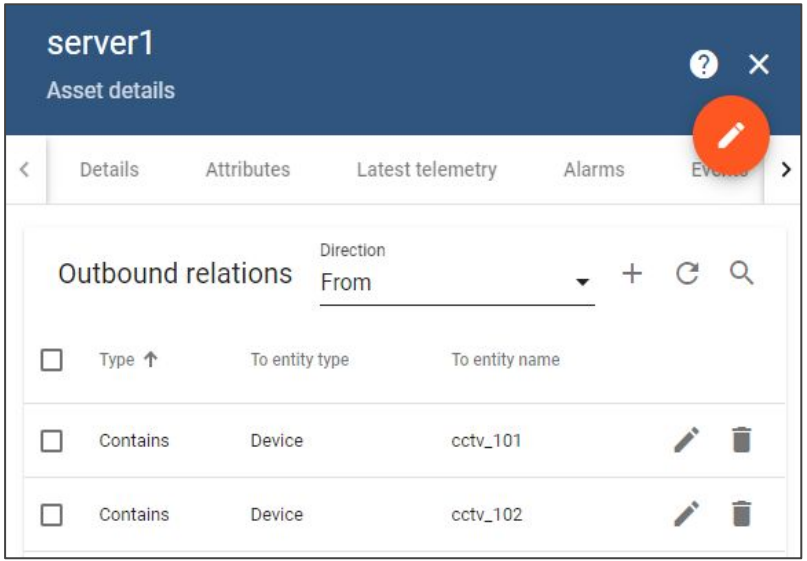
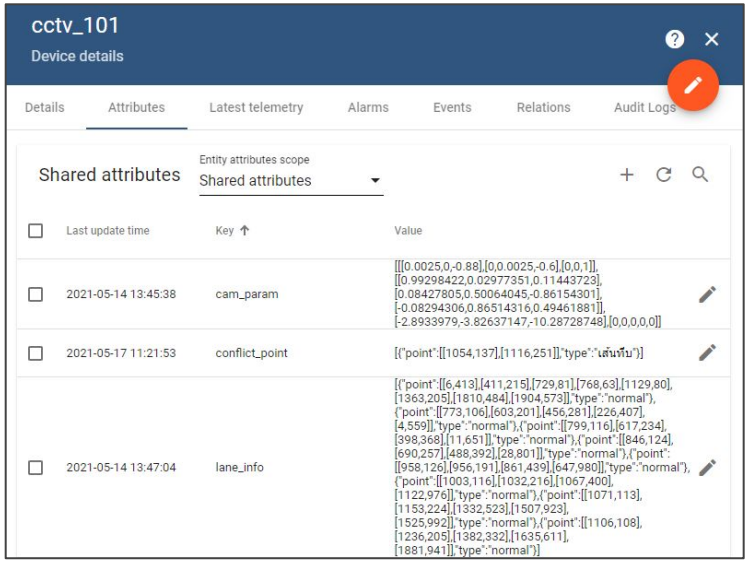
Traffic Analysis System

Progress Report: 21/05/2021

System Configuration

1. Thingsboard Relationship and Attributes





System Configuration

2. Local configuration (config.ini)

config.ini

opt > igi > bin > config.ini

```
[sys]
python = /root/miniconda3/bin/python
# number of gpu
num_gpu = 2
# number of division per gpu
div_gpu = 2

[thingsboard]
asset = server1
url = http://iot.sappasing.com
username = tenant@thingsboard.org
password = tenant
config_update = 5
mqtt_host = 172.23.0.87
mqtt_port = 1883
```

System Configuration

3. Database: CCTV Table

Column	Description
id	ID of CCTV (Integer)
device_id	Thingsboard: Device ID
name	Thingsboard: Name of CCTV
access_token	Thingsboard: Access Token
attributes	Thingsboard: CCTV shared attributes
last_update	Attributes last updated
active	Ready for processing (with valid attributes) Requirement: 'cam_param', 'lane_info', 'speed_marker', 'rtsp_src'

Components

1) **Config Updater (config_update.py)**

   Periodically update CCTV config from Thingsboard to DB

2) **CCTV Processor (cctv_processor.py)**

   Continuously process CCTV data

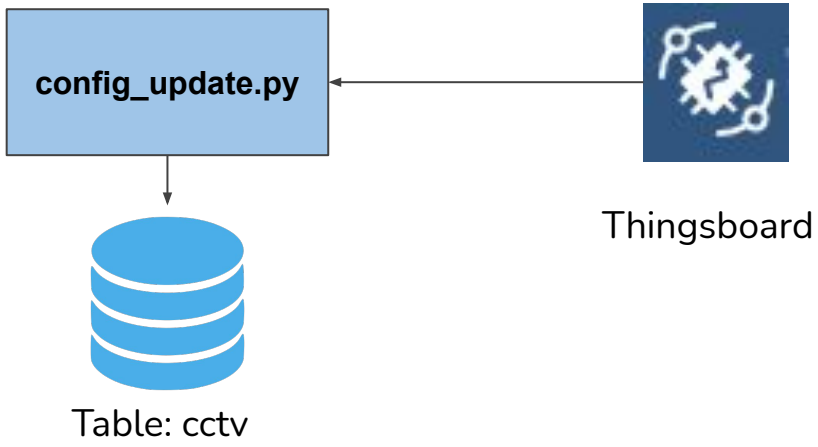
3) **Path Monitor (monitor.py)**

   Watch and process files in the path

Type of Processing

System processing	Per CCTV processing	Concurrent processing
Config Updater	CCTV Processor ● rtsp_download ● rtsp_process ● track ● mod_send_lanespeed ● mod_serve_lanespeed	Path Monitor

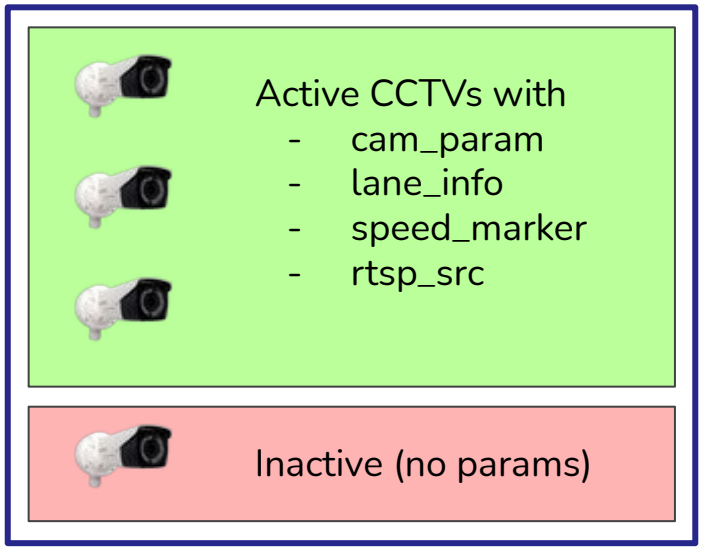
Config Updater



config_update.py

Update CCTVs config from Thingsboard

Devices of ASSET: server_name



CCTV Processor

Table: cctv

cctv_processor.py

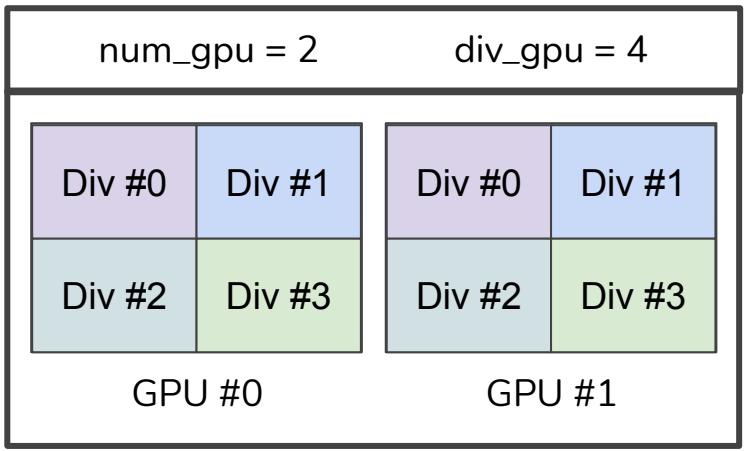
Active: cctv_id

- rtsp_download.py
- rtsp_process.py
- track.py
- mod_send_lanespeed.py
- mod_serve_lanespeed.py

cctv_processor.py

Control processes for each CCTV

Path Monitor



monitor.py

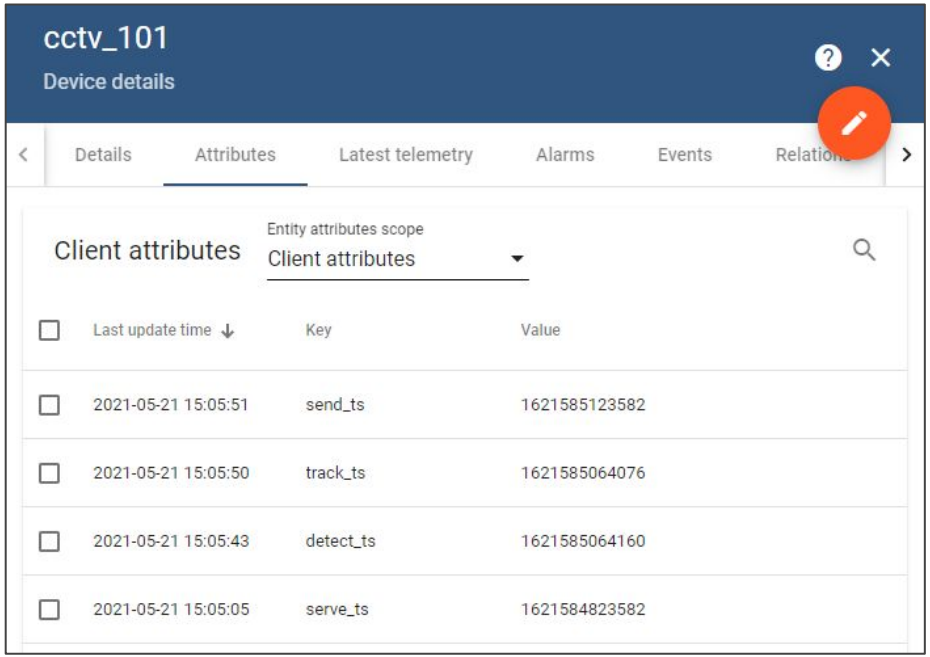
Watch and process files in path

```
monitor.py  <path>
    ├─ monitor.py  <path> <gpu#> <div#>
    ├─ monitor.py  <path> <gpu#> <div#>
    │  ...
    └─ monitor.py  <path> <gpu#> <div#>
```

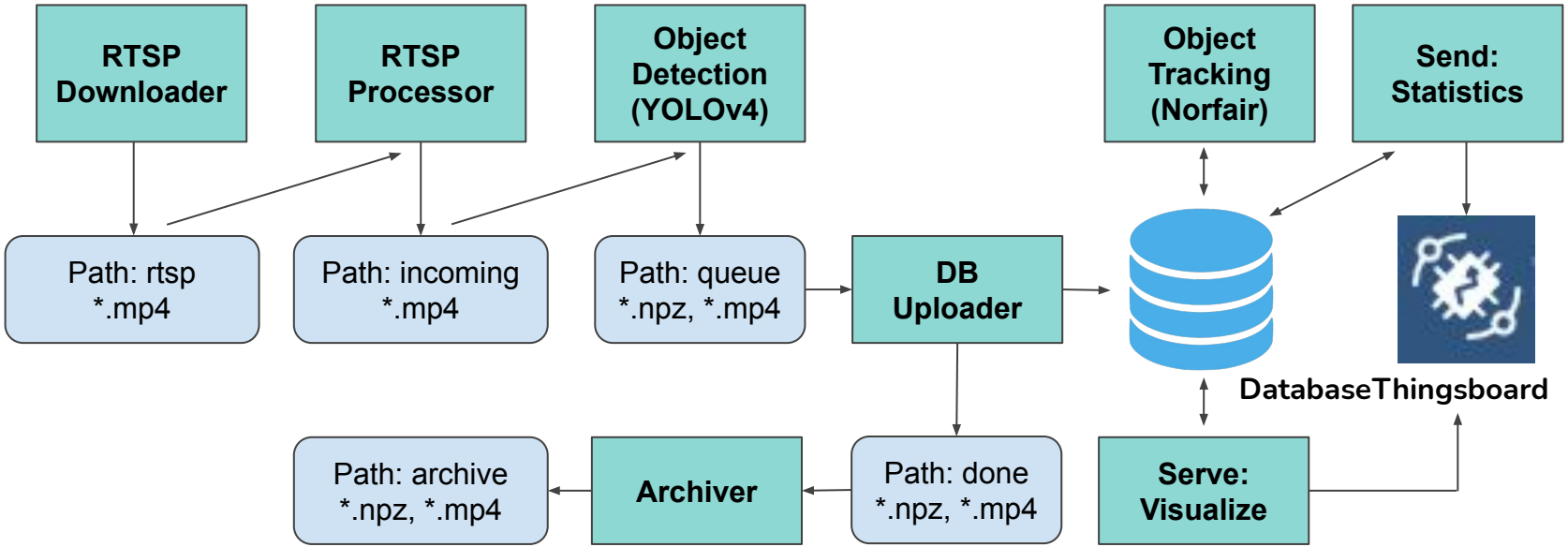
Input Path	Operation	Output Path
incoming	detect.py	queue
queue	upload.py	done
done	archive.py	archive

CCTV Processing Status

Process	Key
RTSP Download / Process	rtsp_ts
Detect / Upload	detect_ts
Track	track_ts
Send Statistic	send_ts
Serve Visualization	serve_ts



Processing Workflow



TODOs

- PyTorch NMS across the classes (detect.py)
- Workflow: LOS Analysis, Reports
- Archiver (archive.py): path [done ➡ archive]
- Frame to TS Visualization (Numpy npz: frame no ➡ ts | get_sources)

frame	timestamp
#	###########
#	###########

- Fix milliseconds mismatch in FFmpeg segment (detect.py / upload.py)