SmartPhoto: A Resource-Aware Crowdsourcing Approach for Image Sensing with Smartphones

Yi Wang, Wenjie Hu, Yibo Wu and Guohong Cao
Department of Computer Science & Engineering
The Pennsylvania State University
University Park, PA, 16802, USA
{yuw124, wwh5068, yxw185, gcao}@cse.psu.edu

ABSTRACT

Photos obtained via crowdsourcing can be used in many critical applications. Due to the limitations of communication bandwidth, storage and processing capability, it is a challenge to transfer the huge amount of crowdsourced photos. To address this problem, we propose a framework, called *SmartPhoto*, to quantify the quality (utility) of crowdsourced photos based on the accessible geographical and geometrical information (called *metadata*) including the smartphone's orientation, position and all related parameters of the built-in camera. From the metadata, we can infer where and how the photo is taken, and then only transmit the most useful photos. Three optimization problems regarding the tradeoffs between photo utility and resource constraints, namely the Max-Utility problem, the online Max-Utility problem and the Min-Selection problem, are studied. Efficient algorithms are proposed and their performance bounds are theoretically proved. We have implemented SmartPhoto in a testbed using Android based smartphones, and proposed techniques to improve the accuracy of the collected metadata by reducing sensor reading errors and solving object occlusion issues. Results based on real implementations and extensive simulations demonstrate the effectiveness of the proposed algorithms.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*;
C.4 [**Performance of Systems**]: *Modeling techniques*

Keywords

Crowdsourcing; Image sensing; Photo sharing; Camera sensor; Smartphone

1. INTRODUCTION

Equipped with GPS, orientation sensors, mega-pixel cameras and advanced mobile operating systems, smartphones not only change the way people communicate with each other, but also the way they interact with the world. The popularity of online photo sharing services such as Flickr and Instagram indicates that people are willing to take photos and share experiences with others. Thanks to the cost efficiency, timeliness and pervasive nature of these data, numerous opportunities have been created for applications based on photo crowdsourcing, such as grassroots journalism [3], photo tourism [18], and even disaster recovery and emergency management [12].

Consider an example in post-earthquake recovery. First responders survey the damage by taking pictures and then transfer them back to the remote command and control center. As events occur, photos need to be collected and uploaded as quickly as possible. However, there are strict bandwidth constraints, no matter it is based on mobile ad hoc networks, delay tolerant networks, or partly damaged cellular networks. Then, how to make use of the limited bandwidth to upload the most useful photos becomes a challenge.

Another example can be found in our daily life. A map service provider can enhance user experience by showing photos of interesting objects around the world, for example, landmarks like famous buildings. Data can be obtained from visitors taking photos via their smartphones. Once the photos are uploaded and processed, other map users can have virtual tours. Due to the existence of many useful applications, people are sharing billions of photos taken by smartphones. Photos are often geographically correlated and this correlation can be used to enrich traditional map experience. However, the sheer amount of photos poses big challenges for image processing and storage at the server end. Fully understanding the semantic of each photo by traditional resource intensive image recognition techniques would be a luxury if not impossible. Therefore, how to identify the most relevant data and eliminate redundancy becomes an important issue.

The major challenges faced by these applications are as follows. The first is how to characterize the quality (usefulness) of crowdsourced photos in a way that is both meaningful and resource friendly. Most content-based image processing techniques such as [6, 26, 25] may demand too much computational and communication resources at both the user and server ends. On the other hand, existing solutions from description based techniques either categorize photos based on user defined tags, or prioritize them by the GPS location [20]. Obviously, tagging each photo manually is not convenient and may discourage public participation. GPS location itself may not be sufficient to reveal the real point

MobiHoc'14, August 11–14, 2014, Philadelphia, PA, USA.

http://dx.doi.org/10.1145/2632951.2632979.

of interest. Even at the same location, smartphones facing different directions will have different views.

To address these issues, we propose a framework to quantify the quality of crowdsourced photos based on easily accessible geographical and geometrical information, called *metadata*, including the orientation and position of the phone, and the field-of-view (FoV) of the camera. Intuitively, a good photo coverage should have multiple views of the target and cover as many *aspects* as possible. Specifically, given a set of targets and photos, we consider an aspect of a target to be properly covered if it is within a proper range of a photo's viewing direction (defined in Section 2). Then we measure the quality of a photo by *utility*, which indicates how many aspects are covered. The utility is calculated based on the metadata, which can be practically obtained via various embedded sensors in most off-the-shelf smartphones. They are independent of the image content, and hence the computation is very fast and resource friendly compared to traditional content based approaches.

With the above model, we address challenges brought by the resource constraint, which is referred to as the *Max-Utility problem*. Resource constraint of bandwidth, storage and processing capability limits the number of photos that can be uploaded to the server. Given the metadata of the candidate photos, how to find a given number of photos such that the total utility is maximized? Note that this is different from traditional maximization problems on sensor coverage in which a target is covered as long as it is inside the sensing range. Here photos taken at different view points cover different aspects of the target. The total utility depends on how many aspects can be covered and how they are covered, which makes the problem unique and complicated. We also consider online selection/optimization to address the requirements of time critical applications.

Another challenge to be addressed is how to remove the redundancy and find the most representative photos. In general, the amount of candidate photos is significant and redundancy occurs if multiple photos are taken at similar locations and from similar angles. The less number of photos is selected, the less amount of bandwidth, storage and processing capability is needed. In the *Min-Selection problem*, given the coverage requirements of the targets, we want to find the minimum set of photos that satisfy the requirements.

Our contributions are summarized as follows. We propose *SmartPhoto*, a novel framework to evaluate and optimize the selection of crowdsourced photos, based on the collected metadata from the smartphones. We formulate the Max-Utility problem for bandwidth constrained networks, and then extend it into an online optimization problem. We also study the Min-Selection problem for redundancy reduction. Moreover, we propose efficient solutions, and find the performance bounds in terms of approximation or competitive ratios for the proposed algorithms.

We have implemented SmartPhoto in a testbed using Android based smartphones. We make use of multiple embedded sensors in off-the-shelf smartphones, and propose a series of methods to fuse data, correct errors, and filter out false information, to improve the accuracy of the collected metadata. Finally, the performance of the proposed algorithms are evaluated through real implementations and extensive simulations.

The rest of the paper is organized as follows. Section 2 introduces the basic concepts and the model. Section 3 studies the Max-Utility problem and Section 4 studies the Min-Selection problem. Section 5 presents the implementation of the testbed. Performance evaluations are presented in Section 6. Section 7 reviews related work and Section 8 concludes the paper.

2. PRELIMINARIES

Consider a scenario in which a set of predefined targets are to be monitored by a group of people or reporters. They use smartphones to take photos and transfer them back to the processing center. However, due to the limited bandwidth available, only a small number of photos can be transferred. For this reason, reporters first transmit the metadata of the photos, which is extremely light weight compared to the original image. After that, the server runs optimization algorithms to determine what photos to be actually transferred and notifies the reporters to transmit the photos.

We first describe the models used in SmartPhoto to characterize targets and photos. Then the concept of utility is introduced. The idea is based on the observation that a good photo should cover as many aspects of the targets as possible. For an aspect to be properly covered, the target should be in a photo whose viewing direction is not too far away from the direction to which the aspect points. This is similar to the face recognition problem in computer vision: as the angle between the object's facing direction (the aspect) and the camera's viewing direction (the vector from the camera to the object) becomes wider, the detection rate of the recognition algorithm will drop dramatically [4, 14]. The utility defined in this section precisely indicates how many aspects of the target are properly covered.

2.1 Targets and Photos

At the beginning of each event, the application server distributes the information of the interested targets to the public users. The set of targets are denoted by $T = \{T_1, \ldots, T_m\}$. T_i also represents the location of the i-th target if there is no ambiguity. An *aspect* of the target, denoted by $\vec{v}$, is a vector that can be represented by an angle in $[0, 2\pi)$ with 0 degree indicating the one pointing to the right (east on the map). For ease of presentation, this angle is denoted by $arg(\vec{v})$ and is calculated by using arithmetic modulo 2π. For any angle $\alpha \in [0, 2\pi)$, $vec(\alpha)$ represents the corresponding vector.

Given a set of photos: $P = \{P_1, \ldots, P_n\}$, each photo P_j is stored locally and it can be registered to the server with a tuple $(l_j, r_j, \varphi_j, \vec{d_j})$, called the *metadata* of the photo. Here l_j is the location where the photo is taken. To simplify notations, we also use P_j to represent the location if there is no ambiguity. r_j and φ_j are two internal parameters of the camera used to take the photo. r_j is the effective range of the camera, and φ_j is the field-of-view (FOV, represented in angle) of the camera lens. $\vec{d_j}$ is the orientation of the camera when the photo is taken. Note that $\vec{d_j}$ is the normal vector derived from the camera lens and vertical to the image plane. It can be acquired by using various sensors embedded in the smartphone. Details of obtaining these geographical information on the smartphone will be given in Section 5. As shown in Figure 1(a), the metadata defines the effective coverage range of the photo.

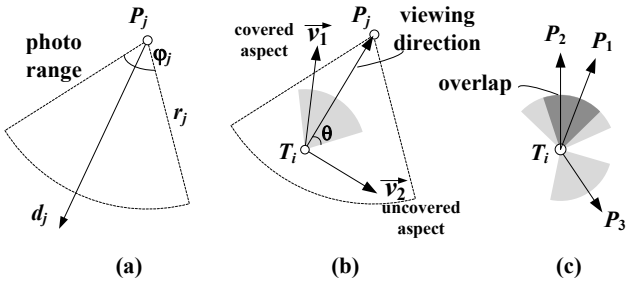


Figure 1: Photo, target and utility.

2.2 Photo Utility

For a target T_i and a photo P_j, T_i is said to be covered by P_j if P_j's range includes T_i. An aspect $\vec{v}$ of T_i is covered if the angle between $\vec{v}$ and $\overrightarrow{T_iP_j}$ is smaller or equal to a predefined angle θ called *effective angle*. Here $\overrightarrow{T_iP_j}$ is the viewing direction of the camera towards the target when the photo is taken[1]. Further, the utility of a photo P_j can be defined based on how many aspects of T_i are covered by this photo.

Definition 1. *[Utility] Given a target T_i and a photo P_j covering the target, the utility of P_j on T_i, denoted by $U_{P_j}(T_i)$, is the portion of aspect that is covered by P_j, i.e., $U_{P_j}(T_i) = \int_0^{2\pi} 1_{P_j}(v)dv$, where $1_{P_j}(v) = 1$ if $\vec{v}$ is covered by P_j, or 0 otherwise.*

Accordingly, the utility of a set of photos $P' = \{P_j : 1 \leq j \leq k\}$ regarding target T_i is the total portion of aspect that is covered by the photos of P', i.e., $U_{P'}(T_i) = \int_0^{2\pi} 1_{P'}(v)dv$, where $1_{P'}(v) = 1$ if $\vec{v}$ is covered by any P_j from P', or 0 otherwise.

Finally, the total utility of the photos regarding all targets $T = \{T_1, \ldots, T_m\}$ is the sum of the utility regarding each target. It is normalized by dividing the total number of targets, i.e., $U_{P'}(T) = \frac{1}{m}\sum_{i=1}^{m} U_{P'}(T_i)$.

For example in Figure 1(b), for target T_i, its aspect $\vec{v_1}$ is covered by photo P_j but aspect $\vec{v_2}$ is not. As a result, if there is only one photo covering the object, the utility is two times the effective angle θ (indicated by the gray area in Figure 1(b)). If there are multiple photos covering the same target, possible overlap (darker area in Figure 1(c)) among photos' coverage needs to be identified and removed. In that case, the overlap can only be counted once towards the total utility, which is reflected by gray area in Figure 1(c).

3. MAX-UTILITY WITH BANDWIDTH CONSTRAINT

In this section, we study the Max-Utility problem and its extension to an online optimization problem.

3.1 Max-Utility Problem

In the scenario described in Section 2, the bandwidth constraint determines the number of photos that can be selected. The problem is defined as follows.

[1]Intuitively, it should be from P_j to T_i, but $\overrightarrow{T_iP_j}$ is used for ease of calculation.

Definition 2. *[Max-Utility Problem] Given a set of m targets with known locations $T = \{T_1, \ldots, T_m\}$ and n photos $P = \{P_1, \ldots, P_n\}$ with known metadata, also given a predefined positive integer $B(\leq n)$, the problem asks for a selection of B photos P' out of the n candidates, such that the total utility of the selected photos $U_{P'}(T)$ is maximized.*

3.1.1 Conversion to Maximum Coverage

Without loss of generalization, we first consider a single target T_i and use the coverage interval $I_i = [0, 2\pi)$ to indicate its aspect to be covered. Let $P = \{P_1, \ldots, P_n\}$ be the set of all photos covering T_i. Then for each P_j, if T_i is covered by P_j, the coverage of P_j on T_i (gray sector in Figure 1(b)) can be represented by a sub-interval of $[0, 2\pi)$, i.e.,

$$S_j \triangleq [x_j, y_j] = [arg(\overrightarrow{T_iP_j}) - \theta, arg(\overrightarrow{T_iP_j}) + \theta] \qquad (1)$$

Note that the angles are always calculated by using arithmetic modulo 2π. Here the two end points x_j and y_j are called dividing points, which divides I_i into two parts: one is S_j and the other is $I_i - S_j$. If there are more photos by which T_i is covered, there would be more dividing points.

If there are multiple targets, every target corresponds to a coverage interval $I_i = [0, 2\pi)$ and each I_i is divided into sub-intervals by the corresponding dividing points. Let $U = \{e_1, \ldots, e_w\}$ be a universe set with each element representing a sub-interval and w being the total number of them. The weight of the element is the length of the sub-interval. For each photo P_j, a subset of U can be generated based on what sub-intervals are covered by it. Let S_j denote this subset. Then we have proved the following lemma:

Lemma 1. *A solution to the Max-Utility problem can be obtained by solving the following problem: given a universe set U of (non-negative) weighted elements, an integer B and a collection of subsets $S = \{S_1, \ldots, S_n\}$, find B subsets such that the total weight of the elements covered by the selected subsets is maximized.*

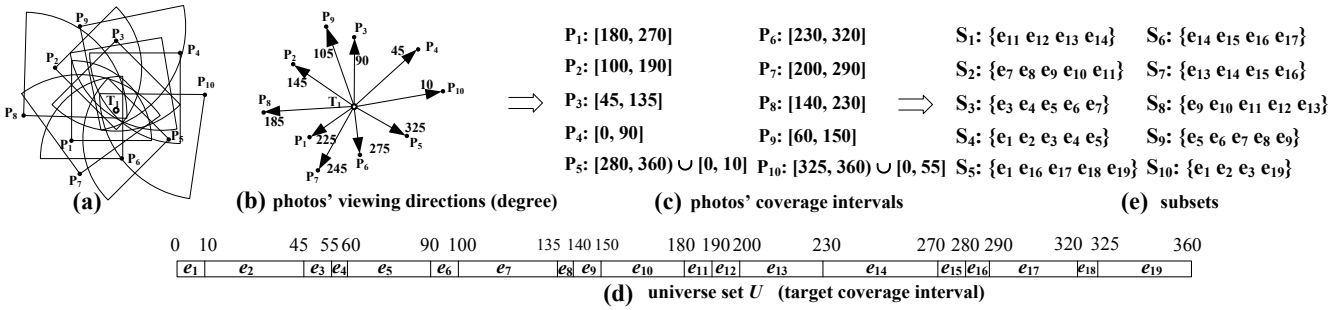
3.1.2 Greedy Selection Algorithm

The general maximum coverage problem is proved to be NP-hard [8]. A greedy algorithm can be used to find a solution. It works as a multi-round selection process. In each round, the weighted contribution (utility) of every unselected photos is calculated. The photo with the most contribution to the total utility is selected. If there are more than one photos with the most contribution, the one with the lowest index is selected. Once a photo is selected, it will be removed from the selection. The elements (sub-intervals) covered by the selected photo will be removed from future consideration. The selection process runs until B photos have been selected or every aspect of all targets has been covered, whichever comes first.

Theorem 1. *Let U_{opt} be the optimal value of the total utility that can be achieved by any B photos from P. Let U_{greedy} be the total utility achieved by the greedy selection algorithm. Then*

$$U_{greedy} \geq [1 - (1 - \frac{1}{B})^B] \cdot U_{opt} > (1 - \frac{1}{e})U_{opt}$$

Proof. From Lemma 1, a selection of B subsets implies a valid selection of B photos. Moreover, the total utility of the photos is maximized if and only if the corresponding



(a) (b) photos' viewing directions (degree) (c) photos' coverage intervals (e) subsets

(d) universe set U (target coverage interval)

Figure 2: The conversion into a set system.

subsets has the maximum total weight. On the other hand, the subsets selected by the greedy selection can yield a total weight that is at least $(1 - 1/e)$ times the optimal value [8]. Therefore, the total utility of the selected photos is also lower bounded by $(1 - 1/e)$ times the maximum total utility. □

An Example: Figure 2 shows an example of one target and 10 photos. Suppose $\theta = 45, B = 3$. Each photo's position is shown in Figure 2(a). The arrows in Figure 2(b) indicate photos' viewing directions, and the number beside the arrow (e.g., 10 beside $\overrightarrow{T_1P_{10}}$) indicates the angle of the viewing direction of the photo (e.g., P_{10}), which has been defined in Section 2. Based on this, each photo's coverage interval is calculated and shown in Figure 2(c) according to Equation (1). Then target T_1's coverage interval $I_1 = [0, 2\pi)$ is divided into sub-intervals by the endpoints of all photos' coverage intervals (Figure 2(d)). This is the universe set U which is composed of weighted elements from e_1 to e_{19}, and the weight of each element is reflected by its length. Finally, each photo's coverage interval is converted into a subset S_i of elements (Figure 2(e)).

We select 3 photos to maximize the total utility. Initially, each S_i has a weight of $2\theta = 90$, and hence S_1 is selected due to the smallest index. Elements $e_{11}, e_{12}, e_{13}, e_{14}$ are removed from U. Second, the weight of each of S_3, S_4, S_5, S_9 and S_{10} is still 90, but for the others the weights become: S_2 is 80; S_6 is 50; S_7 is 20 and S_8 is 40. Obviously, S_3 is selected. Then elements e_3, e_4, e_5, e_6, e_7 are removed from U. Finally, we consider the remaining subsets. The weights of S_5, S_6, S_7, S_8 are unchanged, but S_2 drops to 45, S_9 drops to 15 and S_{10} drops to 80. Therefore, the last selected photo is S_5. The final selection is S_1, S_3, S_5, corresponding to P_1, P_3, P_5, and the total achieved utility is 270.

3.2 Online Max-Utility Problem

For applications like crisis management, due to the urgency, the server should not wait for the metadata of all photos to come in before it begins the selection. Instead, it should start selecting photos from the beginning based on available metadata, and then gradually improve the photo coverage by continuously and periodically selecting photos when new ones become available.

3.2.1 Problem Statement

Let time be divided into transmission periods. At the beginning of each period, based on the available metadata and the available bandwidth in this period, the server makes decision on what photos to be uploaded in this period, and then notify the users to transfer the photos. Let t_i be the i-th period and B_i be the number of photos that can be uploaded in the i-th period. Then let A_i be the set of available photos (but not being uploaded yet) at the beginning of t_i. Finally, the selected photos to be uploaded in t_i is denoted by C_i. The problem is defined as follows.

Definition 3. *[Online Max-Utility Problem] Given a set of m targets with known locations $T = \{T_1, \ldots, T_m\}$, and the set of available photos A_i at the beginning of each period t_i, and suppose the event happens at period t_0, how does one select the set of photo C_i for each period in an online manner, such that $C_i \subseteq A_i$ and $|C_i| \leq B_i$, and at the end of each period t_i, the total utility of all the selected photos up to t_i, i.e., $U_{C_0 \cup \ldots \cup C_i}(T)$ (defined in Definition 1), is maximized?*

Note that the length of the period is a parameter determined by the application, e.g., how urgent the event is and how often new photos should be collected, etc. The bandwidth constraint B_i can vary from one period to another and not necessary to be constant.

3.2.2 Online Selection Algorithm

In each period, all the photos available up to present are considered. Finding the ones that can maximize the increase of total utility is easy when the number of photos is small, and an enumeration of all possible combinations can always deliver the optimal solution. However, as the process continues and more and more photos are available, computation cost would become prohibitively high.

Our solution is to use the approximation algorithm proposed for the original Max-Utility problem. At the beginning of each period, the server selects photos one by one greedily such that each one maximizes the increase of total utility, until it reaches the number imposed by B_i. Note that the conversion into a weighted set system is the same as before except that the aspects covered by photos selected in previous periods should be excluded. After that, the selected photos will be transferred immediately during the period. The performance of the online selection algorithm is evaluated later in Section 6.2.

4. ACHIEVING BEST UTILITY WITH MIN-SELECTION

In this section, we consider another important scenario: the number of photos is minimized while the total utility is to be above a required level. In many practical applications, the major obstacle is to deal with the sheer amount of raw data (photos) obtained via crowdsourcing. Thus, it is desirable to remove the redundancy and only keep the minimum selections of photos that can satisfy the coverage requirement.

4.1 Problem Statement

Each target T_i is associated with a coverage requirement, represented by a coverage interval $I_i = [a_i, b_i], 0 \leq a_i, b_i < 2\pi$. The requirement is met if any aspect $\vec{v}$ chosen from I_i is covered. The problem is defined as follows.

Definition 4. *[Min-Selection Problem] Given a set of m targets with known locations $T = \{T_1, \ldots, T_m\}$ and n photos $P = \{P_1, \ldots, P_n\}$ with known metadata, also given the coverage requirements for the targets: $I = \{I_1, \ldots, I_m\}$, the problem asks for a minimum selection of photos out of the n candidates, such that the coverage requirement for each target is met.*

Note that in this problem if the requirement can not be met due to the insufficiency of the original set of photos, the best achievable utility will be used as the criteria. Here the best achievable utility on a target is the utility of all photos on it, and the best achievable total utility on all targets is the sum on each targets normalized by the number of targets.

4.2 Min-Selection Algorithm

In the following description, it is assumed that the coverage requirement of each target can be satisfied by the whole set of photos. Then the following theorem shows the main result of our findings.

Theorem 2. *Suppose the target's coverage requirement can be satisfied by all photos in the pool and let N_{opt} be the minimum number of photos to satisfy the requirement. There exists N_{approx} photos that can be found in polynomial time such that each target's requirement can be met by these photos and moreover, $N_{approx} \leq O(\log n)N_{opt}$.*

PROOF. We prove this by constructing the selection using a greedy algorithm.

First, we use a conversion process that is similar to Section 3.1.1. Here each target T_i's coverage requirement I_i is partitioned into sub-intervals by the dividing points, and the dividing points are the end points of the coverage intervals (sub-intervals) of the photos like before. After this preparation, all the sub-intervals are numbered, and can be represented by elements that altogether form an universe set $U = \{e_1, \ldots, e_w\}$, where w is the total number of sub-intervals. Then for each P_j, there is a subset $S_j \subset U$ which is comprised of the elements corresponding to the sub-intervals covered by P_j. Based on this, the problem of finding the minimum photo selection can be converted to the following problem:

Given a universe set U and a collection of subsets of U: $S = \{S_1, \ldots, S_n\}$, and assume $\cup_j^n S_j = U$, how to find a subset S' of S such that $\cup_{S_j \in S'} S_j = U$ and $|S'|$ is minimum?

This is an instance of the Set Cover problem, which has been proved NP-hard [8]. Thus for the Min-Selection problem, we can solve it by an approximation algorithm based on the greedy selection.

Specifically, the algorithm begins by selecting the photo (some S_j) that covers the most number of sub-intervals (elements). Once a photo is selected, it will not be removed. The sub-intervals covered will not be considered in the future. Photos are selected one by one based on how many new sub-intervals can be covered. Each time, the photo covering the most number of new sub-intervals is selected. Ties can be broken arbitrarily, e.g., by giving priority to the one with smaller index. The process stops if all sub-intervals (elements of U) is covered or no more photos can be selected (i.e., either photos are all selected or no more benefit can be achieved).

Once the photos are found, it is obvious all the elements in U is covered which implies the requirement of all targets are satisfied. By using similar argument from Theorem 3.1 in [8], it is easy to see the number of selected photos is upper bounded as shown in the theorem. □

An Example: Again, we use Figure 2 to illustrate the above idea. Consider the problem settings in Figure 2(a) and suppose the required coverage for T_1 is $[0, 360)$. The construction of the universe set and all the subsets are shown in Figure 2(b)-(e). The universe set U consists of 19 elements. The selection works on the subsets S_i.
First, photo S_2 is selected as it covers 5 new elements $\{e_7, e_8, e_9, e_{10}, e_{11}\}$. It has the most elements covered and the smallest index. Then S_5 can be selected as it covers 5, the most number of new elements $\{e_1, e_{16}, e_{17}, e_{18}, e_{19}\}$. In the third round, S_3 can be selected, as it covers 4 new elements $\{e_3, e_4, e_5, e_6\}$. After that, S_1, which covers 3 new elements $\{e_{12}, e_{13}, e_{14}\}$, is selected. Up to now, 17 out of the total 19 elements have been covered. The remaining two are e_2 and e_{15}. To cover e_2, S_4 is selected. Then S_6 is selected to cover e_{15}. The final selection is $S_1, \ldots, S_6$, which correspond to the following 6 photos: $P_1, P_2, P_3, P_4, P_5, P_6$.

The above discussion can be easily applied to the scenario of multiple targets. In that case, each target corresponds to a set U_i of elements (sub-intervals). Elements of all U_i will be considered to determine if a particular S_j can yield the most coverage. The algorithm stops if elements of all U_i are covered or no more progress can be made.

5. TESTBED IMPLEMENTATION

A prototype of SmartPhoto has been implemented in a testbed using Samsung Nexus S running Android 2.3.6, Samsung Galaxy S III running Android 4.0.4, and Google (LG) Nexus 4 running Android 4.2.

In the testbed, the smartphones take photos with the metadata automatically recorded. The metadata is a tuple comprised of a GPS location, a range indicating how far the photo can cover, a field-of-view (FoV) angle of the camera taking the photo and an orientation vector indicating the facing direction of the camera lens. After the photo has been taken, the smartphone uploads the metadata of the photo to a centralized server, which is a PC in our lab running the photo selection algorithm. Then the server notifies the smartphones to upload the selected photos. In this section, we present the technical details on how to obtain the metadata, how to improve the accuracy of orientation measurement, and how to deal with occlusion and out-of-focus issues.

5.1 Metadata Acquisition

One critical issue is how to get the metadata from off-the-shelf smartphones. The location can be directly acquired via the built-in GPS receiver. The camera's field-of-view is accessible via the Android camera API [1]. The range is a little trickier as it depends on the resolution of the lens (and the image sensor), the zooming level (or focal length) and the requirement of the application. Applications requiring a survey of large scale buildings may find the photos useful

even if they are taken a hundred meters away by a lower resolution camera, while others may require closer look at the object and hence may exclude photos taken more than a few meters away. In our experiment, as the subjects are buildings on campus, 50 meter is used as a reference range. We find that for our purpose, objects in photos taken within this range are generally recognizable.

Orientation is a key factor that has not yet been fully taken advantage of in previous works. The way used to characterize the orientation in the Android system is to first define a local and a world coordinate system[2], and represent the orientation as a rotation matrix. The rotation matrix is used to transform a local coordinate tuple into a global one. Another way to represent the rotation is to use a three tuple called *azimuth, pitch,* and *roll*, which respectively indicate the phone's rotation around the Z, X and Y axes [13]. The two representations are equivalent and the orientation tuple (i.e., the angles) can be derived from the rotation matrix. In the following description, we use R to denote the rotation matrix.

In Android system, the rotation matrix can be directly obtained based on accelerometer and magnetic field sensor readings. The accelerometer measures the phone's proper acceleration along the three axes in the phone's local coordinate system, including the influence of the gravity. The magnetic field sensor provides the readings measuring the ambient magnetic field along the three axes in the local coordinate system. The coordinates of both the gravity and the ambient magnetic field are known in the world coordinate system. Thus, by combining the above readings and facts, the orientation of the phone can be obtained. Let us call this the "basic" method, and let the result be denoted by R_{basic}.

5.2 Techniques to Improve Accuracy

The rotation matrix R_{basic} is susceptible to noise and errors. It fluctuates quickly due to the vibration of accelerometer's reading. Also, the magnet field sensor's reading is easily affected by nearby magnet objects. Even worse, R_{basic} responses slowly to quick rotation of the phone. Thus, we propose several techniques to improve the accuracy of the orientation.

5.2.1 Hybrid Method

Apart from the accelerometer and the magnetic field sensor, gyroscope is now available in most smartphones, and it can also be used to measure the rotation matrix.

Gyroscope measures the angular rotation speeds along all three axes in the phone's local coordinate system. By integrating (multiplying) the angular speed with the time interval between two consecutive sensor readings, we can obtain the rotation vector, which indicates the change of orientation in terms of rotation angles around the three axes. It can also be used to obtain the rotation matrix (denoted by ΔR_{gy}). Given an initial rotation matrix, which can be obtained from

[2]In a world coordinate system, Z axis is perpendicular to ground and points to the sky; Y is tangential to the ground and points towards the magnetic north pole; X is the vector product of Y and Z. In the phone's local coordinate system, Z is perpendicular to the phone screen and points outward; the X axis is along the width of the phone and the Y axis is along the length [1, 13].

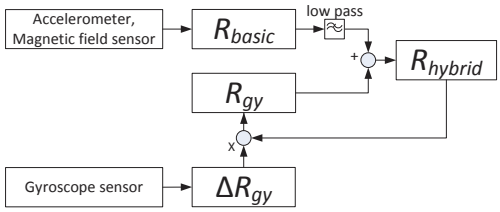


Figure 3: Hybrid method.

R_{basic}, we can get the new rotation matrix, denoted as R_{gy}, by $R_{gy} = R_{gy} \times \Delta R_{gy}$.

However, the cumulative error caused by the integration in R_{gy} can become greater and the result would drift as time goes by. In fact, the orientation derived from R_{gy} alone usually drifts over 10 degrees in about 20 seconds in our lab test.

Thus, we propose a hybrid method which combines the readings from the above sensors to improve the accuracy of orientation, as shown in Figure 3 and explained as below.

First, a simple Infinite Impulse Response (IIR) low pass filter is used on R_{basic} to remove the short term vibration, i.e.,

$$R'_{basic} = R_{basic} + \mu \cdot (R^{prev}_{basic} - R_{basic})$$

where R_{basic} is the current reading and R^{prev}_{basic} is the previous reading from the basic method, and $\mu \in [0, 1]$ is an adjustable parameter balancing the current and previous values. In practice, we find $\mu = 0.3$ is good for our purpose.

Second, we combine R'_{basic} and R_{gy} to take advantage of both values; that is,

$$R_{hybrid} = \nu \times R_{gy} + (1 - \nu) \times R'_{basic}$$

We find $\nu = 0.9$ works well.

Third, R_{hybrid} is the output, and it will also be used as the initial matrix input for the computation of a new R_{gy}.

5.2.2 Enhancement by Orthonormalization

We exploit the orthonormal property of the rotation matrix to further improve the accuracy of orientation. In a valid rotation matrix, any pair of columns (or rows) of the rotation matrix are orthogonal, i.e., with unit length and vertical to each other. However, this property may be violated as errors occur. Thus, the rotation matrix R_{hybrid} obtained from the above method can be further calibrated by an orthonormalization process (e.g., the Gram-Schmidt process [24]) to get an enhanced rotation matrix $R_{enhanced}$.

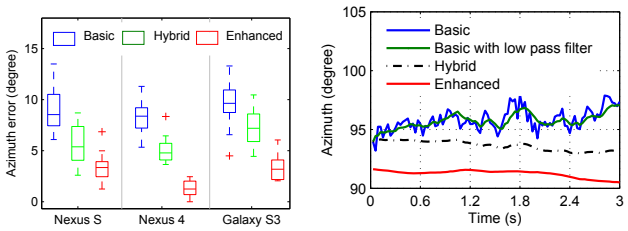
Specifically, consider a 3×3 rotation matrix: $R_{hybrid} = [\alpha_1, \alpha_2, \alpha_3]$, with α_i being a column vector. Let the inner product of the two vectors α and β be $< \alpha, \beta >= \sum_{i=1}^{n} \alpha_i \beta_j$, where $n = 3$ is the dimension.

First, R_{hybrid} is orthogonalized by

$$\begin{aligned}
\xi_1 &= \alpha_1 \\
\xi_2 &= \alpha_2 - \frac{< \alpha_2, \xi_1 >}{< \xi_1, \xi_1 >} \xi_1 \\
\xi_3 &= \alpha_3 - \frac{< \alpha_3, \xi_1 >}{< \xi_1, \xi_1 >} \xi_1 - \frac{< \alpha_3, \xi_2 >}{< \xi_2, \xi_2 >} \xi_2
\end{aligned}$$

Second, the above ξ_i's are normalized by

$$\beta_i = \frac{\xi_i}{\sqrt{< \xi_i, \xi_i >}}, i = 1, 2, 3$$



(a) (b)

Figure 4: Orientation errors.

Then, the final rotation matrix is

$$R_{enhanced} = [\beta_1, \beta_2, \beta_3]$$

Comparisons: To verify the effectiveness of the optimization techniques, we measure the orientation using three different methods: the "basic" method, the "hybrid" method, and the "enhanced" method, and compare their results. We place the phone in a horizontal plane, so the orientation is reflected by the azimuth value. Then we rotate the phone 30 degrees and measure its azimuth reading against a commercial compass. Each measurement is repeated 50 times and the statistics are calculated. Figure 4(a) compares the measurement errors (in degree) by these three methods. The short bar in the middle of each box is the median value of the azimuth reading error, and the lower and upper side of the box are the first (25%) and third (75%) quartile, which is denoted by $Q1$ and $Q3$. Then the lower limit is calculated by $Q1-1.5*(Q3-Q1)$ and the upper limit is $Q3+1.5*(Q3-Q1)$. More details about the average error and standard variance of each method are listed in Table 1.

We find that the hybrid method can reduce the average measurement error by 37% compared to the basic method, and the enhanced method can further reduce the measurement error by more than 40% compared to the hybrid method. Also, new phones (e.g., Nexus 4), with more advanced hardware and OS, are more accurate with less variance. For all these phones, with our enhanced method, the average azimuth reading error is under 3.5 degrees, and the error can be reduced to 1.3 degree with the Nexus 4 phone.

Table 1: Average error in azimuth (degree)

	Nexus S	Nexus 4	Galaxy S III
Basic	9.1(±2.0)	8.2(±1.5)	9.6(±2.4)
Hybrid	5.7(±1.9)	5.1(±1.3)	7.3(±1.7)
Enhanced	3.4(±1.4)	1.3(±0.7)	3.4(±1.3)

To understand the effectiveness of these techniques clearly, we show the measurement results of these methods when the phone is turned to 90 degree, and the results are illustrated in Figure 4(b). As can be seen, the basic method oscillates frequently. The hybrid method improves the accuracy compared to the basic method but still carries the reading errors. With orthonormalization, the enhanced method can significantly improve the accuracy of orientation.

5.3 Occlusion and Out-of-focus

After a photo is taken, we assume that the user will visually check if the object appears in the photo, as most people do. However, if the user does not check the photo, and the object is blocked by unexpected obstacles such as a moving vehicle, the photo will not be useful to the server. Even if the user checks the photo and the object is clear, it may be

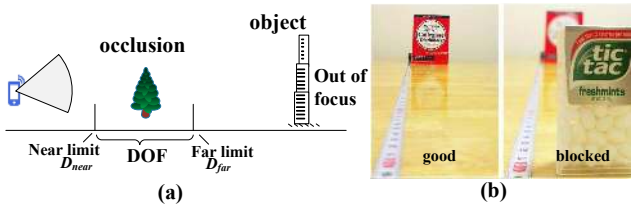


Figure 5: The use of DOF.

different from what the server is expecting. For example, the server may expect the photo to be about a building, but the user may be looking at a tree in front of the building. Although in the two scenarios, the smartphone may produce the same metadata (e.g., the same facing direction), the content could be very different, and the one focusing on (blocked by) the tree is useless for the server's task. Besides this problem, there are many other occasions that the interested targets are out-of-focus. Uploading these photos will waste lots of bandwidths and storage spaces.

We use a feature called *focus distance*, which is provided by many new smartphones with focusing capability, to solve the problem. The focus distance is the distance between the camera and the object perfectly focused in the photo. Note that the real distance between the camera and our interested target can be calculated by GPS locations. Thus in an ideal case, if the two distances do not match, the target is out-of-focus and the photo should be excluded from consideration.

The measurement of the focus distance is sensible to errors. A slight offset does not necessarily mean the target is out-of-focus. In fact, in photography the distance between the nearest and farthest objects that appear acceptable sharp in a photo is called the Depth-Of-Field (DOF). DOF is determined by four parameters: focal length (f), focus distance (v_o), lens aperture (A), and circle of confusion (CoC). Among these parameters, focal length and lens aperture are built-in and readable from the Android API. CoC (denoted by c) is a predefined number which determines the resolution limit for our application. Focus distance changes from case to case but obtainable from Android API. Therefore, we can calculate the near/far limit of DOF (Figure 5(a)) by the following formulas:

$$D_{near} = \frac{v_o(H-f)}{H+v_o-2f}$$

$$D_{far} = \frac{v_o(H-f)}{H-v_o}$$

where $H = \frac{f^2}{Ac} + f$ is the hyperfocal distance [15].

After a photo is taken, the distance between the target and the camera (phone) is compared with the above two values. If the target falls into the DOF, the photo is considered valid; otherwise, it will be dropped. For example, consider the two photos in Figure 5(b). The dictionary is the interested target. In the left photo, the near and far limit of DOF is 85cm and 105cm respectively. In the right photo, the near and far limit of DOF is 5cm and 10cm respectively. The distance between the camera and the dictionary is 100cm. Based on these parameters, it is clear that the target falls into the DOF in the left photo. From the figure we can see, the dictionary is clear in the left photo but blocked by another object in the right photo. Note that this filtering is done at the user side and the metadata of unqualified photos will not be sent to the server.

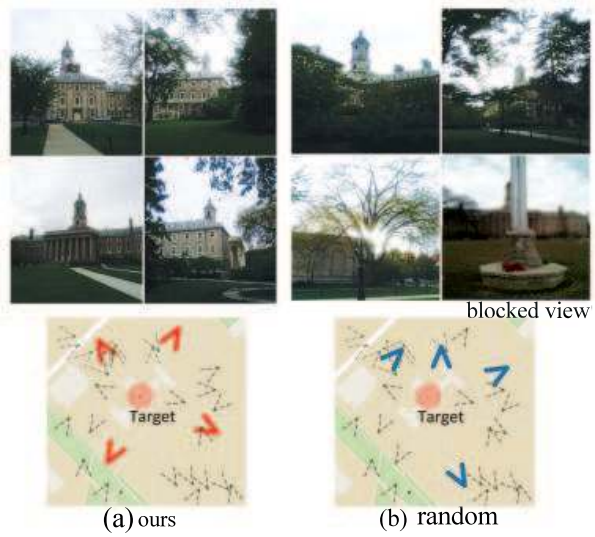


Figure 6: Demo results.

Discussions: Photos can be of low quality due to various reasons. Over-exposure or under-exposure causes images to be too bright or too dark; camera movement and shutter speed affect how severe the image is blurred; the quality of lens and digital sensors is also important. These factors are neither included in our coverage model, nor considered by the DOF method. They can be analyzed only by image processing. Thus, before photo selection, some efficient image processing techniques [7] may be applied at the user end to achieve basic quality control, such as detecting photos that are too dark or severely blurred, and thus low quality photos will not be considered in the photo selection. However, existing image processing techniques are computationally expensive, and thus should be carefully adapted considering the resource limitations of mobile devices. Note that our approach is not meant to replace the role played by image recognition algorithms, but to serve as an important complement to improve the utility of the collected photos, especially when there are resource constraints.

6. PERFORMANCE EVALUATIONS

In this section, we first show a real world demo using the smartphone testbed, and then evaluate the performance of the photo selection algorithms by extensive simulations.

6.1 Demo in a Real-World Example

The above testbed is used in a real-world example to demonstrate the effectiveness of the proposed photo selection algorithm. In this demonstration, a landmark (a bell tower) is the target. Photos are taken by using the reprogrammed smartphones around the target with the metadata automatically recorded. The metadata of all photos are later uploaded into a centralized desktop server. There are 30 photos in total. Although most of them are taken around the target, some are not facing the target, and some are blocked by trees or other objects. Also, the distribution is not uniform, due to the reality that people are likely to take pictures of the front (more attractive) side of the building.

After the metadata is retrieved, the Max-Utility problem is solved by choosing 4 photos. The results are shown in Figure 6(a), with the image shown at the top and the positions and orientations of the photos are shown at the bottom. Here the positions and orientations of the original 30 photos are marked as dotted "V" shape, and the selected photos are marked by bold lines. As a comparison, the 4 photos chosen by a random selection algorithm are shown in Figure 6(b). It can be seen obviously that the 4 photos chosen by our algorithm cover the target from 4 different locations well separated from each other, with each one from a totally different angle. The bell towers are viewable from all 4 photos. On the other side, only 2 photos chosen by the random algorithm cover the target. The third photo is facing away from the target, which is because the random selection does not consider the orientation. In the fourth photo (at bottom right), the target is blocked by a **flagpole**, which is bad for random selection. Note that this photo's orientation (in bottom map) does not reveal the real situation. However, based on the DOF information, the target is out of focus. Hence, it is filtered out at the user end and is not considered by our selection.

6.2 Simulation Results

In this section, we evaluate the photo selection algorithms through simulations. The target is randomly distributed in a 100m by 100m square area. Photos are assumed to be taken at random positions, with random orientations from 0 to 2π, in a larger area that is a 200m by 200m square, with the target area in the center.

During the simulation, the random selection algorithm is used for comparison. For a fair comparison, the random selection excludes any photos that have no target covered, but only consider photos that cover at least one target, i.e., relevant photos. Note that a more naive selection could be blindly selecting photos without considering this.

6.2.1 Results on Max-Utility

In the first part, we evaluate the performance of our algorithm on addressing the Max-Utility problem. Intuitively, with more bandwidths (larger B), better coverage of the targets (total utility) can be achieved. As shown in Figure 7(a), both our algorithm (denoted by "ours") and the random selection (denoted by "random") achieve more utility (y coordinate) as B (x coordinate) increases. The total utility achieved by all photos (denoted by "best achievable") is also shown to provide an upper bound. The difference between our selection and the random selection is significant and the advantage of our algorithm is obvious especially when B is smaller, i.e., bandwidth is more constrained. Although the performance of both algorithms converges to the best-achievable utility as B becomes larger, the convergence of ours is much faster.

Figure 7(b) shows how the total utility changes as the number of candidate photos increases while other factors including bandwidth ($B = 20$) remain unchanged. The advantage of our algorithm is significant across the range. Considering the bandwidth limitation (only 20 photos can be selected to cover 30 targets), the difference between the utility achieved by ours and the best achievable level is small. Moreover, our algorithm can take advantage of the increasing density of photos, and improve its performance as the number of photos increases.

6.2.2 Results on Online Max-Utility

In this part, the algorithm for the online Max-Utility problem is evaluated. We first observe how the total utility

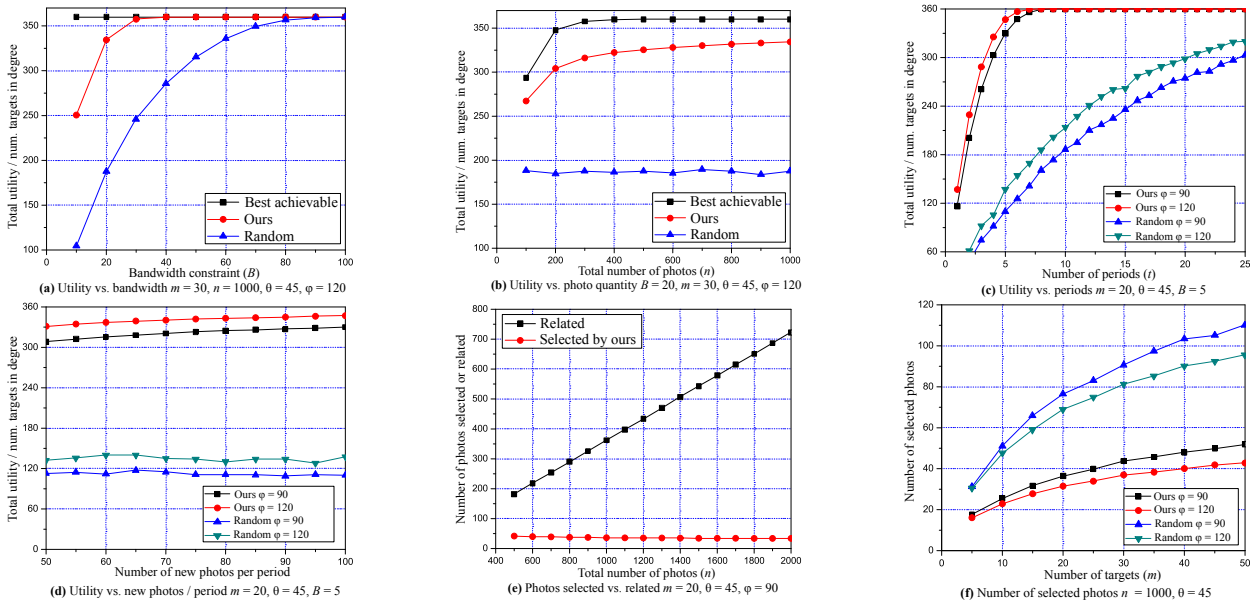


Figure 7: Simulation results: (a)-(b) Max-Utility; (c)-(d) Online Max-Utility; (e)-(f) Min-Selection.

changes as the number of periods increases in Figure 7(c). Here, we use the same target distribution as above, and there are 100 new photos available in every period. Photo parameters are shown in the figure. The normalized total utility at the end of each period is recorded. As can be seen, for both our algorithm and the random algorithm, the total utility increases as the number of periods increases. However, for our algorithm, it quickly approaches to 360. It is actually above 350 after t_7, which means by that time, almost all aspects of the targets are covered by the selected photos. The random algorithm takes much longer (after t_{25}) to reach that level of coverage. Thus, our algorithm is more responsive and effective.

Next, we vary the number of new photos from 50 to 100, with other parameters the same as above except the number of periods which is now fixed to be 5. The total utility of the selected photos after t_5 is shown in Figure 7(d). As the number of available photos increases, the selection algorithm has more choices. As a result, the total utility improves and approaches 360 in our algorithm. In comparison, the performance of the random algorithm is flat (a little fluctuated due to randomness) and very low. Given the same time period, the total utility of our algorithm is much higher than the random algorithm across the range.

6.2.3 *Results on Min-Selection Problem*

In this part the Min-Selection problem is studied. In reality, the given pool of photos can be very large and the number of relevant photos (i.e., photos covering at least one target) can increase very fast as the total number of randomly taken photos increases. Then, a careful selection of photos can greatly reduce the redundancy. Figure 7(c) shows the effectiveness of our selection algorithm on reducing the redundancy. There are 20 targets on the field, and the camera parameters are shown in the figure. As the total number of photos varies from 500 to 2000, the number of related photos (denoted by "related") increases linearly. However, the number of photos selected by our algorithm (denoted by "selected by ours") to achieve the same coverage does not increase. It actually decreases slightly since our algorithm takes advantage of the increased density of the photos and improves its efficiency.

The algorithms are also evaluated under the situation that the number of targets (m) varies from 5 to 50, while the total number of photos is fixed to be 1000 and all other factors remain the same. As shown in Figure 7(d), the algorithms have to select more photos to cover the increased number of targets. However, the number of photos selected by our algorithm remains very low, and the increasing speed is much slower as the number of targets increases, which is much better than the random algorithm.

7. RELATED WORK

The mass adoption of camera sensors and other position sensors on smartphones makes photo taking and sharing via online social networks much easier and more enjoyable. It creates opportunities for many applications based on camera sensor networks, which have received much attention recently in research [2, 16, 19, 11, 17]. One basic problem is how to characterize the usefulness of the image data and how to optimize the network to achieve better quality of information. However, very little effort has been devoted to this field. One problem studied is called pan and scan [10], which is proposed to maximize the total coverage in a camera sensor networks. For camera sensor placement, various optimization models and heuristics are studied in [9]. However, the coverage model is relatively simple, depending only on the distance between the target and the object, which does not consider the uniqueness of photo coverage.

Our work is inspired by the full-view coverage model which was originally proposed in [22] and later extended in [21, 23]. An object is considered to be full-view covered if no matter which direction the object faces, there is always a sensor whose sensing range includes the object and that sensor's viewing direction is sufficiently close to the object's facing direction. Although our work is based on the full-view coverage model, our model is more general and we study vari-

ous optimization problems on the tradeoffs between resource constraints and photo coverage.

Another interesting work is PhotoNet [20], which is a picture delivery service that prioritizes the transmission of photos by considering the location, time stamp, and color difference, with the goal of maximizing the "diversity" of the photos. Compared to their model, we consider direction and angle information, and develop techniques to obtain them through off-the-shelf smartphones. These are very important and unique features for photos and enable us to develop much finer optimization models. Moreover, the solutions to our optimization problems are rigorously analyzed.

It is also worth mentioning that there has been much progress in content-based image retrieval techniques (see [6] for a good survey). These techniques have also been used for images obtained from mobile users. One example is to build photo annotated world maps and create 3D models of the objects from 2D photos via online social networks [5]. Some other interesting works have been done for image retrieval/search on smartphones, e.g., [25]. However, most of these works involve power-intensive computation at both user and server end, and some demands human validation to be included into the cycle [25]. These techniques are challenged by the content diversity and the resource constraints.

8. CONCLUSIONS

We proposed a resource-aware framework, called SmartPhoto, to optimize the selection of crowdsourced photos based on the accessible metadata of the smartphone including GPS location, phone orientation, etc. With this model, a remote server can efficiently evaluate and select photos from mobile users under severely constrained resources such as bandwidth, storage, computational power and device energy. Three optimization problems regarding the tradeoffs between photo coverage and resource constraints have been studied. One proposed algorithm is to maximize the total utility subject to the bandwidth constraint. It has been further extended into an online selection algorithm that can periodically select and collect photos to satisfy the requirement of time critical applications. We also provided solutions to minimize the bandwidth usage while satisfying the coverage requirement. The approximation bounds of the algorithms are theoretically proved. We have implemented SmartPhoto in a testbed using Android based smartphones and a desktop as the remote server, and proposed techniques to improve the accuracy of the collected metadata and mitigate the occlusion and out-of-focus issues. Results based on real implementations and extensive simulations validated the effectiveness of the proposed algorithms.

9. ACKNOWLEDGMENT

This work was supported in part by Network Science CTA under grant W911NF-09-2-0053.

10. REFERENCES

[1] SensorManager - Android Developer. `http://developer.android.com/reference/android/hardware/SensorManager.html`.

[2] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury. A survey on wireless multimedia sensor networks. *Comput. Netw.*, 51(4):921–960, 2007.

[3] N. Alon and J. H. Spencer. *We the Media: Grassroots Journalism by the People, for the People.* O'Reilly Media, Sebastopol, California, 2006.

[4] V. Blanz, P. Grother, P. J. Phillips, and T. Vetter. Face recognition based on frontal views generated from non-frontal images. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 454–461, 2005.

[5] D. Crandall and N. Snavely. Modeling people and places with internet photo collections. *Commun. ACM*, 55(6):52–60, June 2012.

[6] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):5:1–5:60, May 2008.

[7] S. Gabarda and G. Cristobal. Blind image quality assessment through anisotropy. *Journal of the Optical Society of America A*, 24(12):B42–B51, 2007.

[8] D. S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, 1996.

[9] E. Hörster and R. Lienhart. On the optimal placement of multiple visual sensors. In *Proc. ACM International Workshop on Video Surveillance and Sensor Networks (VSSN)*, 2006.

[10] M. P. Johnson and A. Bar-Noy. Pan and scan: Configuring cameras for coverage. In *IEEE INFOCOM*, 2011.

[11] F. Li, J. Barabas, and A. L. Santos. Information processing for live photo mosaic with a group of wireless image sensors. In *Proc. ACM/IEEE Conference on Information Processing in Sensor Networks (IPSN)*, 2010.

[12] S. Liu, L. Palen, J. Sutton, A. Hughes, and S. Vieweg. In search of the bigger picture: The emergent role of on-line photo-sharing in times of disaster. In *Proc. ISCRAM*, 2008.

[13] R. Meier. *Professional Android 2 Application Development, 2nd Ed.* Wiley Publishing, Inc., 2010.

[14] P. J. Phillips, W. T. Scruggs, A. J. O'Toole, P. J. Flynn, K. W. Bowyer, C. L. Schott, and M. Sharpe. FRVT 2006 and ICE 2006 large-scale results. *National Institute of Standards and Technology, Tech. Rep. NISTIR 7408*, 2007.

[15] S. F. Ray. *Applied Photographic Optics.* ocal Press, Oxford, UK, 3 edition, 2002.

[16] B. Rinner and W. Wolf. A bright future for distributed smart cameras. *Proceedings of the IEEE*, 96(10):1562 – 1564, 2008.

[17] Y. Shen, W. Hu, M. Yang, J. Liu, and C. T. Chou. Efficient background subtraction for tracking in embedded camera networks. In *Proc. ACM/IEEE IPSN*, 2012.

[18] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *Proc. ACM SIGGRAPH*, 2006.

[19] S. Soro and W. Heinzelman. A survey of visual sensor networks. *Advances in Multimedia*, 2009:1–22, 2009.

[20] M. Y. S. Uddin, H. Wang, F. Saremi, G.-J. Qi, T. Abdelzaher, and T. Huang. Photonet: A similarity-aware picture delivery service for situation awareness. In *Proc. IEEE Real-Time Systems Symposium (RTSS)*, 2011.

[21] Y. Wang and G. Cao. Barrier coverage in camera sensor networks. In *Proc. ACM Mobihoc*, 2011.

[22] Y. Wang and G. Cao. On full-view coverage in camera sensor networks. In *Proc. IEEE INFOCOM*, 2011.

[23] Y. Wang and G. Cao. Achieving full-view coverage in camera sensor networks. *ACM Transactions on Sensor Networks (ToSN)*, 10(1), 2013.

[24] G. Williams. *Linear Algebra With Applications.* Jones & Bartlett Learning, 2012.

[25] T. Yan, V. Kumar, and D. Ganesan. Crowdsearch: exploiting crowds for accurate real-time image search on mobile phones. In *Proc. ACM MobiSys*, 2010.

[26] C. Zhu, K. Li, Q. Lv, L. Shang, and R. P. Dick. iScope: personalized multi-modality image search for mobile devices. In *Proc. ACM MobiSys*, 2009.