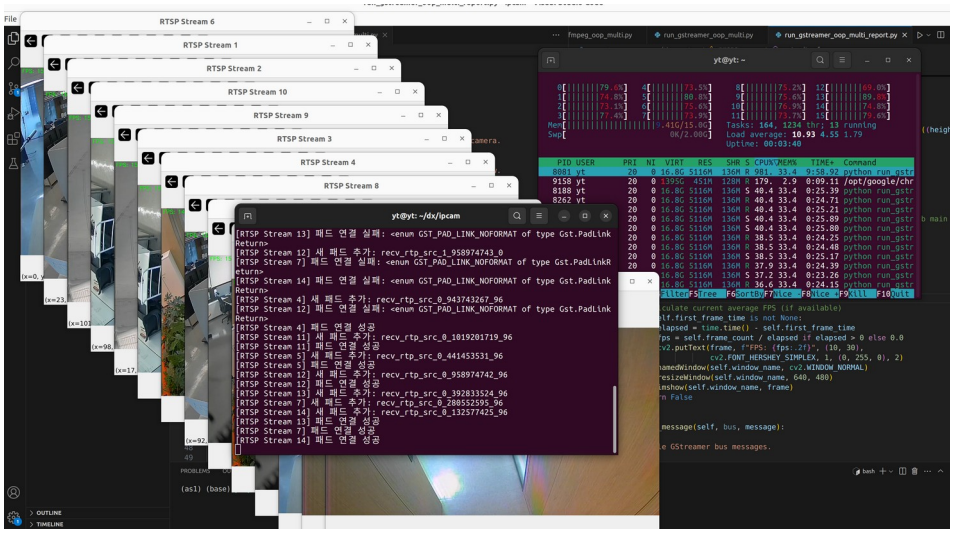
Elements[i].link(~~) 에서 실제로 연결하는 역할을 하고 성공하면 true 반환하도록 되어있음

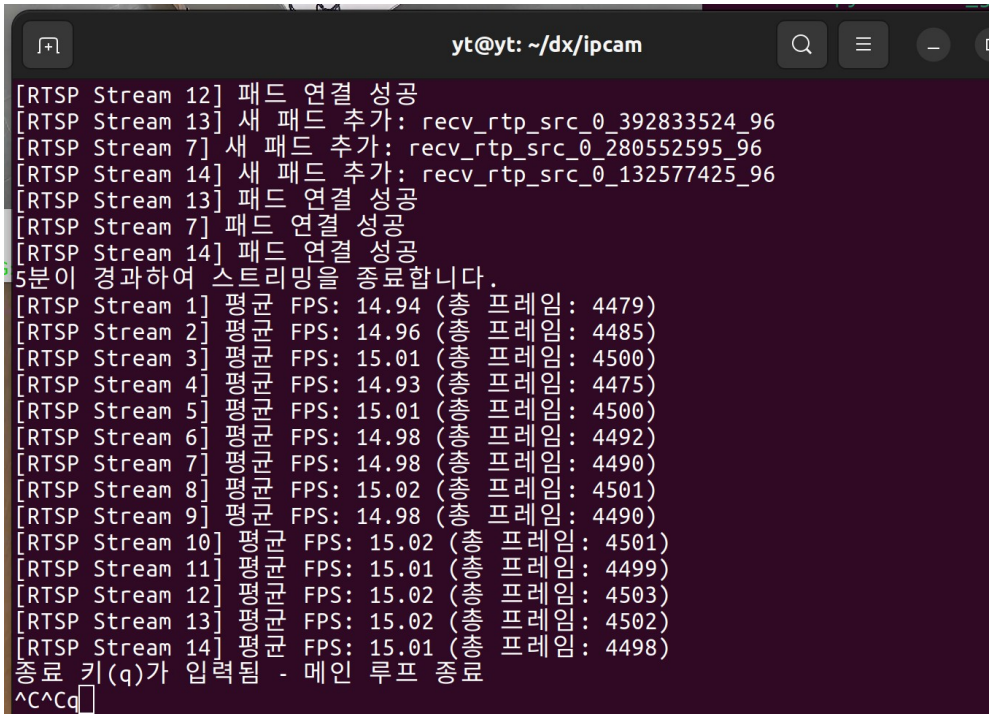
21 개 시도 시에 메모리 초과로 인한 process killed

```
[RTSP Stream 18] 디페이로더의 sink pad가 이미 링크되어 있음
[RTSP Stream 19] 패드 연결 성공
[RTSP Stream 20] 새 패드 추가: recv_rtp_src_0_360204915_96
[RTSP Stream 14] 디페이로더의 sink pad가 이미 링크되어 있음
[RTSP Stream 10] 새 패드 추가: recv_rtp_src_0_2007821588_96
[RTSP Stream 13] 패드 연결 성공
[RTSP Stream 20] 패드 연결 성공
[RTSP Stream 16] 새 패드 추가: recv_rtp_src_0_901496664_96
[RTSP Stream 21] 패드 연결 성공
[RTSP Stream 10] 패드 연결 성공
[RTSP Stream 16] 패드 연결 성공
[RTSP Stream 21] 디페이로더의 sink pad가 이미 링크되어 있음
Killed
(as1) yt@yt:~/dx/ipcam$
```

그래서 14 개로만 비교분석 진행함


RTSP Stream 5
RTSP Stream 11
RTSP Stream 14
RTSP Stream 13


RTSP Stream 6
RTSP Stream 1
RTSP Stream 2
RTSP Stream 10
RTSP Stream 9
RTSP Stream 3
RTSP Stream 4
RTSP Stream 8
yt@yt: ~/dx/ipcam
[RTSP Stream 13] 패드 연결 실패: <enum GST_PAD_LINK_NOFORMAT of type Gst.PadLink
[RTSP Stream 12] 새 패드 추가: recv_rtp_src_1_958974743_0
[RTSP Stream 7] 패드 연결 실패: <enum GST_PAD_LINK_NOFORMAT of type Gst.PadLink
[RTSP Stream 14] 패드 연결 실패: <enum GST_PAD_LINK_NOFORMAT of type Gst.PadLink
[RTSP Stream 4] 새 패드 추가: recv_rtp_src_0_943743267_96
[RTSP Stream 12] 패드 연결 실패: <enum GST_PAD_LINK_NOFORMAT of type Gst.PadLink
[RTSP Stream 4] 패드 연결 성공
[RTSP Stream 11] 새 패드 추가: recv_rtp_src_0_1019201719_96
[RTSP Stream 11] 패드 연결 성공
[RTSP Stream 5] 새 패드 추가: recv_rtp_src_0_441453531_96
[RTSP Stream 5] 패드 연결 성공
[RTSP Stream 12] 새 패드 추가: recv_rtp_src_0_958974742_96
[RTSP Stream 12] 패드 연결 성공
[RTSP Stream 13] 새 패드 추가: recv_rtp_src_0_392833524_96
[RTSP Stream 7] 새 패드 추가: recv_rtp_src_0_280552595_96
[RTSP Stream 14] 새 패드 추가: recv_rtp_src_0_132577425_96
[RTSP Stream 13] 패드 연결 성공
[RTSP Stream 7] 패드 연결 성공
[RTSP Stream 14] 패드 연결 성공
yt@yt: ~
run_gstreamer_oop_multi_report.py
(as1) (base)



Conditions

- **버퍼링이 활성화된 상태**
- **약 0.5 초의 최대 지연이 기본값**
- **14 개의 rtsp 커넥션**

Memory 5gb-> 10gb (5gb 증가)

FPS value

	case1	case2	case3
cam1	14.94	15.00	14.99
cam2	14.96	14.97	14.97
cam3	15.01	14.98	14.99
cam4	14.93	14.96	14.97
cam5	15.01	14.97	14.95
cam6	14.98	14.95	14.97
cam7	14.98	14.97	14.97
cam8	15.02	15.01	15.01
cam9	14.98	14.99	15.00
cam10	15.02	15.01	15.01
cam11	15.01	15.01	15.01
cam12	15.02	15.02	15.01
cam13	15.02	15.01	15.02
cam14	15.01	15.01	14.99

3. GStreamer 기능

영상처리 및 분석 플러그인(filter):

블러효과주기(frei0r), 모션감지(motioncells),

객체감지의 경우 Intel OpenVINO 기반의 GStreamer Video Analytics(GVA) 플러그인인 gvadetect

분석 결과나 디버깅 정보를 영상 위에 텍스트로 표시하고 싶을 때(textoverlay)

Ex) 이미지 resize 사용시

```
src        = Gst.ElementFactory.make("rtspsrc", "source")
depay      = Gst.ElementFactory.make("rtph264depay", "depay")
parse      = Gst.ElementFactory.make("h264parse", "parse")
decoder    = Gst.ElementFactory.make("avdec_h264", "decoder")
convert    = Gst.ElementFactory.make("videoconvert", "convert")
scale      = Gst.ElementFactory.make("videoscale", "scale")
capsfilter = Gst.ElementFactory.make("capsfilter", "capsfilter")
sink       = Gst.ElementFactory.make("appsink", "sink")
```

Ex) 하드웨어가속 디코더 사용예시 Nvidia 기반

```
# 요소 생성
src    = Gst.ElementFactory.make("rtspsrc", "source")
depay  = Gst.ElementFactory.make("rtph264depay", "depay")
parse  = Gst.ElementFactory.make("h264parse", "parse")
# 하드웨어 가속 디코더 사용 (예: NVIDIA의 경우 nvdec_h264)
decoder= Gst.ElementFactory.make("nvdec_h264", "decoder")
convert= Gst.ElementFactory.make("videoconvert", "convert")
sink   = Gst.ElementFactory.make("autovideosink", "sink")
```