BRISTOL STREET VIEW PROJECT

FACULTY OF ENGINEERING SUMMER INTERNSHIP 2018

CHRIS GORA

(2ND YEAR MENG COMPUTER SCIENCE STUDENT)

DR ANDREW CALWAY

(READER IN COMPUTER VISION)

MOTIVATION FOR THE PROJECT AND THE INTENDED OUTCOME

To create a platform to collect and store thousands of 360 degree images of Bristol.

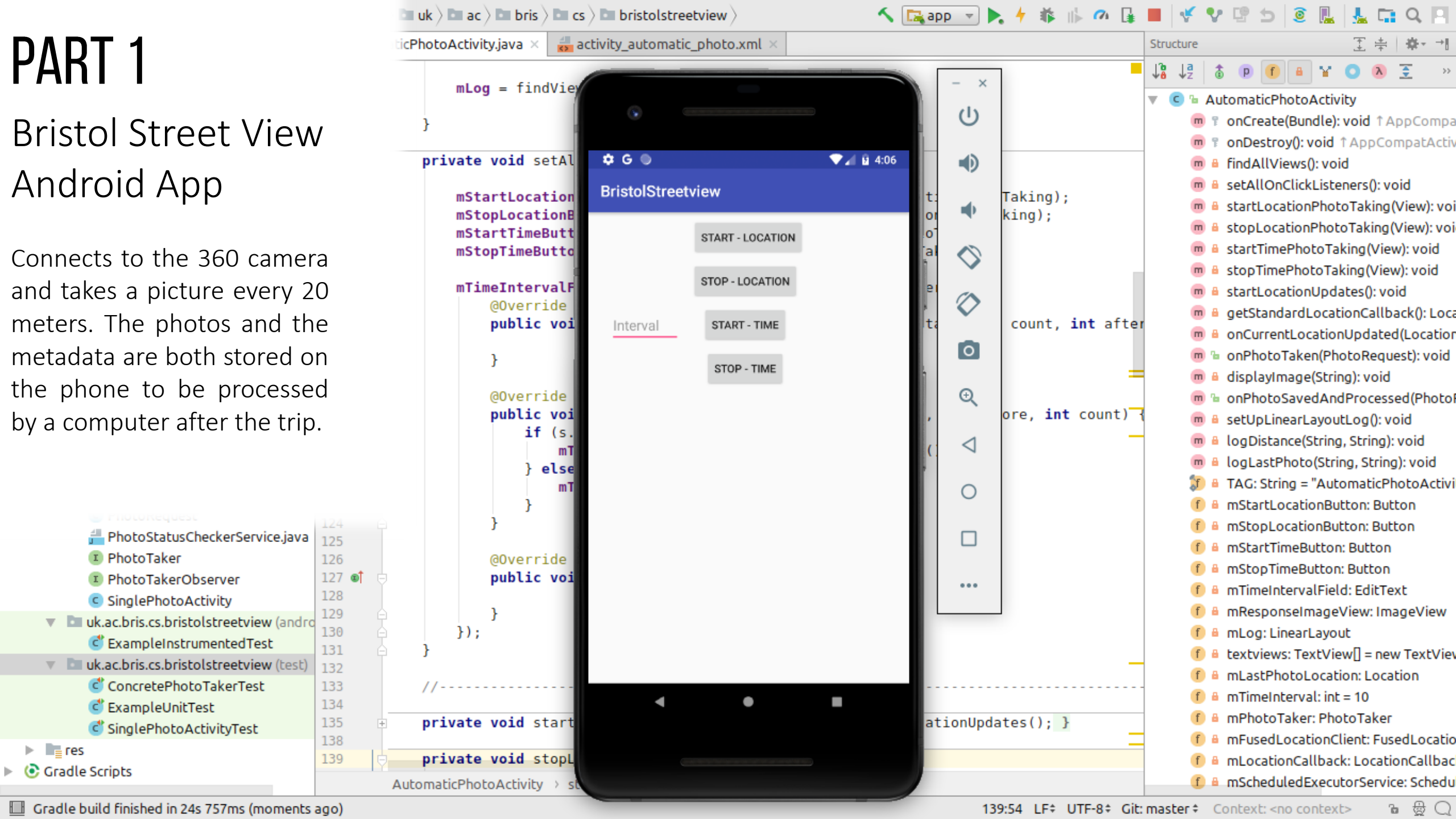
They will then be used to train an algorithm capable of finding the location of each image just by looking at its key features.

The solution consists of 4 key parts…

PART 1

Bristol Street View Android App

Connects to the 360 camera and takes a picture every 20 meters. The photos and the metadata are both stored on the phone to be processed by a computer after the trip.

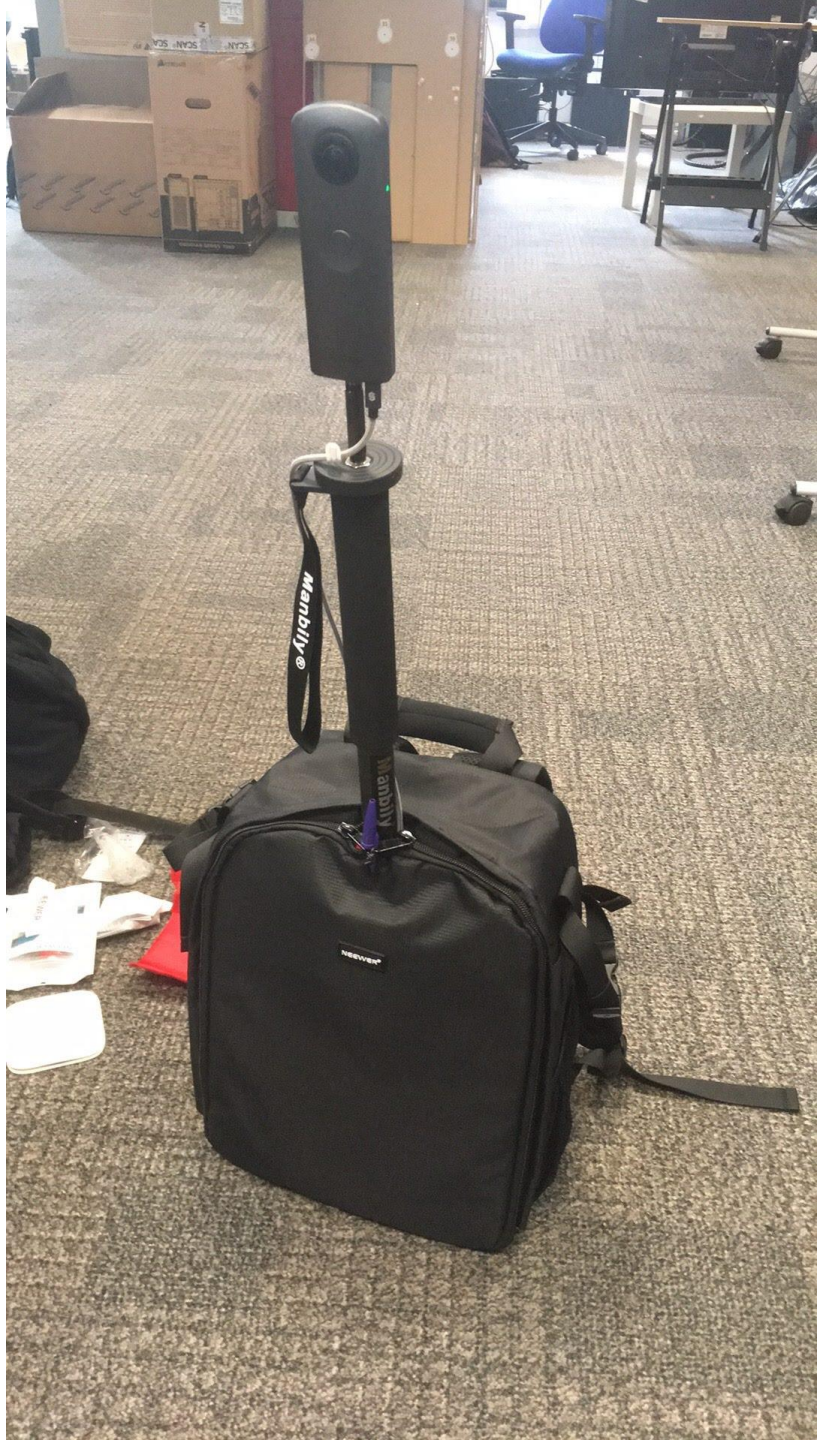


PART 2

The Backpack

Carries the Ricoh Theta V 360 degree camera on a monopod.

Also has a power bank to ensure the camera and the phone don’t run out of battery in the middle of the journey.



PART 3

Desktop Client

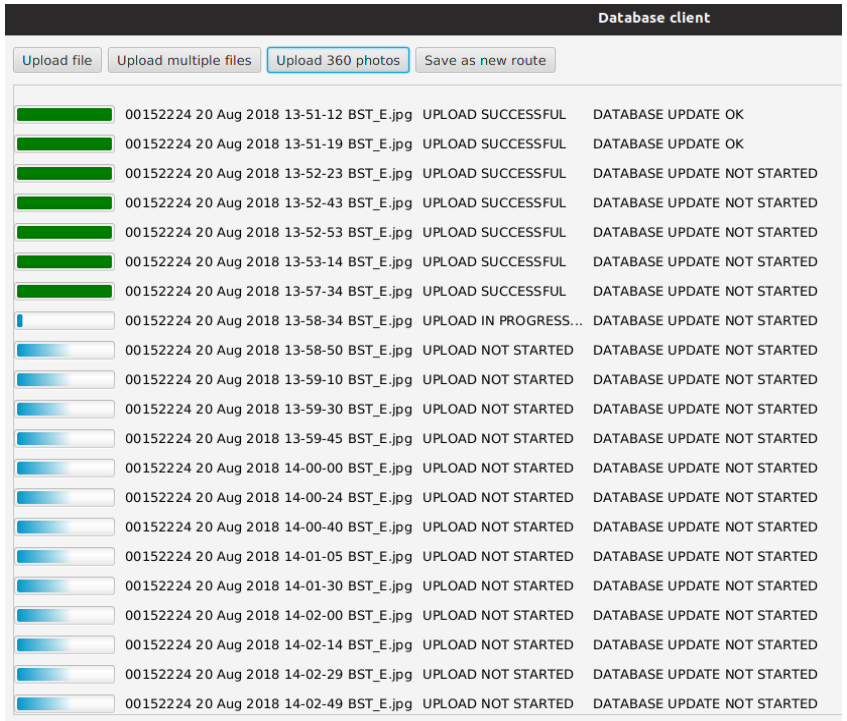
```
BSV DB CLIENT 1.0

Usage: <main class> [-deghsvV] [--deleteAll] [--geo=<geographicSearchRadius>]
                    [--latitude=<latitude>] [--longitude=<longitude>]
                    [--maxGeoResults=<maxGeoResults>] [-r=<route>]
                    [-u=<folderToUpload>] BUCKET [IDS...]
      BUCKET              Folder where the processed images are or will be stored.
      [IDS...]            IDs of files to process (what will be done depends on
                            selected options).
      --deleteAll         Delete all elements from the bucket.
      --geo=<geographicSearchRadius>
                          Conduct a geographic search within the given range (in
                            meters).
      --latitude=<latitude>
                          Set latitude for the geographic search
      --longitude=<longitude>
                          Set longitude for the geographic search
      --maxGeoResults=<maxGeoResults>
                          Maximum number of results allowed from the geographical
                            search. Defaults to 100
  -d, --delete            Delete the photos with the corresponding ids.
  -e, --debug             Debugging output.
  -g, --saveAsGpxAfterUpload
                          Collect all just uploaded files into a GPX file
  -h, --help              Show this help message and exit.
  -r, --route=<route>     Route ID that will be associated with the uploaded pictures.
  -s, --save              Save photos with the corresponding ids to the output
                            directory.
  -u, --upload=<folderToUpload>
                          Upload 360 degree images from the given folder.
  -v, --verbose           Verbose output.
  -V, --version           Print version information and exit.
```

PART 3A

Desktop Client: Uploading

After a journey the desktop client processes all of the stored images. It copies over the files and reads in all available metadata into the database



```
62 DONE: 4caed987eddf4d6984cedd15b831537f-00152224 30 Jul 2018 13-45-47 BST_E.jpg
63 DONE: f20d764425ce49eda4a451ac696e7e0e-00152224 30 Jul 2018 13-28-40 BST_E.jpg
64 DONE: b0cb143881d44baba9b88ff5c536bde8-00152224 30 Jul 2018 13-47-41 BST_E.jpg
65 DONE: 69bd7e6c8da44365b03a6f6c7accd209-00152224 30 Jul 2018 13-26-50 BST_E.jpg
66 DONE: 4277af7ebf94493fbe501635298433b2-00152224 30 Jul 2018 13-42-19 BST_E.jpg
67 DONE: 1b64b189716b4f2d8a691eeecd1ab1e9-00152224 30 Jul 2018 13-43-39 BST_E.jpg
68 DONE: 0367548cbed94277a8c0bb8c1bb80917-00152224 30 Jul 2018 13-46-20 BST_E.jpg
69 DONE: 398dd4cb372a41abbd490ee2db6bbd02-00152224 30 Jul 2018 13-30-13 BST_E.jpg
70 DONE: d76c932cfb8c475ab6b321d4c62f813f-00152224 30 Jul 2018 13-23-33 BST_E.jpg
71 DONE: 83f24456b2094982a098743f96c790c2-00152224 30 Jul 2018 13-25-44 BST_E.jpg
72 DONE: 1b11df2d816948c898704ed6d5e2153c-00152224 30 Jul 2018 13-23-05 BST_E.jpg
73 DONE: 5ea0262fb9b1472ea671c590df9689c2-00152224 30 Jul 2018 13-41-51 BST_E.jpg
74 DONE: 158e4cd08b644cd68661972e5f574b06-00152224 30 Jul 2018 13-46-09 BST_E.jpg
75 DONE: ee0d52d36cea46519192cfd8dfb154c0-00152224 30 Jul 2018 13-26-44 BST_E.jpg
76 DONE: 9ff39002e9124c7e8cd04553bdc2dc6c-00152224 30 Jul 2018 13-37-40 BST_E.jpg
77 DONE: 662e95ece54241b38d8bba0486aeeed9-00152224 30 Jul 2018 13-41-16 BST_E.jpg
78 DONE: 2bc4f32ce2bf4f459a0f6caad6acd47d-00152224 30 Jul 2018 13-38-56 BST_E.jpg
79 DONE: 8f3cf78993ea48dba198a2db54bf0561-00152224 30 Jul 2018 13-23-16 BST_E.jpg
80 DONE: GPX_2_259faebb3f0c4699b89caa90e4fef8b3.gpx
INFO: LocalStorageConnection: RTREE Saved
Processing... 100% |████████████████████████████████████████| 80/80 (0:00:04 / 0:00:00)
```

PART 3B

Desktop Client: Retrieving photos with SQL

Processed images can be retrieved using an SQL query and the java client

```
1   USE bristol_streetview_schema;
2
3   SELECT id FROM (
4
5     SELECT * FROM Photo
6     WHERE bucketName = "bsv"
7     AND routeId = 2
8     ORDER BY photoTimestamp
9
10  ) a;
11
```

Database > @localhost > schemas > bristol_streetview_schema > Photo

BucketHandlerTest.trip2UploadTest

MainCLI.java | bristol_streetview_schema.Photo [@localhost] | BucketHandlerTest.java | README.txt | script.sql | init.sql

79 rows | Tx: Auto

Filter criteria>

id	height	width	photoTimestamp	uploadTimestamp	latitude	longitude
0367548cbed94277a8c0bb8c1bb80917	2688	5376	2018-07-30 13:46:20	2018-09-15 16:09:42	51.45631419999773	-2.602839700002547
0386fbc250eb4d1ea005c3251162b5f8	2688	5376	2018-07-30 13:47:24	2018-09-15 16:09:40	51.45610310000237	-2.602473299999069
082fa7cf0c5d4e0ea9ba392a58a48ff7	2688	5376	2018-07-30 13:29:46	2018-09-15 16:09:41	51.45714530000223	-2.6014936
0a370862a00046a19a5ae19ee57a334e	2688	5376	2018-07-30 13:27:45	2018-09-15 16:09:39	51.457572899997466	-2.6032672999975452
158e4cd08b644cd68661972e5f574b06	2688	5376	2018-07-30 13:46:09	2018-09-15 16:09:42	51.4563165999978	-2.602602100001196
171417e9656e453295cd6d4b528ab4fe	2688	5376	2018-07-30 13:48:14	2018-09-15 16:09:39	51.45572233606558	-2.6031185999981123
174b9d4294ab4638984782561d6d24a5	2688	5376	2018-07-30 13:22:56	2018-09-15 16:09:41	51.456374800000006	-2.6029837000021634
183c82f2b7df4fb2a2f236408412aa2e	2688	5376	2018-07-30 13:28:02	2018-09-15 16:09:39	51.4574876	-2.6030148
19203b8d9f364056a477975d8e0d762a	2688	5376	2018-07-30 13:41:38	2018-09-15 16:09:41	51.45678109999713	-2.6007246000021667
1b11df2d816948c898704ed6d5e2153c	2688	5376	2018-07-30 13:23:05	2018-09-15 16:09:42	51.456468899998576	-2.603005400002366
1b64b189716b4f2d8a691eeecd1ab1e9	2688	5376	2018-07-30 13:43:39	2018-09-15 16:09:42	51.45652630000177	-2.6016896000000003
1bd8f99304c3408482d55b988dccb77f	2688	5376	2018-07-30 13:41:05	2018-09-15 16:09:39	51.456906300002714	-2.6006072000000002
1e0aeb57bb704828aeb07ac906e92681	2688	5376	2018-07-30 13:26:27	2018-09-15 16:09:40	51.45740489999864	-2.603996
222d2999eed046e78a8275872c115d75	2688	5376	2018-07-30 13:24:53	2018-09-15 16:09:41	51.456978699998864	-2.6035305
25274c86226044859bac0b200b479221	2688	5376	2018-07-30 13:22:40	2018-09-15 16:09:40	51.45624730000198	-2.6030626
2ab652049fc6406d88dbc63c74963beb	2688	5376	2018-07-30 13:23:52	2018-09-15 16:09:41	51.456954	-2.6033575
2bc4f32ce2bf4f459a0f6caad6acd47d	2688	5376	2018-07-30 13:38:56	2018-09-15 16:09:42	51.457274100000966	-2.6006088999989108
2dc4e998364c44fcaac09dc46676233c	2688	5376	2018-07-30 13:23:39	2018-09-15 16:09:40	51.456860900002724	-2.603235399997972
2fb7eb650a1249218201e4d23532253f	2688	5376	2018-07-30 13:42:29	2018-09-15 16:09:39	51.4564175	-2.6011920999975895
31fd70f4fde842de985ac802ab50efbb	2688	5376	2018-07-30 13:27:35	2018-09-15 16:09:40	51.4576442000021	-2.603358611111111
3329d50708794a7da3117b49469db4d5	2688	5376	2018-07-30 13:27:29	2018-09-15 16:09:40	51.45773430000224	-2.6034379000019694
34d4a15bd44f4b60a43c0a802aa48fa6	2688	5376	2018-07-30 13:27:23	2018-09-15 16:09:40	51.45784769999879	-2.6035046
356644fe7884487586a893cb36c8a1fe	2688	5376	2018-07-30 13:48:40	2018-09-15 16:09:41	51.455555555555556	-2.6031560000000002
3732387068c842a3a867ffe56b8eb5ce	2688	5376	2018-07-30 13:28:51	2018-09-15 16:09:40	51.4569776	-2.602360299998491
398dd4cb372a41abbd490ee2db6bbd02	2688	5376	2018-07-30 13:30:13	2018-09-15 16:09:42	51.457166	-2.60099820000188
401b7c3c5a064ce3a507790d73248c2e	2688	5376	2018-07-30 13:47:30	2018-09-15 16:09:40	51.45593269999837	-2.602631799998144
4277af7ebf94493fbe501635298433b2	2688	5376	2018-07-30 13:42:19	2018-09-15 16:09:42	51.456515	-2.601202900002226
4caed987eddf4d6984cedd15b831537f	2688	5376	2018-07-30 13:45:47	2018-09-15 16:09:42	51.45624019999816	-2.602278600001848
4d0b200aa30845a9a8ed1d0d934d4306	2688	5376	2018-07-30 13:49:03	2018-09-15 16:09:40	51.45536779999797	-2.603085
4ed417bfc63f4770a9083f67fbd17330	2688	5376	2018-07-30 13:44:00	2018-09-15 16:09:40	51.4564432	-2.601634600001882
52400608a6394c7a9fb127c47eca83b8	2688	5376	2018-07-30 13:25:27	2018-09-15 16:09:40	51.45723739999804	-2.6035032
541f36277b42466f93c3464d9800e23d	2688	5376	2018-07-30 13:37:53	2018-09-15 16:09:39	51.4573764	-2.6005745
546c16c46804439fa2b46164aae8d3c5	2688	5376	2018-07-30 13:49:15	2018-09-15 16:09:39	51.4552715	-2.603030303030303
557a7f41f74e4a2aa3e7e2e28a502532	2688	5376	2018-07-30 13:28:57	2018-09-15 16:09:40	51.4569379000009	-2.602227400002314
58d21e4eb1554024a8865c49a29a5184	2688	5376	2018-07-30 13:29:19	2018-09-15 16:09:41	51.45702790000239	-2.601779399997808
5d66026c530341fba2d13f83d875ccd4	2688	5376	2018-07-30 13:21:45	2018-09-15 16:09:41	51.45619869999904	-2.6031033999978463
5e529c1177b04588b39f74a1e69f5230	2688	5376	2018-07-30 13:50:04	2018-09-15 16:09:40	51.455555555555556	-2.603137700004408
5ea0262fb9b1472ea671c590df9689c2	2688	5376	2018-07-30 13:41:51	2018-09-15 16:09:42	51.45670669999759	-2.6008685000000002

Tests passed: 1 (2 minutes ago)

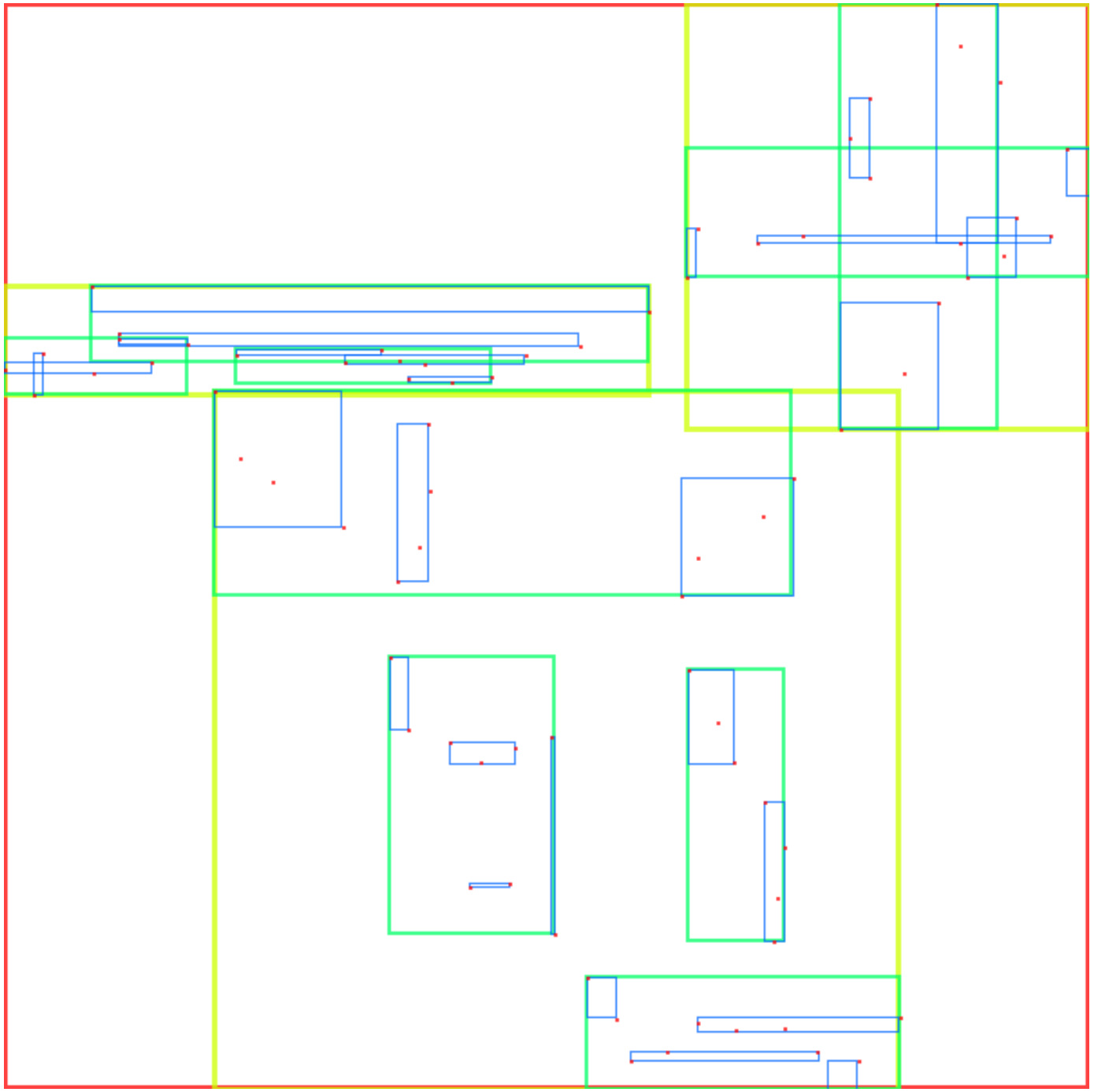
PART 3C

Desktop Client: Retrieving photos with a geographic search

More importantly, processed images can also be retrieved using a geographic search (finding images nearest to the given latitude and longitude.

This uses a Java implementation of an RTree. It's a tree-like data structure designed for storing multi-dimensional information.

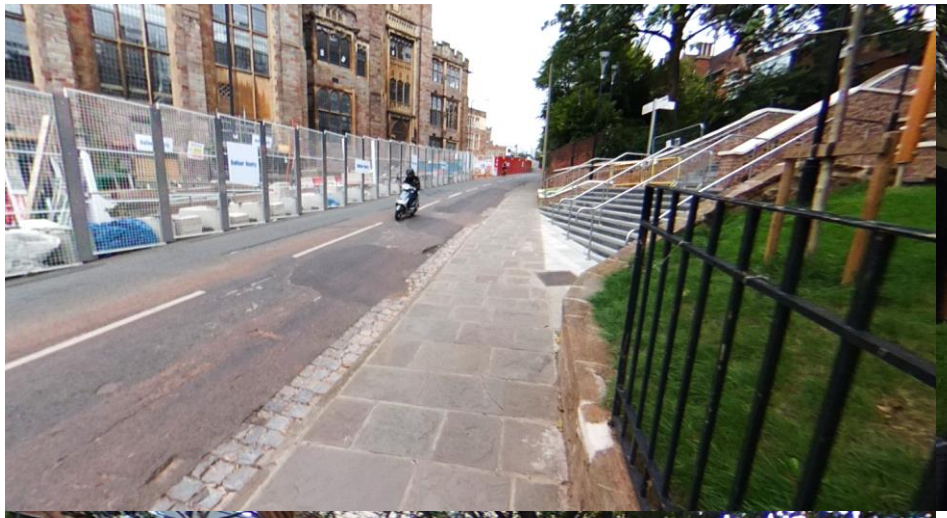
Image on the right represents our RTree after it processed 78 images from a test trip.

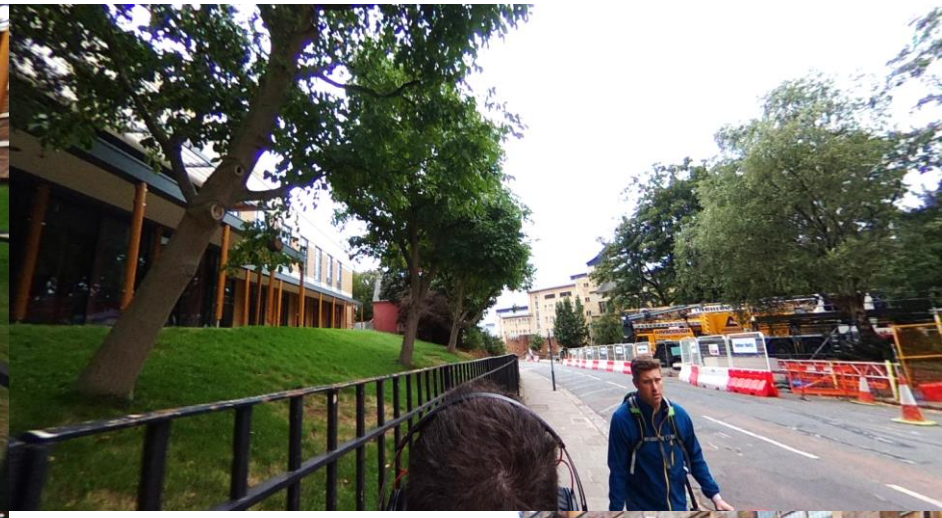


PART 4

Gnomonic Projections

A python library adapted from GitHub, which allows 360 degree images to be converted into Front, Back, Left and Right projections – they look exactly as if they were taken with a normal phone camera.





SUMMARY AND KEY OUTCOMES

Throughout the project I had to use:

- The Android framework
- Java
- MySQL Databases

I also had to use many good Software Engineering practices, such as design patterns to make sure the project didn’t become impossible to continue after the first few weeks.

I’m most proud of making the Android app. It required manually connecting to the camera using HTTP requests and also accessing GPS and phone sensor data – all of which I wasn’t familiar with before this summer.

NEXT STEPS AND FUTURE OF THE PROJECT

The created platform is fully functional and ready to be used in real life.

Andrew Calway and his PhD student will soon begin using the app on trips around Bristol in order to collect images for their localisation algorithm.