Photo2Trip: Generating Travel Routes from Geo-Tagged Photos for Trip Planning*

Xin Lu[1], Changhu Wang[2], Jiang-Ming Yang[3], Yanwei Pang[1], Lei Zhang[2]
[1]Tianjin University, Tianjin, P.R.China, 300072
[2]Microsoft Research Asia, Beijing, P.R.China, 100190
[3]Microsoft Corporation, Shanghai, P.R.China, 200241
{luxin, pyw}@tju.edu.cn, {chw, jmyang, leizhang}@microsoft.com

ABSTRACT

Travel route planning is an important step for a tourist to prepare his/her trip. As a common scenario, a tourist usually asks the following questions when he/she is planning his/her trip in an unfamiliar place: 1) Are there any travel route suggestions for a one-day or three-day trip in Beijing? 2) What is the most popular travel path within the Forbidden City? To facilitate a tourist's trip planning, in this paper, we target at solving the problem of automatic travel route planning. We propose to leverage existing travel clues recovered from 20 million geo-tagged photos collected from www.panoramio.com to suggest customized travel route plans according to users' preferences. As the footprints of tourists at memorable destinations, the geo-tagged photos could be naturally used to discover the travel paths within a destination (attractions/landmarks) and travel routes between destinations. Based on the information discovered from geo-tagged photos, we can provide a customized trip plan for a tourist, i.e., the popular destinations to visit, the visiting order of destinations, the time arrangement in each destination, and the typical travel path within each destination. Users are also enabled to specify personal preference such as visiting location, visiting time/season, travel duration, and destination style in an interactive manner to guide the system. Owning to 20 million geo-tagged photos and 200,000 travelogues, an online system has been developed to help users plan travel routes for over 30,000 attractions/landmarks in more than 100 countries and territories. Experimental results show the intelligence and effectiveness of the proposed framework.

Categories and Subject Descriptors

H.3.5 [**Online Information Services**]: Web-based services

General Terms

Algorithms, Experimentation

*This work was performed at Microsoft Research Asia.

Keywords

Photo2Trip, Geo-Tagged Photos, Travel Route Mining, Trip Planning

1. INTRODUCTION

The prosperity of tourism has made travel increasingly popular in people's everyday lives. Before traveling to an unfamiliar location, most people have questions about how to plan their trips. For example:

-*"I will arrive at Seattle on Jun. 3rd and plan to have a tour there. But I am not familiar with that city. Is there any travel route suggestion to visit the most famous places of interest in one day?"*

-*"I want to have a two-day trip in Seattle, US to visit and taste Seattle's Best Coffee. I desperately need help for trip planning."*

-*"I am going to visit the Forbidden City in Beijing, China at the end of May. Who can offer me a route within the large palace?"*

Although users can search for related travel guide or ask questions in web-based communities, the process is generally not efficient and the results may not be customized. The most common way for current users to find answers for trip planning is probably to read travelogues one by one. However, as each travelogue only records individual footprints during a trip, it is very time-consuming for users to manually summarize tens of travelogues and find a proper travel route for his preference. Moreover, since the information provided by travelogues is usually unstructured and varies from person to person, from language to language, it is extremely hard for users to follow. In this case, an automatic and interactive travel route planning service is highly desired to plan a customized trip according to users' preferences.

In practice, automatic trip planning is a very complex task, which depends on many factors, such as travel duration, travel cost, visiting time, tourist's age and physical condition, and individual interests, out of which some are difficult to model and predict. As the first trial on this task, we target at dealing with the following preferences of users in this work, i.e., travel location (e.g., Beijing, Paris, or New York), travel duration (e.g., two-day trip or five-day trip), visiting time (e.g, summer, winter, March, and October), and destination preference (e.g., prefer historic or prefer scenery sites). Users are enabled to interactively adjust any part of the suggested plans if they have any requirements that the suggested plans do not meet.

In general, for discovering and recommending travel routes, an ideal system needs to answer the following three questions: 1) How to discover popular travel paths and estimate typical stay time within a destination (attraction/landmark)? 2) How to select and organize the destinations in a location to a travel route under certain travel duration constraint? 3) How to meet tourists' diverse preference requirements?

It is not a trivial task to answer the above three questions. However, we have observed that, as the footprints of tourists at memorable destinations, the user-generated geo-tagged photos encode rich travel-related information. Moreover, these photos on the web are publically available and sufficient to cover most of countries and landmarks in the world. This motivates us to address the problem of travel route planning by leveraging geo-tagged photos. However, there are still difficulties we need to face. Although the geo-tagged photos of a tourist could reveal his/her travel clues to some extent, they can only be considered as a discrete and incomplete path. This is because tourists usually take photos at discrete positions along their travel paths and upload only a small portion to their web albums. To address this problem, we propose to aggregate all the photos taken by multiple tourists at the same location, and deliberately design an algorithm to recover as many as possible travel routes by merging incomplete travel paths from multiple tourists.

To the best of our knowledge, little existing work in literatures has systemically investigated the automatic travel route planning problem. Most existing research efforts on user generated content [9, 7, 2, 13] mainly focus on landmarks recognition, scene visualization and recommendation. They ignore the relationships among various destinations and treat each destination independently.

In this paper, we target at solving the automatic trip planning problem. A real-time trip planning system is built, in which the three aforementioned questions are effectively answered. This system contains the following three modules:

- **Destination Discovering** Discovering worldwide destinations and mapping users' diverse preferences to discovered destinations is the fundamental step of interactive travel route planning. To achieve this, more than 20 million geo-tagged photos and 200,000 travelogues are leveraged to map geo-tagged photos, destination style, and visiting time preferences to worldwide destinations, which are used in the customized and interactive trip planning module.

- **Internal Path Discovering** Discovering popular paths within a destination (called internal paths), e.g., paths in Forbidden City, has two important effects. On the one hand, it could guide a tourist to walk and take photos along the most attractive paths. On the other hand, it makes it possible to discover the representative stay time in a destination, which plays an important role in the trip planning module. To discover internal paths, we develop a novel **Internal Path Discovering (IPD) Algorithm**, which can merge incomplete paths encoded in geo-tagged photos from different individuals to reconstruct some representative paths. The discovered paths provide more complete footprints within each destination, based on which we can suggest tourists with typical travel paths and stay times in a destination.

- **Trip Planning** We propose a novel **Travel Route Suggestion (TRS) Algorithm** to suggest customized trip plans among destinations by considering travel location, travel duration, visiting time, and other preferences of users as inputs. First, a customized destination graph is dynamically constructed according to user's preferences, in which the nodes of the graph represent destinations. Second, the route planning problem is formulated as a graph analysis problem, and then solved by a dynamic programming algorithm. As a result, the proposed system can provide a customized travel plan to a tourist, which not only suggests multiple destination sequences, but also provides detailed travel information for each destination including representative travel paths and stay times within the destination. Moreover, the system can also automatically update travel plans in real time to respond to tourists' interactions.

Based on 20 million geo-tagged photos and 200,000 textual travelogues crawled from the Web, we have developed an online service to help users plan travel routes for over 30,000 sights/landmarks in over 100 countries.

To the best of our knowledge, it is the first research work to systematically investigate the automatic trip planning problem. It is also the first system that could interactively help users plan travel routes. Experimental results show the intelligence and effectiveness of the proposed framework.

2. RELATED WORK

Little existing work targets at solving the problem of automatic trip planning. Although [4] and [12] proposed to generate a tour guide from blogs, they did not consider users' preferences to automatically make a trip plan.

Some related work focuses on landmark mining using user-generated texts or photos. [10] mined city landmarks from blogs by exploiting graphic models, while [8], [13], [11] and [5] attempted to visualize, recognize, describe and summarize a scene or a landmark by leveraging geo-tagged photos.

The tremendous number of publically available geo-tagged photos greatly motivated us to address the automatic travel route planning problem. The geo-information associated with photos makes it possible to discover a tourist's travel route and thus recommend to other users.

The data source we used distinguishes our work from previous research on trajectory mining [3, 14, 6], in which the GPS trajectory data was used. [3] mined the trajectories of moving object, and demonstrated its usefulness in the analysis of traffic flows. [6] focused on sub-trajectories discovering by clustering techniques. [14] proposed to extract interesting locations and classical travel sequences using GPS trajectory data. However, GPS trajectory data is comparatively difficult to obtain and therefore is still not readily available. In this case, geo-tagged photos are a good data source to solve the automatic travel route planning problem.

3. CONCEPT DEFINITION

Destination In this work, destinations refer to popular places, such as *attractions*, *sights* or *landmarks*, within a city or a region. If an attraction or landmark is only an individual building such as the Space Needle, the *destination* also includes certain regions outside it from which tourists could also enjoy the trip to this building.

Fragment & Path The travel path of a tourist within a destination refers to the footprints encoded in geo-tagged

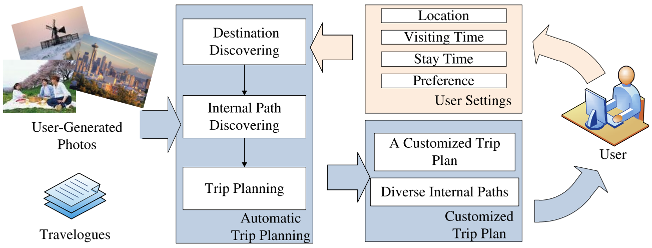


Figure 1: The illustration of the proposed trip planning framework.

photos he/she has taken and uploaded. Usually an *individual path* recovered from a tourist's uploaded geo-tagged photos is far from the *real path* he/she walked along within this destination. Thus, in order to distinguish the *individual path* and the *mined path* from multiple users using the proposed algorithm in this work, when there is ambiguous, we use *fragment* to denote the aforementioned *individual path*, and use *path* to represent the *mined path*. When there is no ambiguous, *path* could refer to all kinds of the aforementioned paths.

Route Different from the *path* which is regarded as the footprints within a destination, a *route* represents a sequence of destinations. The travel route together with the typical stay time and travel path within each destination along this route results in a brief trip plan for tourists.

4. TRAVEL ROUTE MINING

4.1 Framework Overview

The proposed framework for travel route mining is illustrated in Fig. 1. The basic inputs of our framework are visiting location and user preferences, including travel duration, visiting time, and destination style.

Three modules are designed to generate representative travel routes to meet a user's requirements. First, in the *destination discovering* module, thousands of worldwide destinations are discovered from 20 million geo-tagged photos based on a clustering algorithm. The textual travelogues and geo-tagged photos are leveraged to map users' preferences to these destinations. Then, in the *internal path discovering* module, we discover typical travel paths and stay times within a destination by introducing the *Internal Path Discovering* algorithm. Finally, *trip planning* module aims to provide both suggested travel routes among destinations and representative internal travel paths within each destination. A novel *Travel Route Suggestion* algorithm is proposed to generate trip plans according to users' requirements.

The suggested trip plans can be automatically updated in response to users' new requirements in an interactive manner, i.e., updating preferences, adding/removing interested/uninterested destinations, or adjusting stay time at each destination. As a result, according to each user's specified requirements, a trip plan with typical internal paths could be obtained.

In the following subsections, we will introduce the above three modules of our framework in detail.

4.2 Destination Discovering

4.2.1 Destination Clustering

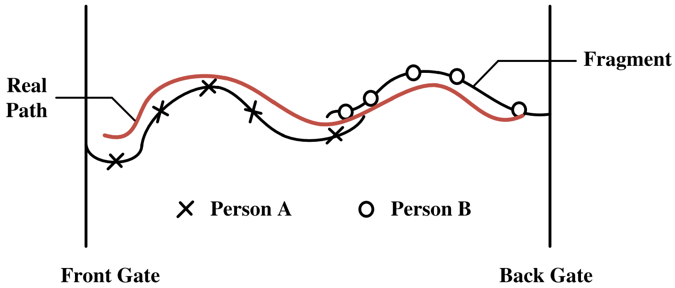


Figure 2: Motivation illustration for the Internal Path Discovering algorithm. Although both Person A and B walked from the front gate to the back gate of Forbidden City, both of them only uploaded and shared five photos onto the Web.

In order to generate travel routes for most popular locations in the world, our system first discovers popular destinations all over the world from 20 million geo-tagged photos crawled from web albums. Using the longitude and latitude as the feature of a photo, MeanShift Clustering Algorithm [1] is used to cluster the 20 million geo-tagged photos into over 300,000 clusters, from which the top 10% biggest clusters are preserved and considered as destinations. The destination distribution all over the world we have discovered is shown in Fig. 7 in Section 5.2.1.

4.2.2 Destination Naming

After destination clustering, each destination is now represented by a set of geo-tagged photos without a destination name. To facilitate the interaction with users, it is necessary to associate each destination with a textual name.

We achieve this task by leveraging the gazetteer together with the location popularity information mined from the aforementioned travelogues. In the gazetteer, each destination name is associated with the center coordinate (represented by longitude and latitude) of this destination. Therefore, we name each photo cluster based on the distance between cluster center and the coordinate of the destination name. When there are multiple destination names matching one photo cluster, we select the most popular name in the 200,000 travelogues.

4.2.3 Preference Discovery

In order to generate customized trip plans, we need to associate each destination with users' potential preferences such as destination style and popular visiting time.

1) ***Destination Style Discovery***
Based on the 200,000 textual travelogues crawled from Weblogs and professional travel websites, we could mine the top style terms such as beach, historic site and bar for each destination as introduced in [9], whose details will be omitted in this work due to space limitation. The examples of style terms for different destinations are shown in Table 1 in Section 5.2.2.

2) ***Popular Visiting Time Discovery***
Each destination would have its best or popular visiting time in a year. We could estimate this information for each destination from the number of tourist who visited this destination in each time period. Example monthly statistical analysis is shown in Fig. 8 in Section 5.2.2.

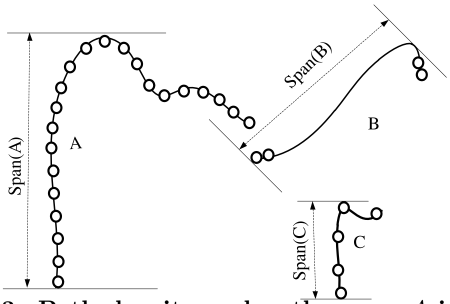


Figure 3: Path density and path span. A **is a path with relatively large path span and high photo density.** B **has relatively large path span but sparse photos. Photos on** C **is densely distributed but of short path span. The quality of** A **is better than** B **and** C**.**

4.3 Internal Path Discovering

In this section, we introduce a novel *Internal Path Discovering* (IPD) algorithm to discover typical paths and stay times within a destination.

4.3.1 Motivation

In real cases, a user usually takes photos at discrete positions along his/her travel path, out of which only a small part might be uploaded to web albums. Thus, geo-tagged photos uploaded by one user usually indicate *incomplete footprints* along his/her real travel path. See Fig. 2 for an illustration. Although both *Person A* and *B* walked from the front gate to the back gate of the Forbidden City, both of them only uploaded and shared five photos onto the Web. Only using the geo-tagged photos of either *Person A* or *Person B*, we cannot recover their complete walking paths. Moreover, estimated popular stay time using these incomplete paths in the Forbidden City will be much smaller than the real popular stay time, which will be verified in Section 5.3.2. However, if we can identify that the two tourists walked along the same path, after merging the two individual fragments together, we can obtain a more complete path from the front gate to the back gate of the Forbidden City, as shown in Fig. 2.

4.3.2 Internal Path Discovering Algorithm

1) Path Quality and Popularity

Before introducing how to merge incomplete individual fragments into more complete paths, we first introduce the properties of a path for path merging and ranking, i.e., *path quality* and *path popularity.*

Path quality represents the degree of a path or a fragment approaching to the "ideal" path. An "ideal" path is defined as a path along which a virtual user takes and uploads photos at any time and position after he/she enters a destination. That means an ideal path should have an unlimited photo density and a relatively large span. Thus, we define photo density $\rho(r)$ and path span $l(r)$ to describe the quality of a path r. Photo density refers to the number of photos per unit length on the path, and path span is the maximum Euclidean distance between the geo-coordinates of any two photos of path r, which are given by the following equations:

$$\rho(r) = \frac{\#photo \ \text{ in } \ r}{\sum_{i=1,\ldots,N-1} EuclideanDist(I_i, I_{i+1})}, \quad (1)$$

$$l(r) = \max_{I_i \in r, I_j \in r} EuclideanDist(I_i, I_j), \quad (2)$$

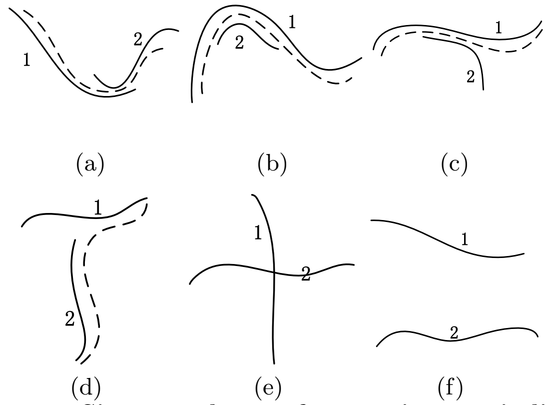


Figure 4: Six general cases for merging two individual fragments, where fragments in (a), (b), (c) and (d) are expected to be merged as corresponding dotted lines, while those in (e) and (f) should not be merged.

where $\{I_1, I_2, ..., I_N\}$ are the geo-tagged photos on path r with taken time $I_1 < I_2 < ... < I_N$.

In practice, original fragments always have limited *photo density* and a shrinked *path span*, as shown in Fig. 3. Therefore, the quality $q(r)$ of a path r is defined as an increasing function of photo density $\rho(r)$ and path span $l(r)$.

Path popularity $pop(r)$ represents the popular degree of a path that previous tourists walked along, which is defined by the number of tourists who have walked along the path. For a path merged by multiple fragments, $pop(r)$ is just the number of fragments that generate the path.

The final *path score* of path r is defined as a linear function of *path quality* and *path popularity* subject to the following constrains:

$$\lim_{\rho(r)\to\infty} s(r) = pop(r), \quad (3)$$

which means that if a path is an ideal path, we will only consider its popularity for ranking. Thus, we use the following two equations to calculate path quality $q(r)$ and path score $s(r)$:

$$q(r) = 1 - \exp\{-l(r)\rho(r)\}. \quad (4)$$

$$s(r) = pop(r) + q(r) - 1. \quad (5)$$

We first describe how to merge two fragments by leveraging path quality information, then describe the internal path discovering algorithm.

2) Fragment Merging

We should answer the following two questions: 1) how to decide whether two fragments could be merged together, and 2) how to merge two fragments.

Six general cases are listed in Fig. 4, where the fragments in (a)-(d) are expected to be merged. The dotted lines indicate the possible merging results. (e) is not expected to be merged as the directions of the two fragments are disagreed with each other. In (f), the distance between the two fragments is too large to be considered as on the same path. The distance of two fragments is defined as the distance of the closest photo pairs, denoted as $AnchorPhotos$, shown in Fig. 5. We use the twofold GPS locating error (20 meters used in this work) as the threshold to prevent pairs of fragments with large distance from merging.

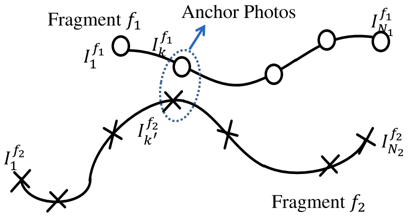


Figure 5: Merging two fragments f_1 and f_2. Photos on the fragments are organized in the chronological order. There will be six ways to merge the two fragments, i.e., f_1, f_2, $I_1^{f_1}$->$I_k^{f_1}$->$I_{k'}^{f_2}$->$I_1^{f_2}$, $I_1^{f_1}$->$I_k^{f_1}$->$I_{k'}^{f_2}$->$I_{N_2}^{f_2}$, $I_{N_1}^{f_1}$->$I_k^{f_1}$->$I_{k'}^{f_2}$->$I_{N_2}^{f_2}$, and $I_{N_1}^{f_1}$->$I_k^{f_1}$->$I_{k'}^{f_2}$->$I_1^{f_2}$, from which we will select the one with the highest *path quality* as the candidate path.

Algorithm 1 : Internal Path Discovering

Input: N fragments $\{f_1, \ldots, f_N\}$, in which each fragment contains N_i geo-tagged images: $I_1^{f_i}, \ldots, I_{N_i}^{f_i}$

Initialize the path collection: $\mathcal{R} = \emptyset$
for i = 1 **to** N **do**
  Initialize merged path: $f' = f_i$
  for $j \in \{1, 2, \ldots, N\} \setminus \{i\}$
    $f' =$ Merge(f', f_j) (see Fragment Merge part)
  end for
  if the number of photos in $f' > 5$
    $\mathcal{R} = \mathcal{R} \cup \{f'\}$
  end if
end for
return candidate path set $\mathcal{R}$

Moreover, from Fig. 5 we can see that, there are always multiple ways to merge two fragments. We calculate the qualities of these paths according to Eqn. 4 and select the one with the highest score as the merging result.

*3) **Path Discovering***

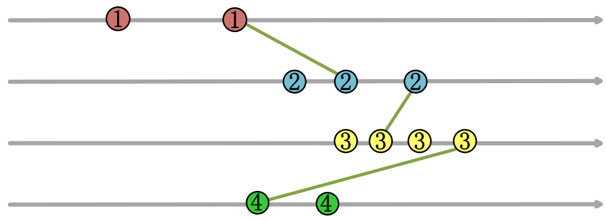
In this section, we first introduce how to obtain the candidate paths from a collection of fragments based on the aforementioned two-fragment merging algorithm, and then discuss how to rank these candidate paths to get the most representative ones.

Given N fragments $f_1, f_2, ..., f_N$ in a destination, the candidate paths could be obtained by merging these fragments together using Algorithm 1. The strategy is to pick each fragment as the target candidate path, and then merge other fragments with this path one by one using the aforementioned *Fragment Merging* method. In each two-fragment merging step, the output merged path is used to update the target candidate path. By considering each of the N fragments as a target candidate path, we could finally generate no more than N candidate paths, out of which duplicate ones will be removed.

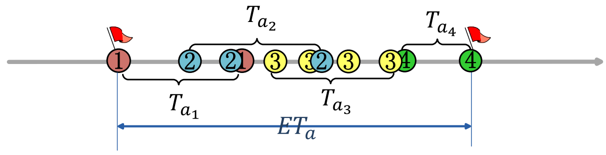
To obtain the most representative paths within a destination, we rank all candidate paths according to their *path score* $s(r)$, and select the top ones as the typical paths in the corresponding destination.

4.3.3 Stay Time Discovering

In this section, we first introduce how to calculate the time span of a discovered typical path, based on which the typical stay time in the corresponding destination could be discovered.



(a) Incomplete fragments with different timelines



(b) Discovered path with a common timeline

Figure 6: Timeline of discovered path. Balls with the same color and number represent the photos uploaded by the same user. Notice that all the fragments are incomplete and follows different timelines. (a) Incomplete fragments with different timelines. Links between *Anchor Photos* are shown. (b) Discovered path, in which all photos are aligned to the same timeline. ET_a is estimated using the time span of the photos, which is much closer to the real case compared with the stay times reflected only by individual fragments, i.e., T_{a_1}, T_{a_2}, T_{a_3} and T_{a_4}.

Notice that a path is merged from different fragments, which could have different timelines. For example, photos taken in the same position in different fragments (or say *AnchorPhotos*) might be taken in different times. Thus, in order to get the right time span of a path, the timelines of all fragments contributing to this path should be aligned to make the *AnchorPhotos* of two fragments have the same time. Based on the common timeline, we use the time span of all photos related to this path as the discovered stay time of this path. For example, as shown in Fig. 6, four incomplete fragments with different timelines are merged into a more complete path with a common timeline. From Fig. 6 we can see that, the path discovered by the Internal Path Discovering algorithm is much more complete, based on which the stay time is estimated.

Based on the statistical analysis of stay times, a stay time distribution could be obtained for each destination. Example result is shown in Section 5.3.2. For each possible stay time, we can also obtain a list of typical internal paths ranked according to the ranking principle discussed at the end of Section 4.3.2.

The time cost for a trip consists of two parts. One is the stay time in all destinations along the trip, which is discussed above. The other is the passing time, which could be obtained using the similar approach as in the stay time estimation. We will omit the details of this part due to space limitation.

4.4 Trip Planning

In this section, we propose a novel Travel Route Suggestion (TRS) algorithm to generate travel route plans for tourists. A typical trip plan targeted in this work is supposed to have the following output. "*The suggested travel routes of a one-day trip in Beijing: three hours in Forbidden City → two hours in Tian An Men Square → two hours in Qian Men. Typical internal paths related to corresponding suggested stay time in each destination are also provided for reference.*" The system also enables users to identify their preferences in advance or change the suggested trip plan in an interactive manner.

In order to mine this kind of trip plans according to users' preferences and previous tourists' experiences encoded in photo/travel collections, we need to answer the following questions: 1) How to choose typical destinations in a location? 2) How to order these selected destinations in the trip? 3) how to manage the stay time in each destination? 4) how to take into account of a user's preference? It should be noted that the above four questions are highly related with each other and cannot be easily solved separately. For example, we need to consider the typical stay time of each destination when we recommend the visiting destinations. If a user only has 5 hours to visit 2 places of interest, it might be improper to recommend him/her a route which costs more than 8 hours for most previous tourists.

In order to answer the above questions and generate a customized trip plan for a user, in this work, we formulate the aforementioned trip planning task as a graph analysis problem, which could be solved by a dynamic programming algorithm. Under this formulation, the destinations now correspond to the nodes $\mathcal{V}$ on the directed graph $G(\mathcal{V},\mathcal{E})$, and the transition from one destination to another corresponds to the transition on the graph. Thus, the problem turns to be how to find the optimal path on the graph $G(\mathcal{V},\mathcal{E})$, along which the total score is maximized subject to the constraint that the total time cost is less than or equal to travel duration set by the user.

In the following subsections, we first introduce how to dynamically construct the directed graph $G(\mathcal{V},\mathcal{E})$ according to users' preferences, followed by how to apply dynamic programming on the graph to generate customized trip plans.

4.4.1 Dynamic Graph Construction

1) ***Nodes***

Each destination is split into several nodes according to the typical stay times mined in the last section. For example, if the typical stay times in a destination are 2 hours, 3 hours, and 4 hours, with stay time probability 0.4, 0.5, and 0.1, there will be three nodes which contains different stay time property. As more nodes will lead to more time cost in trip planning, to find a trade-off solution, we only consider the stay times when their probabilities normalized by the maximal probability are higher than 0.6. In the algorithm, we also prevent the same destination with different stay times from appearing in the same route.

Each node v_i in graph $G = (\mathcal{V},\mathcal{E})$ has three attributes: the stay time t_i, node score s_i, and destination id $dest_i$, where t_i is introduced in Section 4.3.3, and s_i is determined by four factors: destination popularity S_{pop}, stay time weight w_i, destination style preference score S_{dsp}, and visiting time preference score S_{vtp}. $dest_i$ is the destination which node v represents. Two nodes may represent the same destination with different stay times.

Destination popularity S_{pop} is the number of tourists who have visited this destination in historical records. The stay time weight w_i is the stay time probability normalized by the maximal one. Destination style preference score S_{dsp} is obtained as in [9], which is the probability of the style term given the destination. We make a monthly statistic for the visiting time preference score S_{vtp}, which considers both the absolute number of tourists in that month and the ratio of the number in that month to the total number of tourists in that destination, details of which will be omitted due to space limitation.

Finally, the score of node v_i is defined as

$$s_i = (S_{pop} + \alpha S_{vtp} + \beta S_{dsp}) \times w_i, \tag{6}$$

where α and β are two parameters to make S_{vtp}, S_{pop}, and S_{dsp} have the same scale, which are both practically set to be 100 in the implementation.

2) ***Edges***

For each pair of nodes v_i and v_j, we make an edge e_{ij} to connect them, which has two attributes: edge score s_{ij} and passing time t_{ij} between v_i and v_j. The score s_{ij} is equal to the number of people who have sequentially visited $dest_i$ and $dest_j$ in a single trip. For instance, for a historical trip $A \to B \to C \to D$, the occurrences of tuples (A,B), (B,C), (C,D), (A,C), (B,D) and (A,D) are counted. The passing time t_{ij} is computed according to Section 4.3.3. The edges with zero scores will be removed from the graph.

4.4.2 Dynamic Programming for Trip Planning

Given the graph $G(\mathcal{V},\mathcal{E})$ and travel duration T specified by the user, the trip planning problem is interpreted as how to find the optimal path (in terms of total score of nodes and edges) with *time cost* (total stay and passing time of nodes and edges) less than or equal to T.

Thus the whole process is to calculate the scores of the paths between all pairs of nodes given time $t = step \le T$ and then calculate these scores given the time $t = (step + step) \le T$. It finishes when $t \le T$ and $t + step > T$.

To make it more clear, we use the function $f(v_i, v_j, t)$ to denote the score of the optimal route between nodes v_i and v_j, with time cost on the route less than or equal to t. $\mathcal{R}_{ij}^t$ is the set of nodes on the route. The goal is to compute $f(v_i, v_j, T)$ for every v_i and v_j, and then choose the best several routes for suggestion. We will show that it can be solved by a dynamic programming algorithm.

Suppose for all $t' < t - step$, the score function $f(v_i, v_j, t')$ are already known. It is easy to proof that, the optimal score of $f(v_i, v_j, t)$ can be decomposed into the computation of two sub-problems $f(v_i, v_k, t')$ and $f(v_k, v_j, t - t' - t_k)$. Therefore, computing the function has optimal substructure and overlapping sub-problems, and can be solved by applying dynamic programming. Thus, we have:

$$f(v_i, v_j, t) = \max_{\substack{v_k \in V, t' \le t \\ \mathcal{R}_{ik}^{t'} \cap \mathcal{R}_{kj}^{t''} = \{dest_k\}}} \begin{array}{c} f(v_i, v_k, t') + f(v_k, v_j, t'') \\ -s_k, \end{array} \tag{7}$$

where $t'' = t - t' - t_k$, and

$$\mathcal{R}_{ij}^t = \mathcal{R}_{ik^*}^{t^*} \cup \mathcal{R}_{k^*j}^{t'-t^*-t_k} \cup \{dest_{k^*}\}, \tag{8}$$

where v_{k^*} and t^* make Eqn. 7 achieve the maximum value. $dest_k$ is recorded into $\mathcal{R}_{ij}^t$ to avoid repeatedly visiting the same destination, as shown in Eqn. 8.

In the implementation, we initialize the graph as follows.

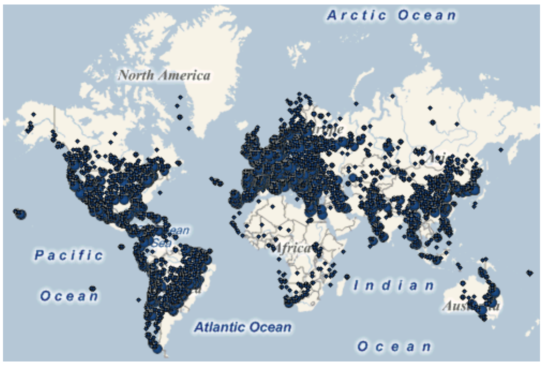


Figure 7: Discovered worldwide destinations using 20 million geo-tagged photos.

We set $\mathcal{R}_{ij}^{t} = \{i, j\}$ for all $e_{ij} \in \mathcal{E}$, and for all $t \geq 0$, we have

$$f(v_i, v_j, t) = \begin{cases} s_i + s_j + s_{ij}, & e_{ij} \in \mathcal{E}, t \geq t_{ij}, dest_i \neq dest_j \\ -\infty, & \text{otherwise} \end{cases}$$

where $t = t_i + t_j + t_{ij}$. After initializing the score matrix, we apply the dynamic programming algorithm introduced above to calculate the maximal score for each pair of two nodes within the travel duration T.

The time complexity of this algorithm is $O(|\mathcal{V}|^3(T/step)^2)$. In the implementation, we set *step* to be 1 hour.

5. EXPERIMENTS

In this section, we evaluate the proposed framework in terms of destination discovering, internal path discovering, and trip planning.

5.1 Datasets

We collected about 20 million geo-tagged photos from Panoramio (`http://www.panoramio.com/`), whose additional information such as taken time and photographer ID were also collected. Meanwhile, about 200,000 travelogues written in English or Chinese were crawled from Weblogs such as *Windows Live Spaces*, and professional travel websites like *TravelPod*, *IgoUgo*, *TravelBlog*, and *Ctrip*.

5.2 Destination Discovering

5.2.1 Destination Discovery

Over 30,000 destinations (attractions/landmarks) in more than 100 countries and territories have been discovered for travel route planning based on the 20 million geo-tagged photos. The coverage of discovered destinations all over the world is shown in Fig. 7. From Fig. 7, we can see that the discovered destinations using geo-tagged photos cover five continents, i.e., Asia, Europe, Australia, America, and Africa. Statistical data show that more than 30,000 destinations covering over 100 countries and areas are discovered.

To evaluate the accuracy of destination naming to the photo clusters, we randomly select ten photos from each of twenty randomly selected clusters. We consider the destination name of a photo cluster as correct if the contents of more than 50% photos in this cluster are exactly about this destination. Notice this is a relatively strict measure, since even the photo is taken within this destination, if the labeler cannot see the typical attraction or landmark in the photo, it will be labeled as wrong. The results show that the accuracy of destination naming is about 90%.

Table 1: Examples of discovered destination styles

Destination	Style terms
Times Square	broadway island village square
Walt Disney World	park resort magic epcot animal
Central Park	park subway skate rink garden
Statue of Liberty	island ferry statue boat liberty
Grand Canyon	canyon grand rim hike park
Golden Gate Bridge	bridge bay alcatraz cable gate
Pearl Harbor	memorial attack battleship
Niagara Falls	fall mist maid water cave island
The White House	monument memorial president
Hearst Castle	william pool house garden

5.2.2 Preference Discovery

1) Destination Style Discovery

We used the same destination style discovering algorithm as introduced in [9], which reported that the accuracy of top 20 discovered style terms for a destination is higher than 80%. In Table 1, we show some discovered destination style terms for some destinations in the United States.

2) Popular Visiting Time Discovery

Fig. 8 shows the visiting popularity distribution of six example destinations in China and the United States in different months of a year. From Fig. 8 we can see that the popular visiting time for Millennium Park is summer; for Walt Disney World in Orlando is December; and for Fragrant Hill the best visiting time is autumn. For Washington Monument and Forbidden City, all the time in a year is good for visiting.

5.3 Internal Path Discovering

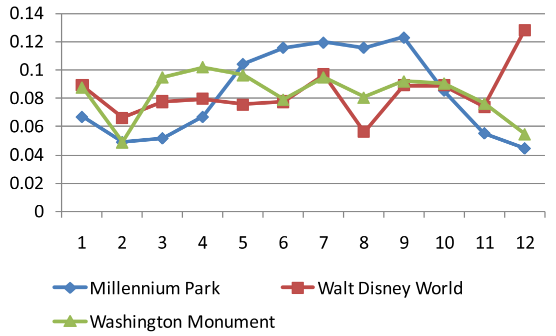
5.3.1 Internal Path Discovering

In order to evaluate the internal path discovering algorithm, we manually find fifteen *website recommended internal paths* from trip planning forums or official websites of destinations as the groundtruth. They are destinations in China and the United States , including West Lake in Hangzhou, Square Market in Lijiang, Bund of Shanghai, Temple of Heaven, Forbidden City, Summer Palace, Tian An Men Square, Fragrant Hill, Millennium Park, Central Park, Statue of Liberty, Grand Canyon, Battery Park, Ellis Island, and Pike Market.

We first check whether the ground truth path or its similar paths (i.e., more than 80% overlap with it) have been discovered by the proposed Internal Path Discovering (IPD) algorithm, or exist in the original fragments. If it is discovered by IPD rather than by the original fragments, IPD wins; while if it exists in the original fragment collection rather than the discovered path collection using IPD, IPD loses. If both the two collections contain the ground truth path, we compare the rankings of the path in the two collections. The one with higher ranking wins. IPD algorithm ranks the paths by their quality and popularity, as introduced in Section 4.3.2. Individual fragments are ranked by the number of photos on it. As a result, IPD won 6 times, lost 1 time, and broke even for other 8 times.

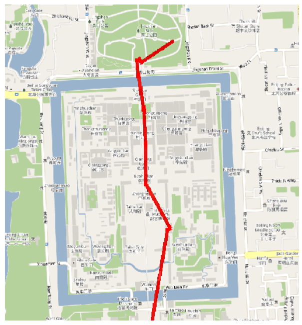
Besides the superiority in discovering website recommended path, compared with the original fragments, the IPD algorithm could discover more diverse paths within a destination which is different a lot from the website recommended path. On the other hand, although a small part of fragments might happen to be similar to the website recommended

(a) Example destinations in China.



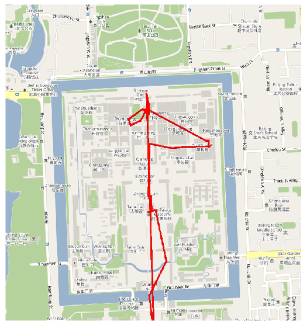
(b) Example destinations in the United States.

Figure 8: Visiting popularity distribution in a year. (a) shows example destinations in China. (b) shows example destinations in the United States.



(a) A 2-hour path.

(b) A 3.5-hour path.



(c) A 4.2-hour path.

(d) A 5-hour path.

Figure 9: Discovered internal paths in Forbidden City with different stay time. (a) 2-hour path; (b) 3.5-hour path; (c) 4.2-hour path; and (d) 5-hour path.

path, most of the individual fragments are incomplete and are difficult to cover other typical paths. In order to verify this point, we ask 20 users to manually compare the top 10 discovered paths with the top 10 individual fragments in the 8 destinations at which the two collections broke even in discovering the website recommended path. The users are asked to label "much better", "better", "equal", "worse", or "much worse" for each comparison. For example, "much better" means that discovered paths are much better than the original fragments. Out of the $20 \times 8 = 160$ comparisons, we get 16 "much better", 74 "better", 42 "equal", 28 "worse", and 0 "much worse".

Fig. 9 and Fig. 10 show the typical discovered paths with different stay times in the Forbidden City and Millennium Park. Comparing with the original fragments as shown in Fig. 11, we can see that the discovered paths are more complete and could reveal more details.

Another advantage of the discovered paths is that they can help accurately discover the typical stay time in a destination, which will be verified in the next subsection.

5.3.2 *Typical Stay Time Discovering*

Typical stay time in a destination could be estimated based on the discovered internal paths. To evaluate the stay time discovering algorithm, two baseline algorithms, i.e., the average time span of individual fragments (AVE) and Gaussian model-based method (GMB) are introduced. GMB supposes that individual stay time in a destination is drawn from Gaussian distribution, but the stay time is translated and scaled due to the noise. We have trained a model on labeled destinations and thus estimated the stay time for other destinations.

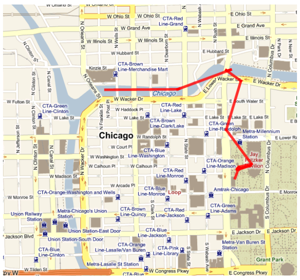
We asked 20 labelers from China to label the typical stay time of the following 10 destinations: Bund of Shanghai, Sunshine Rock, Nan Putuo Temple, Badaling Great Wall, Summer Palace, Forbidden City, Longmen Grottoes, City God Temple, Tiger Leaping Gorge, and Ancient Culture Street. The labelers are only allowed to label the destinations they have visited. Each destination was at least labeled by three labelers. The mean value of labeled typical stay times was used as the groundtruth.

The stay time differences in terms of hours between the ground truth and that estimated by each algorithm were calculated for each destination. The results are shown in Fig. 12, from which we can see that for most destinations IPD performs the best. GMB is better than AVE, since the imprecision of the stay time is usually caused by the incomplete footprints and it could be naturally considered as a kind of noise and modeled by GMB. However, GMB still cannot handle all cases in a uniform framework, since the learned parameters of GMB seem quite sensitive to different destinations. AVE has inferior performance compared with the other two methods, since no effective techniques are leveraged to deal with the problem caused by incomplete individual fragments. The results further confirm the assumption that individual photos from Web albums show incomplete footprints and the proposed IPD algorithm overcomes this challenge to a large extent.

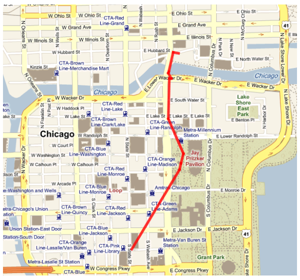
Fig. 13 (a)-(d) show the stay time distributions of tourists who visited Forbidden City, Bund of Shanghai, Madison Square Garden Center, and Golden Gate Bridge. We can see that, the stay time distribution discovered by IPD is more reasonable and robust than the other two methods.

5.4 Trip Planning

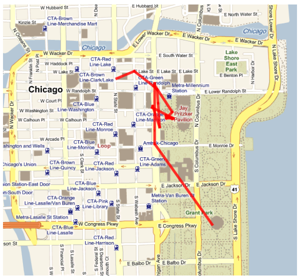
We conducted extensive user study to evaluate the proposed trip planning framework. Twenty graduate students were asked to complete the task as follows. First, they in-



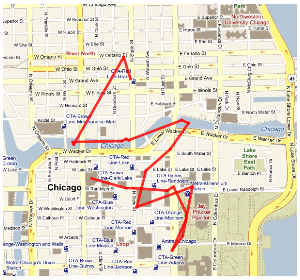
(a) A 1.6-hour path.



(b) A 2.2-hour path.

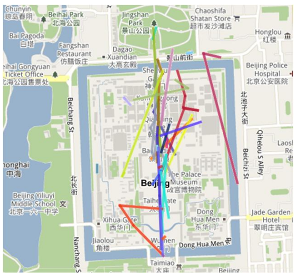


(c) A 3.6-hour path.

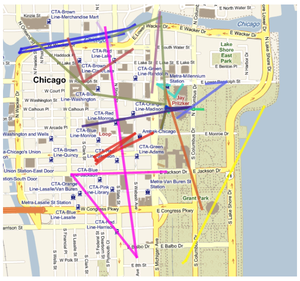


(d) A 5-hour path.

Figure 10: Discovered internal paths in and around Millennium Park with different stay time. (a) 1.6-hour path; (b) 2.2-hour path; (c) 3.6-hour path; and (d) 5-hour path.



(a) Forbidden City



(b) Millennium Park

Figure 11: Individual fragments. (a) Sample fragments in the Forbidden City. (b) Sample fragments in and around Millennium Park. This figure is better viewed in color. Fragments with different colors are from different individuals.

put requests on travel location, travel time duration, visiting time, and destination styles, to trigger trip plans. They were asked to score the suggested routes using 1 to 5 ratings, and select the best plan made by different approaches for each destination. Then, if they want, they can change any of the input, add/delete destinations to make a customized and satisfactory trip plan. Similarly, the students who have never been to the destination were not allowed to label it.

Four aspects are asked and evaluated for travel route planning: (1) ***representativeness*** (i.e., to what extent destinations in the recommended route reflect the culture and characteristics of the location.); (2) ***diversity*** (i.e., to what extent destinations in the recommended route provide rich information about the location.); (3) ***rationality*** (i.e., to what extent destinations are reasonably arranged within time requirement.), and (4) ***overall satisfaction*** (i.e., are you satisfied with the suggested route?).

To evaluate the proposed interactive trip planning method, we compared the following three algorithms: 1) the proposed Travel Route Suggestion (TRS) algorithm, 2) TRS, except

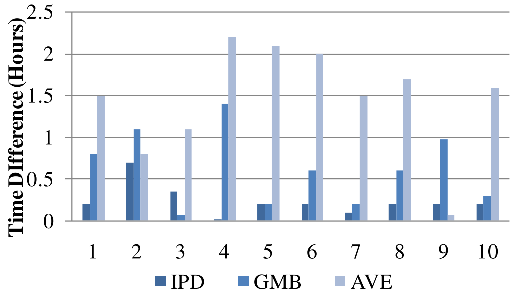
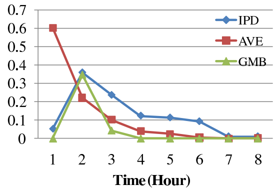
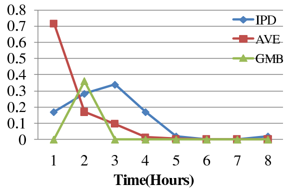


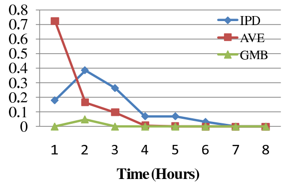
Figure 12: Popular stay time evaluation based on the ground truth given by native long-time residents in different cities. The x-coordinate indicates the index of used destinations. From 1 to 10 are Bund of Shanghai, Sunshine Rock, Nan Putuo Temple, Badaling Great Wall, Summer Palace, Forbidden City, Longmen Grottoes, City God Temple, Tiger Leaping Gorge, and Ancient Culture Street.



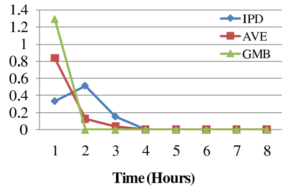
(a) Forbidden City



(b) Bund of Shanghai



(c) Madison Square Garden Center



(d) Golden Gate Bridge

Figure 13: Stay time distributions using different approaches. The distribution is determined by the number of people who visited the destination under certain stay time. (a)-(d) show the distribution in Forbidden City, Bund of Shanghai, Madison Square Garden Center, and Golden Gate Bridge.

that the individual fragments are used for stay time estimation, denoted by TRS-AVE, 3) Random Trip Planning, denoted by RTP. RTP randomly selects destinations in a location and randomly arranges their orders. Ten cities in China, i.e., Beijing, Shanghai, Sanya, Tianjin, Dalian, Xiamen, Hangzhou, Harbin, Xi'an, and Chengdu, were used in the user study.

The results are shown in Fig. 14, from which we can see that the trips suggested by TRS contain more representative and diverse destinations, and TRS has organized them more rationally. Although TRS-AVE could select representative destinations, due to the poor stay time estimation in each destination, the destinations could not be well-organized to be a satisfactory trip. RTP is the worst one, which fails in both destination selection and organization.

To illustrate the interactive process of our trip planning system, we shows two example results in Fig. 15. To response to user A who desired to have a two-day trip in Beijing in autumn, TRS automatically recommended the following plan: visiting Fragrant Hill to enjoy red leaves in the

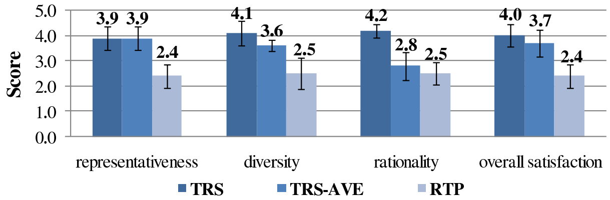


Figure 14: Comparison results of different algorithms for trip planning.

(a) User A (b) User B

Figure 15: Interactive trip planning results. (a) the second day trip in Beijing requested by user A, where dotted lines links the destination (i.e., San Li Tun) added by user. The final trip adopted by user A is 3 hours in ① Forbidden City, 2 hours in ② Tian An Men Square, 2 hours in ③ Qian Men, and 2 hour in ④ San Li Tun. (b) a trip plan with preference of snacks in Shanghai. The trip is 2 hours in ① Bund of Shanghai, 3 hours in ② Oriental Pearl Tower, and 3 hours in ③ Town's God Temple. Town's God Temple is famous for snacks.

first day; visiting Forbidden City (3 hours), Tian An Men square (2 hours) and Qian Men (2 hours) in the second day. The user was generally satisfied with the results, but he had particular interests in the bar culture of Beijing. In this case, he decided to modify the results and planned the two-day trip as follows: on the first day, visiting Fragrant Hill at the day time, and Hou Hai in the evening; on the second day, he preferred to visit the destination we planned at first and then go to San Li Tun at the evening (Hou Hai and San Li Tun are places famous for bar culture). The trip plan on the second day is shown in Fig. 15(a). User A commented on our system that "It is a very interesting system, which provides representative and rational travel routes suggestions. I hope it can include more attractive destinations for young people, such as Huan Le Gu and Nan Luo Gu Xiang".

User B was especially interested in snacks, and she expected to have a one-day trip in Shanghai. The suggested trip plan using our system is shown in Fig. 15(b), which starts from The Bund of Shanghai (2 hours), to the Oriental Pearl Tower (3 hours) and ends at Town's God Temple (3 hours). Town's God Temple is actually very famous for snacks. User B felt enjoyable about the suggested trip plan.

To evaluate the efficiency of the TRS algorithm, we sampled ten cities to make trip plans. As shown in Table 2 and 3, we present average time cost for different numbers of destinations and different time durations. We can see that our system could recommend trip plans for tourists in real-time.

6. CONCLUSIONS

In this paper, we have presented a novel automatic trip planning framework, by leveraging Web scale geo-tagged photos and textual travelogues. Owning to the 20 million geo-tagged photos, an online system has been developed to help users plan travel routes for over 30,000 sites/landmarks in more than 100 countries and territories. To the best of our knowledge, it is the first research work to systematically investigate the trip planning problem. It is also the first system that could interactively help users plan travel routes. Experimental results have shown the intelligence and effectiveness of the proposed framework.

Table 2: Average time cost of one-day trips for different numbers of destinations.

Destination Number	20	30	40	50	60
Time (Seconds)	0.03	0.05	0.11	0.17	0.28

Table 3: Average time cost for different travel durations with destination number fixed to be 60.

Travel Duration	4 hours	8 hours	2 days	3 days
Time (Seconds)	0.11	0.28	0.53	0.75

7. ACKNOWLEDGEMENT

The first author and the fourth author were partially supported by the National Natural Science Foundation of China (No. 60975001), Tianjin Research Program of Application Foundation and Advanced Technology (No. 10JCYBJC07700), and Specialized Research Fund for the Doctoral Program of Higher Education (No. 20090032110028).

8. REFERENCES

[1] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach toward Feature Space Analysis. *TPAMI*, 2002.

[2] D.J. Crandall, et al. Mapping the World's Photos. In *WWW*, 2009.

[3] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory Pattern Mining. In *SIGKDD*, 2007.

[4] H. Kori, et al. Automatic Generation of Multimedia Tour Guide from Local Blogs. In *MMM*, 2007.

[5] I. Simon, et al. Scene Summarization for Online Image Collections. In *ICCV*, 2007.

[6] J.G. Lee, et al. Trajectory Clustering: A Partition-and-Group Framework. In *SIGMOD*, 2007.

[7] L. Kennedy and N. Naaman. Generating Diverse and Representative Image Search Results for Landmarks. In *WWW*, 2008.

[8] N. Snavely, et al. Photo Tourism: Exploring Photo Collections in 3D. In *SIGGRAPH*, 2006.

[9] Q. Hao and R. Cai, et al. Generating Location Overviews with Images and Tags by Mining User-Generated Travelogues. In *ACM MM*, 2009.

[10] R. Ji, et al. Mining City Landmarks from Blogs by Graph Modeling. In *ACM MM*, 2009.

[11] T. Rattenbury, N. Good, and M. Naaman. Towards Automatic Extraction of Event and Place Semantics from Flickr Tags. In *SIGIR*, 2007.

[12] T. Kurashima, et al. Mining and Visualizing Local Experiences from Blog Entries. In *DEXA*, 2006.

[13] Y.T. Zheng, et al. Tour the World: Building a Web-Scale Landmark Recognition Engine. In *CVPR*, 2009.

[14] Y. Zheng, L. Zhang, X. Xie, and W. Ma. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In *WWW*, 2009.