Worldview and Route Planning using Live Public Cameras

Ahmed S. Kaseb, Wenyi Chen, Ganesh Gingade, Yung-Hsiang Lu
School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA

ABSTRACT
Planning a trip needs to consider many unpredictable factors along the route such as traffic, weather, accidents, etc. People are interested viewing the places they plan to visit and the routes they plan to take. This paper presents a system with an Android mobile application that allows users to: (i) Watch the live feeds (videos or snapshots) from more than 65,000 geotagged public cameras around the world. The user can select the cameras using an interactive world map. (ii) Search for and watch the live feeds from the cameras along the route between a starting point and a destination. The system consists of a server which maintains a database with the cameras’ information, and a mobile application that shows the camera map and communicates with the cameras. In order to evaluate the system, we compare it with existing systems in terms of the total number of cameras, the cameras’ coverage, and the number of cameras on various routes. We also discuss the response time of loading the camera map, finding the cameras on a route, and communicating with the cameras.

Keywords: Public Cameras, Live Feeds, Route Planning, Mobile Application

1. INTRODUCTION
How does the Times Square look like right now? Are the streets still snowy on the route to Chicago? Is the shopping mall crowded now? These questions and many more about tourist attractions, traffic, and weather are being frequently asked. To find the answers, some people search online but can find only snapshots taken long time ago. To check the current traffic or weather information, people usually use text-based methods which provide no visual information. Visual information from live feeds can provide timely updates. Moreover, the visual information helps seeing the world, planning routes, and many more applications. In recent years, thousands of public cameras are being installed. However, few mobile applications can provide access to the live feeds from these cameras.

This paper presents a system with an Android mobile application that enables users to watch the live feeds from more than 65,000 public cameras around the world. Users can select the cameras they wish to watch through an interactive map with the geotagged cameras. In addition, users can plan their routes by specifying a starting point and a destination, and hence the application will show the live feeds from the cameras along the route. The system consists of a server, and an Android mobile application. The server maintains a database with the cameras’ information, including their locations and the methods to retrieve the data from these cameras. The Android mobile application retrieves the cameras’ information from the server, and directly communicates with the cameras to get the live feeds without an intermediate server. This reduces the latency and enhances the scalability of the system.

To evaluate the system, we compare it with similar systems in terms of the total number of cameras, the cameras coverage, and the number of cameras on various routes. Our experiments show that this system provides the most cameras with a better coverage than similar systems. The experiments also show that this system provides the most cameras for various selected routes. We also consider the following performance metrics: (i) The response time of loading the world map with the superimposed camera markers depends on the number of cameras. Our experiments show that this response time is 2 seconds for 5,000 cameras, and 11 seconds for 35,000 cameras. (ii) The response time of finding the route from a starting point to a destination, and the cameras along that route depends on both the length and the topology of the route. Our experiments show that this response time does not exceed 1.6 seconds for 1,000-mile routes. (iii) The average response time of retrieving a single frame from various cameras around the world is around 32 milliseconds.

akaseb@purdue.edu; chen345@purdue.edu; ggingade@purdue.edu; yunglu@purdue.edu;
This paper has the following contributions: (i) It allows mobile users to watch the live feeds from more than 65,000 public cameras. Users can watch tourist attractions, streets, malls, etc. (ii) It handles the heterogeneity of the cameras, i.e. they are of different brands (Axis, Panasonic, etc.), resolutions, frame rates, etc. Heterogeneity is a key challenge since the mobile application has to handle the different methods to communicate with cameras of various brands. (iii) It allows travelers to plan their routes by showing the live feeds from the cameras along the route between a starting point and a destination. (iv) It enhances the scalability and reduces the latency by eliminating the need for an intermediate server between the mobile devices and the cameras.

2. RELATED WORK

Similar systems include Beat the Traffic,\textsuperscript{1} INRIX Traffic,\textsuperscript{2} Google Street View,\textsuperscript{3} Open Street View,\textsuperscript{4} Flickr Map,\textsuperscript{5} World Webcams,\textsuperscript{6} Worldscope Webcams,\textsuperscript{7} traffic camera maps of Departments of Transportation,\textsuperscript{8,9} etc. Research projects like AMOS\textsuperscript{10} also collect snapshots from public outdoor cameras. Table 1 compares and summarizes the differences between our system and similar systems. To the best of our knowledge, our system is the first mobile application that provides live video feeds from geotagged cameras and supports route planning as well. Moreover our system provides the most cameras among all similar systems.

<table>
<thead>
<tr>
<th>System</th>
<th>Content</th>
<th>Updating Rate</th>
<th>Coverage</th>
<th>Mobile</th>
<th>Route Plan</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beat the Traffic</td>
<td>Images</td>
<td>Often</td>
<td>USA</td>
<td>Yes</td>
<td>Yes</td>
<td>Android application</td>
</tr>
<tr>
<td>INRIX Traffic</td>
<td>Images</td>
<td>Often</td>
<td>USA</td>
<td>Yes</td>
<td>Yes</td>
<td>Android application</td>
</tr>
<tr>
<td>Google Street View</td>
<td>Images</td>
<td>Very rarely</td>
<td>Mostly USA, and Europe\textsuperscript{11}</td>
<td>Yes</td>
<td>No</td>
<td>3D view of the environment</td>
</tr>
<tr>
<td>Open Street View</td>
<td>Images</td>
<td>User-dependent</td>
<td>Worldwide</td>
<td>No</td>
<td>No</td>
<td>Crowd-sourced images</td>
</tr>
<tr>
<td>Flickr Map</td>
<td>Images</td>
<td>User-dependent</td>
<td>Worldwide</td>
<td>Yes</td>
<td>No</td>
<td>Crowd-sourced images</td>
</tr>
<tr>
<td>World Webcams</td>
<td>Videos</td>
<td>Real-time</td>
<td>108 cameras</td>
<td>Yes</td>
<td>No</td>
<td>Crowd-sourced webcams</td>
</tr>
<tr>
<td>Worldscope Webcams</td>
<td>Mostly images</td>
<td>Often</td>
<td>29,000 cameras</td>
<td>Yes</td>
<td>No</td>
<td>Limited number of videos</td>
</tr>
<tr>
<td>AMOS</td>
<td>Images</td>
<td>Often</td>
<td>19,000 cameras</td>
<td>No</td>
<td>No</td>
<td>Mostly in USA</td>
</tr>
<tr>
<td>Our System</td>
<td>Images, Videos</td>
<td>Real-time</td>
<td>65,000 cameras</td>
<td>Yes</td>
<td>Yes</td>
<td>Provides the most cameras</td>
</tr>
</tbody>
</table>

Table 1: Comparison between our system and similar systems. The columns lists the system, the type of content provided, the rate of updating the content, the coverage of the content, the availability of a mobile version, the availability of the route planning feature, and additional remarks.

Beat the Traffic\textsuperscript{1} and INRIX Traffic\textsuperscript{2} are mobile applications that show traffic cameras along the route to a given destination. However they only cover the USA and traffic cameras usually do not provide live video feeds. Our system has better world-wide coverage, and it supports live video feeds. Google Street View\textsuperscript{3} has a large set of images captured by data-collection vehicles to build a 3D panoramic view of locations around the world. However many images are obsolete (more than 5 years old in some cases). For example, Figure 1 compares our system’s view of Purdue’s Wang Hall building with the corresponding obsolete Google Street View and Google Maps Satellite View images. Moreover, Google Street View is only limited to images and does not provide live video feeds.

For traffic monitoring purposes, Departments Of Transportation (DOT) such as New York City DOT\textsuperscript{8} and Massachusetts DOT\textsuperscript{9} deploy cameras along main roads. The DOT’s websites provide maps which users can browse and watch the live feeds from the cameras. AMOS\textsuperscript{10} is a research project that provides snapshots from
around 19,000 outdoor cameras using a web-based map. Our system is different from the DOTs and AMOS as we provide more cameras, a route planning feature, and a mobile application.

In our previous work,\textsuperscript{12} we built CAM\textsuperscript{2} as a system for large-scale analysis of distributed cameras. We discovered more than 65,000 online public cameras. The cameras are distributed worldwide and are deployed by government agencies, universities, companies, or individuals. In this paper, we use the camera database introduced by CAM\textsuperscript{2} to build a mobile application that allows users to watch the world and plan their routes.

3. METHODOLOGY

3.1 System Architecture

The system consists of two main components as shown in Figure 2: (i) A server that maintains the camera database, and (ii) A mobile application that shows the live camera feeds and searches for the cameras along different routes. The mobile application is able to handle a large number of heterogeneous cameras, i.e. they have different types, brands, frame rates, and resolutions. The mobile application uses Google Maps API to show a map with superimposed camera markers, and Google Directions API to get the route between a starting point and a destination.

3.2 The Camera Database

The server maintains a database with the information of more than 65,000 cameras. The information includes the geolocations, resolutions, and frame rates of the cameras. It also includes the methods that should be used to communicate with different types of cameras. The server provides a web service to allow the mobile application to retrieve the cameras’ information from the database. Cameras can be broadly classified into two categories: (i) IP cameras are cameras with known IP addresses, and hence the system can directly communicate with the cameras to obtain the live video feeds. (ii) Non-IP cameras are offered by websites that usually provide only periodic snapshots in low frame rates.

3.3 The Mobile Application

The mobile application consists of five main modules as shown in Figure 2: (i) The server communicator retrieves the cameras’ information from the server. (ii) The camera communicator handles the communication with the heterogeneous cameras. (iii) The map handler interfaces with the external Google Maps and Google Directions APIs. (iv) The route-based camera search module searches for the cameras along a given route. (v) The User Interface (UI) handler controls the flow of the application and handles the interactions with the user.
The mobile application shows an interactive world map with superimposed markers, each of which represents a single camera as shown in Figure 3(a). If the camera density is high in a particular area, markers are clustered into a circle showing the number of cameras in the cluster. Users can navigate the map by moving, zooming in, and zooming out to reach the places they wish to watch. When the user clicks a camera marker, the application shows the live feeds from the camera as shown in Figure 3(b). If the camera supports live video streaming, the application will show the live video feeds. Otherwise, it will show the most recent snapshot. The user can enter two addresses of a starting point and a destination, and the application will show the cameras along the route between them as shown in Figure 3(c).

The mobile application sends two types of requests to the server: (i) When a user launches the application, the application requests the entire list of active cameras along with their geolocations so that it can show the camera map. (ii) When a user clicks a camera marker, the application requests the information of this particular camera so that the application can show the live feeds. Since the cameras are heterogeneous, different information are obtained for different types of cameras. For example, an IP camera requires the IP address, the port number, and the brand-specific path that is used to obtain the live video feeds. A non-IP camera requires the snapshot URL that is used to retrieve the most recent snapshot.

The mobile application directly communicates with the cameras to obtain the live feeds without the need for an intermediate server. This reduces the latency and enhances the scalability of the system. Some cameras support video streaming, while other cameras provides only periodic snapshots (1 snapshot every few minutes in some cases). For video streaming, we use MJPEG (Motion JPEG) because it is supported by most of the IP cameras and it does not have inter-frame correlation, hence a few missed frames do not affect other frames. This is important because the network conditions on mobile devices can be unstable.

The mobile application queries Google Directions API to obtain a route between a starting point and a destination. A route consists of a series of straight line segments whose geolocations are given. To search for the cameras along the route, a rectangular range is defined around each route segment. This range is configurable by the user. Then, the application searches for the cameras whose geolocations are in the rectangular range around any of the route segments as shown in Figure 4.

4. RESULTS AND EVALUATION

We evaluate the number and the worldwide distribution of the cameras in the database. We also evaluate the response time of (i) loading the map with superimposed camera markers, (ii) finding cameras along the route
Figure 3: Snapshots from the mobile application. (a) The interactive world map with superimposed camera markers and clusters. (b) Live video feeds from a camera in New York, USA. (c) The cameras along the route between two addresses in New York, USA.

Figure 4: A route (black lines) is a series of straight line segments. The cameras along the route (black dots) are the cameras that lie in the range around any route segment (shaded areas). Otherwise, cameras are not considered on the route (white dots).

from a starting point to a destination, (iii) retrieving and displaying a single camera frame. For the evaluation, we use a Nexus 7 with the specifications shown in Table 2.

The database contains more than 65,000 geotagged cameras. Figure 5 shows the distribution of the cameras. As can be seen from the figure, the cameras are distributed around the world. Most cameras are in North America and Western Europe, but many cameras are present in Asia, Africa, and Australia.
Table 2: Specifications of the evaluation mobile device

<table>
<thead>
<tr>
<th>Device</th>
<th>Google Nexus 7 (2013)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Qualcomm Snapdragon S4Pro (Quad-core 1.5 GHz Krait)</td>
</tr>
<tr>
<td>GPU</td>
<td>Adreno 320</td>
</tr>
<tr>
<td>RAM</td>
<td>2GB</td>
</tr>
<tr>
<td>OS</td>
<td>Android 4.4.2</td>
</tr>
</tbody>
</table>

Figure 5: Camera Distribution

Figure 6: The response time breakup with different number of cameras
Figure 6 shows how the response time of loading and showing the world map changes with the number of cameras. The horizontal axis represents the number of cameras returned by the server and the vertical axis represents the response time in milliseconds. The response time is divided into three parts: (i) obtaining the results from the server (ii) parsing the results (iii) displaying the markers on the map. It is clear that the response time is considerably affected by the number of cameras. The total response time is less than 2 seconds for 5,000 cameras, and it is shorter than 11 seconds for 35,000 cameras. Parsing the results and converting them to the Google Maps API format consumes most of the overall time.

The response time of the route-based camera search depends on the distance between two given points, and the route topology. This response time includes finding the route between the two given points, searching for cameras along the route, and showing the results on the map. Figure 7 shows the relation between the response time of the route-based camera search and the route length. The horizontal axis represents the route length in miles and the vertical axis represents the application response time in milliseconds. The starting points and the destinations of the routes are different cities in USA, so the routes include both highways and city streets. The response time increases with the route length, but does not exceed 1.6 seconds even for a very long route (1,000 miles). The response time also depends on the topology of the route. In particular, straight routes (with few turns) have shorter response time as the application searches for cameras along every straight segment on the route.

![Figure 7: The response time of searching for the cameras along routes with different lengths](image)

Our experiments show that the average response time to retrieve and show a single camera frame is 32ms. The experiments are conducted using 100 random cameras around the world. This response time is sufficient to provide a high frame rate video feed if the camera allows.

As shown in Table 1, our system is better than similar systems due to one or more of the following reasons: (i) provides the most cameras; (ii) provides live video feeds; (iii) enables users to plan their routes; (iv) is a mobile application. In order to evaluate the route-planning feature of our application, we compare it to the Beat the Traffic Android application. Table 3 compares the results of each application for 5 selected starting point and destination pairs. For first three routes, our application returns more cameras than Beat the Traffic. The fourth route has no cameras from Beat the Traffic as it exceeds the maximum distance supported, while our application has no such restrictions. The fifth route has no cameras from Beat the Traffic as it does not support routes outside of the USA, while our application supports routes worldwide.
<table>
<thead>
<tr>
<th>Starting Point</th>
<th>Destination</th>
<th>Distance (miles)</th>
<th>Our Application, Camera Count</th>
<th>Beat the Traffic, Camera Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purdue University</td>
<td>Chicago</td>
<td>122.0</td>
<td>20</td>
<td>9</td>
</tr>
<tr>
<td>West 42 St. and 9 ave, New York</td>
<td>Lexington ave and e 48 St., New York</td>
<td>1.3</td>
<td>18</td>
<td>0</td>
</tr>
<tr>
<td>Manhattan</td>
<td>Bronx County</td>
<td>8.0</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>New York City</td>
<td>Ohio</td>
<td>538.0</td>
<td>27</td>
<td>Maximum Distance Exceeded</td>
</tr>
<tr>
<td>Eiffel Tower</td>
<td>Meaux</td>
<td>37.0</td>
<td>3</td>
<td>Server Not Responding</td>
</tr>
</tbody>
</table>

Table 3: Comparison of the route-planning results for 5 selected routes between our application and the Beat the Traffic Android application.

5. CONCLUSIONS

In this paper, we present an Android application that allows users to watch the live feeds from more than 65,000 cameras around the world, and to search for cameras along the route from a given starting point to a given destination. The system consists of two main components: a server that maintains the cameras’ information, and a mobile application that shows live feeds from the cameras. The mobile application communicates with the heterogeneous cameras directly without the need for any intermediate server. This enhances the scalability of the system and reduces the latency. The application is evaluated in terms of the response time of loading the map at the beginning, the response time of the route-based camera search, and the response time of retrieving and presenting the visual content of the cameras. The experiments show that the application successfully keeps an acceptable level of performance.

6. ACKNOWLEDGMENTS

This project is partly supported by NSF CNS-0958487. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF. We would like to thank the organizations that provide the visual data. The list is available at https://cam2.ecn.purdue.edu/acknowledgements.

REFERENCES