Dragon Pipeline Technology and APIs Reference V1.0

By:
Xi Sun nathansun@uvic.ca
Jingtong Zhang zhang.jingto@northeastern.edu
Ting Liu yanaltingliu@gmail.com

Supervisor:
Yvonne Coady
Derek Jacob

2022-04-05

Environment

neweds

- IP: 206.12.92.18
- SSH port: 10082
- Server Port: 10085

Postgres

- IP: 206.12.92.18
- Port: 5432
- Username: propval
- Password: BCParks

File Storage (neweds)

HTTP File Server

- Path: http://206.12.92.18:10083/
- Mapping Path: /mnt/data/ (on neweds server)

Lidar Files

- Path: /mnt/data/lidar/
    - montreal
        - number: 684
        - format: .laz
    - northvancouver
        - number: 24
        - format: .laz
    - vancouver(not in use)
        - number: 181
        - format: .las

3Dtile Files

- Path: /mnt/data/3DTiles/

Postgres Table

lidar

This table contains the Vancouver and Montreal information about their point cloud datasets, where

- “name” stands for the name of laz file in the corresponding folder
- “area” specify the name of area that should be either “montreal” or “northvancouver” for now
- “geo_polygon” is the geography type data representing a polygon of the laz.

Moreover, this table is the merge of the lidar_montreal and lidar_north_va below.

```
CREATE TABLE IF NOT EXISTS public.lidar
(
    name character varying COLLATE pg_catalog."default",
    area character varying COLLATE pg_catalog."default",
    geo_polygon geography
)
```

lidar_montreal

```
CREATE TABLE IF NOT EXISTS public.lidar_montreal
(
    name character varying COLLATE pg_catalog."default",
    geo_polygon geometry
)
```

lidar_va (not in use now)

```
CREATE TABLE IF NOT EXISTS public.lidar_va
(
    name character varying(50) COLLATE pg_catalog."default",
    lidar_url text COLLATE pg_catalog."default",
    geo_polygon geometry,
    geo_point_2d geometry
)
```

lidar_north_va

```
CREATE TABLE IF NOT EXISTS public.lidar_north_va
(
```

```
    name character varying COLLATE pg_catalog."default",
    geo_polygon geometry
)
```

tiles_3d

“name” stands for the folder name under /mnt/data/3DTiles/
“geo_polygon” stands for the corresponding polygon extracted from the tileset.json file under the name folder.

```
CREATE TABLE IF NOT EXISTS public.tiles_3d
(
    name character varying COLLATE pg_catalog."default",
    geo_polygon geometry
)
```

APIs

Polygon Intersection (Lidar)

- Request Type: POST
- Content Type: application/json
- Request Route: `http://206.12.92.18:10085/lidar/polygon`
- Request Body Example:

```
{
    "vertices": [
        [
            -73.6508721117234,
            45.454839234446005
        ],
        [
            -73.65089611244841,
            45.463837704177784
        ],
        [
            -73.63810846779934,
            45.46385388133002
        ],
        [
            -73.63808650095535,
            45.454855406551125
        ],
```

```
        [
            -73.6508721117234,
            45.454839234446005
        ]
    ]
}
```

- Response Example:

```
{
    "regions": [
        "http://206.12.92.18:10083/lidar/montreal/293-5035_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/292-5036_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/293-5034_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/292-5034_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/294-5036_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/292-5035_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/294-5035_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/294-5034_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/293-5036_2015.laz"
    ],
    "total_num": 9
}
```

- Status Code:
    1. 415 when content type is not json
    2. 422 when the number of vertices is less than 4 or the first vertice is not the same as last one
    3. 400 when format of the json is not as expected
    4. 500 when SQL execution error

Polygon Intersection (3D tiles)

- Request Type: POST
- Content Type: application/json
- Request Route: `http://206.12.92.18:10085/3dtiles/polygon`
- Request Body Example:

```
{
    "vertices": [
        [
            -117.99999999999999,
            32
        ],
        [
            -117.99999999999999,
            33
        ],
        [
            -117,
            33
```

```
        ],
        [
            -117,
            32
        ],
        [
            -117.99999999999999,
            32
        ]
    ]
}
```

- Response Example:

```
{
    "regions": [
        "http://206.12.92.18:10083/3DTiles/San_Diego/tileset.json"
    ],
    "total_num": 1
}
```

- Status Code:
    1. 415 when content type is not json
    2. 422 when the number of vertices is less than 4 or the first vertice is not the same as last one
    3. 400 when format of the json is not as expected
    4. 500 when SQL execution error

Circle Intersection (Lidar)

- Request Type: GET
- Request Parameters
    1. longitude
    2. latitude
    3. radius(m)

- Request Example

```
http://206.12.92.18:10085/lidar/circle?longitude=
-73.6508961124484l&latitude=45.463837704177784&radius=1200.3
```

- Response Example:

```
{
    "regions": [
        "http://206.12.92.18:10083/lidar/montreal/293-5035_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/292-5036_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/291-5035_2015.laz",
```

```
        "http://206.12.92.18:10083/lidar/montreal/291-5036_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/292-5037_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/293-5034_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/292-5034_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/294-5036_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/292-5035_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/293-5037_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/294-5035_2015.laz",
        "http://206.12.92.18:10083/lidar/montreal/293-5036_2015.laz"
    ],
    "total_num": 12
}
```


- Status Code:
    1. 422 when parameters is invalid
    2. 500 when fetch database error

Circle Intersection (3D tiles)

- Request Type: GET
- Request Parameters
    1. longitude
    2. latitude
    3. radius(m)
- Request Example

```
http://206.12.92.18:10085/3dtiles/circle?longitude=-117&latitude=33&radius=1000.5
```

- Response Example


```
{
    "regions": [
        "http://206.12.92.18:10083/3DTiles/San_Diego/tileset.json"
    ],
    "total_num": 1
}
```


- Status Code:
    1. 422 when parameters is invalid
    2. 500 when fetch database error

Satellites

- Request Type: GET
- Request Parameters:
    1. time (optional)
       UTC time in the format of “**20220327130722**”
    2. ids (optional)
       divide by “,”
    3. names (optional)
       divide by “,”
       ”ids” and “names” cannot be passed at the same time

- Request Examples:
    1. No parameters passed

       When no parameters passed, the server will return the location information of all the satellites at the query time.

       ```
       http://206.12.92.18:10085/satellites
       ```

       Response


       ```
       {
           "24876": {
               "height": 20102673.743293267,
               "latitude": -4.287945914590169,
               "longitude": 119.22267734387897,
               "name": "GPS BIIR-2  (PRN 13)"
           },
       ..........
           "48859": {
               "height": 20185545.092861727,
               "latitude": 41.06069248269737,
               "longitude": 140.3650382297894,
               "name": "GPS BIII-5  (PRN 11)"
           }
       }
       ```


    2. “time” passed only
       Return the location of all the satellites at the specified time
       ```
       http://206.12.92.18:10085/satellites?time=20220327130722
       ```

       Response


       ```
       {
           "24876": {
       ```


```
        "height": 20183287.188112166,
        "latitude": -28.354274484060266,
        "longitude": 115.36595702728775,
        "name": "GPS BIIR-2  (PRN 13)"
    },
.....
    "26360": {
        "height": 20274164.925928675,
        "latitude": 38.19069356086112,
        "longitude": 95.28364181835775,
        "name": "GPS BIIR-4  (PRN 20)"
    }
}
```

3. “names” passed only
   Return the location information of satellites specified by names at query time

```
http://206.12.92.18:10085/satellites?names=GPS BIIR-4  (PRN 20),GPS BIIR-2  (PRN 13)
```

Response

```
{
    "24876": {
        "height": 20086246.212400284,
        "latitude": 1.8022225691356089,
        "longitude": 119.74287242734019,
        "name": "GPS BIIR-2  (PRN 13)"
    },
    "26360": {
        "height": 20201597.3550637,
        "latitude": 53.956650199353284,
        "longitude": 127.60809295474638,
        "name": "GPS BIIR-4  (PRN 20)"
    }
}
```

4. “ids” passed only
   Return the location information of satellites specified by ids at query time

```
http://206.12.92.18:10085/satellites?ids=27704,28474
```

Response

```
{
    "27704": {
        "height": 19558419.40691366,
        "latitude": -49.752890750849,
        "longitude": -70.15122968820828,
        "name": "GPS BIIR-9  (PRN 21)"
    },
```

```
    "28474": {
        "height": 20612314.066432618,
        "latitude": 34.44134154690087,
        "longitude": 137.34470863544175,
        "name": "GPS BIIR-13 (PRN 02)"
    }
}
```


5. “time” and “ids”/”names” passed together
   Return the location information of satellites specified by ids or names at the specified time
   Here we pass time and ids as an example:

```
http://206.12.92.18:10085/satellites?time=20220327130722&ids=27704,28474
```

   Response


```
{
    "27704": {
        "height": 19794406.76068566,
        "latitude": -50.65827015252906,
        "longitude": -116.66407327532237,
        "name": "GPS BIIR-9  (PRN 21)"
    },
    "28474": {
        "height": 20741527.196331706,
        "latitude": 54.3132465327371,
        "longitude": 110.87597404462657,
        "name": "GPS BIIR-13 (PRN 02)"
    }
}
```