Citywatcher: Annotating and Searching Video Data Streams for Smart Cities Applications

Alexey Medvedev[1], Arkady Zaslavsky[2], Vladimir Grudinin[1], and Sergey Khoruzhnikov[1]

[1] ITMO University, Kronverkskiy pr, 49., St. Petersburg, Russia
{alexey.medvedev,grudinin}@niuitmo.ru, xse@vuztc.ru
[2] CSIRO, Melbourne, Australia
arkady.zaslavsky@csiro.au

Abstract. Digital pervasive video cameras can be abundantly found everywhere these days and their numbers grow continuously. Modern cities have large networks of surveillance cameras including CCTV, street crossings and the like. Sometimes authorities need a video-recording of some road accident (or of some other event) to understand what happened and identify a driver who may have been at fault. In this paper we discuss the challenges of annotating and retrieving video data streams from vehicle-mounted surveillance cameras. We also propose and evaluate the CityWatcher application – an Android application for recording video streams, annotating them with location, timestamp and additional context in order to make them discoverable and available to authorized Internet of Things applications

Keywords: Smart City, video streaming, Internet of Things (IoT), crowdsensing, Intelligent Transportation Systems, Vehicle-Mounted Surveillance Camera (VMSC).

1 Introduction

Digital pervasive video cameras can be abundantly found everywhere these days and their numbers grow continuously. Modern cities have large networks of surveillance cameras including CCTV, street crossings and the like. Sometimes we need a video-recording of some road accident (or of some other event) to understand what happened and identify a driver who may have been at fault. Many car owners use Vehicle-Mounted Surveillance Cameras (VMSC, or in other words, car black boxes, video registers or smartphones) for recording their driving and events of interest. These video-recordings are potentially important sources of data. The drivers can use video recording from their devices as evidence in case of a road accident. However, video recordings from their devices can't be retrieved by others and shared. Therefore we highlight the challenge of retrieving such video fragments for evidence collection and the methods of annotating, discovering, retrieving and processing video data streams. In particular, we argue that smartphones can be used for evidence collection in case of a road accident. Besides, smartphone owners can use the proposed

S. Balandin et al. (Eds.): NEW2AN/ruSMART 2014, LNCS 8638, pp. 144–155, 2014.

CityWatcher application for sending their automatically or manually annotated reports about road accidents, or any other city problems, e.g., road potholes and cracks.

Functioning of a modern city strongly depends not only on the city infrastructure, but also on the availability and quality of information for citizens and authorities about different aspects of city life. Various applications enhancing and making easier the city life lead to the concept of a 'smart city', which has received a lot of attention in last few years. The "Smart City" concept essentially means efficiency, effectiveness and resource optimization. Efficiency is largely based on the intelligent management and integrated ICT (Information and Communication Technology) infrastructure, and active citizen participation. This implies a new form of governance, including genuine citizen involvement in public policy in the form of crowdsourcing and crowdsensing [1], [2], [3].

Applications for smart cities are usually divided into six main areas: smart living, smart governance, smart economy, smart environment, smart people and smart mobility [4]. A very important feature for all these types of applications is the data collection, which is primarily based on sensors and sensor networks. Smart city applications strongly need access to this public data through the web for visualizing, transforming and making use of it. Active participation of all stakeholders is also very important for the smart city.

As the number of cars continues to grow, road problems become one of the main issues for city management. In smart cities, Intelligent Transportation Systems (ITS)[5] are used for solving transportation and congestion problems using Information and Communication Technology (ICT). ITSs help in fast problem detection, violations of regulations, traffic analysis, evidence collection, safety provisioning, reducing costs and delays and much more. Therefore, ITSs appear to be one of the most important parts of a smart city.

Rapid problem solving is really important for the smart city. If a citizen encounters a problem, he/she should be able to draw city management attention to it without taking any complicated actions. On the other side, all city services must be ready to receive such requests and take measures to solve the problem. Besides, they must have instruments to ask for some help from citizens. This help is called crowdsourcing. If this help takes a form of getting any information from computers, smartphones or other devices that incorporate sensors, than we can speak about Urban Crowdsensing. Working in close contact with citizens is very important part of moving towards smart cities.

Every year road accidents cause loss of lives, loss of money and serious delays on city streets and freeways. Approximately 50%–60% of the delays on urban freeways are caused by road accidents [6]. Reducing delays by faster accident analysis is a challenging task. This task includes building an evidence collection system that can help the police and city authorities to make more accurate and faster decisions. The best way to achieve it is to provide video recordings from different angles and sources to the decision maker. If such videos are available, cars that participated in an accident can be faster moved out of the road and unblock traffic.

During last decade vehicle-mounted surveillance cameras (VMSC) gained significant popularity[7]. VMSC can also be referenced as cars video registers or black boxes or dash cams. They can help to prove innocence in case of a road accident or just make a video of something interesting and uncommon, like falling of a meteor or a tree, or a tornado. *"In the USA, 15% of 200 million cars carry a black box, while 80% of the compact cars built since 2004 mount black boxes. In Japan, about 6 million of the black boxes mounted on vehicles include 40,000 business vehicles and 20,000 regular passenger cars" [7].*

By now, VMSCs already have a feature of annotating video with street name, as they have a GPS module onboard. The most modern VMSCs have Internet access. It can be predicted, that in future most of such smart cameras will have Internet access. VMSCs are produced by various manufacturers, for example, Garmin[1], Prestigio [2], HP[3]. A typical VMSC is shown in Fig.1(left). Millions of surveillance cameras are deployed, many of them are private. Some people have such cameras at their windows. Also, many people just make recordings with their smartphones. As a smartphone has a camera, it can easily act as a VMSC if it is fixed on a front window.

Many smartphone apps are available in Google Play and Apple AppStore. Examples of such applications include: DailyRoads Voyager[4], Axel Voyager[5], AVR[6], AutoBoy BlackBox[7], CaroO Pro[8] etc. their characteristics will be briefly discussed in the 'related work' section. A smartphone acting as a VMSC is shown in Fig.1(right).

Eventually, the cloud storage would keep videos of many events that can be of interest for city management, police, road services and other governmental organizations. The challenge is in retrieving relevant video streams when they are really needed. Owners of the surveillance camera may not be informed, that an accident happened just in his/her camera view. Owner of a VMSC could pass by without stopping, when an accident happens in front of their car. In such a situation, police has to understand what happened just by listening to stories of accident participants, that not always lead to correct analysis and respective decisions.

Fig. 1. VMSC (left) and smartphone acting as a VMSC (right)

[1] https://buy.garmin.com/en-US/US/shop-by-accessories/other/garmin-dash-cam-20/prod146282.html

[2] http://www.prestigio.com/catalogue/DVRs/Roadrunner_300

[3] http://www.shopping.hp.com/en_US/home-office/-/products/Accessories/Camera_photo_video/H5R80AA?HP-f210-Car-Camcorder

[4] http://www.dailyroads.com/voyager.php

[5] https://play.google.com/store/apps/details?id=net.powerandroid.axel

[6] https://play.google.com/store/apps/details?id=com.at.autovideosregistrator&hl=uk

[7] https://play.google.com/store/apps/details?id=com.jeon.blackbox&hl=ru

[8] https://play.google.com/store/apps/details?id=com.pokevian.prime

According to [8], surveillance video is now the biggest source of Big Data. It is predicted, that the percentage will increase by 65 percent by 2015. As we are adding video recordings from users Internet Connected Objects (ICO), using the Internet of Things terminology, the amount of data increases significantly. One of the most critical challenges is how to transmit and store all this video. First of all, it seems to be very hard to transmit and store all the video recordings from ICOs to some central storage, whether it's cloud or a video repository. Secondly, not many people might want to share their video recordings in some way, as there's no certainty what it will be used for. It is a serious privacy concern. The solution can be found in storing data in the device, where it was recorded. If we would be able to make requests to all possible ICOs, the need to transfer and store the video data will be reduced or eliminated. So, as data stays in ICOs, we are to look at massively distributed Internet of Things system with challenging indexing, search and processing requirements. The challenge is also to communicate with millions of ICOs, make requests and get videos of events that we are interested in. As these ICOs are heterogeneous, opportunistically available and spread all over the region (however big), building such a system becomes a global IoT challenge. One of the key concepts of the system is crowdsourcing [9].

2 Related Work

The process of getting video from surveillance cameras is already well investigated [10]. We focus not on existing road or traffic intersection cameras, as is the case with most current applications, but mainly, on opportunistically available mobile data sources, which belong to citizens. The problem of crowdsourcing with smartphones is discussed in [11]. A prototype of a crowdsourced evidence collection system is presented in [7]. Though, some principles are similar with the proposed CityWatcher system, their prototype is developed for laptops, which is not as ubiquitous as using smartphones and besides, as a major distinction, does not incorporate annotating and discovering relevant video recordings. The number of possible use cases is limited only to requesting the video and the research challenge of how to deal with the heterogeneous world of IoT is not discussed. The problem of crowdsourced video annotation is widely discussed in [12].

As it was mentioned above, there are a number of applications, which allow using an Android smartphone as a VMSC. We will discuss their characteristics and main features, typical for applications of this class. Main characteristics of a typical VMSC emulating application include but aren't limited to: (a) Free or not?; (b) Video format, possibility to choose the format; (c) Cycle video recording; (d) Possibility of making photos; (e) Annotation with timestamp; (f) Annotation with GPS coordinates; (g) Annotation with location information (street name); (h) Map, navigation; (i) Acting as a car's computer (measuring maximum and average speed, etc); (j) Possibility of loading video to the Internet via wireless networks; (k) Indexing significant moments manually or by a sensor; (l) Resource consumption; (m) Possibility of working in background. Some of the mentioned apps are briefly discussed below:

- DailyRoads Voyager – free, easy to use application. Axel Voyager – complicated full featured application, supports navigation functionality and can make indexes of significant moments

- AVR – usual video recording features, image stabilization function, navigation, can send video to the Internet
- AutoBoy BlackBox – full featured video player, displays additional information (speed, compass), very big choice of settings
- CaroO Pro - additional functionality includes features, usual for cars computer

Any of these apps can be used as a VMSC, but none of them support answering requests to share video recordings and make their video recordings discoverable. In the proposed CityWatcher application we focus on request processing and answering, so that interested parties could make use of locally stored and annotated video recordings.

3 CityWatcher Concepts

In this paper we propose, develop, discuss and evaluate the "CityWatcher" Android-based app. First of all, we discuss the use case "Road accident investigation" as illustrated in Fig.2.

Let's assume the following scenario. A road accident happens on a street. Road police has to decide who was at fault, but it is not always clear what happened. But if there was no road camera watching exactly that very spot, police will have to use other evidence which could be much weaker than the video recording. There is a good chance that the accident may have been recorded by someone's VMSC. Surely, the owner could stop the car and give his/her phone number to everyone involved in the accident and then they can get the video recording and present to investigation unit. In real life this is not always so. First of all – a car could be parked, while the driver was absent, if we assume that a VMSC continues recording while a car is parked. Or the driver hadn't seen the accident, or just had no time to stop. Anyway, there is a need to get a video. But how to get it ? The policeman opens a web browser, navigates to a special web site and makes a request to find all video recordings from that location by marking it on a map and specifying the time. Request is passed to a cloud service and linked to the account of a policeman. IoT middleware sets up a new task, which can last for several days or even longer. We can't assume that all smartphones that act as a VMSCs will be on and available via Internet just at the moment. So, the middleware gathers information as smartphones become available. When an available smartphone is discovered, middleware makes a request like "Have you been in 123.456/ 678.678 (lon/lat) on 03.03.14 12.14:45. The smartphone checks its own database. It sends no answers to the server. All the answers are connected with private data, so have to be handled very carefully. If there is no recording from that time and location – nothing happens. If there is – the application asks the user to check, if there is an accident on the record. If the users finds that there is, than he/she can share the fragment with the authorities. The user can also wipe out the sound, so that personal talks are not transferred.

All such answers are linked to the policeman's request, so when he/she gets back after some time he/she finds that he/she can look at the accident from different angles and make a fully qualified decision. We can also think about some incentives to make this service more attractive. People that provide significant help by sharing their video recordings can be rewarded somehow, for example by lowering the transport registration fee or by reducing fine points, or by giving some priorities in a queue for police service. This can depend on current country laws and traditions.

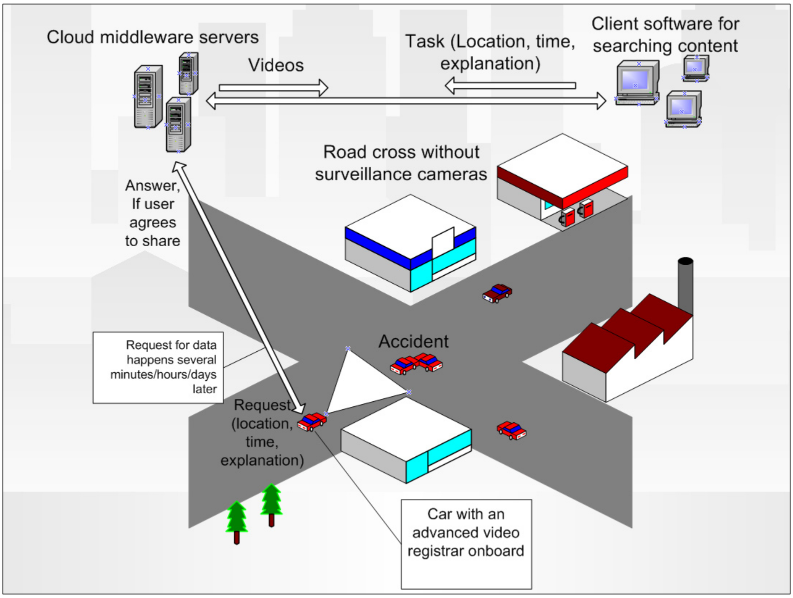

Fig. 2. Use case 1 “Road accident investigation”

As it was already mentioned, the main principle of data acquisition is crowdsourcing. Crowdsourcing seems to be a strong method of collecting data, but there are concerns about manual data processing. Though a lot of dispatching work can be automated, most of it has to be made manually and by professionals. There are also a number of challenges that would arise in any crowdsourcing solution:

- **Redundant data.** In our scenario all videos are complementary as there couldn’t be identical videos from different cars.
- **False data.** Someone’s joke can become a waste of time for the service team. One of the solutions of this problem can be in storing user’s ratings in the database. These ratings can be made by giving marks to loaded videos by personnel which work with them. Requests from users, who had already produced false data, can be blocked.
- **Feedback.** Users who send data want to understand what happens next. If they don’t receive an acknowledgement that their request was processed and their work is needed, they will not send any requests any more. Citywatcher android application has a form with reports on all shared videos, so the user can check if his contribution was used.

Designing, running and automating a processing center is also a big challenge. At first, report processing can be done manually by professional operators. Later, when all the business logic becomes clear, many techniques can be used for process automation and reduction of the need in human resources. First of all, speech and video recognition techniques must be used for determining the context. When this is done, it becomes possible to apply different reasoning techniques for making a decision where to transfer the report. Context reasoning techniques, which will be used in future releases of Citywatcher, can be broadly classified into six categories: (a) supervised learning, (b) unsupervised learning, (c) rules, (d) fuzzy logic, (e) ontological reasoning and (f) probabilistic reasoning. For example, supervised learning methods include artificial neural networks, Bayesian networks, case-based reasoning, decision tree learning and support vector machines [13]. Such methods must be supported by the middleware platform. We will discuss the choice and features of the middleware platform in the next section.

4 CityWatcher Architecture and Implementation

The proposed system is divided into three main parts: ICOs, middleware and client software as shown in Fig.3. ICOs include the smartphones, VMSCs, smart cameras, cars video registers, etc. All participating ICOs record video via a special application.

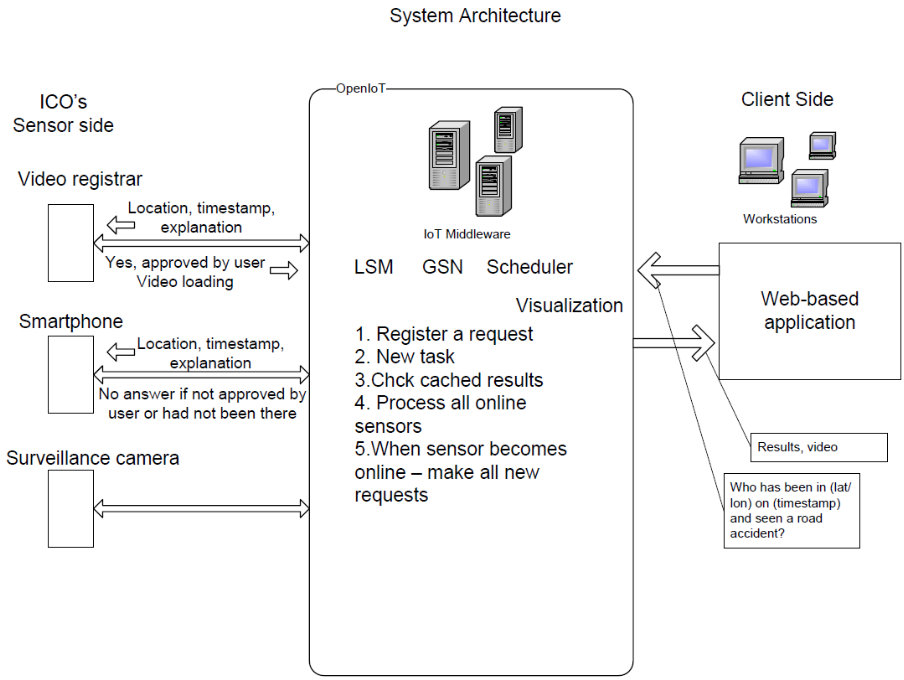

Fig. 3. CityWatcher system architecture

The prototype is an Android app. Application features include Video recording, Video annotating (gps coordinates, time), Self-discovery, auto register with middleware ,Request processing (like 'Have you been(time, location) ?'), Asking the owner to watch and share video, Loading part of the video to the cloud. Screenshots of

CityWatcher application for Android are presented in Fig.4. It is a working prototype app. The design of user-friendly interface will be included in future work.

CityWatcher Middleware is based on **OpenIoT** Platform [14]. OpenIoT project is a new open source middleware platform for intelligent IoT applications. It is an extension to cloud computing implementations and provides functions for managing IoT resources. In this way users can get IoT services including Sensing-as-a-Service.

OpenIoT platform is discussed in [16][17].

The CityWatcher project relates to urban crowdsensing. So we expect the city administrations to become service providers and a host to an OpenIoT platform. We use Global Sensor Network (GSN) to provide middleware functionality for registering and finding ICOs. Using an OpenIoT platform gives us an efficient and advanced way to fusing data sources. Not only VMSC can be used for getting video data. Streets in modern cities already have lots of surveillance cameras. Video streams from those cameras are stored in different systems, as cameras can have different vendors. It will be hard for users to make requests to check all such systems. OpenIoT platform can be used for efficient data fusing, so the user will have to make only one request to the middleware. The integration of CityWatcher application with OpenIoT middleware is shown in Fig.5. **Client software** located in Utility App Plane is a web-based application, which allows a user to make request to the system and view results.

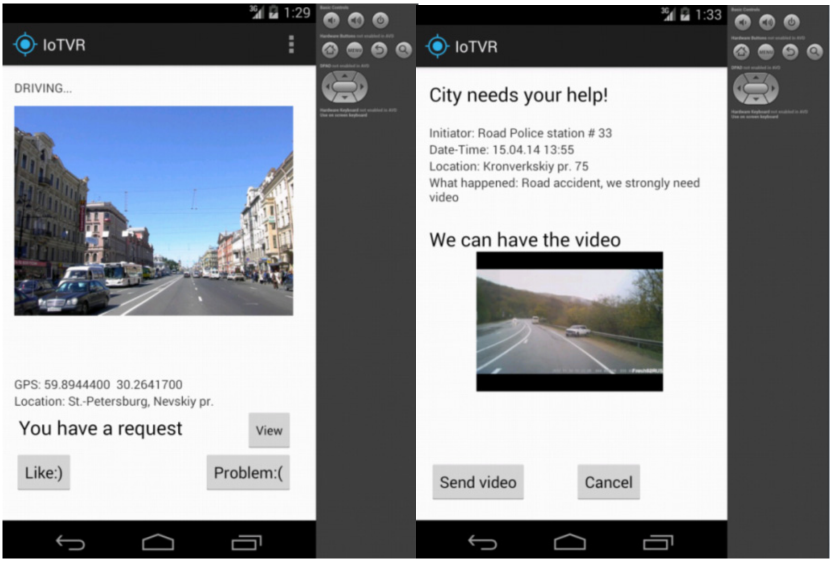

Fig. 4. CityWatcher application GUI

One of the technical problems needed to be solved is identifying the device, on which video is recorded and stored. Firstly, middleware has to be aware of all participating ICO's in the system, as it has to broadcast/multicast a request to all of them, register that the task has been sent and match the answer with the device IF. Secondly, middleware needs to have information about user personal preferences, as the user obtains some incentives for participating and sharing his/her videos. Besides, having information about the ICO owner makes the video fragment more legal in the court. We must also bear in mind that one user can have several devices and one device can be used by different users.

We propose to use the following scheme: every user gets an account in the CityWatcher system. It is a classical login/password pair. With this account a user can start the CityWatcher app. On the middleware side login is linked to full user description/profile, so the user can get his/her incentives from city services. As the video is stored on the users' devices and one user can have multiple devices, we need to uniquely identify every device. IP addresses and MAC addresses are poor candidates for that, as IP address can be changed when the device changes the network, and MAC address, in practice, can be not unique.

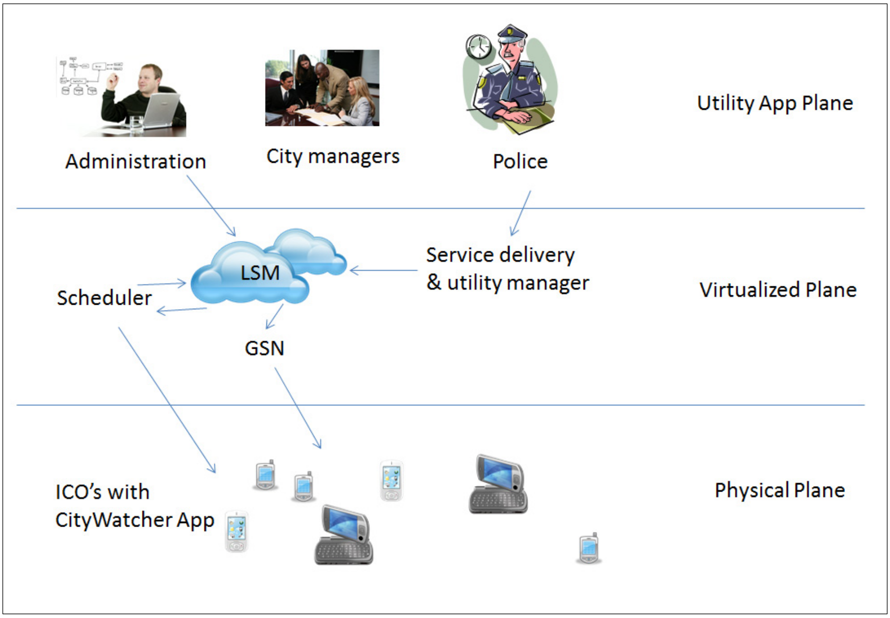

Fig. 5. OpenIoT middleware with CityWatcher Application

To solve this problem a unique device ID can be generated following the CityWatcher installation. It is like a license number for software. Then, all users of one device use this unique ID, but log in with their own accounts. This enables the middleware to distinguish devices for video searching possibilities and also, distinguish users for incentive purposes.

When the video fragment is loaded into the cloud, it should be stored and registered in the database. Additional information about the fragment includes task ID, on which it was found, user ID, device ID and obtaining time. Besides, some information about device like camera resolution and record format is also attached. This make possible to sort videos and firstly view the results with best resolution.

We use SQLite as an engine for storing data about timestamps and coordinates. The CityWatcher app instantly adds record to the database. When a task arrives and there is a need to understand if there is a video on local storage – a request to the database is made.

When the request is made, we search not exactly the same time and coordinates, but add delta values and perform range search. The middleware database structure is omitted in this paper due to space limitations.

As it was mentioned above, video and all the metadata about is stored on the device. This is more secure and gains more trust, than loading all the data to the cloud. On the other hand, searching in devices metadata depends on the online availability of the device. If a device is offline, searching can be slow, as the middleware will have to wait for the device to become available on the network to give it a search task. We can assume, that some users can agree to load the metadata about when and where they have been to the cloud immediately. That makes possible for the middleware to make search in a local database without spreading tasks among thousands of ICOs and waiting for their reply.

5 Conclusion and Future Work

We can think of a scenario, where people not only report problems by pressing the "red button", but, in contrast, they press a "green button", to appreciate things or changes they like. This "likes" can be marked on a map, aggregated and analyzed to understand, which locations in a city produce positive emotions. This can be a little step towards augmented reality. Another challenge is to measure the probability of the first scenario. We can expect only some percentage of cars to carry an ICO with CityWatcher app, so it will be challenging to find out, how often it will be possible to get the video of an accident. User study will be part of the future work. Measuring probability can be achieved by using modern modeling methods like traffic simulators. For example, open source projects like SUMO and Veins can be used [18],[19]. Making a precise model for prediction is a challenging task and can be considered for future work. We are confident that it is possible to extend the CityWatcher application with pervasive object search options. It is hard to imagine a location where ICOs cameras can't reach. ICOs can be deployed anywhere and anytime. This reinforces the motto of pervasive computing "anywhere, anytime, on any device and over any network". While it might still be hard and expensive to put surveillance cameras everywhere, with the advent of mobile computing we can assume that users carry their ICOs/smartphones everywhere they go. ICOs become more and more powerful and video recognition algorithms also are becoming more advanced. For example, we can include some library for car number plate recognition [20]. An ICO can receive a request from the police, searching for a stolen car. ICO analyzes all car number plates that it detects on the road. If a match is found – an alert can be sent back to police

As video recognition algorithms become mature, we foresee more and more scenarios of pervasive object search. Present day smartphones are able to record and store a large amount of video content with location and timestamp annotations, as well as other diverse context. The proper use of sharing this information can help in improving how the smart city runs and functions.

References

[1] Caragliu, A., Del Bo, C., Nijkamp, P.: Smart Cities in Europe. In: Proceedings of the 3rd Central European Conference on Regional Science, Košice (2009)

[2] Hu, M., Li, C.: Design Smart City Based on 3S, Internet of Things, Grid Computing and Cloud Computing Technology. In: Wang, Y., Zhang, X. (eds.) IOT Workshop 2012. CCIS, vol. 312, pp. 466–472. Springer, Heidelberg (2012)

[3] Smart city, http://en.wikipedia.org/wiki/Smart_city (accessed: April 18, 2014)

[4] Giffinger, R., Fertner, C., Kramar, H., Kalasek, R., Pichler-Milanovic, N., Meijers, E.: Smart cities – Ranking of European medium-sized cities. Smart Cities. Centre of Regional Science, Vienna (2007)

[5] Intelligent transportation system, http://en.wikipedia.org/wiki/Intelligent_transportation_system (accessed: April 24, 2014)

[6] Ki, Y.-K., Lee, D.-Y.: A Traffic Accident Recording and Reporting Model at Intersections. IEEE Transactions on Intelligent Transportation Sytems 8(2) (June 2017)

[7] Chae, K., Kim, D., Jung, S., Choi, J., Jung, S.: Evidence Collecting System from Car Black Boxes. School of Electronic Engineering Soongsil University Seoul, Korea

[8] Huang, T.: Surveillance Video: The Biggest Big Data. Computing Now 7(2) (February 2014), http://www.computer.org/portal/web/computingnow/archive/february2014?lf1=929719755f161316415589b16868896

[9] Schuurman, D., Baccarne, B., De Marez, L., Mechant, P.: Smart Ideas for Smart Cities: Investigating Crowdsourcing for Generating and Selecting Ideas for ICT Innovation in a City Context. Journal of Theoretical and Applied Electronic Commerce Research 7(3), 49–62 (2012)

[10] Georgakopoulos, D., Baker, M., Nodine, A.: Event-Driven Video Aware-ness Providing Physical Security. World Wide Web Journal (WWWJ) 10(1) (March 2007)

[11] Chatzimilioudis, G., Konstantinidis, A., Laoudias, C., Zeinalipour-Yazti, D.: Crowdsourcing with Smartphones. IEEE Internet Computing 16(5), 36–44

[12] Vondrick, C., Patterson, D., Ramanan, D.: Efficiently Scaling up Crowd-sourced Video Annotation. A Set of Best Practices for High Quality, Economical Video Labeling. Springer Scince+Business Media LLC (2012)

[13] Perera, C., Zaslavsky, A., Christen, P.: andD. Georgakopoulos, Context Aware Computing for Internet of Things: A Survey. IEEE Communications Surveys & Tutorials Journal (2013)

[14] OpenIoT Architecture Webpage, https://github.com/OpenIotOrg/openiot/wiki/OpenIoT-Architecture (accessed: April 12, 2014)

[15] OpenIoT Webpage, http://www.openiot.eu (accessed: March 12, 2014)

[16] PodnarZarko, I., Antonic, A., Pripužic, K.: Publish/Subscribe Middleware for Energy-Effcient Mobile Crowdsensing. In: UbiComp 2013 Adjunct Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication, pp. 1099–1110 (2013)

[17] What is OpenIoT and what does it can do for you?, http://academics.openiot.eu/ (accessed: April 24, 2014)

[18] Sommer, C., German, R., Dressler, F.: Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. IEEE Transactions on Mobile Computing 10(1), 3–15 (2011)

[19] Arellano, W., Raton, B., Mahgoub, I.: TrafficModeler extensions: A case for rapid VANET simulation using, OMNET++, SUMO, and VEINS. In: 2013 10th International Conference on High Capacity Optical Networks and Enabling Technologies (HONET-CNS), pp. 109–115 (2013)

[20] Buch, N., Kristl, V.S.A., Orwell, J.: A Review of Computer Vision Techniques for the Analysis of Urban Traffic. IEEE Transactions on Intelligent Transportation Systems 12(3)