# Extract Trajectories from YFCC100M

This slides describe the steps to reconstruct user trajectories from photos with geo-tag and time stamp

# Original Dataset: YFCC100M

- YFCC100M photo/video dataset is used to reconstruct user trajectories
- YFCC100M has been released by Yahoo for research and contains
  - 99.3 million photos (49 million are geo-tagged) and 0.7 million videos
  - Each photo/video with corresponding metadata
- Photos and videos are uploaded to Flicker between 2004 and 2014
  * We will not distinguish photos and videos in the following slides
- Example of data entry:

| Attribute | Value |
|---|---|
| Photo_ID | 12019020406 |
| User_ID | 30265340@N00 |
| Date Taken | 2014-01-18 13:10:29.0 |
| Longitude | 145.40 |
| Latitude | -37.87 |
| Accuracy | 16 |
| URL | http://www.flickr.com/photos/ |
| ... | ... |

# Trajectory extraction steps

- From the original YFCC100M data, we reconstruct user trajectories by using the **geo-tagged** photos with corresponding **timestamp**

- Three steps for extracting trajectories:

**Step 1**: Extract photos near Melbourne from YFCC100M dataset

**Step 2**: Construct candidate trajectories from the extracted photos

**Step 3**: Quality control: filter out abnormal trajectories from the candidate trajectories

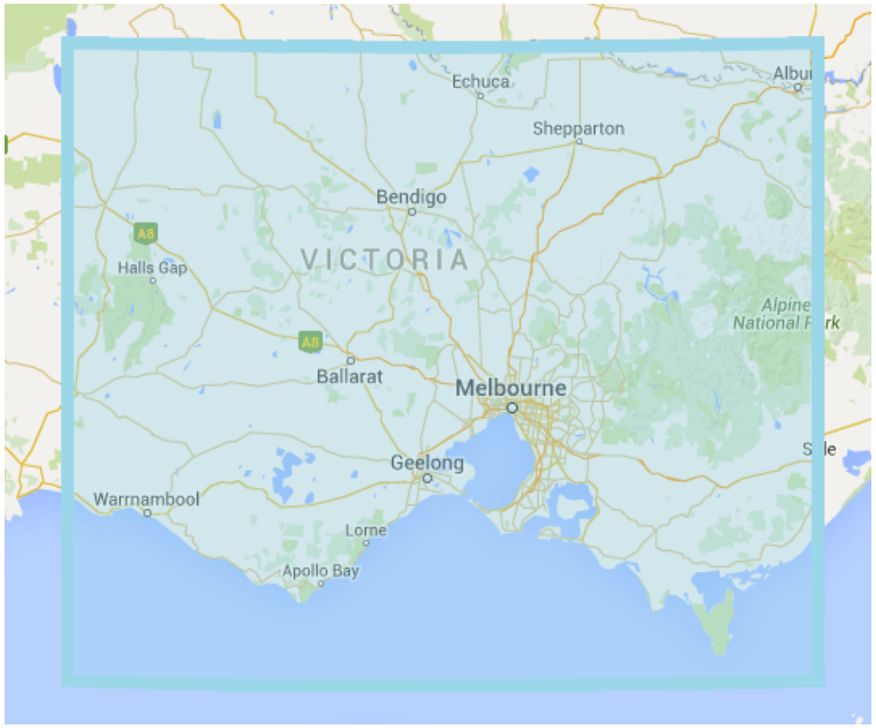
- We will describe the details of each step in the next few slides

Data Construction

# **Step 1**: Extract photos near Melbourne area

- The original YFCC100M dataset contains 100 million photos. We are interested in travel patterns near Melbourne area, so we extract photos taken near Melbourne to reduce the further computational cost

- Precisely, we **extract photos taken in the region below** from the original YFCC100M:




- From the first step, **216,472** photos from **2,982** users are extracted

- Source code of step 1: src/filtering_bigbox.py

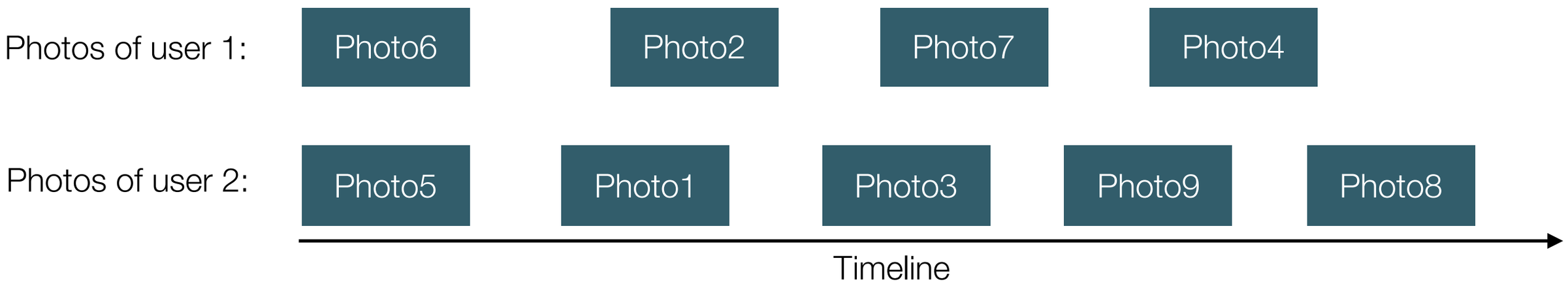
# **Step 2**: Construct candidate trajectories

From the extracted photos, we reconstruct user trajectories using geo-tag and timestamp of photos as follows:

**Step 2.1**: Group the extracted photos by user

**Step 2.2:** Sort the grouped photos by timestamp

Below example, we grouped 9 photos by user, and then sort the photos by timestamp



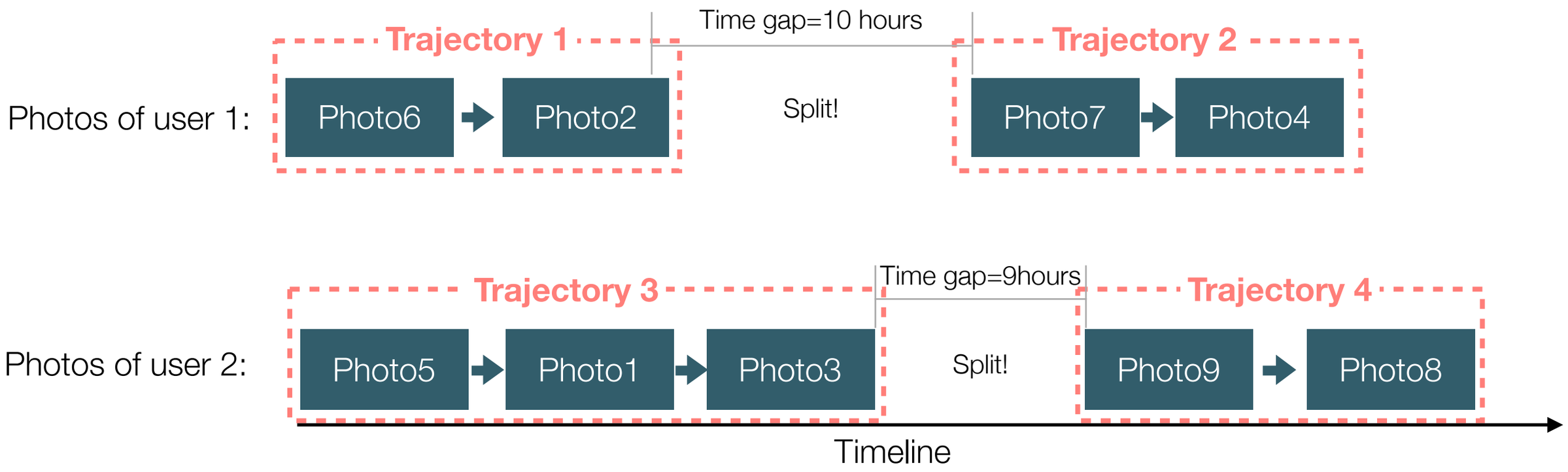

Example of grouped photos by users and then sorted by timestamp

# **Step 2**: Construct candidate trajectories

---

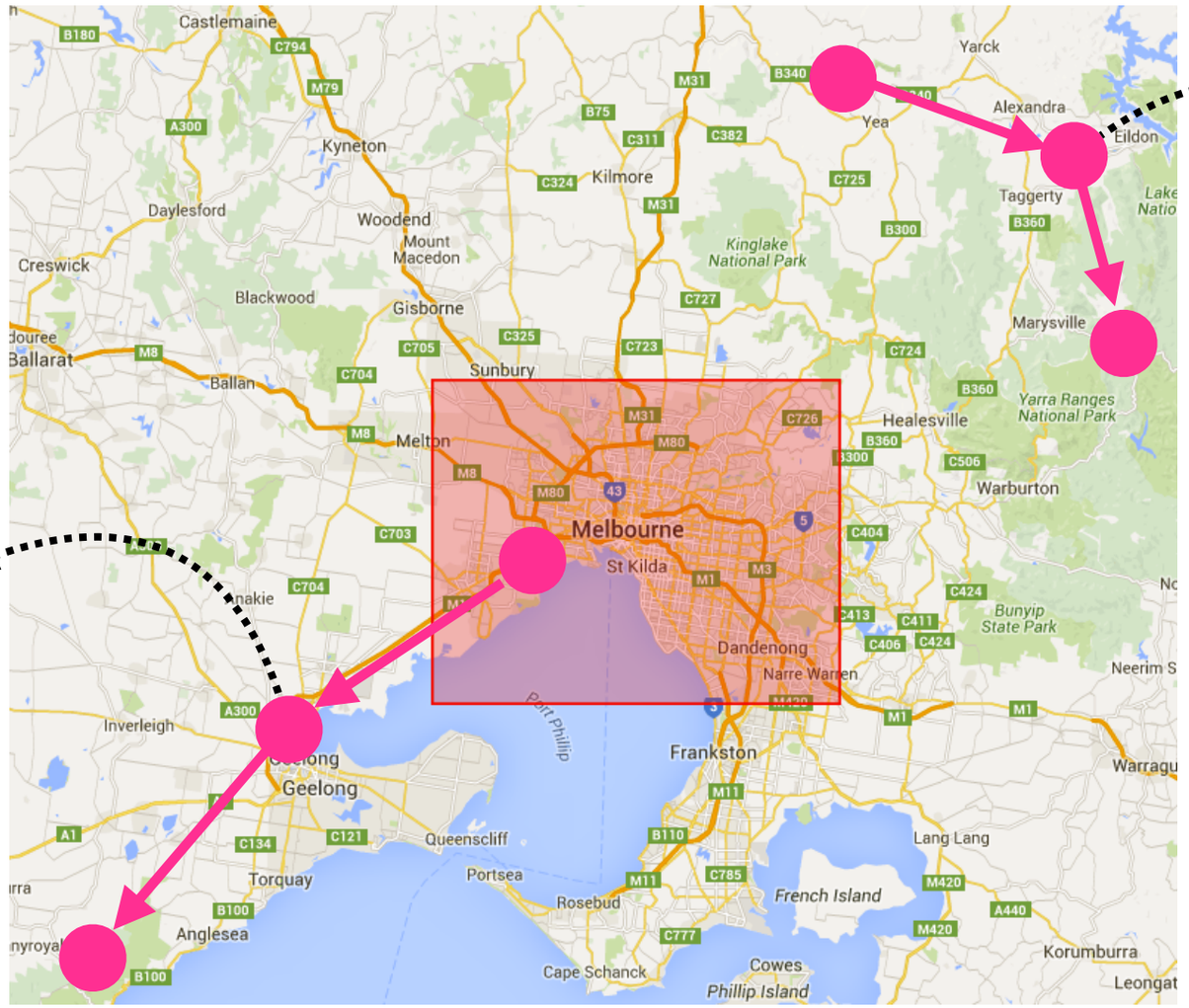
**Step 2.3**: Split the sorted photos into trajectories if the time gap between **two consecutive photos** is greater than **8 hours**

In the following example, we construct 4 trajectories from sorted photos



# **Step 2**: Construct candidate trajectories

**Step 2.4**: We plot the trajectories on map. Keep trajectories **at least one photo** is taken from the central Melbourne area (Red box below). To make sure that the travel is not far from Melbourne




This trajectory will be removed
It is too far from Melbourne

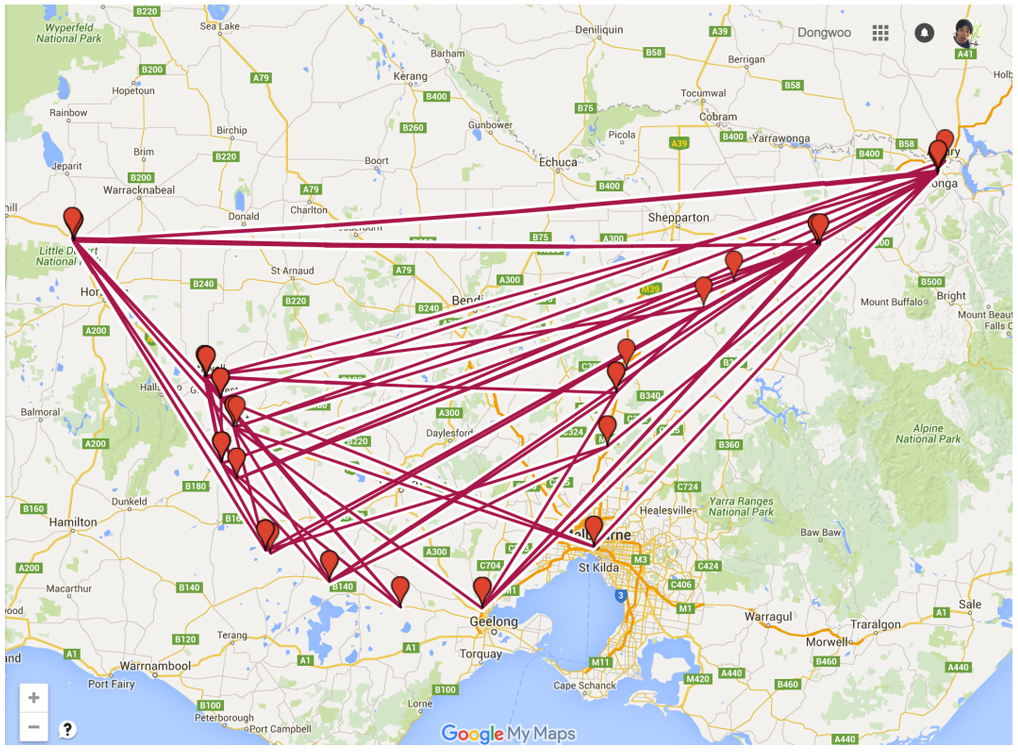
This trajectory will survive
It shows interesting travel pattern from Melbourne

Source code of step 2: src/generate_tables.py

# **Step 3**: Quality Control

- Due to several reasons, the candidate trajectories contains some abnormal, improbable, and meaningless trajectories. We filter out these trajectories based on:
  - **Step 3.1 - Travel time**: Too short travel time is meaningless as a trajectory. These trajectories are filtered out based on travel time
    - Travel time = timestamp of the last photo - time of the first photo
  - **Step 3.2 - Travel distance**: Photos taken from a single location form a single trajectory. Since we are not interested in these trajectories, these trajectories are filtered out based on travel distance
  - **Step 3.3 - Average speed**: Trajectories with improbable speed are filtered out based on their average speed. Caused by GPS and time stamp errors
    - Average speed = total distance / travel time
    - Total distance = sum of distances between consecutive photos
- Source code of step 3: src/trajectory_construction.ipynb

# **Step 3.1 & 3.2**: Example of abnormal travel time & distance




- Trajectory with the highest distance
- Travel distance: 11,936.86 km
- Total time: 0.0 min
- Average speed: NA km/h

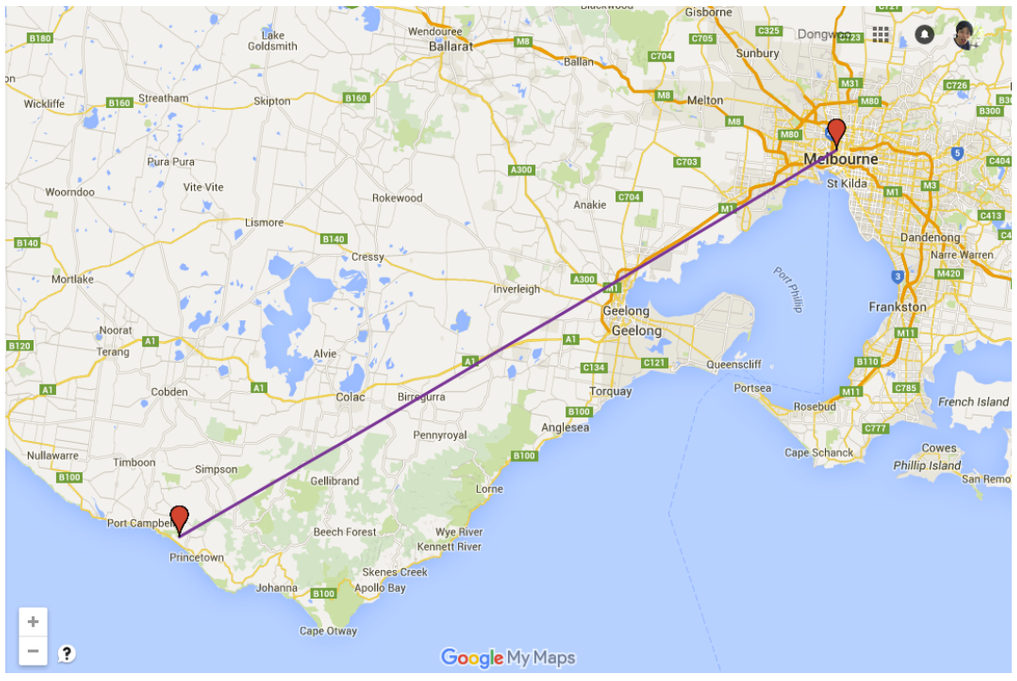

Sample photos from this trajectory

- All photos in this trajectory have the same time-stamp (Total time = 0 min)
- These photos seems to be taken during a train trip. But the timestamp of photos is not correctly captured, therefore the reconstructed trajectory shows random path between photos
- We filtered out all trajectories
  - Less than **30 minutes** of travel time or more than **24 hours** of travel time
  - Less than **500 m** of travel distance

Data Construction - Data Cleaning

# **Step 3.3**: Example of abnormal speed

- We will see some examples of abnormal trajectory visualised on Google map (Clicking the map will open corresponding Google map page)



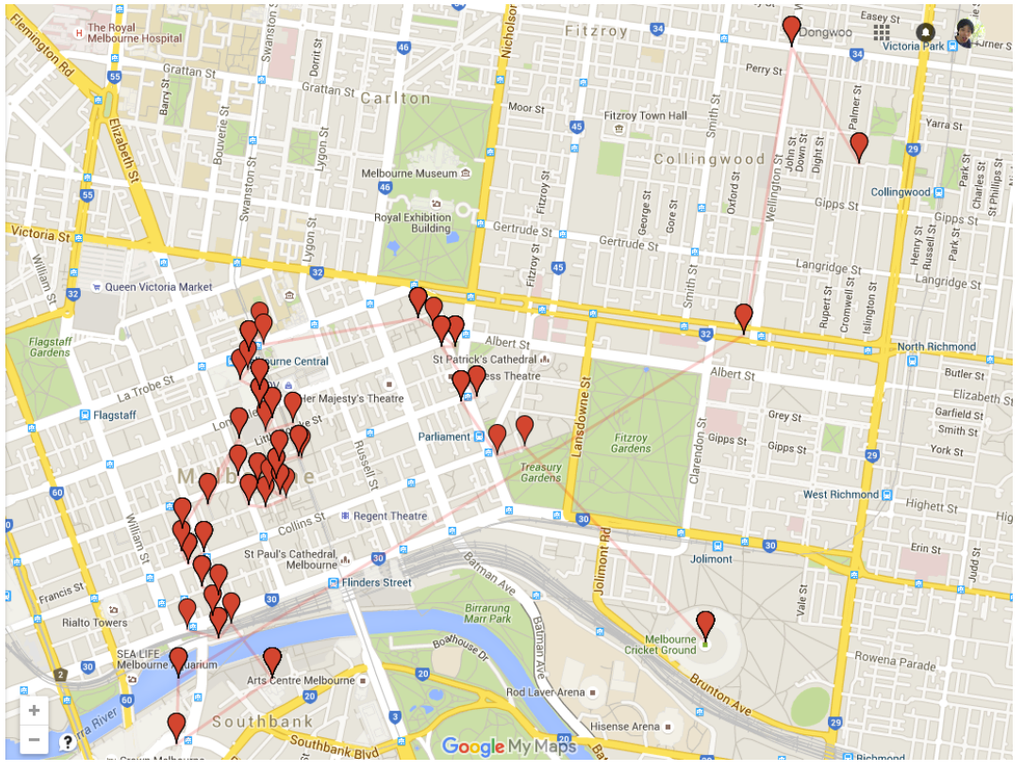

- Trajectory with the highest speed
- Travel distance: 1871.32 km
- Total time: 0.03 min
- Average speed: 3,368,377 km/h

Sample photos from this trajectory

- The improbably high speed is caused by invalid time-stamp
  - All photos in this trajectory are taken in 3 minutes
- We filtered out all trajectories over **100 km/h**
  - We actually use bit more complex algorithm. See the notebook file for details

Reconstructed Trajectory

# Example of interesting trajectory 1



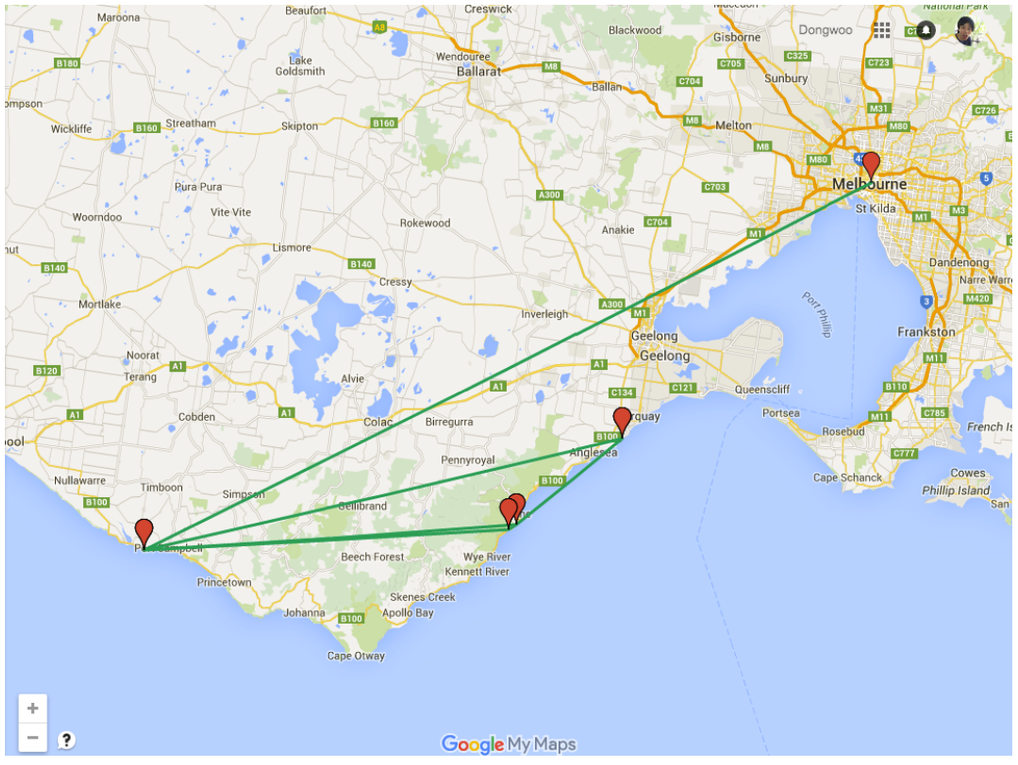

- Trajectory with the most photos
- # of photos: 379
- Total time: 593.5 min
- Average speed: 0.2 km/h

Sample photos from this trajectory

- This is the trajectory with the most number of photos
- The user starts his journey at 11am, walk around city centre, take photos, and finally move to the Melbourne cricket ground at 9pm
- 379 photos show the detail trajectory of traveller in the town centre

Reconstructed Trajectory

# Example of interesting trajectory 2




- Trajectory with Great Ocean Road
- # of photos: 16
- Total time: 1,351 min
- Average speed: 38.4 km/h


Sample photos from this trajectory

- This trajectory starts from the central Melbourne area, and then the user moved to the great ocean road
- There are several other trajectories from the great ocean road in the final trajectory dataset

# Data Statistics Before and After Filtering

| | # of photos | # of users | # of trajectories | # of photos per user | # of trajectories per user |
|---|---|---|---|---|---|
| Before filtering | 151,585 | 2,443 | 26,468 | 62.05 | 10.83 |
| After filtering | 58,146 | 977 | 4,460 | 59.51 | 4.56 |

- The above table shows some basic stats before and after several filtering steps
- 1/3 photos are left, 1/6 of trajectories are left
- Some meaningful trajectories might be filtered out
- Some noisy trajectories might be still in the dataset

# Data files

- After reconstruction & filtering, we generate two data files

1. Trajectory data (data/trajectory_photos.csv): each row represents one photos with corresponding trajectory ID

| Trajectory_ID | Photo_ID | User_ID | Timestamp | Longitude | Latitude | Accuracy | Marker photo=0 video=1 | URL |
|---|---|---|---|---|---|---|---|---|
| 0 | 4581420457 | 10033564@N03 | 010-05-03 16:49:4 | 144.97 | -37.81 | 11 | 0 | http://www.flickr.co |
| 5 | 2889262481 | 10058801@N06 | 008-09-26 22:02:4 | 144.97 | -37.82 | 15 | 0 | http://www.flickr.co |
| 18 | 12322029584 | 100895643@N03 | 014-02-05 23:52:3 | 144.98 | -37.82 | 16 | 0 | http://www.flickr.co |

2. Trajectory statistics (data/trajectory_stats.csv): each row contains some basic statistics about each trajectory

| Trajectory_ID | User_ID | #Photo | Start_Time | Travel_Distance(km) | Total_Time(min) | Average_Speed(km/h) |
|---|---|---|---|---|---|---|
| 1 | 10033564@N03 | 4 | 2010-05-03 16:49:44 | 5 | 49.72 | 10.0 |
| 7 | 10058801@N06 | 19 | 2008-10-11 15:56:41 | 1.37 | 71.53 | 1.15 |
| 10 | 10087938@N02 | 16 | 2008-03-13 01:35:32 | 0.78 | 169.95 | 0.28 |

# Miscellaneous

- All source codes and datasets are available on GitHub repository
  - https://github.com/arongdari/flickr-photo
- Detail data processing and cleaning steps with notebook
  - https://github.com/arongdari/flickr-photo/blob/master/src/trajectory_construction.ipynb
  - **Changing filtering parameters and rerun this notebook will generate different set of trajectory files**
- Data statistics and a way to generate KML files
  - https://github.com/arongdari/flickr-photo/blob/master/src/flickr_analysis.ipynb