Real Time Object Scanning Using a Mobile Phone and Cloud-based Visual Search Engine

Yu Zhong
University of Rochester
Rochester, NY 14627 USA
zyu@cs.rochester.edu

Pierre J. Garrigues
IQ Engines, Inc.
2501 9th St, Berkeley, CA 94710 USA
pierre@iqengines.com

Jeffrey P. Bigham
University of Rochester
Rochester, NY 14627 USA
jbigham@cs.rochester.edu

ABSTRACT
Computer vision and human-powered services can provide blind people access to visual information in the world around them, but their efficacy is dependent on high-quality photo inputs. Blind people often have difficulty capturing the information necessary for these applications to work because they cannot see what they are taking a picture of. In this paper, we present *Scan Search*, a mobile application that offers a new way for blind people to take high-quality photos to support recognition tasks. To support real-time scanning of objects, we developed a key frame extraction algorithm that automatically retrieves high-quality frames from continuous camera video stream of mobile phones. Those key frames are streamed to a cloud-based recognition engine that identifies the most significant object inside the picture. This way, blind users can scan for objects of interest and hear potential results in real time. We also present a study exploring the tradeoffs in how many photos are sent, and conduct a user study with 8 blind participants that compares *Scan Search* with a standard photo-snapping interface. Our results show that *Scan Search* allows users to capture objects of interest more efficiently and is preferred by users to the standard interface.

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation]: User Interfaces; K.4.2 [Social Issues]: Assistive technologies for persons with disabilities

General Terms
Human Factors, Design, Experimentation.

Keywords
Real time object scanning; Accessibility; Blind user; Mobile.

1. INTRODUCTION
Many everyday tasks require object identification, yet many objects are indistinguishable without visual information. For example, many food products have the same size and packaging, and so the only way to tell them apart is by looking at the labels. A quick and accurate visual scan by a sighted person can help blind people with the minor problems and finish more daily tasks independently. Blind people often have workarounds that can render individual problems into mere nuisance, but, collectively, small problems can lead to decreased independence [7].

Many applications from both research and industry have been designed to help blind people recognize objects around them, either by applying computer vision [10, 11, 16] or human computation [5, 7, 26]. Most of these applications have a photo-snapping interface – a button in the interface acting as a camera shutter which triggers photo taking and subsequently object recognition events. When the input photo has good quality and abundant information, these applications can work well to provide the user a good recognition result. But the photo-taking interface on current mobile phones is not friendly to blind users, as very few smart phones have acoustic feedback in the photo-taking interface. This fact often leads to difficulty for blind people to correctly frame the camera and take a picture with the target object at a good position. Even when the camera is perfectly framed and the object distance is good that most area of the object facing the camera is inside the frame and in focus, there may not be enough information inside the photo to identify an object, for example, the camera is facing the wrong side of a food product and there are only advertisements or nutrition facts in the photo.

Difficulties in blind photography can make assistive services less beneficial to blind people than they could be. Workers powering systems like VizWiz [7] can suggest camera positioning guidance to help the blind user to take a better photo for the next run, but it can take several runs (and several minutes) to recognize an object, resulting a much longer time to complete individual object identification task than desired [4]. Services powered by computer vision usually lack of this feature in large part due to the difficulty in constructing automatic technologies that can do this well.

In this paper we introduce *Scan Search*, a project aiming at enabling real time object scanning for blind people to help them quickly and accurately identify everyday objects. Blind people use *Scan Search* on their existing camera phones. The application automatically extracts good quality frames from the camera feed and sends those frames to the IQ Engines' web service for identification. IQ Engines is a cloud-based visual search engine with a large public dataset containing several million images of packaged goods, print media, brand logos, etc. [6] Unlike most current assistive object identification applications, *Scan Search* does not have a photo taking button. Blind users open the application, put the object they want identified in front of the camera and start scanning from different angles and distances for real-time identification. *Scan Search* intelligently decides which frames to process to conserve computational resources, as opposed to other applications that fully process each frame. It leverages a cloud-based visual search engine to address general

ASSETS'13, October 21–23, 2013, Bellevue, Washington, USA.

scenarios, as opposed to only OCR [18], currency recognition [10] or bar code scanning [11] offered by other applications.

Since *Scan Search* works in a real-time scanning fashion, it can save time for blind users who may otherwise need to figure out the right position of camera in order to take a single high-quality photo and then wait for feedback. According to our user study, *Scan Search* allows blind user to identify a food product with a success rate of 91.7% as opposed to a photo-snapping interface with a success rate of only 62.5% with the same image recognition mechanism.

The *Scan Search* application is efficient on computational and networking resources on the iPhone, as the visual search engine of IQ Engines [6] doesn't need high-resolution input images. In our experiments the required bandwidth was below 50 KB/s. Therefore it can be deployed on a large range of smart phones as long as they have a camera with reasonable resolution. Given the prevalence of smartphones and their better accessibility over feature phones [2], *Scan Search* can potentially benefit a large population with visual impairment.

The Contributions of this paper include: *(i)* an efficient algorithm that automatically extracts good quality, information-rich frames from continuous camera video stream; *(ii)* a mobile application, *Scan Search*, that enables blind users to scan everyday objects for real-time identification result; *(iii)* and, a usability study that shows *Scan Search* is preferred by blind users over standard photo-snapping interfaces for its effectiveness and efficiency in taking good photos and identifying objects.

2. RELATED WORK

2.1 Accessibility on Mobile Phones

For the most of the past few decades, mainstream cellphones have been inaccessible to blind people. Blind people had to rely on separate screen reading software like Mobile Speak Pocket (MSP) [13] to have best access to the phones. Such software has been limited due to its high price (several hundreds of dollars in addition to the price of a smart phone).

In the past few years, many smart phone manufacturers have started to develop their own screen reading software that allows blind people to use their phones and either include the software into the operating system or ship it for free. For example, Apple's iPhone (available on 3GS and later models) now has VoiceOver[1], Android-powered (starting from 4.0) smartphones now support "Eye-Free"[2] multi-touch interactions, and Nokia has also released a free screen reader in their online application store since October, 2011[3]. Touchscreen devices like iPhone were once thought to be inaccessible to blind users, but well-designed, multi-touch interfaces leverage the spatial layout of the screen and can even be preferred by blind people [8]. The iPhone has proven particularly popular among blind users, which is why we developed the first version of *Scan Search* application on the iOS.

With the accessibility of smart phone platforms improving, not only existing applications with graphic interfaces such as web browsers are becoming more accessible to blind people, there are also more and more applications designed for blind users now. Including but not limited to GPS navigation and way-finding applications, OCR readers, currency/color recognizers, and also many object identification applications.

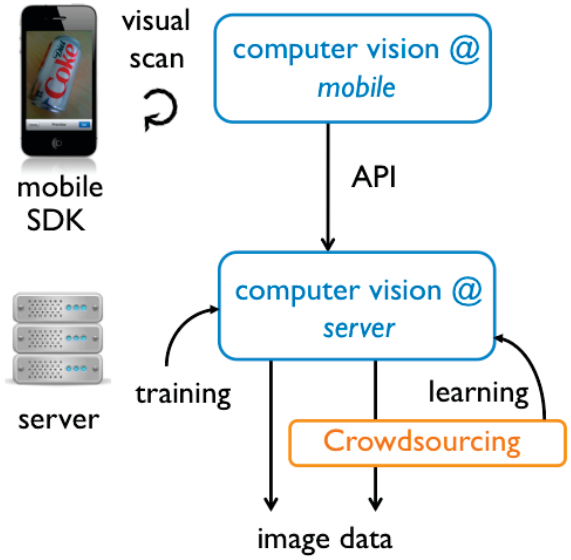

Figure 1. 3-step IQ Engines visual search flow, to achieve real time user experience, *Scan Search* only uses the computer vision server now but it is scalable.

2.2 Object Identification for Blind People

Object identification is an important and frequent task in people's daily lives, and often acts as a critical first step of completing more complicated tasks. Although many objects can be identified without visual information, for example, with tactual features, many objects are only differentiable by visual characteristics, such as two cans of the same size and tactual feels and different labels. Although blind people often have work-arounds for those small problems or can seek help from sighted persons, collectively those small problems can lead to decreased independence and less efficiency, sometimes even big frustrations. Prior study shows that identification is the most common visual challenge for which blind people seek help from access technology (41% of four categories of questions) [4].

Access technology helps blind people with object identification through two kinds of approaches, either computer vision powered automatic services or human-powered services. Computer vision powered services generally have faster response time and better availability but are limited in scope and error-prone [7], while human-powered services are more flexible and economical. All the services require a certain level of input photo quality to provide satisfactory results.

2.2.1 Computer Vision Powered Services

On mobile phones, there are many accessible object identification applications, which employ different computer vision algorithms to identify objects inside the input photos captured by blind users in real time.

Some of the applications run all object identification tasks on the mobile phones with a local dataset. They are faster and don't require network connection, but only work in a specific scope because computational and storage resources on mobile phones are not comparable to those on desktop or web servers. For instance, the LookTel Money Reader [10] can identify currency

[1] http://www.apple.com/accessibility/voiceover.

[2] https://code.google.com/p/eyes-free/.

[3] http://conversations.nokia.com/2011/10/27/nokia-rolls-out-new-screen-reader/.

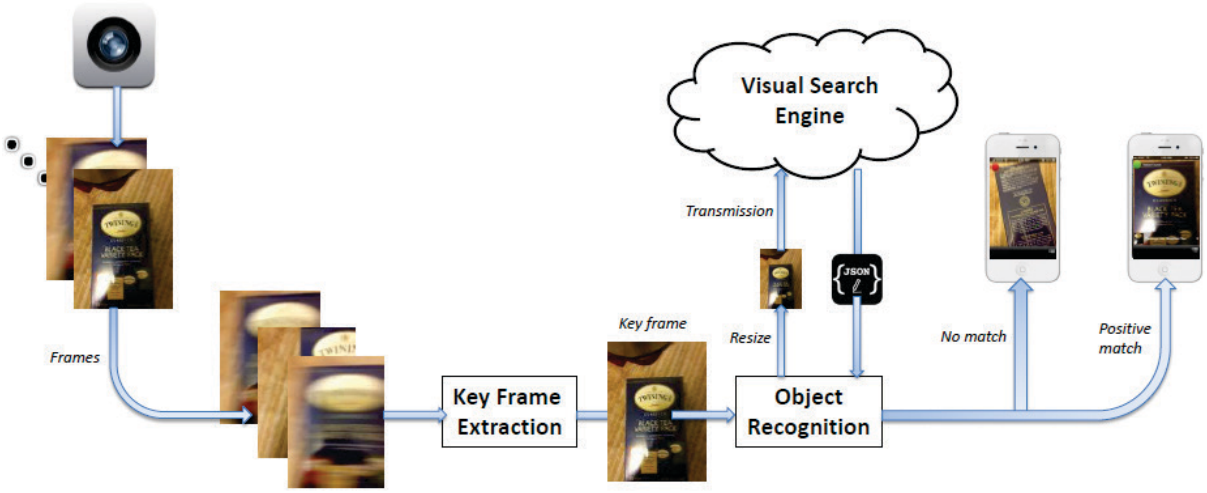

Figure 2. The Scan Search application reads frames from the camera buffer, extracts key frames from the stream, sends key frames to the cloud-based visual search engines and presents recognition results in the interface.

denominations instantly (less than 0.5 second) but only work with currency of five countries.

Another kind of object identification applications combines local image search with web visual search engines. Since web servers can store much bigger datasets and run heavy computational tasks much faster than mobile phones, they can work with a wide range of objects and still respond in real time. For example, Omoby [16] passes locally unrecognized photos to their own visual search engines, which have millions of trained images for a further match. Given a good network connection, they can return remote search results in less than 2 seconds.

Although often helpful in blind people's daily lives, computer vision services often fail or required a long time for blind users to identify an object because of low recognition rates. Because blind people have no access to visual information, it can be difficult for them to frame a good photo with important optical features inside, and those applications can't provide any camera framing guidance. Most of the time, a blind user will need to try multiple times before s/he gets the identification result if s/he succeeds.

2.2.2 Human-Powered Services

Since human-powered services like VizWiz [7] use real human to do visual scan on the input photos, they can work with a wider range of objects and have lower requirement of image quality. But the main advantage of those services is that they provide camera positioning guidance to blind users in order to answer the blind users' questions, in this context, identifying objects.

Although human computation cannot act as fast as computer algorithms in that it requires time to recruit online workers and wait for them to complete the tasks, with well-designed infrastructure they can still provide near real-time responses. For instance, VizWiz can answer a question in less than 30 seconds if a steady pool of workers is maintained [7].

In practice, there are several factors affecting performance of human-powered services, such as lower availability of online workers during some time periods in a day and malicious workers. And even with camera guidance, it may take several runs for a blind user to take a photo with necessary information inside. Aggregating time of sending photos and feedbacks back and forth can be longer than desired and sometimes frustrating.

2.3 Blind Photography

A number of published articles [1, 3, 20, 21, 24] have shown that blind people take photos for multiple reasons, including sending to remote sighted people for feedback and for general object recognition [3]. Despite this, most current camera interfaces are only marginally accessible, which leads to poor-quality photos that are blurry, tilted or improperly framed. For instance, more than 17% of the questions sent to VizWiz could not be answered because the photo quality was too poor [4].

There have been many efforts [3, 14, 15, 19, 22, 25] to assist blind people better using the inaccessible cameras and also some technology [9, 28] potentially can be used to facilitate this task. For instance, the system developed by M. Vázquez, et al. [15] to help visually impaired users aim a camera can effectively assist blind users to frame a better photo by applying optical region of interest algorithm to suggest better framing for blind users. Despite this breadth of work, blind people still take lower quality photos than do sighted people [15], indicating the need for more research in this area. Even high-quality photos taken by blind people can be insufficient for many uses – for instance, a high-quality photo of the back of a box may not show its label. *Scan Search* helps to solve these problems by giving blind users direct feedback on what is shown in the camera.

2.4 The IQ Engines Visual Search Engine

In our *Scan Search* system, we used the IQ Engines cloud-based image recognition service, which is built on top of both a public dataset containing millions of images and a private dataset created by each user. Searching works best for flat objects and packaged goods including but not limited to beer, wine labels, logos, print ads, books, CD/DVDs, posters and artwork.

The IQ Engine service also has options to process local visual search queries on iOS and Android phones before remote search engine queries are triggered and to pass unrecognized images to human-powered service for a guaranteed response. In order to maintain a controlled experiment condition to evaluate our real-time scanning interface, those two options are not enabled in our *Scan Search* application. With local search and human-powered service disabled, a single visual query takes less than 1 second to finish given a good network connection and since it only accepts

photos with resolution ranging from 200x200 to 800x800 and it is also network efficient as mentioned before.

IQ Engines handles a visual search query by first matching the input photo (a frame chosen by *Scan Search*) against a local image dataset on the mobile phone (if enabled) (Figure 1). It then sends the photo to a cloud-based server for remote matching against images in both private and public datasets. If there is still no matching result it will then be forward to human-powered service that takes less than 10 seconds to respond (if enabled) or return a "No Match" result.

3. SCAN SEARCH

Scan Search is an iPhone application designed for use with the VoiceOver available on the iPhone 3GS and later models. The interaction to identify objects with real-time scanning is simple and intuitive, and fully accessible for blind users.

3.1 System Description

As shown in Figure 2, the *Scan Search* system has two modules that work together to facilitate real time object identification while scanning objects. The first one is a key frame extraction module that will be described in detail in the next section. It runs on continuous camera video stream to retrieve high quality frames. The quality of a frame is defined as the stableness of the camera at the time the frame is recorded and the richness of visual characteristic features (indicated by the green points on the interface in Figure 2). The second one is an object recognition module, which sends key frames to the visual search engine for recognition results and subsequently presents the returned results with both visual and audio feedbacks. All matching results are then stored in an accessible history table in the order of picture taking time for blind users to further review the objects identified and differentiate pictures and corresponded results.

When users start *Scan Search*, it starts to read frames from the buffer of the iPhone camera and process each frame to determine whether it is good enough to be considered a key frame. If a frame passes the validation process it is immediately resized and encoded as a 640x480 JPG file and sent to the IQ Engines visual search engine which is described in detail in the previous section. Visual and audio hints are available at the time of key frame sending events. Then the application continues with another incoming frame without waiting for the asynchronous recognition query to finish. Once a recognition result arrives at the phone, the application alerts the user with both visual and audio feedbacks. If the result is a positive match with one of the objects in the dataset then the match is stored in the history table for further review.

3.2 Key Frame Extraction

3.2.1 Design of Algorithm

Most mainstream phones now have a camera that can capture frames at a rate of 30 fps. It is impractical to send all frames in the camera buffer to a cloud-based visual search engine because of network bandwidth limitation. Even if possible, it is inefficient because most buffered frames are blurry or improperly framed images captured by the camera when the user is adjusting phone position. Sending the whole buffer would result in a huge waste of both network and computational resources. In addition, for single object identification, the visual search engine actually needs only one good image with abundant optical characteristics. To efficiently and accurately retrieve such good images from continuous video, we designed a lightweight optical algorithm that runs on phones to enable real time key frame extraction.

The heuristic of extracting a high quality key frame is the same as taking a photo with a handheld camera. We want the camera to be steady and well-focused at the actual photo-taking time; we also want as much visual information as possible to be included in the frame; and we don't want to take too many pictures for the same scene which is inefficient and a waste of further processing resources. And since the computational capacity is limited on mobile platforms, we cannot perform heavy calculations. Bearing those principles in mind, we leverage the lightweight Lucas–Kanade optical flow method [12] to efficiently track feature points. The amount of feature points in each frame is used as an indicator of optical information richness of that correspondent frame and the estimated movements between a specific frame and its previous one serves as an indicator of stableness of the camera.

With the Lucas-Kanade algorithm benchmarking stableness of the camera and optical information richness of frames, as shown in Figure 3, our algorithm runs continuously on video stream and break the stream into segments, each segment represents a scene whose optical information is significantly different from its neighbors'. In each segment, at most one good frame is extracted as the key frame in each scene in order to ensure efficiency.

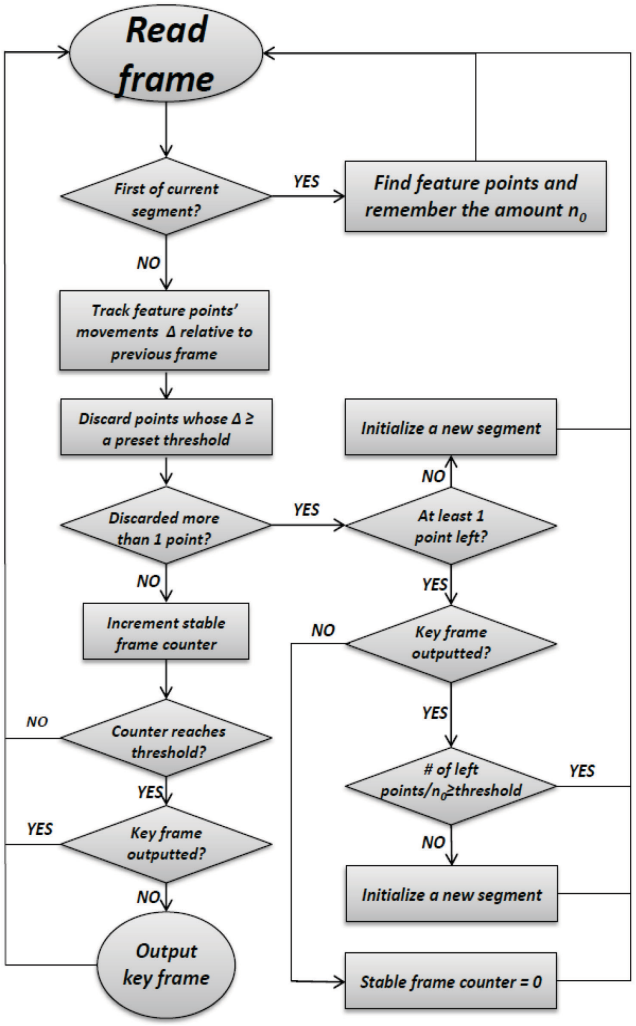

Figure 3. Flow chart of the key frame extraction algorithm.

3.2.2 Implementation

As a popular optical flow-tracking algorithm, the Lucas-Kanade method is included in the OpenCV [17] library, which is available on several platforms including iOS (in 2.4 and later versions). It has been widely used since firstly proposed in 1981 [12]. Nowadays most mainstream smart phones have enough computational power to apply the algorithm in real time, for instance the 800MHz A5 processor on the iPhone 4S model can process 240x320 grayscale frames at a rate of 15-20 fps.

Our implementation of the key frame extraction algorithm is in C and can be compiled with OpenCV library on many platforms, including iOS, Android and Linux, which means it can be easily ported and embedded into different applications. In this paper we evaluate and optimize the performance of our algorithm on iOS in order to better serve blind users of *Scan Search*. However, we have also successfully tested the algorithm on Android and we believe the evaluation and optimization discussed in the next section are also applicable to the other platforms.

3.3 Algorithm Evaluation and Optimization

The performance of our key frame extraction algorithm depends both on the device hardware and on the parameter settings, which is why we want to evaluate and optimize our algorithm implementation before putting it into the *Scan Search* application. As illustrated in Figure 3, there are three parameters/thresholds in our key frame extraction algorithm. They are:

a. Movement threshold: The threshold of a point's movement between two consecutive frames, any point moved a distance smaller than this threshold will be considered a stationary point and kept in subsequent computation, otherwise discarded. In order to make the design compatible with different camera resolutions, movement threshold is defined as a percentage of either width or height of the frame, whichever is smaller. For instance, if the frame size is 640x480, a movement threshold of 1% means a stationary point can move at most (480*1%)-1 = 3 pixels.

b. Initialization threshold: The threshold of the percentage of left stationary points in a specific frame compared to the amount of points in the first frame in the current segment of the video stream. Since in the same segment points can only be discarded because of significant movements, this percentage will drop from 100% (the first frame) gradually. When the percentage becomes lower than the threshold, we consider a new scene is being captured and thus switch to a fresh segment.

c. Stableness threshold: The threshold of the number of stable frames needed before a frame is considered to be the key frame of this segment. A specific frame is considered stable if and only if in this frame no points are discarded because of significant movement comparing to the previous frame, when a frame is categorized as stable the stable frame counter will increment by 1. Once the counter reaches this threshold, the current stable frame will be outputted as the key frame of this segment.

In order to obtain an ideal performance of our algorithm, we analyzed how each parameter affects the performance. Unlike the other two parameters, the stableness threshold largely depends on the processor speed. Generally, the faster the device running the algorithm, the larger the best-fit stableness threshold is. For instance, most phone cameras can capture a clear and well-focused image after being held stably for 0.5-1 second, if the a frame takes 0.1 seconds to process, a stable frame counter of 5-10 indicates a good time to extract the key frame, while a processing speed of 20 fps corresponds to a stableness threshold of approximately 10-20. Thus this parameter should be set according to processor specifications, after testing with different iPhone models, we have found the best empirical stableness parameters for the iPhone 4, 4GS and 5 as listed in Table 1.

Table 1. Empirical stableness threshold settings

Model	Processor [27]	Threshold
iPhone 4	A4 (clock speed unrevealed)	10
iPhone 4S	A5, 800 MHz	15
iPhone 5	A6, 1.3 GHz	20

In order to make the algorithm portable to other mobile platforms without empirical threshold settings, we have also implemented a dynamic stableness threshold adjusting mechanism, which automatically sets the threshold according to the following formula formed with the logic described above.

$$T_s = \frac{0.8}{\left(\sum_{i=0}^{n} t_i\right)/n}$$

Where T_s is the stableness threshold, t_i is the time used to process i-th frame and n is the total number of frames processed so far.

The movement and initialization thresholds are more complicated, because they are far less dependent on hardware performance and more directly affect the quality and quantity of key frames extracted from the same video stream.

When adjusting the movement threshold we face a tradeoff between quality of key frames and number of redundant key frames with similar information. As shown in Figure 2, lower movement threshold means fewer points can be considered stationary thus it is harder for a frame to pass the stableness test, leading to better quality of key frames. However, it also means that in a certain stream segment, stationary points will drop below the initialization threshold faster, leading to more segments and subsequently more key frames extracted from the same stream.

For initialization threshold, there is also a tradeoff which is between the thoroughness of visual information scanning and amount of redundant key frames. The logic in Figure 2 shows lower initialization threshold leads to smaller difference in optical characteristics between two consecutive segments, leading to more thorough information retrieval and also more key frames extracted from the same stream.

To better understand the tradeoffs related to the two parameters, we did two experiments to evaluate the performance of our algorithm with different parameters and optimized the algorithm.

3.3.1 Experiment Designs

The two experiments differed in the dataset used by the visual search engine. The first one was conducted with a controlled private dataset with only trained images of objects used in the experiment, while the second was conducted with the very large public dataset to better evaluate expected performance in practice.

For the first experiment, the subject objects were three cans of food with the same size and tactual feel but different labels. We first took 13 pictures of each object from different angles and distances and trained them in a private dataset for image matching.

We then recorded an approximately 20-second video of each object, which simulated a scanning of the object by moving the camera around the object and zooming in/out from different angles. A script was used to extract key frames from the video with different parameter settings and send the images extracted to the visual search engines to match against entries in the private dataset for object identification results.

For the second experiment, the subject object was a single canned food. We used the same script to process a 30-second scanning video of the object to extract key frames and match them with trained images in the large-scale public dataset.

To measure the performance of our algorithm with different parameters, each visual query result was recorded. Movement threshold ranged from 1% to 10% with an interval of 1% and initialization threshold was chosen from 1%, 5%, 10 and 20%.

3.3.2 Results and Discussion

For each parameter setting, we counted the number of key frames extracted and then calculated the percentage of frames inside which objects were successfully identified, for simplicity the percentage is referred as identification rate later. Higher percentage of successfully identified frames is regarded as an indicator of better quality of the key frames extracted in the experiment while more extracted key frames is a signal of both more thorough information retrieval and more redundancy. The relationship between the parameters and those two performance measurements were then analyzed as below.

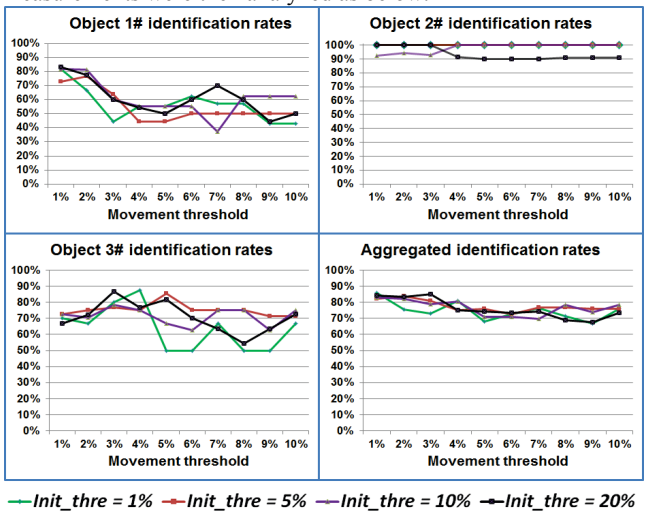

Figure 4. Identification rates in controlled dataset with different movement and initialization thresholds.

As shown in Figure 4, the identification rate varies largely for different objects in the first experiment. This is because some objects have more distinguishable visual features, such as logos with distinct edges. However, overall linear regression on the aggregated results shows that movement threshold significantly predicted identification rate (b = -1.11, $t(37)$ = -4.90, p < .001), on the other hand, initialization threshold didn't, together they explained a significant proportion of variance in identification rates (R^2 = .39, $F(2, 37)$ = 12.03, p <. 001). The finding of negative significant coefficient of movement threshold conforms to our theory of the effects of each parameter which is described in the previous section. Furthermore, because the private dataset is small and controlled, most key frames were correctly identified by the engine (average identification rate is 76.23%, σ = 5.07%).

When looking at the amount of key frames extracted with each parameter setting, we found our theory of a negative correlation between numbers of key frames and movement threshold correct, as well as a positive correlation with initialization threshold. Linear regression on the aggregated results was used to verify our

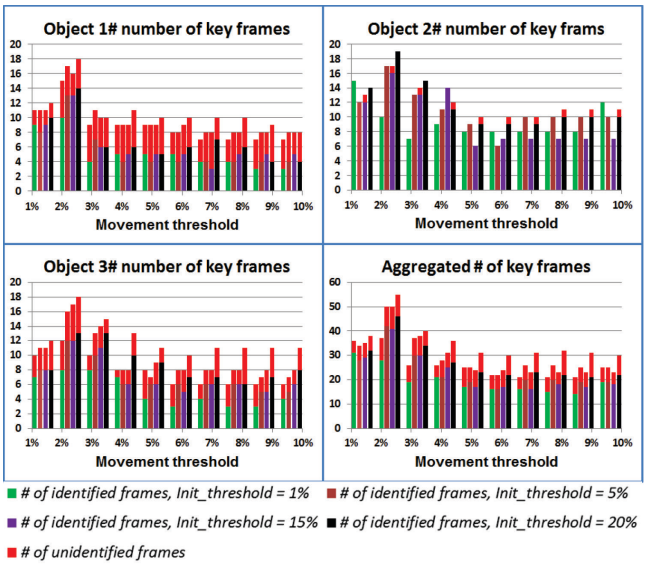

Figure 5. Number of key frames extracted with different movement and initialization thresholds, length of a bar is the total number of key frames, red means unidentified frames, otherwise identified frames.

findings. Specifically, as shown in Figure 5, higher movement threshold significantly predicted fewer key frames (b = -204.85, $t(37)$ = -7.15, p < .001), while higher initialization threshold significantly predicted more key frames (b = 40.59, $t(37)$ = 3.51, p < .01), together they explained a significant proportion of variance in number of key frames (R^2 = .63, $F(2, 37)$ = 31.705, p <. 001). Again, for individual object the result varies because of differences in visual features and video taking positions but the same trend can still be seen in each individual result. On average, 30.45 frames were extracted from three videos together (σ = 8.25), and the total length of the three videos are 60 seconds, resulting to a frame extracting speed of approximately 1 frame every 2 seconds, given the file size of a 640x480 jpg is at most 100 KB, the maximum network uploading bandwidth needed is 50 KB/s which can be supported by most wireless connections, e.g. EDGE.

From Figure 5, we also noticed that although movement threshold is negatively correlated with number of key frames, the peaks of frame number almost always appear at 2% movement threshold. Our explanation for the bump between 1% and 2% movement threshold is that in the pre-processing step, each frame is converted to a 160x240 grayscale image to alleviate burden on the tracking algorithm. Thus 1% movement threshold means a stationary point can move at most (160*1%) - 1 = 0 pixel, which essentially means it cannot move at all. But camera shake is a common problem of handheld photographs especially in low light situations, e.g., inside buildings [23], therefore 1% is too strict that in most segments of the video stream a key frame cannot be extracted, which counterbalanced the effects of more segments.

The second experiment conducted with the large-scale public dataset gave us more insight into the expected performance of our algorithm in practice. As shown in Figure 6, we can see the same

correlation between movement/initialization threshold and identification rate as well as number of key frames.

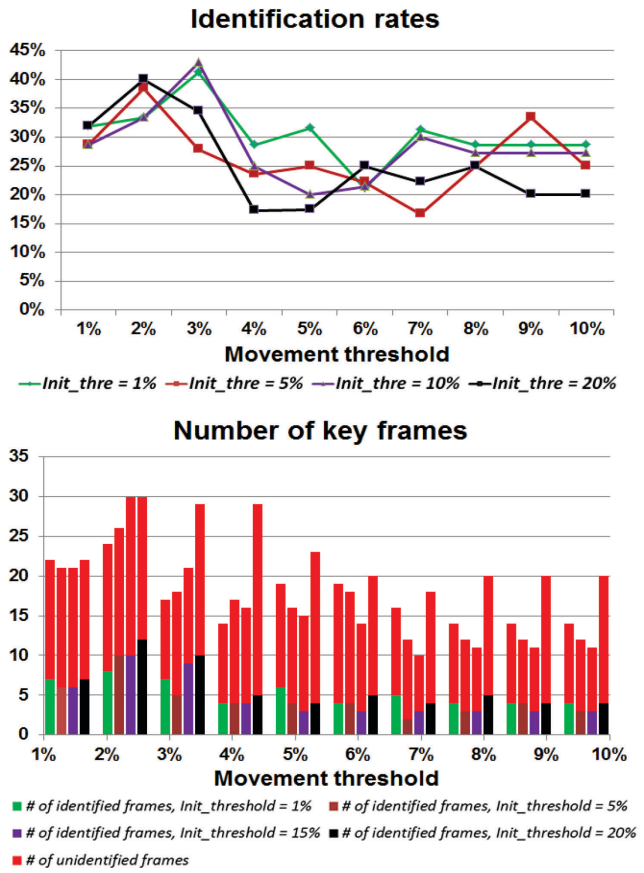

Figure 6. Results of the experiment with public dataset.

Data analysis corroborated our observation, as a linear regression showed that movement threshold significantly predicted identification rates (b = -0.94, $t(37)$ = -2.96, $p < .01$) and number of key frames (b = -126.36, $t(37)$ = -6.62, $p < .001$), on the other hand, initialization threshold significantly predicted only number of key frames (b = 32.72, $t(37)$ = 4.24, $p < .01$). Together they explained a significant proportion of variance in both identification rates (R^2 = .24, $F(2, 37)$ = 5.77, p <. 01) and number of key frames(R^2 = .63, $F(2, 37)$ = 30.89, p <. 001).

We confirmed this on public dataset our theory of parameter settings is still valid. We also successfully identified the subject object in the video with each and every parameter setting. But we are more interested in the differences between private to public datasets. The most obvious change is that identification rate dropped from >70% to <45% (μ = 27.66%, σ = 6.35%) because the public dataset does not have as many trained images of the subject objects as private dataset. We also observed that some queries returned ambiguous or false positive results, e.g., "Canned food" while the ground truth is "Progresso Vegetable Classics".

Bearing the reality issues with using the large-scale dataset and lack of access to visual information of blind users in mind, we decide to choose a parameter setting that allows some redundancy in change of more thorough scanning in order to enable blind users to identify objects more easily with the ability to filter false positive results. But we do not want to sacrifice key frame quality too much, therefore we defined movement threshold as 2% and initialization threshold as 10% in *Scan Search* application.

4. USER STUDY

To explore the effectiveness of *Scan Search* in assisting blind users to identify objects in their everyday lives and to compare the scanning interface with standard photo-snapping interface, we conducted a study with 8 blind people (6 male and 2 female). The age of our participants ranged from 21 to 52 (μ = 30.88). The study was conducted remotely from the blind participants' homes using their own iPhones. The phones used were iPhone 4 (1), iPhone 4S (4) and iPhone 5 (3). Participants were paid $5 each, consented online, and not otherwise affiliated with this project.

As a control condition, we developed another object identification application without the key frame extraction algorithm. In the control application, users have to push a button to take pictures like the way they would use Omoby [16] or Taptapsee [26]. Before the study, the participants were briefed on how both applications worked, and used each application to identify an object shown in an image opened in their web browser. During the study, they were asked to find and then identify three differently shaped everyday objects: *(i)* a bottle of water/light drink/beer, *(ii)* a can of food, and *(iii)* a frozen dinner or a carton of milk. All of the objects used in the trials were first confirmed to exist in the public dataset so that failed trials would be due to poor quality of pictures sent to the visual search engine and not because of a lack of appropriate trained images. The participants were encouraged to take photos from different distances, angles and camera orientations and did not receive instructions from us.

Participants used both *Scan Search* and the control application to identify objects in each of the 3 categories (6 trials per participant). To alleviate short-term memory of object positioning, the order of tasks and applications were randomized. Each object identification task was limited to 5 minutes. Tasks that exceeded the time limit were considered failed and discontinued. All task completion times were recorded. A completion time was defined as the interval between the time a user starts trying to identify an object and the time s/he receives a satisfactory result (defined as either being accurate or containing enough information for her/him to use another service to identify the object). For example, an accurate description of the product or a bar code number.

5. RESULTS AND DISCUSSION

All participants completed the experiments with network connections ranging from slow EDGE to high speed Wi-Fi and on average each image matching on the cloud took less than 1 second. 11 of 48 total trials failed, and most (9) of the failed trials occurred in the control condition (standard photo-snapping interface). One of the failed cases with control condition was found when reviewing trial images that a false positive was accepted by the participant, others are all due to time out. Thus it's easier for blind users to identify objects with *Scan Search* than other photo-snapping applications. The success rate of scanning interface (91.67%) was significantly higher than that of photo-snapping interface (62.5%), $t(46)$ = 6.29, p = .016. The average time taken per identification task with *Scan Search* was 73.2s as compared to 126.4s with the control, which is 42% less. The difference was not detectably significant, in part because of large variation in completion time. We found that some trials succeeded quickly because of a lucky starting position of the camera that captured a distinct area of the target object with less than three photos, for instance, the UPC label. In these cases, both applications worked just as well because no search was required. Thus, we did further analysis on only those trials that took more

than 5s to complete. All of these trials produced more than 3 pictures with the last one correctly identified the object, suggesting a visual search which is challenging for blind people was actually performed. For those trials *Scan Search* needed 24.43s each in average while photo-snapping interface took 75.57s, which means a blind user could successfully locate the visual information needed to identify objects faster with *Scan Search.* The difference was significant ($t(12) = 5.99$, $p = .031$).

The quality of photos taken by the participants were also better when using our scanning interface because the key frames extracted were guaranteed by the application to be non-blurry and well-focused. It is also one reason that blind users succeeded in more trials with the scanning interface even though the average number of photos taken in each trial were almost the same with scanning interface (11.4) and photo-snapping interface (14.1), $t(46) = 0.41$, $p = .523$. It suggests that our algorithm is no more likely to overwhelm users with too many pictures. Another observation worth noting is that blind users have largely different levels of camera using skill. Therefore we believe audible guided exploration of visual scene can be very helpful, especially for those not familiar with photography.

At the end of the study, participants were asked to take a short survey about their preferences between scanning and standard photo-snapping interfaces and given general feedback on the two applications. 7 participants said they "strongly prefer" and 1 said "prefer" scanning interface over photo-snapping interface, and 6 participants would like to continue using *Scan Search* in their daily lives because of "fast and good results" while the other 2 said they "possibly", one of the participants was "surprised that it can recognize objects with random scanning".

6. CONCLUSION AND FUTURE WORK

In this paper, we have contributed an algorithm, which can extract high-quality and visually-rich frames from continuous camera video, experiments that evaluate and optimize the algorithm, an accessible real-time scanning application with which blind people can identify everyday objects around them and usability studies that show our approach works better than the current standard. Most camera interfaces lack of accessibility for blind people even though many accessible mobile applications are picture-based. *Scan Search* improves blind users' experience in multiple areas.

More designs and studies are presently being conducted on combing the key frame extraction algorithm with other technology, such as crowdsourcing and real time camera framing assistance, to create or improve more accessible applications and address the image dataset scalability. Interface improvements of *Scan Search* are also ongoing to enable end-users to train datasets on both phone and cloud in order to customize visual searches. For the next stage, we plan to continue our research on both application and algorithm levels. Specifically, we'd like to refine *Scan Search* based on feedback and then distribute it on the public market to better understand its potential real-world benefits. On the other hand, we'll improve our algorithm to take more optical features into account and compare it with other key frame extraction methods, for instance, naive sampling.

7. ACKNOWLEDGMENTS

This work was supported by National Science Foundation Awards #IIS-1116051 and #IIS-1149709, and by National Institutes of Health Award R44EY019790.

8. REFERENCES

[1] Blind with Camera. http://www.blindwithcamera.org/.
[2] Burton, D. (2011). The Current State of Cell Phone Accessibility.
[3] C. Jayant, H. Ji, S. White, and J. Bigham. Supporting blind photography. ASSETS 2011, 2011.
[4] E. Brady, M. Morris, Y. Zhong, S. White, and J. Bigham. Visual challenges in the everyday lives of blind people. CHI 2013, 2013.
[5] E. Brady, Y. Zhong, M. Morris, J. Bigham. Investigating the appropriateness of social network question asking as a resource for blind users. CSCW 2013, 1225-1236, 2013.
[6] IQ Engines. http://www.iqengines.com.
[7] J. Bigham, C. Jayant, H. Ji, G. Little, A. Miller, R. C. Miller, R. Miller, A. Tatrowicz, B. White, S. White, and T. Yeh. Vizwiz: Nearly real-time answers to visual questions. UIST 2010, 2010.
[8] Kane, S. K., J. Bigham, and J. Wobbrock. Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques. ASSETS 2008, 73–80, 2008.
[9] Ko, J. and Kim, C. Low cost blur image detection and estimation for mobile devices. Proc. ICACT 2009, IEEE Press (2009), 1605–1610.
[10] LookTel Money Reader, http://www.looktel.com/moneyreader.
[11] LookTel Recognizer, http://www.looktel.com/recognizer.
[12] Lucas, B. and Kanade, T. 1981. An iterative image registration technique with an application to stereo vision. In Proceedings of the International Joint Conference on Artificial Intelligence, pp. 674– 679.
[13] Mobile speak screen readers. Code Factory, 2008. http://www.codefactory.es/en/products.asp?id=16.
[14] M. Vázquez and A. Steinfeld. An assisted photography method for street scenes. Proceedings of the 2011 IEEE Workshop on Applications of Computer Vision, 2011.
[15] M. Vázquez, and A. Steinfeld. Helping visually impaired users properly aim a camera. ASSETS 2012, 95-102, 2012.
[16] Omoby, https://www.iqengines.com/omoby/.
[17] OpenCV, http://opencv.org.
[18] Pleco, http://www.pleco.com/.
[19] P. Sanketi, and J. Coughlan. Anti-blur feedback for visually impaired users of smartphone cameras. ASSETS 2010, 2010.
[20] Seeing Beyond Sight. http://www.seeingbeyondsight.org/home/.
[21] Seeing with Photography Collective. http://www.seeingwithphotography.com/swpc_home.html.
[22] S. Harada, D. Sato, D. W. Adams, S. Kurniawan, H. Takagi, C. Asakawa. Accessible Photo Album: Enhancing the Photo Sharing Experience for People with Visual Impairment. CHI 2013, 2127-2136, 2013.
[23] S. Harmeling, M. Hirsch, and B. Schölkopf. Space-variant single-image blind deconvolution for removing camera shake. ICCV 2011, 2011.
[24] Sight Unseen, UCR/California Museum of Photography, http://www.cmp.ucr.edu/exhibitions/sightunseen.
[25] S. White, H. Ji, and J. Bigham. EasySnap: real-time audio feedback for blind photography. UIST 2010, 409-410, 2010.
[26] TapTapSee, http://www.taptapseeapp.com/.
[27] Wikipedia, http://en.wikipedia.org.
[28] X. Hou, and L. Zhang. Saliency detection: a spectral residual approach. CVPR 2007, 2007.