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

Yibo Wu, *Student Member, IEEE*, Yi Wang, Wenjie Hu, *Student Member, IEEE*, and Guohong Cao, *Fellow, IEEE*

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. Four optimization problems regarding the tradeoffs between photo utility and resource constraints, namely Max-Utility, online Max-Utility, Min-Selection, and Min-Selection with k-coverage, 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.

Index Terms—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 [20], and even disaster recovery and emergency management [14].

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 [7], [27], [28] may demand too much computational and communication resources at both user and server end. 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 [22]. 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 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

- Y. Wu, W. Hu, and G. Cao are with the Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802. E-mail: {yxw185, wwh5068, gcao}@cse.psu.edu.
- Y. Wang is with the Google Inc., Mountain View, CA 94043. E-mail: yuw124@cse.psu.edu.

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. We also consider having certain level of redundancy in case better coverage is needed.

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 study the Min-Selection problem for redundancy reduction, and also extend it to the case where better coverage (e.g., k-coverage) is desired. 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 post-disaster 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, only a small number of photos can be transferred due to the limited bandwidth caused by damage to base stations or overwhelming cellular traffic. For this reason, reporters first transmit the metadata of the photos, which is extremely lightweight compared to the original images. 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], [16]. 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 ith 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.

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 Fig. 1a, the metadata defines the effective coverage range of the photo.

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
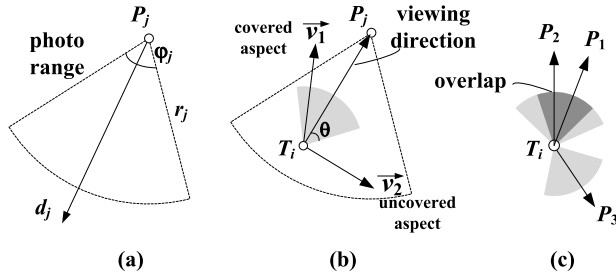
Fig. 1. (a) Metadata defines the effective coverage range of a photo. (b) An aspect is covered if it is close to the viewing direction. (c) Coverage overlap shows the redundancy in the photos.

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 Fig. 1b, target T_i is covered by photo P_j. Its aspect $\vec{v_1}$ is covered by P_j but aspect $\vec{v_2}$ is not. In fact, P_j covers all the aspects in $[arg(\overrightarrow{T_iP_j}) - \theta, arg(\overrightarrow{T_iP_j}) + \theta]$ (gray sector in Fig. 1b). Thus, P_j's utility is 2θ. If there are multiple photos covering the same target, possible overlap (darker area in Fig. 1c) 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 the gray area in Fig. 1c.

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.

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.*

---

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

3.1.1 Conversion to Maximum Coverage

Without loss of generality, 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 Fig. 1b) can be represented by a sub-interval of $[0, 2\pi)$, i.e.,

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

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 [10]. 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*

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

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 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 [10]. Therefore, the total utility of the
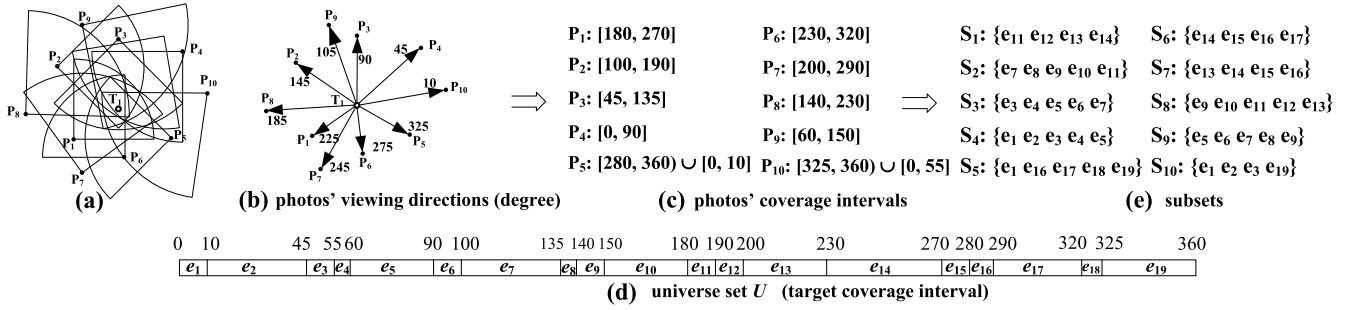
P_1: [180, 270] P_6: [230, 320] S_1: {e_{11} e_{12} e_{13} e_{14}} S_6: {e_{14} e_{15} e_{16} e_{17}}

P_2: [100, 190] P_7: [200, 290] S_2: {e_7 e_8 e_9 e_{10} e_{11}} S_7: {e_{13} e_{14} e_{15} e_{16}}

P_3: [45, 135] P_8: [140, 230] S_3: {e_3 e_4 e_5 e_6 e_7} S_8: {e_9 e_{10} e_{11} e_{12} e_{13}}

P_4: [0, 90] P_9: [60, 150] S_4: {e_1 e_2 e_3 e_4 e_5} S_9: {e_5 e_6 e_7 e_8 e_9}

P_5: [280, 360] $\cup$ [0, 10] P_{10}: [325, 360] $\cup$ [0, 55] S_5: {e_1 e_{16} e_{17} e_{18} e_{19}} S_{10}: {e_1 e_2 e_3 e_{19}}

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

(d)   universe set U (target coverage interval)

Fig. 2. The conversion into a set system.

selected photos is also lower bounded by $(1 - 1/e)$ times the maximum total utility.   □

An Example. Fig. 2 shows an example of one target and 10 photos. Suppose $\theta = 45°, B = 3$. Each photo's position is shown in Fig. 2a. The arrows in Fig. 2b indicate photos' viewing directions, and the number beside the arrow (e.g., 10 beside $\overrightarrow{T_1 P_{10}}$) 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 Fig. 2c 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 (Fig. 2d). 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 (Fig. 2e).

We select three 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 degree, but for the others the weights become: S_2 is 80 degree; S_6 is 50 degree; S_7 is 20 degree and S_8 is 40 degree. 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 degree, S_9 drops to 15 degree and S_{10} drops to 80 degree. 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 degree.

Theorem 2. *For the Max-Utility problem, the worst case time complexity of the greedy algorithm is $O(Bn^2m)$.*

Proof. Since there are n photos and m targets, calculating the coverage intervals of each photo on each target takes $O(mn)$ time. Then, a target can be covered by at most n photos, so there are $O(n)$ coverage intervals on a target. Sorting the endpoints of those coverage intervals gives us the universe set, which takes $O(mn \log n)$ time for all m targets.

Next, photo selection is done in B steps. There are n candidate photos to consider in the first step, but the number decreases by 1 after each step. With assumption $B \ll n$, the number of candidate photos to consider is $\Theta(n)$ for any of the B steps. In total, we consider candidate photos for $\Theta(Bn)$ times. Now let's look at how much time it takes to handle one candidate photo. For each of the m targets, we need to add the weights of the elements covered by that photo. The number of those elements is

proportional to the total number of elements on the target, which is in the order of $O(n)$. For example, given effective angle $\theta = 45°$, a photo always covers 1/4 of the total aspects, so the elements it covers are 1/4 of the total number of elements. This means calculating the utility of one candidate photo takes $O(mn)$. Getting these together, the selection process takes $O(Bn \cdot mn) = O(Bn^2m)$ time.

To summarize, the worse case time complexity is $O(mn + mn \log n + Bn^2m) = O(Bn^2m)$.   □

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 ith period and B_i be the number of photos that can be uploaded in the ith 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 is not necessarily a constant.

3.2.2 Online Selection Algorithm and Analysis

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.

Then we prove the competitive ratio [5] of the above online selection algorithm.

Theorem 3. *For any given integer s, let U_{online} be the total utility of the selected photos up to period t_s by the online selection algorithm, and let U_{opt} be the total utility of the optimal set of selected photos, subject to the constraints in Definition 3, then $\frac{1}{2}(1 - (1 - 1/B_{min})^{B_{min}}) \cdot U_{opt} \leq U_{online}$, where $B_{min} = \min\{B_1, \ldots, B_t\}$, and $\frac{1-(1-1/B_{min})^{B_{min}}}{2} \geq \frac{(1-1/e)}{2} \approx 0.32$.*

Proof. To simplify the presentation, we first define some notations used in the proof. Let C_i denote the photos selected by our algorithm in t_i and C_i^{opt} denote the photos selected by an optimal algorithm including a possible offline algorithm. Meanwhile, let $SC_i = \bigcup_{j=1}^{i-1} C_j$ and $SC_i^{opt} = \bigcup_{j=1}^{i-1} C_j^{opt}$.

Let $a(C_i)$ denote the aspects (of the targets in T) that are covered by C_i but not covered by any photos in SC_i. And the size of the set $a(C_i)$ is $|a(C_i)| = U_{SC_i \cup C_i}(T) - U_{SC_i}(T)$, i.e., the amount of total utility increased by adding C_i to the existing selection. Similarly, we can define $a(C_i^{opt})$ and its size in regarding to the selection by the optimal algorithm. From this definition, we have $U_{opt} = |a(\bigcup_{i=1}^{s} C_i^{opt})|$ and $U_{online} = |a(\bigcup_{i=1}^{s} C_i)|$.

For any period t_k, let $I_k = a(\bigcup_{i=1}^{k} C_i) \cap a(\bigcup_{i=1}^{k} C_i^{opt})$, i.e., aspects that are covered by both our selection of photos and the optimal selection. Then let $\alpha = 1 - (1 - 1/B_{min})^{B_{min}}$, where $B_{min} = \min\{B_1, \ldots, B_t\}$. We will show by induction that, for any period t_k,

$$\left| a\left(\bigcup_{i=1}^{k} C_i^{opt} \right) \setminus I_k \right| < \frac{1}{\alpha} \left| a\left(\bigcup_{i=1}^{k} C_i \right) \right|. \quad (2)$$

Clearly this is true for t_0. Suppose for t_{k-1}, where $2 \leq k \leq s$, (2) is correct. Then consider $a(\bigcup_{i=1}^{k} C_i^{opt}) = a(SC_k^{opt}) \cup a(C_k^{opt})$. Notice that $a(SC_k^{opt}) \setminus I_k \subseteq a(SC_k^{opt}) \setminus I_{k-1}$, as $I_{k-1} \subseteq I_k$. Thus, according to induction assumption, $|a(SC_k^{opt}) \setminus I_k| \leq \frac{1}{\alpha} |a(SC_k)|$.

Also notice that $a(C_k^{opt}) \setminus I_k \subseteq a(C_k^{opt}) \setminus A(SC_k)$. Because C_k is selected by using greedy algorithm maximizing the increase of total utility, from Theorem 1, we have $(1 - (1 - 1/B_k)^{B_k})|a(C_k^{opt}) \setminus a(SC_k)| \leq |a(C_k)|$. And as $\alpha \leq 1 - (1 - 1/B_k)^{B_k}$, thus, $|a(C_k^{opt}) \setminus I_k| \leq \frac{1}{\alpha} |a(C_k)|$.

Combining the results from above two paragraphs, we have proved (2). From (2), and the fact that $U_{opt} = |a(\bigcup_{i=1}^{s} C_i^{opt}) \setminus I_s| + |I_s \setminus a(\bigcup_{i=1}^{s} C_i^{opt})|$, and the later is smaller than $|I_s| \leq |a(\bigcup_{i=1}^{s} C_i)| = U_{online} < \frac{1}{\alpha} U_{online}$, we have $U_{opt} \leq \frac{2}{\alpha} U_{online}$, which concludes the proof. $\square$

4 ACHIEVING REQUIRED UTILITY WITH MIN-SELECTION

In this section we first study the Min-Selection problem, and then extend it to consider better coverage.

4.1 Min-Selection Problem

We consider a different scenario from the Max-Utility problem: the number of photos is minimized while the total utility is to be above a required level. In many practical applications such as virtual tours in map services, the major obstacle is to deal with the sheer amount of raw data (photos) obtained via crowdsourcing. Thus, it is desirable to remove redundancy and only keep the minimum selection of photos that satisfies the coverage requirement.

4.1.1 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.

4.1.2 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 4. *Suppose the targets' coverage requirements 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, $N_{approx} \leq O(\log mn) N_{opt}$.*

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 [10]. 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 subintervals (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 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 [10], it is easy to see the number of selected photos is upper bounded as shown in the theorem. □

An Example. Again, we use Fig. 2 to illustrate the above idea. Consider the problem settings in Fig. 2a and suppose the required coverage for T_1 is $[0, 2pi)$. The construction of the universe set and all the subsets are shown in Figs. 2b, 2c, 2d, and 2e. 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 five 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 five, 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 four new elements $\{e_3, e_4, e_5, e_6\}$. After that, S_1, which covers three 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 six 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.

Theorem 5. *For the Min-Selection problem, the worst case time complexity of the greedy algorithm is $O(n^3 m)$.*

Proof. The conversion process takes time $O(mn + mn \log n)$. For the selection process, we highlight two differences between the Max-Utility problem and the Min-Selection problem. First, in Max-Utility we select the photo that covers maximum weight of new elements, while in Min-Selection we select the photo that covers maximum number of new elements. Although weight and cardinality are different, they can be calculated by the same number of additions. Thus they are the same with regard to asymptotic running time. The second difference is that Max-Utility finishes with exactly B steps, while Min-Selection may finish in any steps between 1 and n. Considering the worst case, if Min-Selection finishes with n steps, the number of candidate photos it considers in those n steps is $n + (n - 1) + \cdots + 1 = O(n^2)$. Since handling each candidate photo requires time $O(mn)$, the time complexity of the entire algorithm is $O(mn + mn \log n + n^2 mn) = O(n^3 m)$. □

4.2 Min-Selection with k-Coverage

Some crowdsourced photos may contain inaccurate information. This may happen in an emergency situation where photos have to be taken in a very short time, leaving not much time for users to contemplate. Even if metadata can help understand how and where the photo was taken, some real photos may still miss our expectations. The inaccuracy can be caused by various reasons such as image blur due to vibration of the phone, occlusion, color abberation, or simply inaccurate metadata. To reduce the possibility that an important aspect of the object is missed, applications may require some degree of fault-tolerance, which can be achieved with k-coverage.

4.2.1 k-Coverage

In this problem, an aspect is required to be covered k ($k \geq 2$) times. Each target T_i has a coverage requirement $I_i = [a_i, b_i]$, $0 \leq a_i, b_i < 2\pi$, but now I_i needs to be covered k times by the selected photos. The problem is formally defined as follows.

Definition 5 (Min-Selection with k-Coverage). *Given 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: $I = \{I_1, \ldots, I_m\}$ and an integer $k \geq 2$, the problem asks for a minimum number of photos out of the n candidates, such that the coverage requirement for each target is covered at least k times.*

As the original Min-Selection problem can be converted to the set cover problem, the k-coverage problem can be converted in the same way to the set multicover problem. The set multicover problem, as its name suggests, differs from the set cover problem that each element e in the universe set U must appear at least k_e times in the selected subsets of U, where k_e is a positive integer for element e. Note that in our case, $k_e = k$ for any element e.

The greedy algorithm for the original Min-Selection problem can be naturally extended to the case of k-coverage. Specifically, an element is *alive* until it is covered k times. In each step, we select the photo that covers as many alive elements as possible, and then update the aliveness of the elements covered by this photo as appropriate. The selection proceeds until all elements are not alive or no more photos
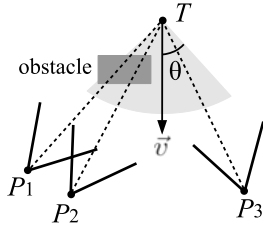
Fig. 3. The idea of 2-coverage+: two photos covering an aspect should have fairly different viewing directions.

can be selected (either photos are all selected or no more benefit can be achieved).

Suppose the overall coverage requirements can be satisfied if all photos are selected. Dobson [8] proved that the above algorithm achieves an approximation ratio of $O(\log mn)$, which means the number of photos selected by the greedy algorithm does not exceed $O(\log mn)$ times the minimum possible number. Note that this approximation ratio is the same as the ratio achieved by the Min-Selection algorithm (Theorem 4).

4.2.2 2-Coverage+

Although the k-coverage model looks good in theory, it may not work well in reality. Consider a scenario shown in Fig. 3. An aspect $\vec{v}$ of the target T is covered by three photos, P_1, P_2 and P_3. There is an obstacle between the target and the photos. By the k-coverage model, if $k = 2$, any combination of two photos among P_1, P_2 and P_3 satisfies the coverage requirement on aspect $\vec{v}$. However, choosing P_1 and P_2 will actually leave aspect $\vec{v}$ uncovered because the target is blocked in photo P_1 and P_2.

The key observation here is that the existence of obstacles affects our choice of photos. Ideally, the server should avoid P_1 and P_2 since they are blocked by an object and do not cover aspect $\vec{v}$. Although we will propose techniques in Section 5 to detect occlusion and filter out photos with occlusion, these techniques are not guaranteed to detect all obstacles. Hence, when selecting photos, the server should take into consideration the potential existence of obstacles. Instead of choosing k photos arbitrarily, the photos with fairly different views should be chosen, so that even one of them is blocked by an obstacle, the other still covers the aspect. Specifically, in the *2-coverage+* model, for each aspect covered by two photos, the angle between their viewing directions should be larger than a predefined threshold α.

Definition 6 (Min-Selection with 2-Coverage+). *Given 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: $I = \{I_1, \ldots, I_m\}$ and a threshold $\alpha \in [0, 2\theta]$, the problem asks for a minimum number of photos such that each aspect in the coverage requirements is covered by at least two photos, and the maximum angle between the viewing directions of those photos is greater than α.*

In this problem, the value of α determines the coverage resistance to obstacles. The larger α is, the farther the two photos are separated, and the less likely they are blocked by the same obstacle. However, α cannot be arbitrarily large. If a photo covers a given aspect $\vec{v}$, its viewing direction must be within a 2θ range, $[arg(\vec{v}) - \theta, arg(\vec{v}) + \theta]$ (light gray area

in Fig. 3). This means α has an upper bound 2θ. As α becomes closer to 2θ, it becomes harder to find two photos satisfying the requirement, because the two photos have to be positioned more precisely so as to both cover the aspect and have far enough viewing directions.

The 2-coverage+ problem is equivalent to the 2-coverage problem when $\alpha = 0$, so it is at least as hard as the k-coverage problem, and thus there is no polynomial time algorithm to find its optimal solution. Similar to the k-coverage problem, we can use a greedy algorithm to find a solution, but with some adjustment to the notion of aliveness. Specifically, the aliveness value of an element can be 2, 1 or 0. $alive = 2$ means that the element has never been covered; $alive = 1$ means that it has been covered at least once, and among the photos covering it, the maximum angle between their viewing directions is less than α; $alive = 0$ means that it has been covered at least twice, and among the photos covering it, the maximum angle between their viewing directions is no less than α.

In each step of the greedy algorithm, for each photo, we count the number of elements whose aliveness value would decrease if the photo were selected. Then we pick the photo with the largest count, and update the aliveness values accordingly. Here the decrease of an aliveness value means previously $alive = 2$ and after selecting the photo $alive = 1$, or previously $alive = 1$ and after selecting the photo $alive = 0$. This selection process continues until all $alive = 0$ or no more photo can be selected (either photos are all selected or no more benefit can be achieved). The performance of the greedy algorithm will be evaluated in Section 6.2.

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 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 [15]. 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

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], [15].
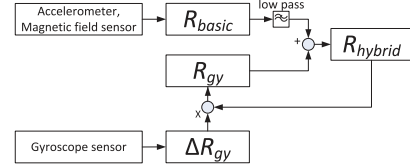

Fig. 4. Hybrid method combines the results of the basic method with gyroscope readings.

ΔR_{gy}). Given an initial rotation matrix, which can be obtained from 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 Fig. 4 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.,

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

where R_{basic} is the current reading and R_{basic}^{prev} 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 [26]) 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

$$\xi_1 = \alpha_1,$$
$$\xi_2 = \alpha_2 - \frac{< \alpha_2, \xi_1 >}{< \xi_1, \xi_1 >} \xi_1,$$
$$\xi_3 = \alpha_3 - \frac{< \alpha_3, \xi_1 >}{< \xi_1, \xi_1 >} \xi_1 - \frac{< \alpha_3, \xi_2 >}{< \xi_2, \xi_2 >} \xi_2.$$
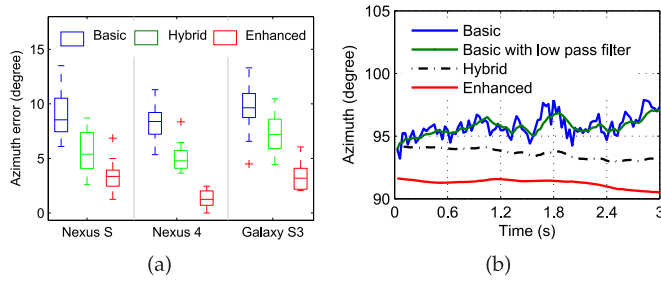
(a) (b)

Fig. 5. Orientation errors of the three proposed methods.

Second, the above ξ_i's are normalized by

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

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. Fig. 5a 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 percent) and third (75 percent) 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 percent compared to the basic method, and the enhanced method can further reduce the measurement error by more than 40 percent 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 Nexus 4.

To understand the effectiveness of these techniques clearly, we show the measurement results of these methods when the phone is turned to 90 degree, as illustrated in Fig. 5b. 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

TABLE 1
Average Error in Azimuth (Degree)

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

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 (Fig. 6a) by $D_{near} = \frac{v_o(H-f)}{H+v_o-2f}$, $D_{far} = \frac{v_o(H-f)}{H-v_o}$, where $H = \frac{f^2}{Ac} + f$ is the hyperfocal distance [17].

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. This filtering is done at the user side and the metadata of unqualified photos will not be sent to the server. As an example, consider the two photos in Fig. 6b. The dictionary is the interested target. In the left photo, the near and far limit of DOF is 85 and 105 cm respectively. In the right photo, the near and far
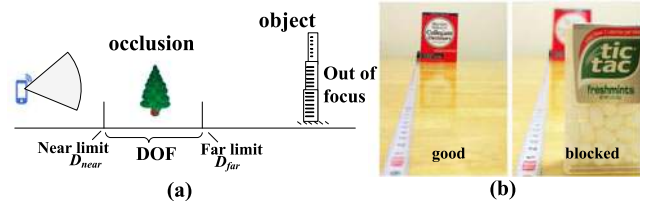

(a) (b)

Fig. 6. Using DOF to detect occlusion and out-of-focus.

(a) ours

(b) random1

blocked view

(c) random2

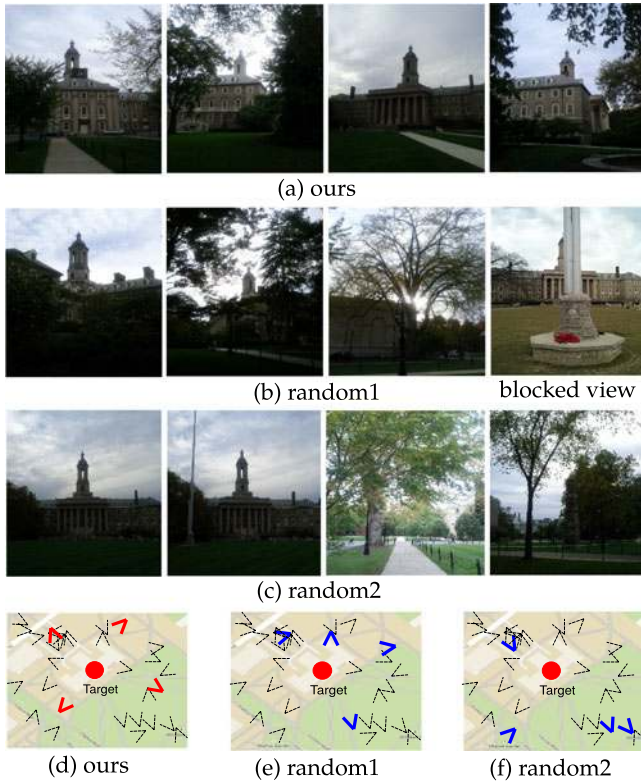(d) ours   (e) random1   (f) random2

Fig. 7. Demo results based on Max-Utility. (a) Photos selected by our algorithm. (b), (c) Photos selected randomly. (d), (e), (f) The locations and orientations of the photos are marked as "V" shapes on the map.

limit of DOF is 5 and 10 cm respectively. The distance between the camera and the dictionary is 100 cm. 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 method can detect most obstacles but not all. If the obstacle is very close to the target (their distances to the camera are almost the same), it is possible that the target is blocked but still in the DOF.

Discussions. Energy is an important issue for mobile devices, especially in post-disaster scenarios. Although various built-in sensors are used to collect metadata, they do not consume too much energy due to the following two reasons. First, metadata are collected only when users are taking photos, and the sensors are inactive most of time. Second, crowdsourcing relies on large number of users to obtain information. A single user does not need to take many photos and thus does not spend too much energy.

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 can only be analyzed by image processing. Thus, before photo selection, some efficient image processing techniques [9] may be applied at the user end to filter out low quality photos. 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 processing algorithms, but to serve as an important
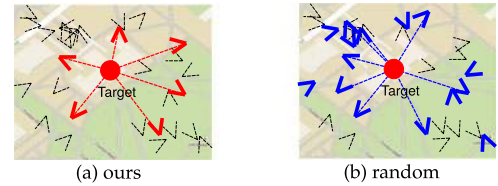

(a) ours   (b) random

Fig. 8. Demo results based on Min-Selection. To cover all aspects of the target, our algorithm uses six photos, and a random selection uses 15 photos.

complement to improve the utility of 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 testbed in Section 5 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 four photos. The photos selected by our algorithm are shown in Fig. 7a. The positions and orientations of the photos are shown in Fig. 7d, where the original 30 photos are marked as dotted "V" shapes, and the selected photos are marked by bold lines. As a comparison, we randomly choose four photos as shown in Fig. 7b, and randomly choose another four photos as shown in Fig. 7c. It can be seen that the four photos chosen by our algorithm cover the target from four different locations well separated from each other, with each one from a totally different angle. The bell tower is viewable from all four photos. In contrast, only two photos in the first random selection cover the target. The third photo faces away from the target, which is because random selection does not consider the orientation. In the fourth photo, the target is blocked by a *flagpole*. This photo is not considered by our algorithm because the target is out of focus according to the DOF information. For the second randomly selected four photos, two of them cover the target but they are very similar and contain redundant information. The other two do not cover the target.

We also demonstrate the effectiveness of our algorithm for the Min-Selection problem where the coverage requirement is from 0 to 360 degree, i.e., all aspects of the target. As shown in Fig. 8a, our algorithm selects six photos to meet the coverage requirement. The angle between any two adjacent viewing directions (dashed lines connecting the photos and the target) is less than 90 degree. Since the effective
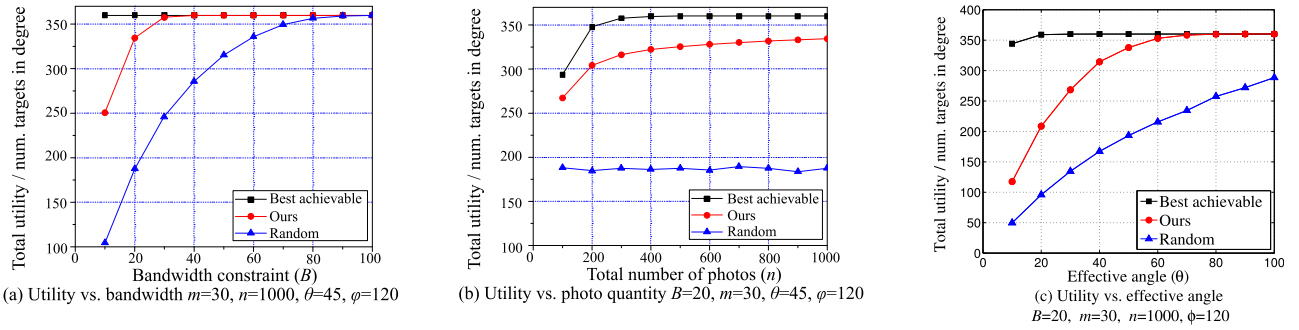
Fig. 9. Simulation results on Max-Utility problem.

angle is set to 45 degree, all aspects are covered. We also compare the performance with a random selection approach, which randomly selects photos one by one until the coverage requirement is achieved. As shown in Fig. 8b, the random approach has to select 15 photos to meet the same coverage requirement. The experiment based on the random selection approach is repeated 100 times, and on average 21 photos are selected to meet the same coverage requirement. This demonstrates that our algorithm can significantly reduce the number of photos selected to achieve the required coverage.

6.2 Simulation Results

In this section, we evaluate the photo selection algorithms through simulations. Targets are randomly distributed in a 100 m by 100 m square area. We generate photo metadata to represent real photos taken by users as follows. The photo locations are uniformly distributed in a 200 m by 200 m square, with the target area in the center. The orientations are randomly distributed from 0 to 2π. The field-of-view is set to 120 degree, and the range is set to 50 m as discussed in Section 5.1.

During the simulation, we compare our algorithms with a random selection algorithm that randomly selects photos at each step, until the bandwidth constraint is reached or the coverage requirement is satisfied. 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 Problem

In the first part, we evaluate the performance of our algorithm on addressing the Max-Utility problem. Intuitively, with more bandwidths, better coverage of the targets can be achieved. As shown in Fig. 9a, both our algorithm ("ours") and the random selection ("random") achieve more utility as the bandwidth B increases. The total utility achieved by all photos ("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.

Fig. 9b 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.

Fig. 9c plots the total achieved utility against the effective angle θ, with all other factors unchanged. As the effective angle increases, the coverage intervals of photos grow accordingly, so the total utility of both our algorithm and random selection increases as expected. Moreover, when the effective angle is between 10 to 40 degrees, the utility achieved by ours increase at a faster rate (the red circle line has larger slope than the blue triangle line). This is because our algorithm tries its best to avoid coverage overlap and thus benefits more from the coverage growth of each single photo. Note that as the effective angle increases, photos have more coverage but also have more coverage overlap. When the effective angle is more than 40 degrees, the utility obtained by our algorithm approaches the best achievable, so its increasing rate cannot be as high as before.

6.2.2 Results on Online Max-Utility Problem

In this part, the algorithm for the online Max-Utility problem is evaluated. We first observe how the total utility changes as the number of periods increases in Fig. 10a. Here, targets are randomly distributed as mentioned at the beginning of Section 6.2, 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 360 degree. It is actually above 350 degree 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 Fig. 10b. As the number of available photos increases, the selection algorithm has more choices. As a result, the total utility improves and approaches 360 degree in our algorithm. In comparison, the
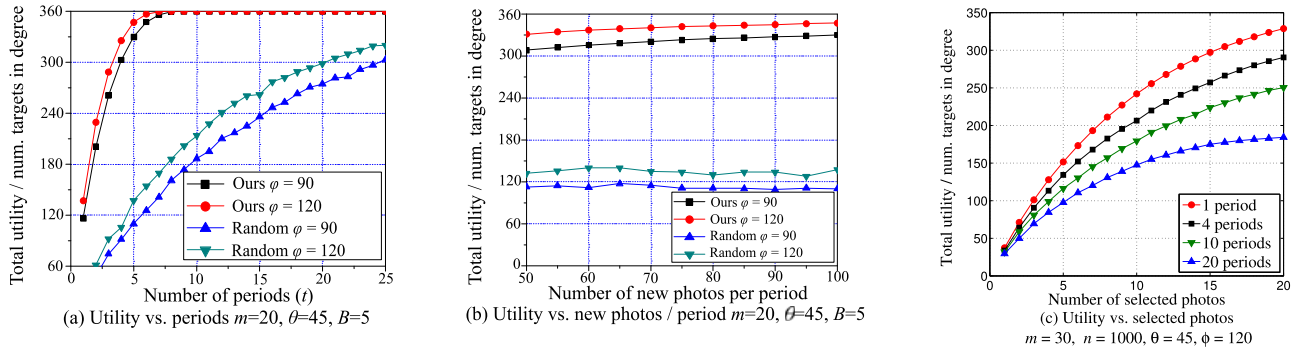
Fig. 10. Simulation results on online Max-Utility problem.

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.

We also compare the original version and the online version of the Max-Utility problem. Fig. 10c shows the utility after each photo is selected in a process of selecting 20 photos. The four lines are four greedy selections with different number of periods but the same number (1,000) of total available photos. The red circle line has 1 period, so it directly selects 20 photos from 1,000 candidates, which represents the algorithm for the original Max-Utility problem. The other three lines have more than one period, so they represent algorithms for the online problem. For example, the black square line has four periods, so in each period it selects $20/4 = 5$ photos from $1,000/4 = 250$ candidates. Our first observation is that for each line, the utility gain of selecting a photo is always positive and non-increasing. This complies with the features of greedy algorithms. Secondly, the performance becomes worse if there are more periods. This is because the online selection algorithm picks a fixed number of photos from each period, even though there are fewer good photos in one period and more in another. Finally, after selecting 20 photos, the utility achieved by the 20-period line is about half the utility achieved by the 1-period line. Note that 20 periods is the most periods possible for selecting 20 photos, so it represents the worst performance of the online algorithm. Therefore, this simulation result matches what we proved in Theorem 3, i.e., the performance bound of the online algorithm is half the performance bound of the original greedy algorithm.

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. Fig. 11a shows the effectiveness of our selection algorithm on reducing the redundancy. There are 20 targets, and all the aspects from 0 to 360 degree are required to be covered. As the total number of photos varies from 500 to 2,000, the number of related photos ("related") increases linearly. However, the number of photos selected by our algorithm ("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 1,000 and all other factors remain the same. As shown in Fig. 11b, 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.

In Fig. 11c we fix the number of targets as 30 and change the amount of aspects to be covered on each target. As expected, the number of photos needed to achieve the required coverage increases as more aspects are to be covered. Interestingly, the increasing rates of the lines rise on the left half but drop on the right half. The reason behind this can be explained as follows. Let us denote the amount of aspects to be covered on a target as x. The coverage
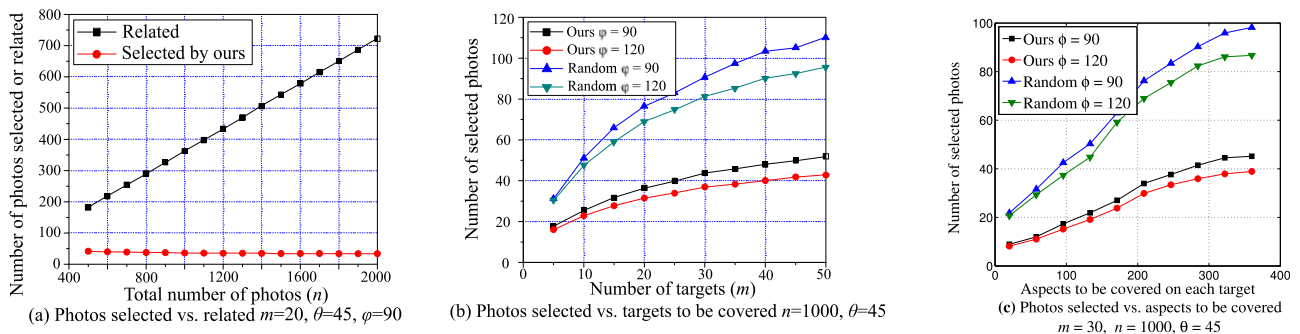

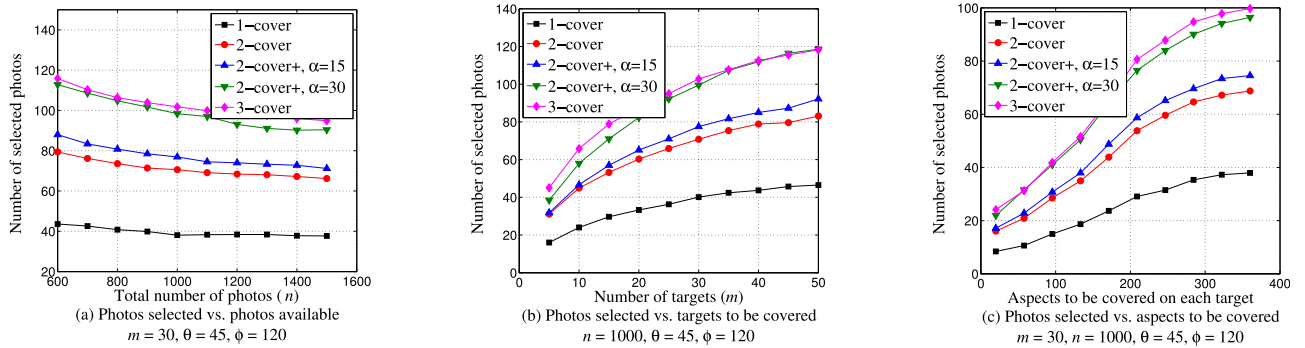
Fig. 11. Simulation results on Min-Selection problem.

Fig. 12. Simulation results on Min-Selection with k-Coverage.

requirement of target i is $I_i = [0, x]$. When x is small (e.g. 30 degree), the photos selected to cover $[0, x]$ also cover more than $[0, x]$, since by definition a photo always covers a 2θ range of aspects on a target (here $\theta = 45°$). Then if we increase x a little bit, not many new photos are needed to achieve the new coverage, since it is almost achieved by previous photos. As x keeps increasing, this advantage becomes weaker because the new coverage requirement $[0, x]$ is no longer within the 2θ range. Then many new photos have to be selected to complete the coverage. That is why the increasing rate rises for small x. On the other hand, when x is large, this advantage becomes stronger as x increases, because the remaining uncovered aspects are now within the 2θ range, and they are more likely to be covered by the selected photos. This explains the drop of the increasing rate for large x.

6.2.4 Results on Min-Selection with k-Coverage

In this part, we study the Min-Selection problem with k-coverage by comparing five different coverage settings. Three of them are based on the k-coverage model, with $k = 1, 2, 3$, respectively. The other two are based on the 2-coverage+ model, with $\alpha = 15$ and 30 degrees. Fig. 12a plots the number of selected photos as a function of the total number of available photos. The other parameters are fixed and shown in the figure. First, it is clear that all algorithms are able to take advantage of the increased number of photos, make better choices, and reduce the number of selected photos. When comparing 1-cover, 2-cover and 3-cover, the number of selected photos is almost proportional to the degree (k) of coverage. This shows that k-coverage does not fundamentally differ from single coverage. It just repeats single coverage for k times. We also compare 2-cover (i.e. 2-cover+ with $\alpha = 0$), 2-cover+ with $\alpha = 15$, and 2-cover+ with $\alpha = 30$. The $\alpha = 15$ line is pretty close to the $\alpha = 0$ line. However, $\alpha = 30$ requires at least 25 percent more photos than $\alpha = 15$. This suggests that the difficulty of achieving 2-coverage+ and the value of α are not linearly related. Once α becomes large, it requires much more photos to achieve the desired coverage.

In Fig. 12b, the number of targets varies from 5 to 50 while the number of available photos is fixed at 1,000. For a target, all the aspects from 0 to 360 degree should satisfy the required level of coverage, either k-coverage or 2-coverage+. We have similar observations as previous figures. The relationships between the lines are similar to those in Fig. 12a, and the increasing trend of the lines is similar to that in Fig. 11b.

In Fig. 12c, we fix the number of targets but change the amount of aspects that should satisfy the required coverage. The relationships between the lines are similar to those in Figs. 12a and 12b, and the increasing trend of the lines is similar to that in Fig. 11c.

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], [13], [18], [19], [21]. 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 [12], 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 [11]. 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 [24] and later extended in [23], [25]. 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 various optimization problems on the tradeoffs between resource constraints and photo coverage.

Another interesting work is PhotoNet [22], 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 from 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 [7] 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 [6]. Some other interesting works have been done for image retrieval/search on smartphones [27]. 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 [27]. These techniques are challenged by the content diversity and the resource constraints.

8   CONCLUSION AND FUTURE WORK

We proposed a resource-aware framework, called Smart-Photo, 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. Four optimization problems regarding the tradeoffs between photo coverage and resource constraints, namely Max-Utility, online Max-Utility, Min-Selection and Min-Selection with k-coverage, have been studied. The approximation bounds of the proposed algorithms 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 and mitigate the occlusion and out-of-focus issues. Results based on real implementations and extensive simulations validated the effectiveness of the proposed algorithms.

As future work, we will consider assigning more weights to targets and aspects that are more important. For example, after a disaster, a building with more potential survivors (e.g. schools, malls) is more important than others, and its possible entrance is more important than other aspects of the building. Hence, photos covering that building and its entrance should have higher priority to be selected. We will also address other research issues such as extending the current photo model to 3D and identifying other photo selection problems based on the requirements of crowdsourcing applications.

REFERENCES

[1] SensorManager—Android Developer [Online]. Available: http://developer.android.com/reference/android/hardware/Sensor-Manager.html, 2014.
[2] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury. "A survey on wireless multimedia sensor networks," *Comput. Netw.*, vol. 51, no. 4, pp. 921–960, 2007.
[3] N. Alon and J. H. Spencer. *We the Media: Grassroots Journalism by the People, for the People*. Sebastopol, CA, USA: O'Reilly Media, 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 Conf. Comput. Vis. Pattern Recog.*, 2005, pp. 454–461.
[5] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[6] D. Crandall and N. Snavely, "Modeling people and places with internet photo collections," *Commun. ACM*, vol. 55, no. 6, pp. 52–60, Jun. 2012.
[7] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: Ideas, influences, and trends of the new age," *ACM Comput. Surv.*, vol. 40, no. 2, pp. 5:1–5:60, May 2008.
[8] G. Dobson. "Worst-case analysis of greedy heuristics for integer programming with nonnegative data," *Math. Operations Res.*, vol. 7, no. 4, pp. 515–531, 1982.
[9] S. Gabarda and G. Cristobal, "Blind image quality assessment through anisotropy," *J. Opt. Soc. America A*, vol. 24, no. 12, pp. B42–B51, 2007.
[10] D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*. Pacific Grove, CA, USA: PWS Publishing Company, 1996.
[11] E. Hörster and R. Lienhart, "On the optimal placement of multiple visual sensors," in *Proc. ACM Int. Workshop Video Surveillance Sensor Netw.*, 2006, pp. 111–120.
[12] M. P. Johnson and A. Bar-Noy, "Pan and scan: Configuring cameras for coverage," in *Proc. IEEE INFOCOM*, 2011, pp. 1071–1079.
[13] 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 Conf. Inf. Process. Sensor Netw.*, 2010, pp. 58–69.
[14] 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. Inf. Syst. Crisis Response Manag. Conf.*, 2008, pp. 140–149.
[15] R. Meier, *Professional Android 2 Application Development*, 2nd ed. New York, NY, USA: Wiley, 2010.
[16] 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.
[17] S. F. Ray, *Applied Photographic Optics*. 3 ed. Focal Press, Oxford, U.K., 2002.
[18] B. Rinner and W. Wolf, "A bright future for distributed smart cameras," *Proc. IEEE*, vol. 96, no. 10, pp. 1562–1564, Oct. 2008.
[19] 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 11th Int. Conf. Inf. Process. Senor Netw.*, 2012, pp. 103–104.
[20] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d," in *Proc. ACM SIGGRAPH*, 2006, pp. 835–846.
[21] S. Soro and W. Heinzelman, "A survey of visual sensor networks," *Adv. Multimedia*, vol. 2009, pp. 1–22, 2009.
[22] 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 Syst. Symp.*, 2011, pp. 317–326.
[23] Y. Wang and G. Cao, "Barrier coverage in camera sensor networks," in *Proc. 12th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2011, pp. 12:1–12:10.
[24] Y. Wang and G. Cao, "On full-view coverage in camera sensor networks," in *Proc. IEEE INFOCOM*, 2011, pp. 1781–1789.
[25] Y. Wang and G. Cao, "Achieving full-view coverage in camera sensor networks," *ACM Trans. Sensor Netw.*, vol. 10, no. 1, pp. 3:1–3:31, 2013.
[26] G. Williams, *Linear Algebra With Applications*. Boston, MA, USA: Jones & Bartlett Learning, 2012.
[27] T. Yan, V. Kumar, and D. Ganesan, "CrowdSearch: Exploiting crowds for accurate real-time image search on mobile phones," in *Proc. 8th Int. Conf. Mobile Syst., Appl. Services*, 2010, pp. 77–90.
[28] C. Zhu, K. Li, Q. Lv, L. Shang, and R. P. Dick, "iScope: Personalized Multi-modality image search for mobile devices," in *Proc. 7th Int. Conf. Mobile Syst., Appl. Services*, 2009, pp. 277–290.
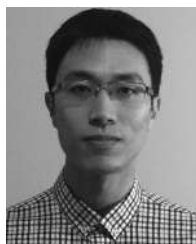
Yibo Wu received the BE degree in information engineering from Shanghai Jiao Tong University in 2008. He is currently working toward the PhD degree in the Department of Computer Science and Engineering, the Pennsylvania State University. His research interests include mobile crowdsourcing, vehicular networks, and opportunistic networks. He is a student member of the IEEE.
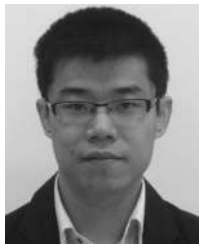
Yi Wang received the BS degree in mathematical sciences from Tsinghua University, and the PhD degree in computer science and engineering from the Pennsylvania State University. His research interests include wireless sensor networks, mobile computing, and distributed systems. He is currently working on software-defined networking for Google Inc.
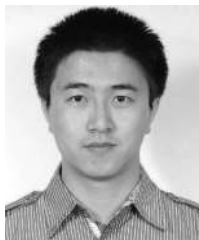
Wenjie Hu received the BS degree in computer science from Tongji University in 2007 and the MS degree in computer science from Tsinghua University in 2010. He is currently working toward the PhD degree in the Department of Computer Science and Engineering, Pennsylvania State University. His research interests include energy management for smartphones, mobile cloud, and mobile video. He is a student member of the IEEE.

Guohong Cao received the BS degree from Xian Jiaotong University, China. He received the MS and PhD degrees in computer science from the Ohio State University in 1997 and 1999, respectively. Since then, he has been with the Department of Computer Science and Engineering, Pennsylvania State University, where he is currently a professor. His research interests include wireless networks and mobile computing. He has published more than 150 papers in the areas of wireless sensor networks, wireless network security, cache management, data access and dissemination, and distributed fault tolerant computing. He has served on the editorial boards of the *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Wireless Communications*, and has served on the organizing and technical program committees of many conferences. He received the US National Science Foundation (NSF) CAREER award in 2001. He is a fellow of the IEEE.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.