Leveraging the Social Breadcrumbs


[adult swim]
xanga
My
AskMen
Ezine
@rticles
PREMIUM
care2
YouTube
amazon
BARNES & NOBLE
Adobe
digg
iMDB
yelp
kaboodle
MULTIPLY
FARK
fanpop!
deviantART

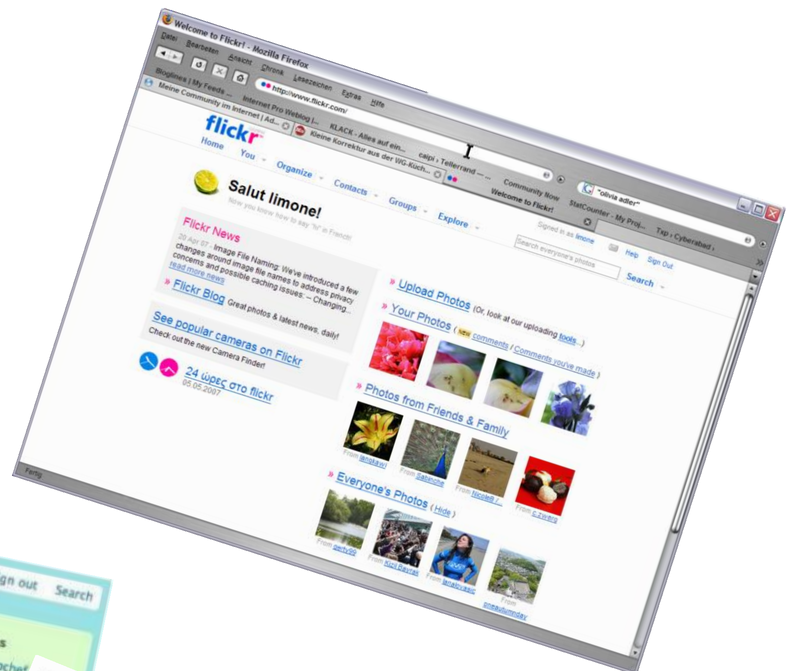
Social Network Service

- Important part of Web 2.0
- People share a lot of data through those sites
- They are of different kind of media
- Uploaded to be seen by other people

- Somehow read-once
- But we want to exploit more other useful information from them
- Through automatic applications

Diverse Services

- We will look through some examples



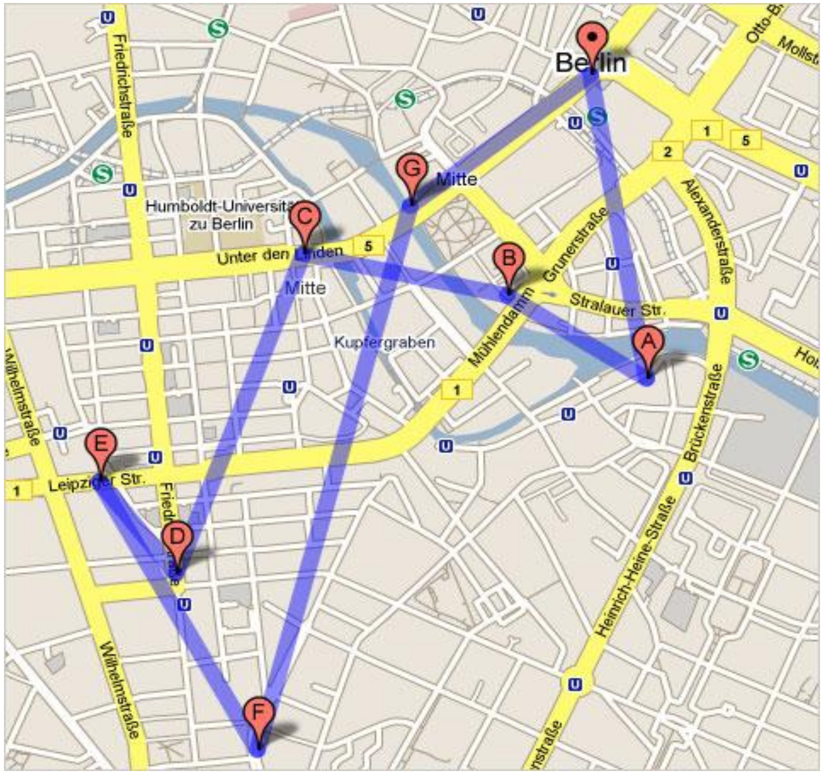
Automatic Construction of Travel Itineraries using Social Breadcrumbs

Problem

- Travel itinerary planning is often difficult
- Traveler must
  - Identify points of interests (POIs) worth visiting
  - Consider the time worth spending at each point
  - Consider the time it will take to get from one place to another
- Compiling an itinerary is both time consuming and requires significant search expertise

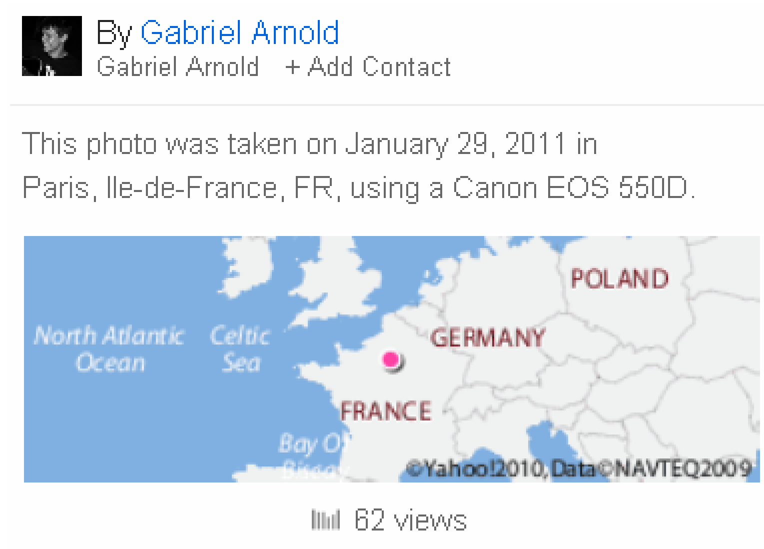
Our Goal

- Automatically construct travel itineraries at a large scale
- Construct itineraries that reflect the “wisdom” of touring crowds
- “Automatically”, and “wisdom of touring crowds”, these are the two main points in this article



Idea

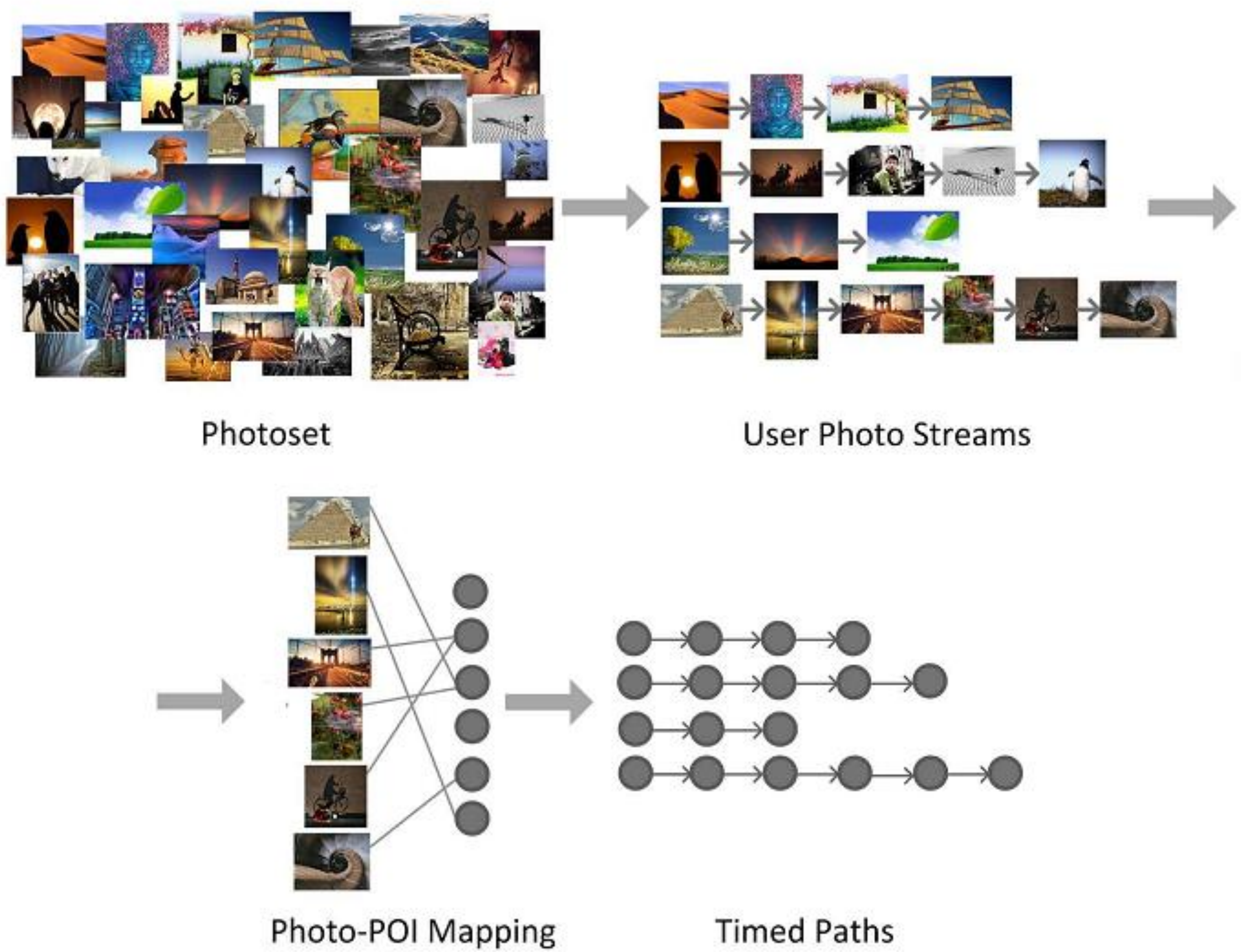
- millions of travelers
- sharing their travel experiences
- through rich media data
- contextual information
  - time-stamped
  - geo-tagged
  - textual metadata



Two Steps

- touristic data analysis
  - analyzing POI visitation patterns from geo-spatial and temporal evidences left by travelers
- touristic information synthesis
  - construct and recommend tourist itineraries at various granularity

Itineraries as Timed Paths



Constructing User Photo Streams

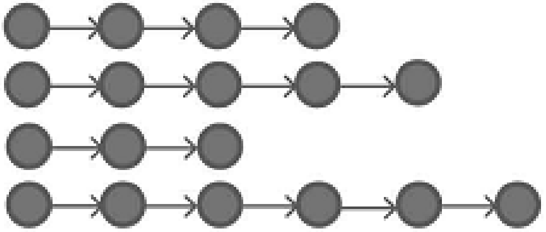
- Pruning away irrelevant photos using these 3 rules
  - Identifying photos of the city
    - semantic tags
  - Filtering residents of the city
    - tourists visit within a short time period
    - a user visits at least two POIs to be considered as a tourist
  - Photo taken time verification
- Sort them by their taken time.
- The result is a collection of city photo streams.

Generating Timed Paths

Photo-POI Mapping

- Photo – POI Mapping : geo-based, tag-based

- *Visit time* : a lower bound on the actual time spent by the particular user at that POI
- *Transit time* : an upper bound on the time it took for the particular user to move from one POI to the next



Timed Paths

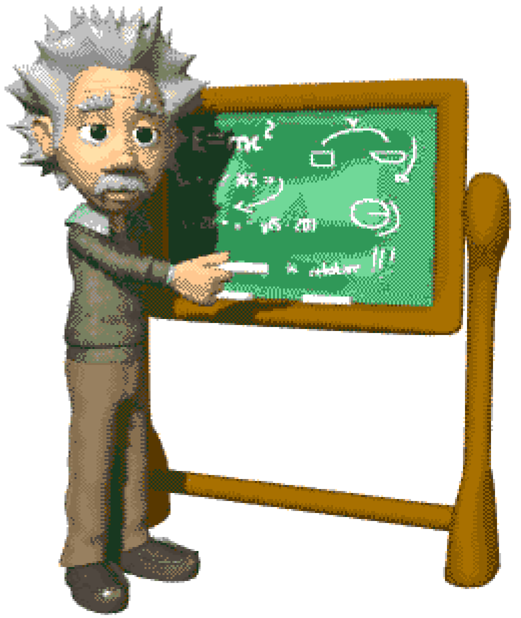
Itinerary Mining Problem (IMP)

- Objective : Find an itinerary in *G* from *s* to *t* of cost at most B maximizing total node prizes
- *G* : Undirected graph of POIs associated with Transit times and Visit times
- *s, t* : either provided by the user or implicitly set by the itinerary application
- *B* : user's time
- *Prize* : product of the popularity and the visit duration

Time **09:00** : Start from **ground zero**
Time **09:00** : Spend 27 minutes at **ground zero**.
Time **09:27** : Transit to **empire state building** (estimated travel time: 52 minutes)
Time **10:19** : Spend 1 hour and 13 minutes at **empire state building**.
Time **11:32** : Transit to **new york public library** (estimated travel time: 15 minutes)
Time **11:47** : Spend 29 minutes at **new york public library**.
Time **12:16** : Transit to **radio city music hall** (estimated travel time: 24 minutes)
Time **12:43** : Spend 51 minutes at **radio city music hall**.
Time **13:34** : Transit to **central park** (estimated travel time: 23 minutes)
Time **13:57** : Spend 40 minutes at **central park**.
Time **14:37** : Transit to **rockefeller center** (estimated travel time: 33 minutes)
Time **15:10** : Spend 37 minutes at **rockefeller center**.
Time **15:47** : Transit to **grand central terminal** (estimated travel time: 22 minutes)
Time **16:09** : Spend 27 minutes at **grand central terminal**.
Time **16:36** : Transit to **chrysler building** (estimated travel time: 6 minutes)
Time **16:42** : Spend 31 minutes at **chrysler building**.
Time **17:13** : Transit to **brooklyn bridge** (estimated travel time: 32 minutes)
Time **17:45** : Spend 36 minutes at **brooklyn bridge**.
Time **18:21** : Transit to **statue of liberty** (estimated travel time: 21 minutes)
Time **18:42** : Spend 42 minutes at **statue of liberty**.
Time **19:24** : Transit to **little korea** (estimated travel time: 26 minutes)
Time **19:50** : Spend 31 minutes at **little korea**.
Time **20:21** : Transit to **ground zero** (estimated travel time: 38 minutes)

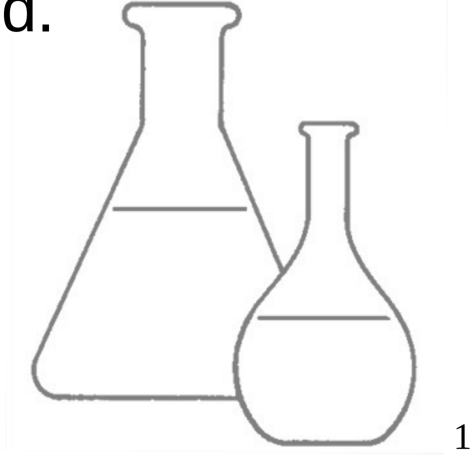
Algorithm to Solve IMP

- The Itinerary Mining Problem is NP-Hard
- Proved by a reduction from the Hamiltonian Path problem
- Reduce IMP to the directed Orienteering problem
- Solve using Chekuri and P´al’s approximation algorithm
  - Recursive greedy algorithm for Orienteering

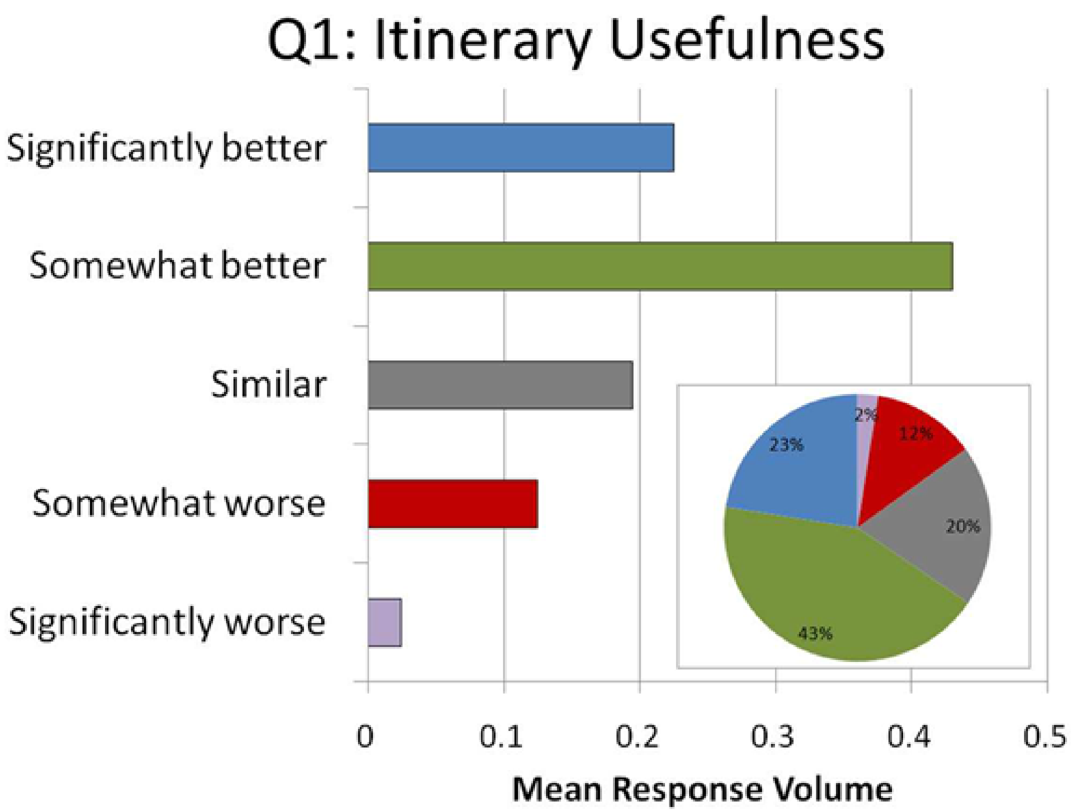


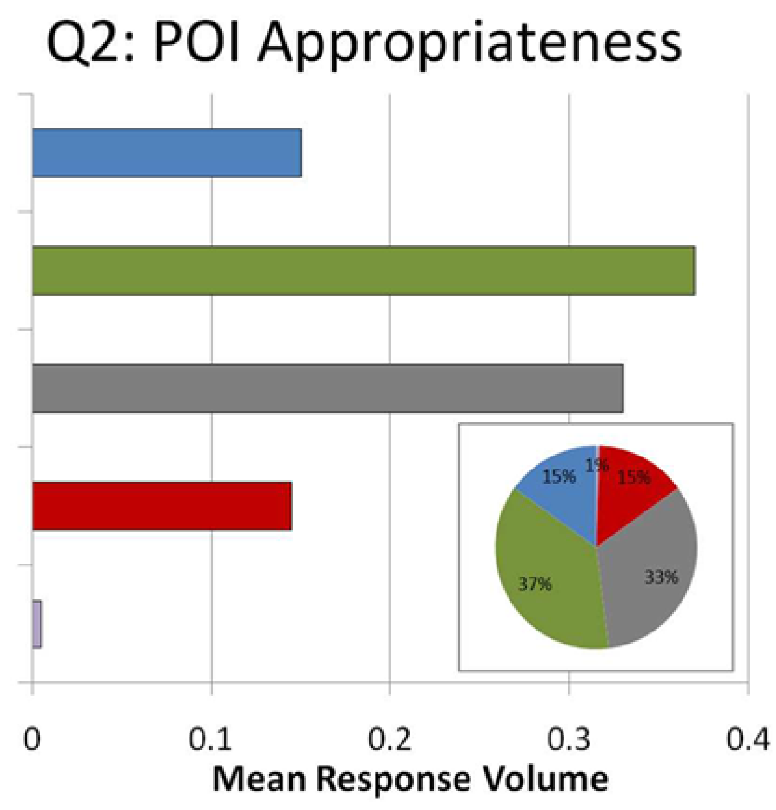
Experimental Methodology

- Design several user studies using the Amazon Mechanical Turk
  - a crowd-sourcing marketplace
  - provides *requesters* the use of human intelligence to perform tasks which computers are unable to do
  - *workers* can then browse among existing tasks and complete them for a monetary payment
- We enforce that only the workers who correctly identify three lesser known POIs of the city, qualify to proceed.



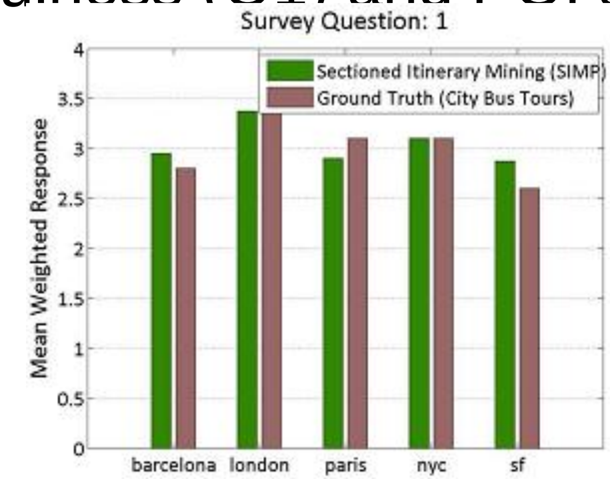
Comparative Evaluation of Itineraries

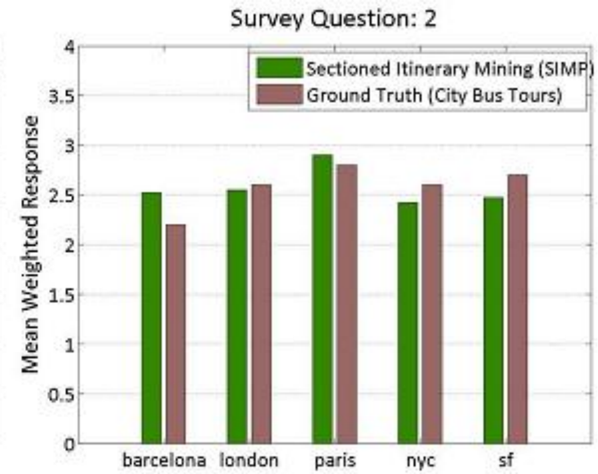




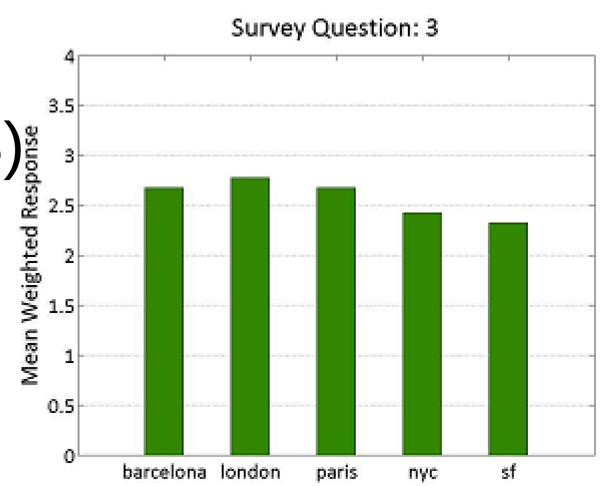
Independent Evaluation of Itineraries

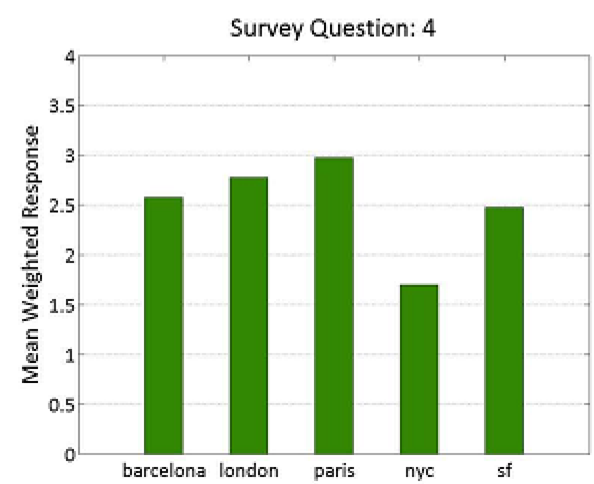
- In terms of overall usefulness (Q1) and POI satisfaction (Q2), IMP itineraries are as good as professionally generated ground truth itineraries





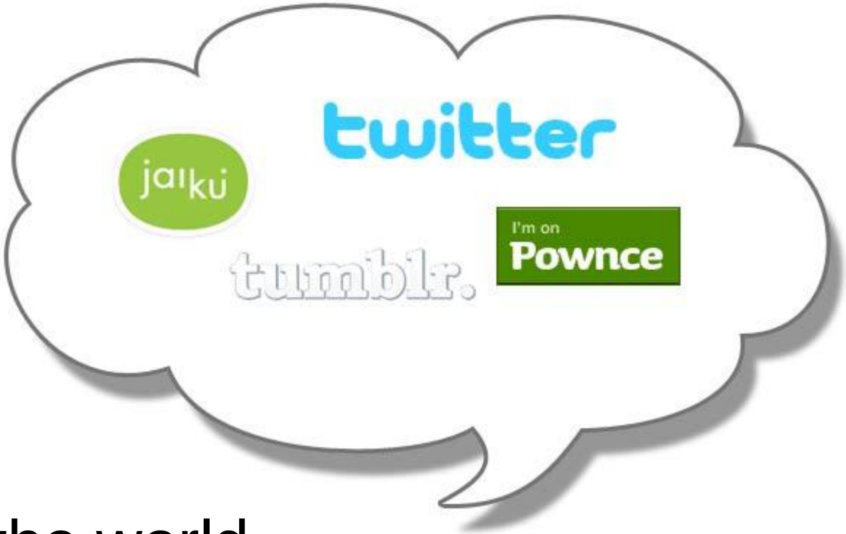
- Workers are generally happy with the visit (Q3) and transit (Q4) times that our system produces





Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors

Microblogging

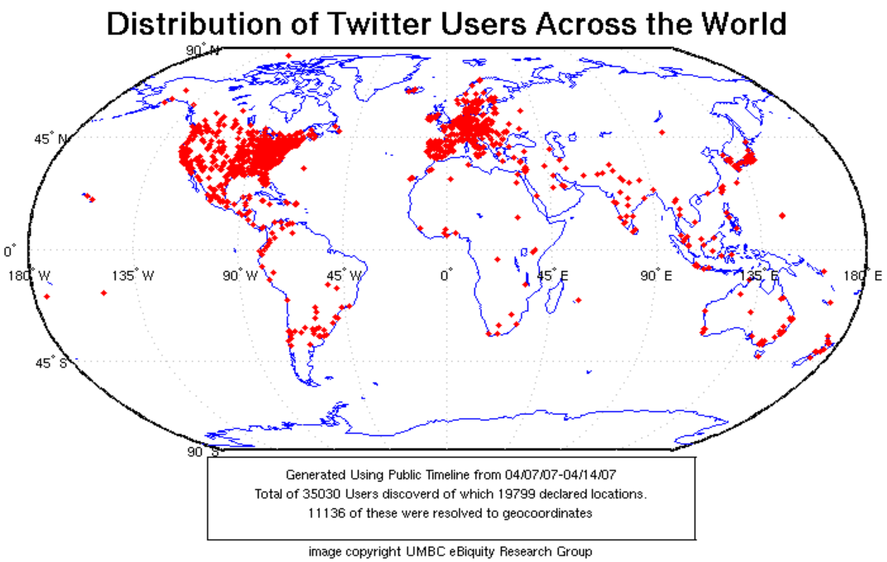


- What I'm doing right now ...
- What I'm feeling right now ...
- What I'm wishing right now ...
- Used by millions of people around the world
- Large number of updates → numerous reports related to events
- Many works done on leveraging this amount of data

Real-time Notification

- Earthquake at August 12, 2009 in Japan
- The first user tweeted about it was Ricardo Duran

Twitter : Network of Social Sensors



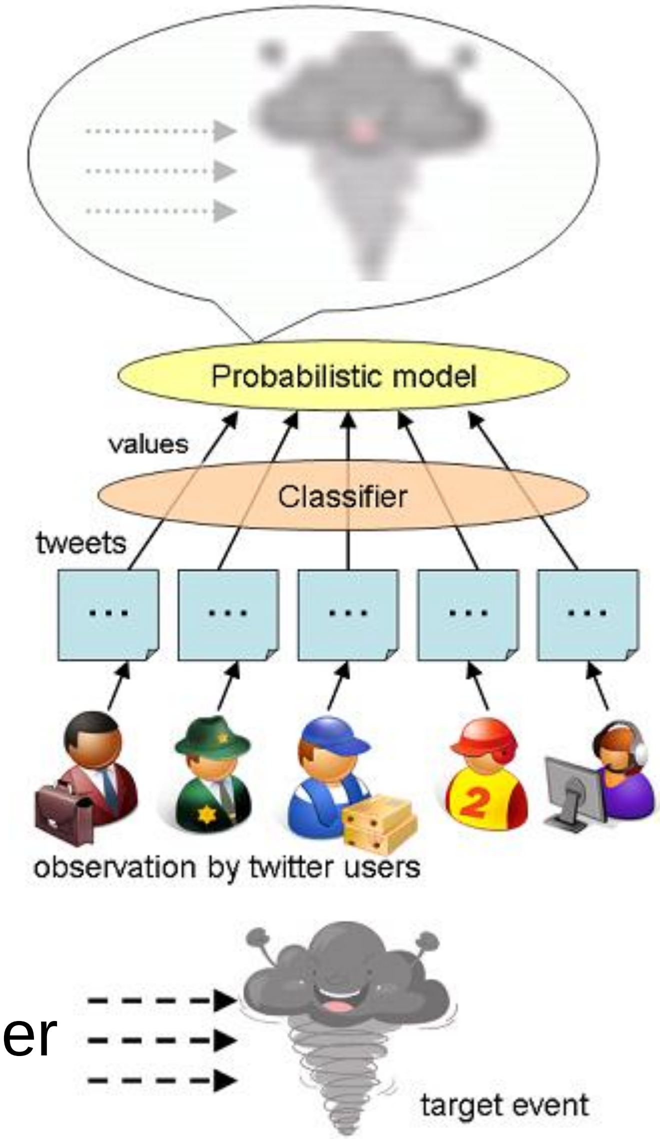
- Each Twitter user as a sensor
- 200 million sensors worldwide
- Tweet sensory information
- Real-time nature
- Huge variety
  - Very active or not
  - Even inoperable or malfunctioning sometimes
- Very noisy compared to ordinary physical sensors

Event Detection

- Visible through tweets: *Earthquakes*,*Typhoons*, *Traffic jams*
  - large scale (many users experience the event)
  - influence people's daily life (they tweet about it)
  - have both spatial and temporal regions
    - Each tweet has its post time
    - GPS data are attached to a tweet sometimes
    - Each user registers his location in the user profile
- Search from Twitter and find useful tweets
  - Using *search.twitter.com* API
- Tweets would be classified as negative class and positive class

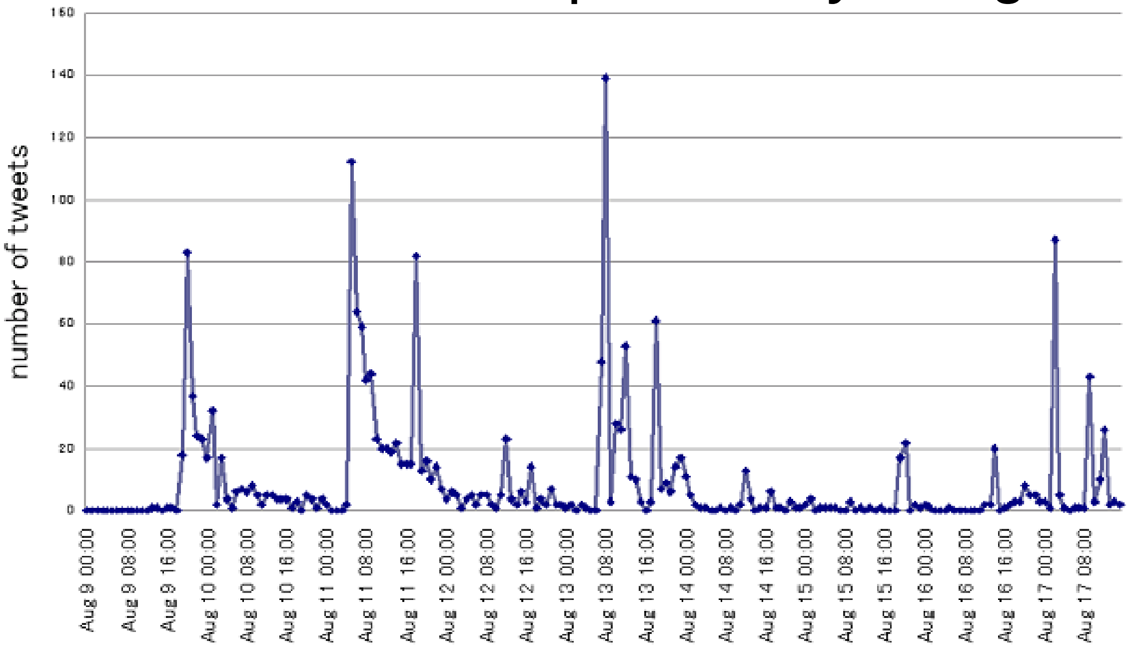
Event Detection (cont.)

- ✔ “Earthquake!”
- ✔ ”Now it is shaking”
- ✗ ”I am attending an Earthquake Conference”
- ✗ ”Someone is shaking hands with my boss”
- Support Vector Machine (SVM), a machine-learning algorithm to classify the tweets
- A probabilistic model used to detect event
- As an application, construct an earthquake reporting system in Japan.
- Numerous earthquakes and the large number of Twitter users throughout the country.



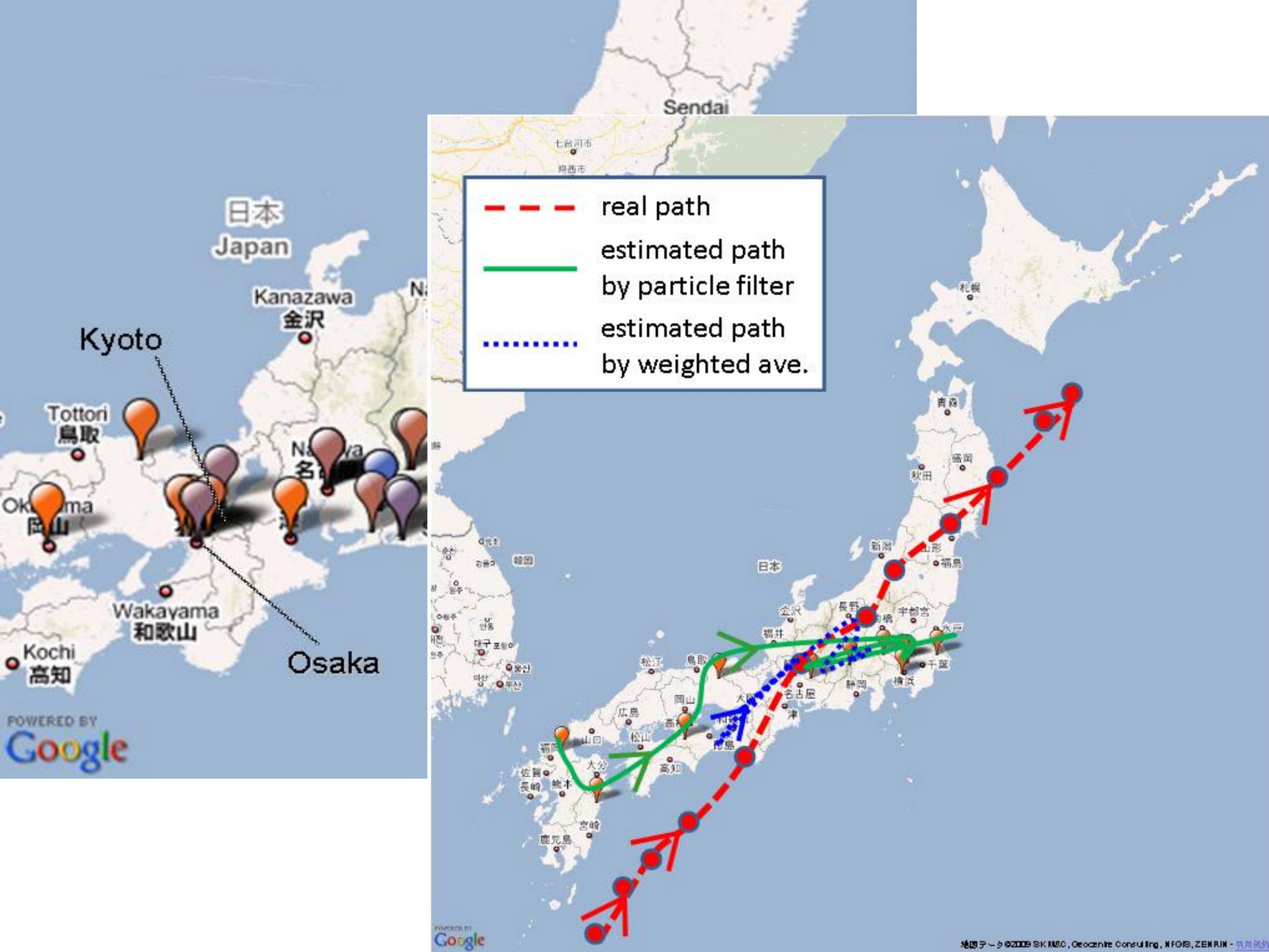
Temporal Model

- The distribution of the number of tweets followed by an event is an exponential distribution
- We can assume that the sensors are i.i.d. when considering real-time event detection such as typhoons and earthquakes
- We consider that an event is detected if the probability is higher than a certain threshold



Spatial Model

- In the paper, implemented models for two cases
  - Location estimation of an earthquake center
  - Trajectory estimation of a typhoon
    - consider both the location and the velocity of an event
- The tracking problem is to calculate recursively some degree of belief in the state at time t, given data up to time t
- Use a Markov process
- We compare Kalman filtering and particle filtering, with the weighted average and the median as a baseline
- Particle filters perform well compared to other methods



Sendai
日本
Japan
Kanazawa
金沢
Kyoto
Tottori
鳥取
Wakayama
和歌山
Kochi
高知
Osaka
POWERED BY Google
real path
estimated path by particle filter
estimated path by weighted ave.

Reporting System

- The greater the number of sensors, the more precise the estimation will be
- The first tweet of an earthquake is usually made within a minute
  - time for posting a tweet by a user
  - time to index the post in Twitter servers
  - time to make queries by our system

- System sent E-mails mostly within a minute, sometimes 20 s
- JMA announcement is broadcast 6 min after an earthquake
- Detected 96% of earthquakes larger than JMA seismic intensity scale 3

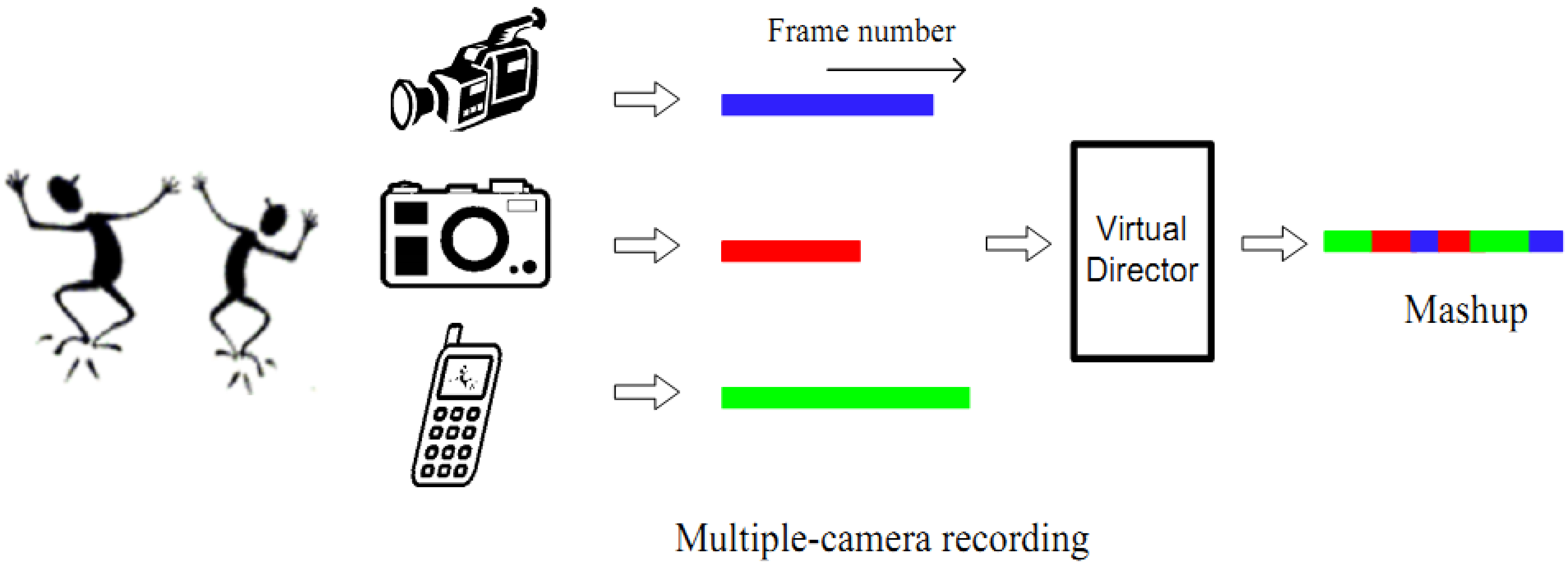
Automatic Mashup Generation from Multiple-camera Concert Recordings

Multi-cam Recording

- It has become common for audiences to capture videos (mobile phones, camcorders, and digital-still cameras) during concerts
- Some are uploaded to the Internet
- Called multiple-camera or multi-cam recordings
- Typically perceived as boring mainly because of their limited view, poor visual quality and incomplete coverage
- *Objective* : To enrich the viewing experience of these recordings by exploiting the abundance of content from multiple sources

Virtual Director

- Automatically analyzes, selects, and combines segments from multi-cam recordings in a single video stream, called mashup

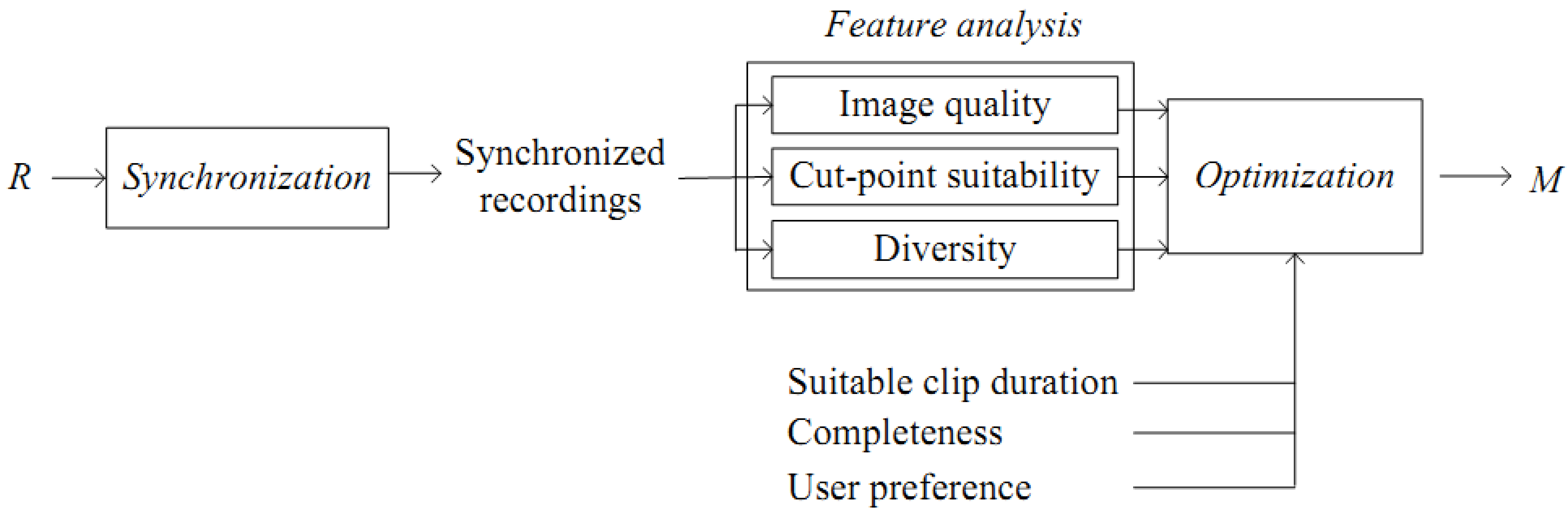


Mashup Requirements

- Constraints
  - Synchronization
  - Suitable segment duration
  - Completeness
- Maximization parameters
  - $Q(M)$: Image quality
  - $\delta(M)$: Diversity
  - $C(M)$: User preference
  - $U(M)$: Suitable cut point

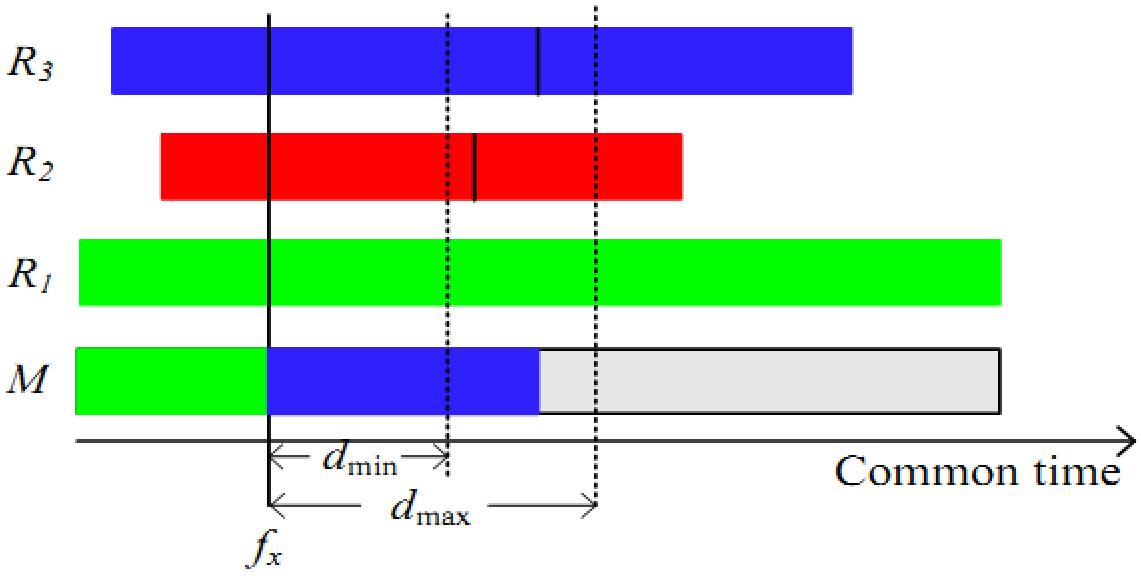
Mashup Generation as an Optimization Problem

- *objective function*
  - MS(M) = aQ(M) + bδ(M) + cC(M) + dU(M)



Optimization

- Search space of multi-cam recording is extremely large
- Developed a greedy algorithm called *first-fit*



Experiment

- Manual mashups created by a professional video editor
- User test with 40 subjects
- The participants have rated the mashups via a questionnaire
- In terms of : *diversity*, *visual quality* and *pleasantness*
- In comparison to the manual mashups the first-fit mashups
  - scores slightly higher in diversity
  - slightly lower in visual quality
  - while both of them score similar in pleasantness
- We conclude that the perceived quality of mashups generated by the first-fit and manual methods are similar

Questions?