e-Turist: An Intelligent Personalised Trip Guide

Božidara Cvetković, Hristijan Gjoreski, Vito Janko, Boštjan Kaluža, Anton Gradišek and Mitja Luštrek
Department for Intelligent Systems, Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

Igor Jurinčič, Anton Gosar, Simon Kerma and Gregor Balažič
Faculty of Tourism Studies – Turistica, University of Primorska, Obala 11a, SI-6320 Portorož, Slovenia
E-mail: boza.cvetkovic@ijs.si

Keywords: tour planning, recommender system, route optimization

Received: June 16, 2016

We present e-Turist, an intelligent system that helps tourists plan a personalised itinerary to a tourist area, taking into account individual's preferences and limitations. After creating the route, e-Turist also offers real-time GPS guidance and audio description of points of interest visited. Here we focus on two main components, the recommender system and the route planning algorithm. We also present some use cases to highlight e-Turist functionalities in different configurations.

Povzetek: Predstavljamo inteligentni sistem e-Turist, ki turistom pomaga izdelati personaliziran načrt poti po določeni turistični regiji. Pri tem upošteva posameznikove želje ter omejitve. Po izdelavi poti e-Turist omogoča tudi vodenje s pomočjo sistema GPS in opis zanimivosti s pomočjo zvočnih datotek. V članku podrobneje predstavimo dve glavni komponenti sistema, in sicer priporočilni sistem ter algoritem za načrtovanje poti. Poleg tega predstavimo nekaj primerov uporabe, ki prikažejo delovanje e-Turista v različnih konfiguracijah.

1 Introduction

Tourism is one of the fastest growing global industries. Though suffering a setback during the late-2000s recession, the tourism sector has seen a robust growth for six consecutive years,[1] with the number of international tourist arrivals growing by 4.4 % from 2014 to 2015 and totalling 1.2 billion worldwide. From 25 million international tourists in 1950, the number is forecast to reach 1.8 billion by 2030.[2] Currently, tourism generates 9 % of global GDP through direct and indirect impact, creates 1 in 11 jobs, and represents 6 % of world's exports. At least 53 % of international tourists (600 million) travelled for holidays, recreation, and other forms of leisure. In addition, the number of domestic tourists is estimated to be between 5 and 6 billion.[2] Tourism represents an important part of economy in individual countries, contributing to 9-13 % of national GDPs in countries such as Italy, Germany, France, and Slovenia, whereas the contribution in countries such as Croatia, Malta, and Iceland is over 23 %.[3]

Tourists can plan their trips either individually or using services provided by tourist agencies. Each approach has advantages and disadvantages. By joining an organised group, little planning is required. Such groups typically employ a licensed tour guide who is familiar with the pre-planned itinerary and individual stops on the route. However, such groups typically focus on a smaller number of prime-level tourist destinations, suitable for large group visits. A fixed itinerary may also conflict with the interests of group members who would prefer spending more time at certain locations or visiting nearby attractions that are not included in the route. Individual tourists are more flexible in designing their itinerary and choosing points of interest (POI) to visit. Individual tourists may also find less-known destinations attractive to visit, often even more so, as they are better suited for smaller groups of people and are more diverse in the activities they offer. However, the initial planning is more complex. The tourists have to compile travel information from various sources, such as tourist guidebooks, websites, tourist information centres, etc. Apart from scattered information sources, they have to consider opening times, geographical distribution of POIs (distances between locations), and the availability of services such as accommodations and restaurants. This makes designing a good itinerary quite difficult. However, since each tourist has unique preferences related to type of activities, choice of food, special interests, and potential limitations due to physical or other impairments, fixed itineraries from a guidebook or similar source are not satisfactory. A platform containing all relevant information about a certain region that would allow simple creation of personalised itineraries could represent a significant advantage both for tourists (by facilitating the planning) and the local tourism sector (by highlighting local POIs and services that would otherwise stay overlooked).

In her extensive research, Molz is introducing the notion of "flashpackers" (affluent interactive travellers with a budget, higher than backpackers, spending freely for activities at their chosen destinations), arguing that

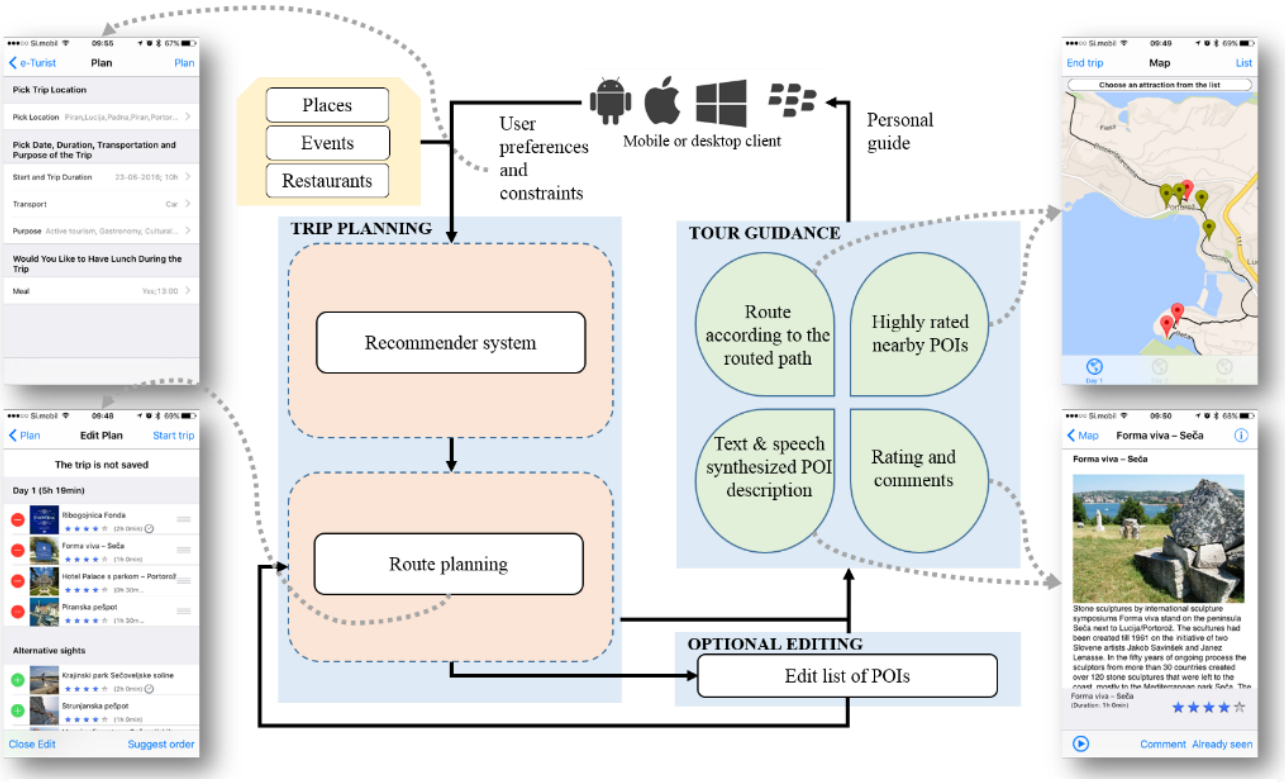

Figure 1. e-Turist system overview.

the rapidly evolving context of new mobile and media technologies has made interactive travel an ever more significant element of modern social life, especially in a mobile world.[4] In the last decade, extensive research has been carried out with the goal of improving tourist experiences,[5] [6] [7] partially stimulated by the fact that smartphones have become powerful computing tools with access to cloud-based services, which allow personalisation and real-time functionality. The potential for such applications is in creating personalised tours which, as discussed above, improve user experience over guidebooks that offer generic visitor tours through a city or a region.[5] Personalisation is achieved through user specifying their preferences and constrains, such as the available time and start and end point of the tour. While the current existing applications use various approaches to offer best experience, they have some common components. Typically, a recommender system is used [5] [6] [7] to create a list of most appropriate POIs. Some implementations are simple, for example recommending the nearest attractions based on location. Other are more advanced and use artificial intelligence approaches, such as various types of filtering (knowledge-based, demographic, hybrid, etc.), automatic clustering algorithms, approximate reasoning methodologies, such as fuzzy logic, or ontologies. Often, a route planning functionality is included. Here, the task of the application is to present the optimal route between POIs. Different approaches of solving the Travelling Salesman Problem (TSP) [8] are implemented in this module.[5] An example of such systems is a pilot study by Garcia et al.,[9] which was developed for the city of San Sebastian in Spain and uses a basic recommender system and route planning. However, the main issue with such systems is that they were mostly developed as pilot projects and/or have not left the academic environment.

On the other hand, there are several mayor players in the tourist industry that use various approaches. TripAdvisor [10] contains extensive lists of attractions with user reviews and also allows users to book flights or hotels. Other applications are more specific. For example, Triposo [11] includes a ranked list of attractions by category, Roadtrippers [12] is designed for car travellers and shows potential POIs close to the planned route, and Route Perfect [13] includes recommendations based on explicit user preferences and a route planning component, but lacks day-to-day sightseeing itineraries in individual cities.

As seen in this quick overview, several approaches have already been developed to help tourists, but we were unable to find a widely-used product that would contain the complete functionality, namely a recommender system, tour routing, and a real-time guiding component. In this paper, we present our solution, called e-Turist (e-Tourist),[14] [15] which is an intelligent platform designed to help individual tourists or small groups prepare a customised sightseeing/travel plan and offer them an experience comparable to one offered by a professional tourist guide. To some extent, the platform promotes smaller tourist-oriented businesses, which is important, since they contribute to local economy and ensure jobs in local environment. e-Turist was initially developed for tourist areas in Slovenia and is now being expanded to include destinations worldwide. The system can be accessed either through a mobile application or a website. Tourists

enter their preferences and the system prepares a customised itinerary which includes the most appropriate points of interest with the shortest route connecting them. In addition, the application guides the tourist using GPS and offers information about attractions either in text or audio format. We discuss the main components of e-Turist, namely the recommender system that creates a list of most interesting places for the tourist to visit, and the route planner which calculates the optimal route between these places. We also present some use cases, which highlight the main functionalities of the system.

2 System overview

The e-Turist system is designed as a web service (software as a service - SaaS) and is accessible either through a web-browser or a smartphone app. The system architecture is shown in Figure 1.

When opening the application, the user enters his preferences regarding the trip. Those may be basic, such as the trip duration, or more detailed, by creating a user profile that includes information such as age, education, or budget (these parameters were considered relevant by tourism experts). The application then creates the proposed itinerary, which is done in two steps. First, the recommender system creates a list of appropriate POIs chosen dynamically according to the user's preferences. In the second step, the route planning module proposes the optimal route between a subset of chosen POIs, based on the available time and some other parameters, such as stops for meal during the trip. The user can then customize the proposed route in one or more steps. The recommender module and the route planning module will be presented in more detail in the following sections.

When on the tour, the application uses real-time guidance based on the GPS data. When reaching each stop, the application offers information about the POI, either in text or audio form. During the tour, the application also highlights nearby POIs so that the user can decide whether to make a detour. At the end, the user can rate the visited POIs with 1–5 stars.

The browser application is essentially identical to the mobile one, with the exception of the GPS guidance being absent. Interchangeable use is possible, such as editing and saving the route via the browser application and opening it on the mobile device.

e-Turist employs an audio module, which offers audio POI descriptions. To create audio descriptions in the Slovenian language, the voice synthesizer Govorec [16] is used. Govorec was developed in collaboration between Jožef Stefan Institute and the company Amebis. For other languages – currently, English, German, and Italian are supported – Microsoft Speech API [17] is used. The audio files are generated automatically when a POI description is entered into the database and are stored there for user access.

The administration module is used to edit the database of POIs. For destinations (regions), geographical area and approximate radius are defined. For each POI, a description, one or more images, and metadata are defined. The metadata includes the type of the POI, opening hours, accessibility, expert rating, suitability for different types of tourists, etc. The administration module highlights missing information in the descriptions, and allows monitoring the visits and user ratings for individual POIs, which allows tourism workers to improve the service.

3 Recommender system

In order to prepare the most suitable itinerary for individual tourists, the recommender system module uses a combination of constraints filtering, knowledge-based recommendation and collaborative filtering, as shown in Figure 2.

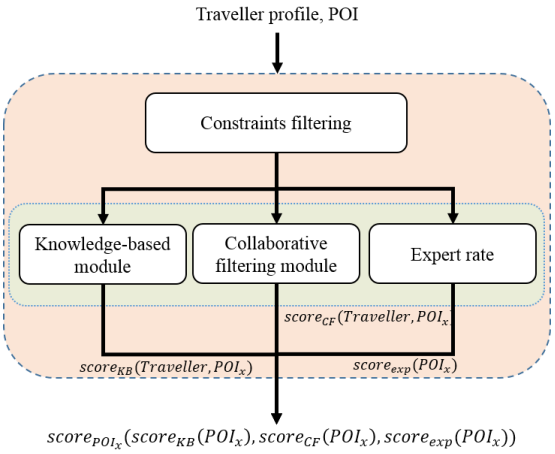

Figure 2. Recommender system schematics.

The **constraints filtering module** utilises "hard" constraints that exclude the POIs that do not meet the user's limitations and key requirements. These constraints are (i) the location, (ii) the purpose of the trip – currently the options are active tourism, cultural heritage, entertainment and gastronomy, (iii) the opening hours of the POIs, and (iv) mobility limitations for physically impaired users. For example, if a user is interested in active tourism, he will not be suggested to visit museums but venues such as adrenaline parks instead.

The **knowledge-based module** computes the distance between each POI and the user based on four sets of expert-defined characteristics: age, education, country of residence, and budget. There are five age groups: age up to 26, 27 to 36, 37 to 45, 46 to 55, and 56 and higher. There are three education groups: primary, secondary, and tertiary, and also three budget groups: low, medium, and high. The country parameter corresponds to countries whose tourists often visit that POI. Each POI is assigned one or more groups for each characteristic, except for budget, where it can only have one value. Each group is assigned numerical value (e.g. 0 is low, 1 is medium, and 2 is high budget), respectively, which are used to compute the Euclidian distance between the user and POI characteristics, later transformed into score of the individual characteristic.

The score for age, education, and country characteristic is computed using Equation 1. The absolute distance between the user characteristic and POI

characteristic is divided by number of groups per characteristic and subtracted from the perfect-fit value of 1. The score for budget is computed with the same approach unless the $user_{budget} \geq POI_{budget}$, in which case the score is always set to the perfect-fit value of 1. In case the characteristic value is not defined for a POI or the user, the score is set to the medium-fit value of 0.5.

$$score_{char} = 1 - \left|\left(\frac{user_{char} - POI_{char}}{char_{groups}}\right)\right| \quad (1)$$

The final score $score_{KB}$ of the knowledge-based module is calculated using Equation 2. The score is afterwards normalised to interval from 1 to 5.

$$score_{KB} = \frac{score_{age} + score_{edu} + score_{country} + score_{budget}}{4} \quad (2)$$

The **collaborative filtering** uses a memory-based approach to assign the $score_{CF}$ for the user. Each instance represents one user. Its feature vector is composed of ratings per POI given by the individual user. In case the user did not rate a POI, that value is defined as a missing value. We used the *k*-nearest neighbour algorithm [18] to find *k* similar users, once a sufficiently high number of users have been recorded in the database. The final score for an individual POI is an average value of scores per POI for the *k*-nearest neighbours.

The final result of the hybrid recommender system is the final score calculated with Equation 3. It is a weighted sum of the knowledge-based rating, the collaborative-filtering rating, and the expert rating (POI rating provided by experts).

$$score = w_1 score_{KB} + w_2 score_{CF} + w_3 score_{exp} \quad (3)$$

We discuss the weight values in the following subsection.

3.1 Experimental evaluation and parameter settings for the recommender module

The recommender system was tested on data of 24 users with different age and background who were given a list of 90 POIs from the Heart of Slovenia region [19] . The users were asked to rate the POIs they are familiar with from 1 to 5 stars. These ratings were then compared to the recommender system modules, with the goal of accurately predicting the rating of POI as would be given by the current user. This helped us define the final weights of Equation 3.

Each recommender system module was evaluated for the performance using the mean absolute error (MAE) over all 90 POIs, as presented in Equation 4. The lower the MAE value, the better the prediction of the modules.

$$MAE = \frac{\sum_{i=0}^{n} |(score_{true} - score_{predicted})|}{n} \quad (4)$$

In the first step, we evaluated the baseline approach which rates all POIs with the score 3, being the medium score. The MAE value of the baseline approach is 1.05 rating.

In the second step, the expert rating was evaluated. It amounted to 0.99 rating.

In the third step, the knowledge-based module was evaluated. The users' budget preferences and demographic data were used to compute the $score_{KB}$. The evaluation of the score estimates per user yielded MAE of 0.98.

Next, we evaluated the collaborative filtering module. Before evaluation we had to define the number of neighbours that would be used by the *k*-nearest neighbours algorithm. We tested the algorithm for *k*=1 to *k*=5 and settled for *k=4* as it returned the best results. The MAE of the collaborative filtering module using 4-nearest neighbours algorithm was 0.87 score.

Before setting the weights of the three modules, we decided that the weight of the expert rating should be smaller than the weights for other ratings, otherwise the recommendations would not be personalised. We decided to assign it to 0.2. Since the difference between the knowledge-based and collaborative filtering module was not large, we gave them equal weights of 0.4 as shown in Equation 5. In case the expert score is missing, the recommender modules are assigned weights of 0.5 (Equation 6), and in case the knowledge-based or the collaborative filtering rating is missing, the 0.8 weight is assigned to the remaining one (Equation 7 and 8). If there is only one score, it is assigned the full weight.

$$score = 0.4(score_{KB} + score_{CF}) + 0.score_{exp} \quad (5)$$

$$score = 0.5 score_{KB} + 0.5 score_{CF} \quad (6)$$

$$score = 0.8 score_{KB} + 0.score_{exp} \quad (7)$$

$$score = 0.8 score_{CF} + 0.score_{exp} \quad (8)$$

The results of the above equations were compared to individual modules. The results are presented in Table 1. The MAE value of the baseline approach is 1.05 score and the MAE of the final rating is 0.86 score, which is better than the MAE of the knowledge-based, collaborative-filtering rating, and expert rating alone.

Approach	**MAE (rating)**
Baseline	1.05
Expert score	0.99
Knowledge-based module	0.98
Collaborative filtering module	0.87
e-Turist final rating	**0.86**

Table 1. Results of the baseline rating, individual modules ratings, and the e-Turist recommender system final rating.

4 Route planning algorithm

The task for the route planning module is to prepare the best route for the tourist from the list of POIs generated

by the recommender system, taking into account the attractiveness of individual POIs and the time limitations of the tourist. This task is reminiscent of the well-known knapsack problem,[20] where the best set of items (in our case POIs) with weights W_i (the time needed to visit the POI) and values V_i (POI attractiveness) have to be chosen so that the overall weight does not exceed the limit and that the total value is maximal. Such problem can be solved in a pseudo-polynomial time with dynamic programming.[21] However, in our practical implementation, the algorithm needs additionally to take into account the time needed to visit the chosen items (POIs) – which is an estimation problem by itself. The path duration estimation problem opens a new sub-problem inside the knapsack problem, i.e., how to find the route with minimal duration, given the POIs. This sub-problem is also a known problem in the theory of computation, called the TSP.[8] Therefore, in our case we have a "modified knapsack problem" where the total value of the chosen items changes dynamically (with each algorithm iteration). Additionally, the path estimation is computationally expensive process, because it requires solving an NP-hard problem, i.e., TSP. Moreover, the final algorithm execution time should be in the range of seconds, because it will be used in a real-time POI recommender application, where the user needs instant feedback from the system. Because of these reasons, two simplifications were proposed: a greedy approach for knapsack problem (POIs ordered by value) and an adapted TSP for path duration estimation (finds near optimal solution).

The first step in our algorithm is the estimation of the importance of an individual POI (how attractive a POI is – POI value). We defined the POI value considering three factors:

- POI's rating (provided by the recommender system)
- POI's visit duration
- POI's local reachability duration

The first factor is a value that is provided by the recommender system ($score$) which varies from 1 to 5 (attractiveness of the POI). The next factor, the POI's visit duration, represents the average time that a tourist needs in order to see the POI, which is defined by an expert in the POI database. The final factor, the POI's reachability duration, is a variable that represents how far a POI is from its nearest neighbours. In other words, if a POI is far from the rest of the POIs, the value for this variable would be greater compared to the reachability duration of the other POIs. Using this information, the POIs that are "outliers" (far from the rest of the POIs) are "punished" because the tourist needs more time to reach them. For the estimation of this variable we used a partial implementation of the Local Outlier Detection (LOF) algorithm. In particular, we used the local reachability distance (*lrd*) metric in order to estimate how far a POI is from its neighbours. The LOF algorithm and its mathematical definitions are described by Breunig et al [22] .

The mathematical definition of the POI importance, which includes all the three factors, is presented in Equation 7.

$$V = \alpha * score + (1 - \alpha) * (1 - P_{norm}) * rate \quad (7)$$

The variable $score$ is the POI's rating, which varies from 1 to 5. The variable P_{norm} represents the normalised value (from 0 to 1) of the P, which is a sum of the POI's visit duration (V_{d}) and the POI's local reachability distance (*lrd*) – note that V_{d} and *lrd* take values in hours:

$$P = V_d + lrd \quad (8)$$

Because the idea is to "punish" the POIs that require longer time to visit and the ones that are far away, the normalised P is subtracted from 1 (the bigger the value of P_{norm}, the less important the POI). α is a parameter regulating the importance of the evaluation value (V) on one side, and the POI's visit duration and POI's local reachability distance (P) on the other side. The empirical analysis of the data showed that 0.5 is a reasonable trade-off value for α. This way, both sides of the equation are equally weighted in the final importance value V. To summarize, the first term in Equation 8 is considered as it is, while the second term is reduced by the factor corresponding to the time needed for the visit ($1 - P_{\text{norm}}$).

In the next step of the algorithm, all the POIs are ordered by the importance value V. Using a greedy strategy, the algorithm then adds items (POIs) in the knapsack until the limit is reached. With each POI added, the weight of the knapsack is checked – if the weight (time duration) of the chosen POI combination is below the maximum weight (total available time of the tourist). In addition, with each added POI, a TSP algorithm estimates the path duration, which is also checked with the time limit of the tourist. This way, a near optimal combination of POIs is found.

Additionally, the algorithm checks for nested POIs, since there may be more than one POI at the same location. For example, the Ljubljana castle additionally has a museum and a tower. Therefore, if the algorithm has chosen some of the nested POIs in the optimal route, it additionally adds all the other nested POIs at that location, and by doing so it also updates the time for the visit and checks the constraints of the knapsack.

After the combination of POIs is found, the algorithm checks whether the user prefers to start from the nearest POI. If this is the case, the order of the POIs is recalculated with a modified version of the original TSP which creates a path using a fixed start POI.

In the final step of the algorithm, it is checked whether the tourist has chosen multiple days for sightseeing. In the case of a multiple-days visit, the POIs are segmented into groups, each group corresponding to one day of the trip. Additionally, it is checked if the user plans a meal at a particular hour of the day. If this is the case and there are restaurant-POIs in the list of POIs, the best (according to the evaluation value and the location) restaurant is chosen. That day's route is modified in such a way that the tourist is near the restaurant during the

Algorithm 1: Route planning algorithm

```
Input:
    allPOIs //POIs from the recomm. system
    transport //the means of transport
    numDays //number of days for the visit
Result:
    FinalPOIs //The near-optimal list of POIs
    AlternativePOIs //Alternative POIs

duration = 0
//order by value V
orderedPOIs = OrderPOIsByValue (allPOIs)

//start adding POIs ordered by the value, add the rest to the alternative list
For POI in orderedPOIs
  If duration + POI.duration < totalDuration
    tempPOIs.add(POI)
    pathDuration = TSP(tempPOIs, transport)
    duration =
    updateDuration(POI.duration, pathDuration)
    If POI is child at location
      tempPOIs.add (POI.parent)
      duration =
      updateDuration(POI.parent.duration)
    If POI is grandchild at location
      tempPOIs.add (POI.grandParent)
      duration = updateDuration
      (POI.grandParent.duration)
  Else
  AlternativePOIs.add(POI)
End

//Find the near-optimal path and time to visit all chosen POIs
Solution = TSP(tempPOIs, transport)

//Take care of the parent-child relation
FinalPOIs= OrderByParentChildRelation (Solution)

//If the user is less than 1 km away from some of the chosen POIs, start from the nearest POI
If StartLocation.DistanceToNearestPOI < 1km
    FinalPOIs= Reorder(FinalPOIs)

//if there are multiple days, segment the solution to multiple lists of POIs, for each day
If numDays > 1
    FinalPOIs= SegmentByDays (FinalPOIs)

Return FinalPOIs, AlternativePOIs
```

previously chosen meal time. This is done by dividing the problem in two segments (before and after the meal) and calling the TSP solver for each. First, for before the meal, the end location is fixed to the restaurant, and next, for after the meal, the start location is fixed to the restaurant.

The pseudo code of the route planning algorithm is given with Algorithm 1.

Since the TSP solves a sub-problem in the general knapsack problem, its execution time needs to be in the range of milliseconds. Therefore, for its implementation, we considered an open-source algorithm [23] that finds a near-optimal solution. It is a greedy approach with additional optimization mechanisms. The empirical tests showed that for our scenario (up to 200 POIs) it almost always finds the optimal solution, and also the execution time is acceptable i.e., several milliseconds. Additionally, we modified the original algorithm so that it can use start and end POI. This option is required when the users wants to start from the closest location to them (e.g. by using the GPS signal of the tourist's smartphone). Also, when the user selects meal-time, the path is divided into two parts: before and after lunch.

In order to decrease the execution time, we call the TSP algorithm only when the time needed to visit all the POIs reaches a predefined threshold of 80% of the total available time. In other words, when the time required to visit the POIs reaches 80% of the total available time, the TSP is called to estimate the exact path duration. Otherwise, every time a POI is added, the path is estimated simply by adding the path duration to the nearest POI. A detailed evaluation of the routing algorithm is a problem on its own and will not be addressed in this paper.

5 Use cases

In this section, we show different aspects of e-Turist functionality. Let us consider three different tourists, Ada, Bob, and Ted. The details about these users are presented in Table 2.

		User		
		Ada	**Bob**	**Ted**
Profile	Age	/	60	/
	Country	/	Germany	/
	Budget	/	high	/
	Education	/	tertiary	/
	Mobility lim.	/	yes	/
Interest	Active tourist	✓	×	×
	Gastronomy	✓	×	×
	Entertainment	✓	×	×
	Cultural heritage	×	✓	✓
	Include lunch	✓	×	✓
Transport	Car	✓	✓	×
	Walking	×	×	✓
Location	Slovenian Istria	✓	✓	×
	The Hague	×	×	✓

Table 2. Users and their interests. Note that Bob creates his profile only in the second step of the use case.

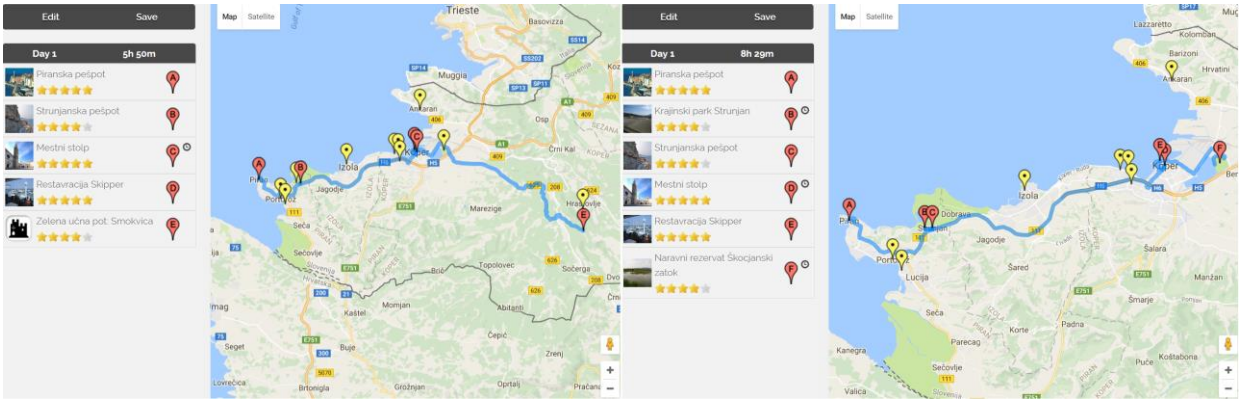

Figure 3. Left: Initial route, proposed for Ada. The sidebar on the left shows the list of the recommended POIs (red balloons), while the right side shows the route on the map (blue line) and other nearby POIs (yellow balloons). Expert ratings are marked with stars. Right: The new route, proposed for Ada, after she manually edits the POI list.

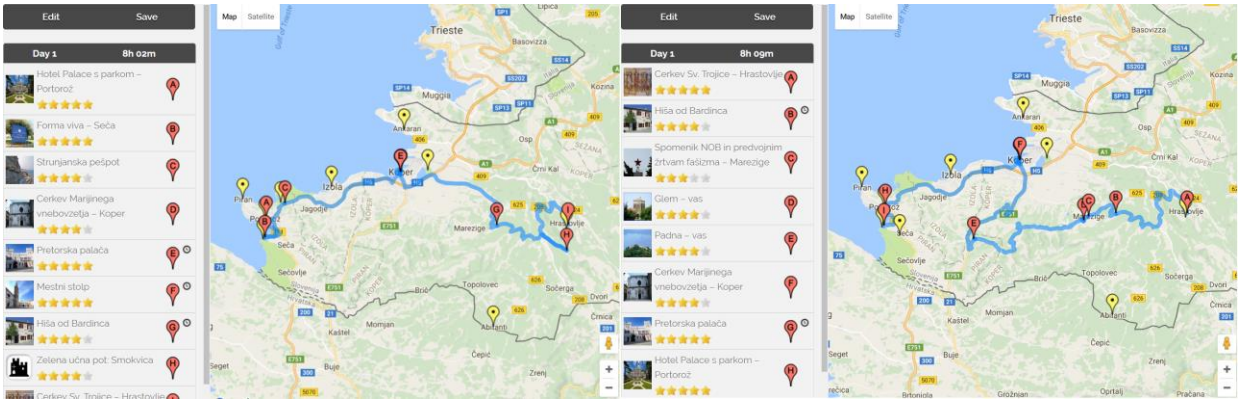

Figure 4. Left: Initial route, proposed for Bob, who is interested in cultural and natural heritage sites. Right: The new route, following Bob's creation of user profile.

Ada and Bob would like to spend a day in the Slovenian Istria region on Thursday, 1 September 2016. Both start the trip at 10 am and plan to spend 8 hours sightseeing. Both of them have a car available as a means of transport and Ada chooses to have lunch around 1 pm. Neither of them has yet created a user profile, which is likely for first-time users. The recommender system will therefore only consider the information from the constraints filtering module and the expert rating. Ada is interested in active tourism, gastronomy, and entertainment while Bob is interested in cultural and natural heritage.

The initial route proposed for Ada can be observed on the left side of Figure 3. After it is created, Ada decides to modify the proposed itinerary by clicking the "Edit" button which allows to remove the suggested and add other nearby/alternative POIs using drag and drop. When clicking "Update", Ada can choose whether to use the POIs in sequence as manually selected or to automatically reorder the sequence into an optimal route. Figure 3, right, presents the new proposed trip after choosing the automatic reordering. This new route requires longer than 8 hours to complete. If Ada decides that this is too long, she can edit the itinerary again, perhaps removing some of the POIs.

On the other hand, Bob has different preferences than Ada, which will reflect in different proposed itinerary (Figure 4, left). It is, however, possible for some POIs to overlap since they may be listed in more than one category, e.g. active tourism and cultural and natural heritage.

In the next step, Bob, who is a 60-year old German with tertiary education, high budget, and mobility limitations, creates his user profile. This allows the knowledge-based module and the collaborative filtering module to modify the list of POIs to better match Bob's profile. The new proposed route for Bob looks rather different (Figure 4, right). The constraints filtering module has first removed POIs that are less appropriate for people with mobility limitations, such as Strunjan trail (Strunjanska pešpot) and Smokvica educational trail (Zelena učna pot: Smokvica). In addition, the knowledge-based module has added the picturesque old villages Glem and Padna to the itinerary, since these are destinations that are likely to appeal to Bob's demographics, and also added the Second World War memorial (Spomenik NOB) - although it only has a 3-star expert rating. Additional alternative POIs appear on the map as well. Since Bob has not rated any POIs yet, the collaborative filtering has no specific ratings (for

Bob) to use for finding similar users and it therefore does not contribute to the POI selection in this use case. Once Bob starts rating visited POIs, the collaborative filtering will contribute to the final rating of the POIs in future trip planning.

One may observe that most POIs in routes for Ada and Bob have 4 or 5-star rating. They both travel by car, therefore they can cover larger distances during the trip, so the algorithm can include several POIs with high ratings.

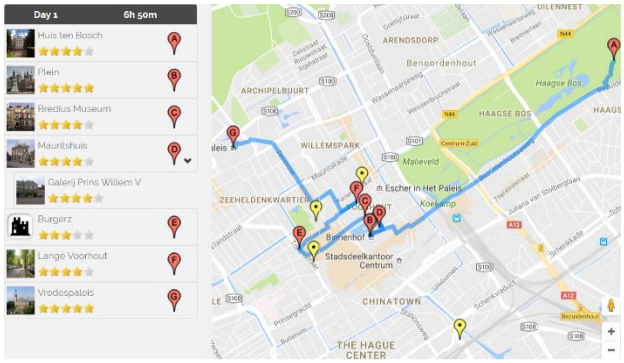

Figure 5. Proposed route for Ted, who explores The Hague on foot.

e-Turist also works on a city level. Let us consider Ted, also a first-time user, who is interested in cultural and natural heritage sites in the city of Hague in the Netherlands. Ted will explore the city on foot and use the same time window as Ada and Bob. The proposed route for Ted is shown in Figure 5. Here, we can also see an example of a nested POI, since Prince William V Gallery is located within the Mauritshuis complex which is a POI on its own.

6 Conclusion

We present a personalised trip planner that aims to improve tourist experience by facilitating the preparation of the trip and offering real-time guiding. e-Turist can be accessed either via browser or mobile application.

The platform is built on a POI database, which is easy to manage and expand. Based on the user profile and interests, the recommender system first creates a list of POIs which the user would find appealing. In order to create the list, the system uses a combination of constraints filtering, expert knowledge and collaborative filtering. In the next step, the route planning algorithm creates the optimal route between a subset of POIs. The algorithm uses the concepts from the TSP problem and the modified knapsack problem. The guiding component contains real-time GPS guidance and audio descriptions of the POIs, created using a speech synthesizer. It also allows the users to rate POIs, which serves for the collaborative filtering and as feedback available to tourism workers in the administration module.

Initial tests of the recommender system turned reasonably accurate (Section 3.1), however, since the current set of registered users is rather small, we expect the collaborative module to perform better in future. The advantage of e-Turist is that is easy to adapt to different regions. Initially, it was developed as a pilot project on two regions in Slovenia but is now being expanded to include regions worldwide. In the future, we also plan to integrate existing POI databases.

Acknowledgement

The e-Turist project was funded by Slovenian ministry of education, science and sport: call for proposals for co-funding of projects developing e-services and mobile applications for public and private non-profit organizations. We would like to thank the Faculty of Tourism Studies – Turistica and Municipality of Litija who provided the data and expert knowledge.

References

[1] UNWTO 2016, http://media.unwto.org/press-release/2016-01-18/international-tourist-arrivals-4-reach-record-12-billion-2015
[2] UNWTO Tourism Highlights, 2015 Edition
[3] Knoema, https://knoema.com/atlas/topics/Tourism
[4] Jennie Germann Molz, (2012). Travel connections: tourism, technology and togetherness in a mobile world. Routledge, New York.
[5] Wouter Souffriau, Pieter Vansteenwegen, Tourist Trip Planning Functionalities: State–of–the–Art and Future, Current Trends in Web Engineering, volume 6385 of Lecture Notes in Computer Science (2010) 474-485
[6] Damianos Gavalas, Charalampo sKonstantopoulos, Konstantinos Mastakas, Grammati Pantziou, Mobile recommender systems in tourism, Journal of Network and Computer Applications 39 (2014) 319–333
[7] Joan Borràs, Antonio Moreno, Aida Valls, Intelligent tourism recommender systems: A survey, Expert Systems with Applications 41 (2014) 7370–7389
[8] Gerhard Reinelt, The Traveling Salesman: Computational Solutions for TSP Applications, Springer Berlin Heidelberg (1994)
[9] Ander Garcia, Olatz Arbelaitz, Maria Teresa Linaza, Pieter Vansteenwegen, and Wouter Souffriau, Personalized Tourist Route Generation, ICWE 2010 Workshops, LNCS 6385, pp. 486–497, 2010. Springer-Verlag Berlin Heidelberg 2010s
[10] TripAdvisor https://www.tripadvisor.com/
[11] Triposo, https://www.triposo.com/
[12] Roadtrippers, https://roadtrippers.com/
[13] Route Perfect, https://www.routeperfect.com/
[14] e-Turist, https://www.e-turist.si/
[15] Igor Jurinčič, Anton Gosar, Mitja Luštrek, Boštjan Kaluža, Simon Kerma, Gregor Balažič, E-tourist: electronic mobile tourist guide. V: Peace, culture and tourism: collection of papers. Novi Sad: Faculty of Sciences, Department of Geography, Tourism and Hotel Management, 2013, str. 182-191.

[16] Amebis Govorec. http://govorec.amebis.si/
[17] Microsoft Speech API, http://www.microsoft.com/
[18] D. T. Larose. k-Nearest Neighbor Algorithm, Discovering Knowledge in Data, pp. 90-106, John Wiley & Sons, Inc, 2005.
[19] Srce Slovenije, http://www.srce-slovenije.si/
[20] Hans Keller, Ulrich Pferschy, David Pisinger, Knapsack Problems. Springer Berlin Heidelberg (2004)
[21] Dynamic Programming Knapsack 0-1 Problem. http://www.geeksforgeeks.org/dynamic-programmingset-10-0-1-knapsack-problem/
[22] Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; Sander, J. (2000). "LOF: Identifying Density-based Local Outliers". Proceedings of the 2000 ACM SIGMOD international conference on Management of data. SIGMOD '00: 93–104.
[23] Travelling salesman problem, Python implementation: https://github.com/dmishin/tsp-solver