RTSP-OpenCV-Docker 카메라 연동

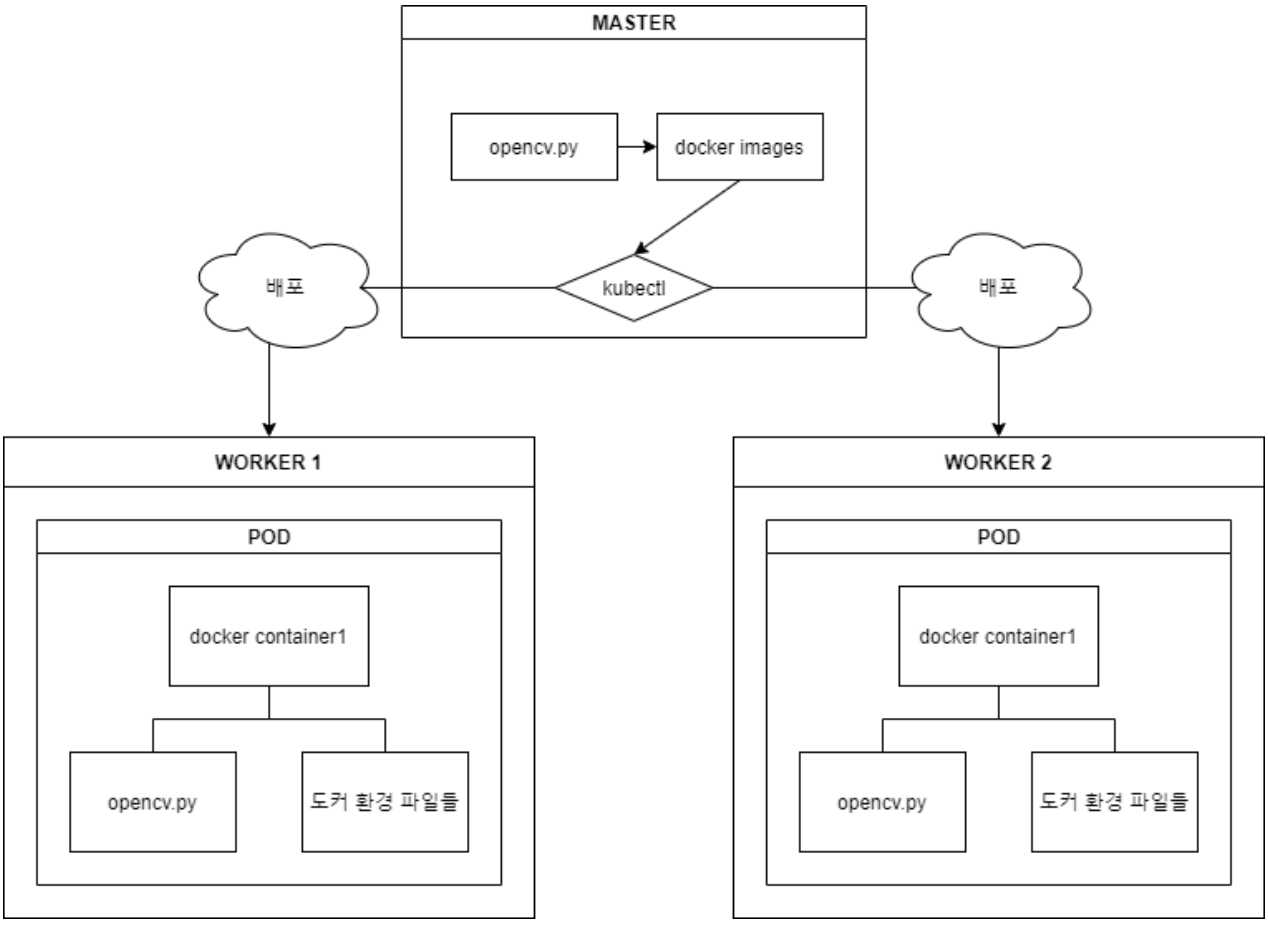
- RTSP 실시간으로 전달되는 웹캠의 데이터를 OpenCV로 실행되게 하고 Docker 컨테이너에 정보를 담아 사용한다
- 추후에 k8s 를 사용하여 worker 에 배포

Spec

rtsp address

- rtsp : // keti : keti1234@192.168.100.60 : 8805 / videoMain -> opencv1.py
- rtsp : // keti : keti1234@192.168.100.70 : 8810 / videoMain -> opencv2.py

구조



진행과정

opencv-python 파일

- 기능 : cv2를 import 하여 해당 rtsp url 을 가지고 실시간 카메라 영상을 보여줌

1. opencv1.py

```
import cv2

url = 'rtsp://keti:keti1234@192.168.100.70:8810/videoMain'
cap = cv2.VideoCapture(url)

while True:
    # Image read
    ret, image = cap.read()
    # image show
    cv2.imshow('stream', image)
    # q 키를 누르면 종료
    if cv2.waitKey(5) & 0xFF == ord('q'):
        break

cv2.destroyAllWindows()
```

2. opencv2.py

```
import cv2

url = 'rtsp://keti:keti1234@192.168.100.60:8805/videoMain'
cap = cv2.VideoCapture(url)

while True:
    # Image read
    ret, image = cap.read()
    # image show
    cv2.imshow('stream', image)
    # q 키를 누르면 종료
    if cv2.waitKey(5) & 0xFF == ord('q'):
        break

cv2.destroyAllWindows()
```

OpenCV-python 파일을 도커라이징

Dockerfile

```
FROM python:3.7
MAINTAINER Josip Janzic <josip@jjanzic.com>
```

```
RUN apt-get update \
    && apt-get install -y \
        build-essential \
        cmake \
        git \
        wget \
        unzip \
        yasm \
        pkg-config \
        libswscale-dev \
        libtbb2 \
        libtbb-dev \
        libjpeg-dev \
        libpng-dev \
        libtiff-dev \
        libavformat-dev \
        libpq-dev \
    && rm -rf /var/lib/apt/lists/*

RUN pip install numpy

WORKDIR /
ADD . /workspace # 로컬의 현재 디렉터리를 workspace 라는 이름의 디렉터리로 복사
ENV OPENCV_VERSION="4.1.0" # 4.1.1 버전이 오류나서 변경한 부분(하지만 아직 오류)
RUN wget https://github.com/opencv/opencv/archive/${OPENCV_VERSION}.zip \
&& unzip ${OPENCV_VERSION}.zip \
&& mkdir /opencv-${OPENCV_VERSION}/cmake_binary \
&& cd /opencv-${OPENCV_VERSION}/cmake_binary \
&& cmake -DBUILD_TIFF=ON \
  -DBUILD_opencv_java=OFF \
  -DWITH_CUDA=OFF \
  -DWITH_OPENGL=ON \
  -DWITH_OPENCL=ON \
  -DWITH_IPP=ON \
  -DWITH_TBB=ON \
  -DWITH_EIGEN=ON \
  -DWITH_V4L=ON \
  -DBUILD_TESTS=OFF \
  -DBUILD_PERF_TESTS=OFF \
  -DCMAKE_BUILD_TYPE=RELEASE \
  -DCMAKE_INSTALL_PREFIX=$(python3.7 -c "import sys; print(sys.prefix)") \
  -DPYTHON_EXECUTABLE=$(which python3.7) \
  -DPYTHON_INCLUDE_DIR=$(python3.7 -c "from distutils.sysconfig import
get_python_inc; print(get_python_inc())") \
  -DPYTHON_PACKAGES_PATH=$(python3.7 -c "from distutils.sysconfig import
get_python_lib; print(get_python_lib())") \
  .. \
&& make install \
&& rm /${OPENCV_VERSION}.zip \
```

```
&& rm -r /opencv-${OPENCV_VERSION}
RUN ln -s \
  /usr/local/python/cv2/python-3.7/cv2.cpython-37m-x86_64-linux-gnu.so \
  /usr/local/lib/python3.7/site-packages/cv2.so
EXPOSE 5001 # 혹시 포트를 사용하게될까바 5001로 컨테이너 포트번호 지정
CMD ["python3", "workspace/opencv1.py"] # 시작하자마자 workspace 에 있는 opencv1.py 실행
```

도커 라이징 - 이미지 만들기

```
$ docker build -f Dockerfile -t sehooh5/opencd-python:latest .
```

도커 이미지 잘 실행되나 확인

```
$  docker run --privileged -it --env DISPLAY=$DISPLAY --
env="QT_X11_NO_MITSHM=1" -v /dev/video0:/dev/video0 -v /tmp/.X11-
unix:/tmp/.X11-unix:ro -p 5002:5001 sehooh5/opencv-python5
```

- 여기서 바로 실행하게되면 오류가 뜬다..위 명령어 + bash 로 들어가서 잘 구성되었는지 확인해준다
- 오류 해결 방법 : 도커 내에서 opencv-python 파일을 다시 받아주면 실행된다

```
 # pip install opencv-python
```

- 오류 해결 후 python 파일 실행해보기

```
#  cd workspace \
#  && python3 opencv1.py
```

도커 이미지를 배포하기 위한 deployment 작성

```
apiVersion: v1
kind: Service
metadata:
  name: opencv-python-service5
spec:
  selector:
    app: opencv-python5
  ports:
  - protocol: "TCP"
    port: 6002
    targetPort: 5002
  type: LoadBalancer
```

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: opencv-python5
spec:
  selector:
    matchLabels:
      app: opencv-python5
  replicas: 3
  template:
    metadata:
      labels:
        app: opencv-python5
    spec:
      containers:
      - name: opencv-python5
        image: sehooh5/opencv-python5:latest
        imagePullPolicy: Always
        command: ['sh', '-c', 'echo 'xxx && sleep 6000']
        ports:
        - containerPort: 5002
```

- 첫번째 단락은 웹 통신을 위한 Service(여기서는 LodaBalancer 타입)
- 두번째 단락은 Deployment 로 데몬셋 형태로 3개의 Pod 을 배포한다(replicas : 3)
- imagePullPoicy 를 Alway로 해줘야 CrashLoopBackOff 에러가 발생하지 않는다. 또한, sleep 명령으로 잠시 쉬게끔 해주어야 한다
- 배포 후 잘 배포되었는지 워커노드가서 container 확인 혹은 마스터에서 pod 확인

Error 및 질의

1. 도커 컨테이너를 실행할 때 자동으로 opencv1.py 파일을 실행하게 했는데 파일 을 실행하는 과정에서 cv2.error 가 발생한다 - 에러슈팅 위에 있음
2. 이렇게 불러온 실시간 영상을 어떻게 활용할 수 있을지?

참고자료

OpenCV를 Docker 에서 활용

- 잘 다룬 예제 및 참고 페이지 : opencv-python Docker image 참고 문서가 있고, 도커 컨테이너 내에서 cv2 실행시 일어나는 오류에 대한 수정이 있음
- 위 예제 활용한 페이지
- Docker hub 에 다른사람들도 참고한 리포 : 위 예제들에서 사용한 DockerHub image
- OpenCV 공식문서

외부에서 Pod의 웹 서비스에 접근하는 방법

- https://developer.ibm.com/kr/cloud/container/2019/03/05/kubernetes_practice/
- **영상 띄울수 있는 컨테이너를 만들고 ServiceType을 NodePort 로 하면 되지않을까?**

Ingress

- 클러스터 내의 서비스에 대한 외부 접근을 관리하는 API 오브젝트
- 일반적으로 HTTP를 관리

OpenStack 사용

- 영어 자료 참고

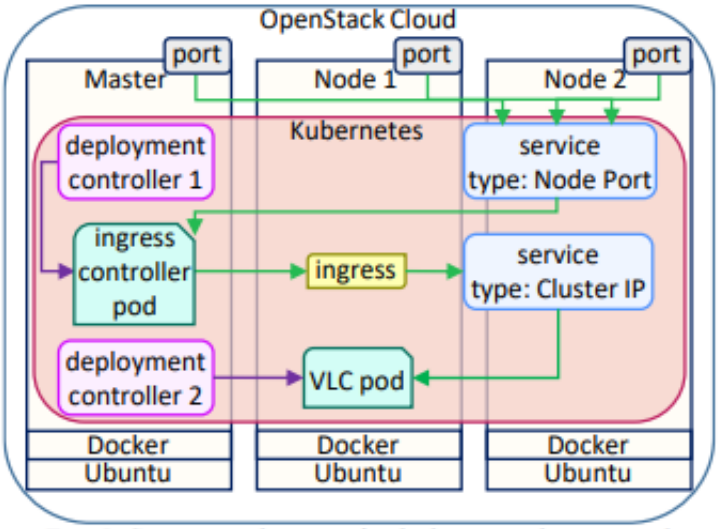


Fig. 5. Concrete architecture for deploying applications with Kubernetes - No-Redundancy redundancy model.

지금 파이썬 배포에 활용하고 있는 사이트

아래내용

1.1 쿠버네티스 프록시를 localhost로 돌리고 API 서버에 접근하는 방법 (kube proxy)

쿠버네티스 라이브러리를 사용하는 wrapper 애플리케이션을 Master 노드의 로컬에 둔 뒤, 이 애플리케이션이 localhost로 접근하면 쿠버네티스 클러스터를 제어할 수 있다.

공식문서 파이썬 배포

나중 flask 사용할 시 참고

실제 OpenCV, RSTP, DOCKER 사용된 프로젝트

웹캠-도커 연결
