Gstreamer

웹캠으로 가능한 매개변수를 설치해야하는 간단한 프로그램

sudo apt install v4l-uitls

파이프라인

소스에서 싱크로 이동 한 다음 중간에 작업을 수행한다.

소스가 동기화와 일치하지 않으면 변환을 수행해야 한다.

1. 소스에서 오디오 톤을 지정하고 스피커를 잘 동기화 하는법

느낌표는 한 모듈을 다른 모듈에 연결하는 것.

# 오디오 테스트

gst-launch-1.0 audiotestsrc ! alsasink

# 오디오 테스트에 관한 모듈 같은거 디테일 같은거 볼 수 잇음.

# 소스템플릿으로 가서 확인해봐.

# 싱크는 모듈에 대한 입력과 같다.

# 소스는 출력과 같다.

# 오디오테스트 소스 인 모듈소스

gst-inspect-1.0 audiotestsrc

속성은 설정할 수 잇음.

설정할 수 있는 것들은 name, parent, blokcsize… , wave 등등 다양함.

# 구현파. 소리 종류 설정.

Gst-launch-1.0 audiotestsrc wave=1 ! alsasink

# frequency 도 설정

Gst-launch-1.0 audiotestsrc wave=1 freq=200 ! alsasink

video/x-raw 비디오 원시 파일.

gst-inspect-1.0

오디오 테스트 형식 포맷 U-8

gst-launch-1.0 audiotestsrc wave=0 freq=300 volume=1 ! audio/x-raw, format=U8 ! alsasink

U18e 포맷

gst-launch-1.0 audiotestsrc wave=0 freq=300 volume=1 ! audio/x-raw, format=U18LE ! alsasink

하면 오류뜸

gst-inspect-1.0 audiotestsrc 보면 U18LE 있는데

gst-inspect-1.0 alsasink로 보면 U18LE를 찾을수 없음.

작업을 하려면 오디오 변환을 해야함. 어떻게 초기화 하냐?

변환기 이용햇 사용 audioconvert!

이렇게 하면 됨.

gst-launch-1.0 audiotestsrc wave=0 freq=300 volume=1 ! audio/x-raw, format=U18LE ! audioconvert ! alsasink

Pad Templates에 Capabiltiies format 보면 U18LE 보면 있다.

입력하고 출력하면 기능로서의 전환을 얻을 수 있음.

오디오 동기화는 매우간단하다.

모듈 이름이 무엇인지 확인하고 gst-inspect 해서 보고

입력으로 원하는 것이 무엇인지 확인 싱크에 대한 캡이 무엇인지 확인하고 수행해야한다.

Audio/x-raw로 유지.

gst-launch-1.0 audiotestsrc wave=0 freq=300 volume=1 ! audio/x-raw, format=U18LE ! audioconvert ! audio/x-raw, format=U8 ! alsasink

입력을 출력으로 변환.

비디오 테스트 !!!!!!!

이미지싱크로 파이핑 하고있다.

Gst-launch-1.0 videotestsrc ! ximagesink

간단한 데모를 할 때.

Gst-inspect-1.0 videotestsrc

원형 얻기. pattern

gst-launch-1.0 videotestsrc pattern=11 ! autovideosink

capabilityies 로 가서 video/x-raw format이 굉장히 많음

gst-launch-1.0 videotestsrc pattern=11 ! autovideosink

이미지 동기화가 안됬어 이미지 동기화가 BGR을 좋아하지 않음.

gst-launch-1.0 videotestsrc pattern=0 ! video/x-raw,format=BGR ! autovideosink

동기화 검사를 해보자

Gst-inspect-1.0 ximagesink

비디오 변환을 할 수 있음 우리는

Gst-launch-1.0 videotestsrc pattern=0 ! video/x-raw, format=BGR ! autovideoconvert ! ximagesink

입력 또는 소스가 있고 출력이 있다.

Autovideoconvert는 어떠한 핸들링을 주지 않음

그래서 수동 비디오 변환을 해야함.

Gst-launch-1.0 videotestsrc pattern=0 ! video/x-raw, format=BGR ! autovideoconvert ! videoconvert ! video/x-raw, width=1280, height=960 ! ximagesink

모든 출력이 모든 소스와 잘 어울리도록 어댑터를 계속 해줘야함.

또 다른 매개변수 테스트

프레임 속도도 변경 가능.

Gst-inspect-1.0 videoconvert

Gst-launch-1.0 videotestsrc pattern=0 ! video/x-raw, format=BGR ! autovideoconvert ! videoconvert ! video/x-raw, width=1280, height=960, framerate=30/1 ! ximagesink

초당 프레임속도 1초당 30

카메라라 프레임 속도를 높일 수 ㄴ없ㅇㅁ

1프레임

Gst-launch-1.0 videotestsrc pattern=0 ! video/x-raw, format=BGR ! autovideoconvert ! videoconvert ! video/x-raw, width=1280, height=960, framerate=1/1 ! ximagesink

2 프레임

Gst-launch-1.0 videotestsrc pattern=0 ! video/x-raw, format=BGR ! autovideoconvert ! videoconvert ! video/x-raw, width=1280, height=960, framerate=2/1 ! ximagesink

라즈베리파이 카메라 쓸 때

Nvidia arguas camera

나는 소스와 동기화를 알고있지만 알아서 컨버트 해줘.

Gst-launch-1.0 nvarguscamerasrc ! autovideoconvert ! ximagesink

젯슨나노 보드에서 포트가 두개있어서 senser-id =0 이렇게 설정해줘야함.

오디오 변환 포트가 필요없을거임. 아마

Nvvidconv 플립한다음에, 캡설정을 하는데1280x 840으로 한다.

640 x 480으로 해보자.

Gst-launch-1.0 nvarguscmaerasrc ! nvvidconv flip-method=2 ! video/x-raw, width=1280, height=840 ! autovideoconvert ! ximagesink

gst-inspect-1.0 nvvidconv 보면

메모리 NVMM 사용해가지고 video/x-raw 출력하는거볼수잇음.

Video/x-raw 속성 보면 format, width, height, framerate 설정 할 수 있음.

Gst-launch-1.0 nvarguscmaerasrc ! nvvidconv flip-method=2 ! video/x-raw, width=1280, height=840, framerate=20/1 ! autovideoconvert ! ximagesink

Gst-launch-1.0 nvargusccamerasrc ! video/x-raw, framerate=21/1 ! nvvidconv flip-method=2 ! video/x-raw, width=1280, height=840, framerate=20/1 ! autovideoconvert ! ximagesink

위에 두개 오류임

카메라에서 제공하는 프레임속도 써야함

문자열로 만들거야

라즈베리파이 팁.

카메라의 모든 작업을 수행하려면

비디오를 3264 픽셀 너비로 출력한다. 또는 2464

바꾸면 되게 안좋음 잘라내서 해야함.

32.64 24.64

Gst-launch-1.0 nvarguscamerasrc ! ‘video/x-raw(memory:NVMM), width=3264, height=2464, framerate=21/1’ ! nvvidconv flip-method=2 ! video/x-raw, width=640, height=480 ! autovideoconvert ! ximagesink

Appsink opencv bgr을 원하므로.

Bgr로 데리고가서 opencv 로 간다.

Python opencv cam set (rasberrypi)

camSet = ‘nvarguscamerasrc sensor-id=0 ! video/x-raw(memory:NVMM) framerate=21/1, format=NV12 ! nvvidconv flip-method=’+str(flip) + ‘ ! video/x-raw, width=’+str(width)+’, height=+str(height)+’, format=BGRx ! videoconvert ! video/x-raw, format=BGR ! appsink’

camSet= ‘v4l2src device=/dev/video1 ! video/x-raw, width=’+str(width)+’, height=+str(height)+’, framerate=24/1 ! videoconvert ! appsink’

# webcam demo

Gst-launch-1.0 v4l2src ! video/x-raw, format=YUY2 ! videoconvert ! ximagesink

Mic demo

Gst-launch openalsrc ! audioconvert ! wavescope style=3 shader=2 ! videoconvert ! autovideosink

# tee demo

Gst-launch-1.0 v4l2src ! video/x-raw, width=320, height=240 ! videoconvert ! tee name =t1 t1. ! queue ! ximagesink t1. ! queue ! videoconvert ! edgedetect ! videoconvert ! ximagesink t1. ! queue ! videoconvert ! motioncells ! videoconvert ! ximagesink t1. ! queue ! videoconvert ! glupload ! gleffects\_sepia ! glimagesink t1. ! queue ! videoconvert ! glupload ! gleffects\_xrat ! glimagesink

# UDP receiver

Gst-launch-1.0 udpsrc port=10000 ! application/x-rtp, encoding-name=H264 ! rtpjitterbuffer ! rtph264depay ! avdec\_h264 ! videoconvert ! ximagesink

# UDP transmitter

Gst-launch-1.0 v4l2src ! video/x-raw, width=320, height=240 ! x264enc tune=zerolatency ! rtph264pay ! udpsink port=10000

gst-launch-1.0 -v v4l2src ! video/x-raw-yuv, framerate=30/1 ! timeoverlay halign=left valign=bottom text=”Stream time:” shaded-background=true ! clockoverlay halign=right valign=bottom shaded-background=true time-format=”%Y/%m/%d %H:%M:%S” ! ffmpegcolorspace ! x264enc tune=”zerolatency” bitrate=256 noise-reduction=10000 ! rtph264pay ! udpsink port=50000 host=127.0.0.1

gst-launch-1.0 udpsrc port=50000 caps=”application/x-rtp,media(string)video, clock-rate=(int)90000, encoding-name=(string)H264m payload=(int)96” ! gstrtpjitterbuffer ! rtph264depay ! ffdec\_h264 ! ffmpegcolorspace ! ximagesink sync=false

opencv 사용하여 비디오에서 이미지 추출

udp 기반 소켓 프로그래밍

서버 및 클라이언트

Server.py

*# This is server code to send video frames over UDP*

**import** cv2, imutils, socket

**import** numpy **as** np

**import** time

**import** base64

BUFF\_SIZE **=** 65536

server\_socket **=** socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM)

server\_socket.setsockopt(socket.SOL\_SOCKET,socket.SO\_RCVBUF,BUFF\_SIZE)

host\_name **=** socket.gethostname()

host\_ip **=** '192.168.1.102'*# socket.gethostbyname(host\_name)*

**print**(host\_ip)

port **=** 9999

socket\_address **=** (host\_ip,port)

server\_socket.bind(socket\_address)

**print**('Listening at:',socket\_address)

vid **=** cv2.VideoCapture(0) *# replace 'rocket.mp4' with 0 for webcam*

fps,st,frames\_to\_count,cnt **=** (0,0,20,0)

**while** True:

msg,client\_addr **=** server\_socket.recvfrom(BUFF\_SIZE)

**print**('GOT connection from ',client\_addr)

WIDTH**=**400

**while**(vid.isOpened()):

\_,frame **=** vid.read()

frame **=** imutils.resize(frame,width**=**WIDTH)

encoded,buffer **=** cv2.imencode('.jpg',frame,[cv2.IMWRITE\_JPEG\_QUALITY,80])

message **=** base64.b64encode(buffer)

server\_socket.sendto(message,client\_addr)

frame **=** cv2.putText(frame,'FPS: '**+**str(fps),(10,40),cv2.FONT\_HERSHEY\_SIMPLEX,0.7,(0,0,255),2)

cv2.imshow('TRANSMITTING VIDEO',frame)

key **=** cv2.waitKey(1) **&** 0xFF

**if** key **==** ord('q'):

server\_socket.close()

**break**

**if** cnt **==** frames\_to\_count:

**try**:

fps **=** round(frames\_to\_count**/**(time.time()**-**st))

st**=**time.time()

cnt**=**0

**except**:

**pass**

cnt**+=**1

Client.py

*# This is client code to receive video frames over UDP*

**import** cv2, imutils, socket

**import** numpy **as** np

**import** time

**import** base64

BUFF\_SIZE **=** 65536

client\_socket **=** socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM)

client\_socket.setsockopt(socket.SOL\_SOCKET,socket.SO\_RCVBUF,BUFF\_SIZE)

host\_name **=** socket.gethostname()

host\_ip **=** '192.168.1.102'*# socket.gethostbyname(host\_name)*

**print**(host\_ip)

port **=** 9999

message **=** b'Hello'

client\_socket.sendto(message,(host\_ip,port))

fps,st,frames\_to\_count,cnt **=** (0,0,20,0)

**while** True:

packet,\_ **=** client\_socket.recvfrom(BUFF\_SIZE)

data **=** base64.b64decode(packet,' /')

npdata **=** np.fromstring(data,dtype**=**np.uint8)

frame **=** cv2.imdecode(npdata,1)

frame **=** cv2.putText(frame,'FPS: '**+**str(fps),(10,40),cv2.FONT\_HERSHEY\_SIMPLEX,0.7,(0,0,255),2)

cv2.imshow("RECEIVING VIDEO",frame)

key **=** cv2.waitKey(1) **&** 0xFF

**if** key **==** ord('q'):

client\_socket.close()

**break**

**if** cnt **==** frames\_to\_count:

**try**:

fps **=** round(frames\_to\_count**/**(time.time()**-**st))

st**=**time.time()

cnt**=**0

**except**:

**pass**

cnt**+=**1

sudo sysctl -w net.inet.udp.maxdgram=65535

**Gst-launch-1.0 -v vl2src device/dev/video0 ! “image/jpeg, width=320, height=240, framerate=30/1” ! rtpjpegpay ! multiudpsink clients= ip 주소**

**스트리밍**

**멀티미디어 파일을 다운로드함과 동시에 실행하는 방법 또는 기술**

**영상재생에 필요한 만큼 다운로드 받아지면 계속 추가함.**

**규약 혹은 통신이 필요하게 되는데**

**스트리밍 프로토콜이라함.**

**대표적 스트리밍 프로토콜**

**MPEG의 MPEG-DASH**

**Apple의 HTTP LIVE Streaming**

**HLS은 뭐냐**

**미디어 컨테이너 포맷: .mp2ts**

**매니페스트 파일 : .m3u8**

**1. H.264 동영상 파일을 10초 정도의 작은 단위로분할**

**2. 분할된 파일의 재생 순서가 작서되 ㄴ매니페스트 파일 생성.**

**3. HLS를 지원하는 브라우저나 애플리케이션에서 매니페스트 파일을 읽어 재생 시작.**

**재생은 매니페스트 파일에 작성돼 있는 분할 된 동영상 파일을 순차적으로 읽어드리며 진행**

**동영상 도중부터 재생할경우 해당하는 시간의 동영상 파일을 읽어냄.**

**HTTP 서버 실행**

**Python -m SimpleHTTPserver**

**멀티 비트레이트에 대해서 알아보자**

**비디오 플레어에서 자주 볼 수 있는 화질 선택 기능과 밀접한 관련이 있음.**

**유튜브 실시간 스트림의 해상도 및 비트 전송률**

**Text, table

Description automatically generated with medium confidence**

**Graphical user interface, text

Description automatically generated**

**1400 kbps 이하면 852 480 으로 480p**

**2800 kbps 이하면 1280 720 으로 720p**

**5000 kbps 이하면 1920 1080으로 1080p**

**Text

Description automatically generated**

**Mpeg-dash**

**매니페스트 파일 -> xml 로 .**

**MSE 로 만들어서 할 수 있음.**

**Diagram

Description automatically generated**

**이걸로한다고 하면**

**영상을 계속 스트리밍을 하고 로컬에 웹 계속**

**스트리밍을 하면서**

**3초단위로 mp4**

**동여상을.**

**1,2,3,4,5,6,7,8**

**스트리밍 파트랑 / 영상 확인 부분.**

**영상 확인 싸운게 있엇다.**

**라벨링해서 뜬다는거 보여줘야되자나.**

**스트리밍 클라이언트 -> 웹에다 스트리밍,.**

**모델 3초단위 mp4 파일.**

**그 영상을 3번파일에**

**검출 영상에**

**3번 파일을 업로드하는?**

**실해시켯을때 바운딩 박스 이렇게 쳐지느?**

**스트리밍 하는 영상만 보이고**

**검출이되면서 스트리밍을 한다는게 너무 어려울거같고.**

**스트리밍 / 우리가 시간을 확인하는거 .**

**동영상 검출이되면 바로 알림을 보내는 핸드폰 알림보내는.**

gst-launch-1.0 rtspsrc location="$RTSP\_PATH" latency=500 ! rtph264depay ! h264parse ! nvv4l2decoder <b>enable-max-performance=1</b> ! nvoverlaysink overlay-x=800 overlay-y=50 overlay-w=640 overlay-h=480 overlay=2

$ gst-launch-1.0 rtspsrc location="$RTSP\_PATH" latency=500 ! rtph264depay ! h264parse ! omxh264dec ! nvoverlaysink overlay-x=800 overlay-y=50 overlay-w=640 overlay-h=480 overlay=2