RegionSpeak: Quick Comprehensive Spatial Descriptions of Complex Images for Blind Users

Yu Zhong[1], **Walter S. Lasecki**[1], **Erin Brady**[1], **Jeffrey P. Bigham**[1,2]

Computer Science, ROC HCI[1]
University of Rochester
Rochester, NY 14627 USA
{zyu, wlasecki, brady}@cs.rochester.edu

HCI and LT Institutes[2]
Carnegie Mellon University
Pittsburgh, PA 15213 USA
jbigham@cmu.edu

ABSTRACT

Blind people often seek answers to their visual questions from remote sources, however, the commonly adopted single-image, single-response model does not always guarantee enough bandwidth between users and sources. This is especially true when questions concern large sets of information, or spatial layout, *e.g.*, where is there to sit in this area, what tools are on this work bench, or what do the buttons on this machine do? Our RegionSpeak system addresses this problem by providing an accessible way for blind users to *(i)* combine visual information across multiple photographs via image stitching, *(ii)* quickly collect labels from the crowd for all relevant objects contained within the resulting large visual area in parallel, and *(iii)* then interactively explore the spatial layout of the objects that were labeled. The regions and descriptions are displayed on an accessible touchscreen interface, which allow blind users to interactively explore their spatial layout. We demonstrate that workers from Amazon Mechanical Turk are able to quickly and accurately identify relevant regions, and that asking them to describe only one region at a time results in more comprehensive descriptions of complex images. RegionSpeak can be used to explore the spatial layout of the regions identified. It also demonstrates broad potential for helping blind users to answer difficult spatial layout questions.

ACM Classification Keywords

H.5.2 [Information interfaces and presentation]: User Interfaces – *Input devices and strategies*.; K.4.2 [Computers and Society]: Social Issues – *Assistive technologies for persons with disabilities*.

Author Keywords

Visual questions; crowdsourcing; stitching; accessibility.

INTRODUCTION

Crowdsourcing answers to visual questions can help blind and low vision users better access the world around them [3, 4]. However, most current approaches only permit users to take a single photograph [3], which can lead to several issues: *(i)* users often have difficulties in framing the correct information for their questions, *(ii)* workers do not give consistent levels of details in responses —especially those which are broad or open ended —which means that users get unpredictable answers from the system (Figure 1), and *(iii)* because question complexity cannot be automatically determined a priori, it is hard to compensate crowd workers appropriately.

Figure 1. An example of a real image from a VizWiz user, submitted with the question "What am I looking at that is in front of me?" Questions like this burden crowd workers with lengthy answers, and are hard to answer using text-based spacial descriptors alone. Our RegionSpeak system allows workers to mark regions in an image and answer sub-sets of the questions. This makes the task more approachable for workers, and provides more and better information to end users, who can browse answers using a touchscreen interface that allows them to get a sense of the spacial orientation the corresponds to each label in the image.

Systems such as VizWiz [3] struggle with these aspects because of their single-image, single-response model. Chorus:View [16] overcomes many of these problems by engaging users in *continuous* interactions with the crowd via voice and video to help reduce the overhead associated with multi-turn interaction. However, video-based approaches are expensive and difficult to scale. They can also be cumbersome for end users who must actively wait (e.g., holding the camera steady) while the crowd determines a response.

Our goal is to account for the large set of tasks that fall somewhere between the ideal case for single images in VizWiz, and the continuously-engaged interaction of Chorus:View. We analyze examples of 1,000 single-image visual questions

CHI 2015, April 18 - 23 2015, Seoul, Republic of Korea

http://dx.doi.org/10.1145/2702123.2702437

is read to the user via VoiceOver [1] (on iOS) or TalkBack [2] (on Android). RegionSpeak is designed to run on and extend existing question answering platforms such as VizWiz, which has answered 70,000+ questions since May 2011.

Chorus:View

VizWiz effectively answers self-contained visual questions, but it struggles in settings where more context is needed than can be provided in a single image, or where the information needed is hard to locate when taking an image. Prior work has shown that these types of questions make up 18% of the questions asked to VizWiz [6], and that does not account for all of the questions that users avoid asking because they have learned the system does not address them well.

Chorus:View [16] is a system that addressed many of these problems by engaging users and the crowd in a continuous interaction by streaming video to workers and allowing chat-style responses to be provided at any point workers felt was appropriate. This more conversational interaction can answer many questions users commonly have in a fraction of the time of the original VizWiz system, and as a result, was strongly preferred by most users in lab studies.

Chorus:View provides users with continuous, real-time feedback by keeping a group of workers around synchronously with the user. It is best for tasks that require constant attention, i.e., navigation. For tasks that benefit from asynchronous assistance, i.e., the identification and description questions frequently asked by blind users [6], Chorus:View is too expensive and heavyweight. Since Chorus:View keeps a group of workers active during the answering process, the price is prohibitive (order of $1 per question [16]) for deployment. RegionSpeak avoids these issues by collecting information upfront (instead of interactively), and presenting a user interface in which a user can explore multiple answers spatially. By combining image stitching and labeling, RegionSpeak provides a tradeoff between the benefits of short and cheap single-question services like VizWiz, and more thorough but expensive services like Chorus:View. They are useful in different settings.

Computer-aided photography

Using computer vision techniques to assist blind and low-vision camera users has been widely studied and evaluated. Prior work shows with appropriate interface and feedback, quality of photos taken by the targeted users can be substantially improved (over 50% in [8]). Those photography applications are also easy to learn and use, therefore almost always preferred by study participants. [8, 22]. In addition, Kane, *et. al.* also explored the possibilities of using computer vision to assist screen reading and gained success[10].

Panoramic image stitching algorithms has been extensively studies by computer vision researchers, various algorithms [7, 19, 21] excel in different performance aspects such as speed, seamlessness and order invariance. Built-in camera applications on iOS and Android both support panorama photoshooting. However, most commercially applied algorithms are not fully automatic, requiring human input or restrictions on the image sequence in order to establish matching images. For example, when using the built-in panorama interface on iOS, the user has to move the camera towards one direction and keep the vertical center line of view port steady. When using the equivalent interface on Android, the user has to move camera center to several fixed points on the screen. Panorama apps leverage these input constraints to yield high quality and wide angle photographs, but it is difficult for blind users to satisfy these constraints because available feedback has previously been visual. The algorithm used in RegionSpeak stitching interface is designed to be nearly-automatic, robust order of input images, rotation and illumination, therefore puts minimal input restrictions on blind users.

CAPTURING ADDITIONAL VISUAL INFORMATION

We begin by addressing the information capture and image framing problem faced by many blind users. Prior work has explored how to help guide blind users to frame objects appropriately in images [4, 8]. However, these approaches are based on using computer vision to guide users to a specified object, which does not take into account the specific information that needs to be included to answer a given question. For example, they may help a user to correctly frame a container in an image (Figure 3), but cannot help a user know if the usage instructions are visible to workers.

Stitching Images to Increase Context

To benefit from the additional information workers are able to see when answering a question using video, without adding the additional delay, we introduce *image stitching*.

Image stitching can increase information conveyed by blind users as objects captured in multiple images are sent to workers who have a better chance to locate essential information to answer the question in a quick, light-weight dialog turn.

Interface Details

RegionSpeak does not require aesthetically appealing photographs, but does need to have an accessible interface. Therefore, we employed a nearly-automatic and the least demanding stitching algorithm in OpenCV[3] which is similar to [7]. Our experiments show that even when results are not seamless (Figure 2), our algorithm is able to create satisfactory panorama without loss of visual information.

Combining the automatic stitching algorithm and Zhong *et. al*'s key frame extraction algorithm [24], we created an panorama interface for RegionSpeak which has no restriction that needs visual inspection. Users of RegionSpeak can move the camera in any direction, and the key frame extraction algorithm will detect substantial changes in view port and alert users to hold their position to capture a new image. RegionSpeak then takes a photo automatically when the view port is stabilized and gives users an audio cue to move on. Users can stop exploring after three photos are taken. A pilot study showed that some users would continue to take photos if not given a limit, so we added a limit of six photos to automate

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

[2] http://goo.gl/zbdZsD

[3] http://opencv.org

Figure 3. A sequence of questions asked to VizWiz. The user has trouble framing the entire bottle in a single image while still being close enough to allow workers to read the instruction text. Worse, because each image is sent to a potentially different worker, different parts of the text that are captured are out of context for the worker trying to answer the question for each image. RegionSpeak allows users to capture more information in a single image by adding *image stitching* to VizWiz. In this instance, the first two images can be stitched as well as the other three.

the process. Six photos was derived as a limit of latency tolerance from observation of the pilot users experience.

All the photos are then sent to our server which stitches them together. While the stitching algorithm could be performed locally on mobile devices, speed became an issue. The stitching algorithm can stitch four 640x480 images in 4-5 seconds on a desktop with 1.3 GHz i5 CPU, but the same task takes approximately 20 seconds to finish on an iPhone 4S (which clocks at 800MHz). Remote stitching keeps users from having to wait for the process to complete on their devices, keeping the total time of the question-asking interaction low.

Evaluation

To evaluate the usfeulness of stitching, we needed to see if blind people would be able to learn and use this new interface, as it requires a certain level of camera skill. To explore the effectiveness and usability of the stitching interface, we conducted a study with 10 blind people (9 male, 1 female) to compare it with single picture approaches. The study was conducted remotely from the blind participants' home using their own iPhones. The phones used were iPhone 4S (2), iPhone 5 (3), and iPhone 5S (5). Participants were recruited via mailing lists from previous studies and Twitter announcements, they were each paid $10, consented online, and are not otherwise affiliated with this project.

In order to fairly compare stitching with the single photo interface, we developed an experimental application which replaces the single photo interface of VizWiz with the stitching interface to compare both. All of our participants had used VizWiz before the study and reported experiences of having trouble capturing enough information with VizWiz to get an answer. Before the study, participants were briefed on how to use the stitching interface with a $\sim$5 minute introduction session. Then we allowed the participant to familiarize himself with the stitching interface by asking a random question about a stitched image of immediately surrounding environment, followed by a controlled experiment to compare the stitching interface with conventional single photo interface. At the end of each session, a questionnaire was issued to collect preference data. Each session lasted 30-50 minutes depending on individual camera skills.

The experiment was a 2×3 within-subjects factorial design. The first factor was the capturing interface: single photo versus stitching. The second factor was three tasks:

- **Printed document:** Asking for the author of a book.
- **Screen:** Reading user name off a faked log in window.
- **Description**: Getting description of on-table objects.

We chose these three tasks because reading information and getting descriptions are documented visual challenges blind people face everyday [6]. Additionally, these objects tend to have wide flat surfaces which users often find difficult to capture. In order to limit interactions to a reasonable length, we set a maximum task time of 10 minutes, as was done in [16]. Trial duration was recorded with second timestamps. If the correct information was not found by the time limit, the task was considered incomplete. For these tasks, the user expects an answer immediately, *e.g.*, the user wouldn't want to wait spend more than 10 minutes to find out a user name (classified as “urgent“ in [6]).

In order to evaluate performance, task completion time was measured as mean trial duration, calculated as the elapsed time from the opening of application to reception of a satisfiable answer (including time taken to capture photos). If a trial timed out, the task completion time is undefined and excluded from analysis. Iterations were defined as the number of times the participant had to repeat the process of question asking before the task was completed or timed out.

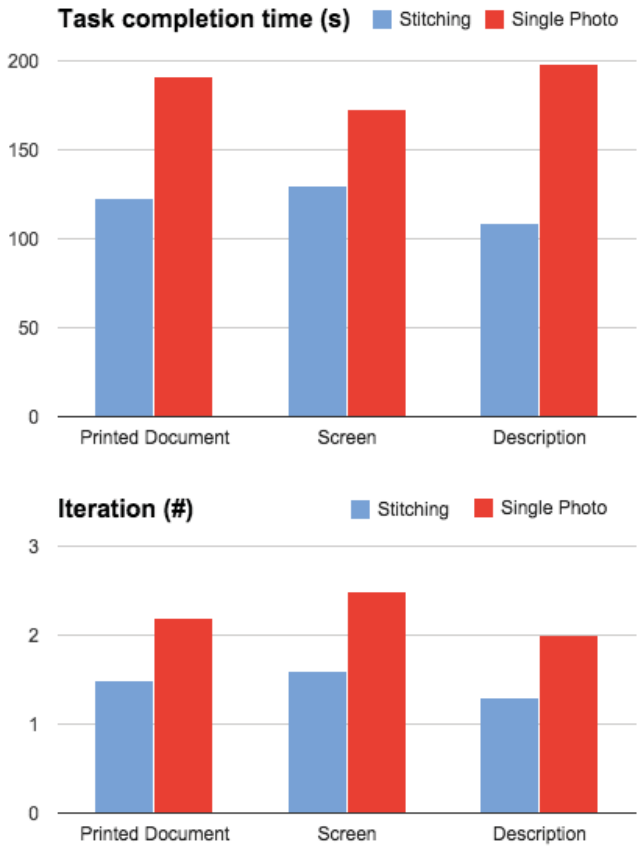

Figure 4. Mean target completion times and iteration numbers show the stitching interface outperformed the single photo interface.

Task completion time data were logarithmically transformed to correct for violations of normality as common practice. We analyzed the time data using ANOVA, with interface and task type modeled as fixed effects.

Results

All participants completed the tasks using the stitching interface within the 10 minute limit, with an average time of 121.1 seconds, while VizWiz failed 1 of the user name reading tasks, with an average time of 187.4 seconds. The difference was significant, $F_{1,51} = 8.424$, $p < .01$. The average number of Q&A iterations it took for stitching (mean = 1.48) to yield right answers was also significantly lower than single photo interface (mean = 2.24), $F_{1,52} = 7.04$, $p = .01$. The results confirmed that with the stitching interface blind users were likely to capture more visual information in each dialog turn and save time (by 35.4%) and iterations (by at least 1 on average) in subsequent interactions, detailed means of time and iteration numbers are shown in Figure 4. There were no interaction effects found on either task completion time or number of iterations required.

When the stitching algorithm fails to produce a panorama (because of insufficient overlap between input images), the application sends all captured images and shows them together to web workers. The success rate of our stitching algorithm during experiments was 83.3%, suggesting most blind people's camera skill is enough to operate the stitching interface.

User Feedback

Participants were asked to answer a survey about the stitching app and to provide their feedback after the experiments. All of them expressed that stitching is easy to understand, learn and use. The participants also all preferred using the stitching interface when taking photos for all task types in our study. Although we did not quantify objects identified in the description tasks, it was noted by multiple participants that with stitching interface they received more comprehensive descriptions of the content of the images. All participants also showed desire to continue using stitching interface after the experiments and "*look forward to seeing it released to the general public*". The feature participants liked most in the stitching interface was the audio guidance which allows "*easy identifying of many things*" and really helps with blind people's photo taking troubles, especially when looking for a specific but small piece of information. It was also mentioned that the fact our stitching interface "*cleverly put different images together figures out the orientation of them*" gave them more freedom of interaction.

Discussion

As seen in both related work [24] and our experiments, blind people usually have worse phototography skills than sighted people, so an accessible camera interface with few restrictions and rich guidance is crucial to help them take better photos. Existing camera interfaces often fail to provide assistance, resulting in poor performance of photo-based assistive applications. We observed that when users of a single-photo interface followed crowd workers' instructions to adjust the camera, they often overdid the adjustments, having to retake photos for several times until they succeeded. With the stitching interface, however, users felt much easier and less stressful to follow framing instructions.

Several further improvements to the stitching interface were directly derived from participants' feedback. One suggestion from the participants was to include a a button or gesture to pause and resume the stitching process, to make the interaction more natural. Another technical limitation discovered in the experiments was the application's difficulty with stitching photos of non-flat surfaces, *e.g.*, cylinders.

While participants appreciated the stitching interface, and preferred it for all tasks in the study, they suggested that stitching should not replace the existing single photo camera interface, but should exist as an option that can be used when needed, especially for those blind users whose camera skill is good enough to capture visual information most of the time without using assistive technology.

ELICITING COMPLETE VISUAL DESCRIPTIONS

Image stitching allows users to capture more context to show to workers, while still maintaining a lightweight interaction. However, additional visual information is not useful unless sighted workers describe it completely back to the blind user. To improve the question-asking interaction further and increase bandwidth of visual information in Q&A interations, we focus on how to elicit more detailed responses and spatial

Figure 5. Images used to evaluate both iterative and parallel labeling approaches, which cover a range of scenarios observed in VizWiz usage.

information from the crowd, while being able to accurately compensate them for their efforts.

Queries are defined by users, and thus have a wide range of variance. For example, an image of a menu could be associated with the question *"how much does the 6" Turkey Breast sandwich cost?"* (to which the answer could be *"$4.99"*) or with the question *"What does this board say?"* (to which the best answer likely involves transcribing the entire menu from the image). Because it is not possible to reliably classify the amount of information required or provided for a given query automatically, both VizWiz and Chorus:View pay a standardized fee for each answer, and do not have a means of scaling payments to match worker effort. In the long term, this could lead to disincentivizing the available workforce.

Below, we first describe the set of 5 images used to test RegionSpeak, and initial worker responses to them. Then, we present two methods to increase the amount of detail in answers without overloading any one worker: an iterative workflow so that workers can build on one another's responses [2], and a parallelized workflow where users can focus on small portions of the image at once.

Methodology and Initial Responses

We analyzed ∼1,000 questions VizWiz couldnt answer because the single photo submitted with the question did not contain enough information or had too much information for a single worker. Five representative photos were created to simulate those questions (Figure 5): a set of five packages of food (simulation of Figure 1), a simple diagram on a whiteboard, a set of buttons on a microwave, a menu from a restaurant, and a picture of an outdoor scene in a commercial area.

We began by asking workers on Mechanical Turk to describe five different images (Figure 5) with general questions such as *"Can you describe what you see in this image?"* We sampled a broad set of workers by running experiments with minimum qualification and during various periods of the day.

We collected responses from ten workers for each of these images (for a total of 50 data points). We began by counting the character length of answers, to provide a quantitative measure of answer detail. To measure the content more accurately than using answer length alone, we also had 2 researchers code the answers for all images. The following attributes were coded for, and the Cohen's Kappa scores for 50% (25 answers) of the initial dataset are shown beside each:

- **Validity:** If an object is described as being in the image, is it actually shown? Answers could be completely correct (no incorrect element named in the answer), partially correct (at least one correct and one incorrect element), or incorrect (no correct elements). Cohen's kappa of 0.95.
- **Minimalist:** Does the answer appear to be the answer requiring the least effort, even if valid (e.g., answering *"can"* when describing a can of Del Monte green beans)? Cohen's kappa of 0.69.
- **Distinct Items:** How many distinct items are named in the answer? Cohen's kappa of 0.85.
- **Details:** How many explanatory details provided beyond the core description (e.g., answering *"mug"* counts as 0 additional details, answering *"a green mug that doesn't have any liquid"* counts as two additional details details)? Cohen's kappa of 0.88.
- **Spatial Information:** How many spatial cues are provided in the answer? Cues can be either object locations or relative layouts, i.e., a cup on the table would yield one spatial descriptor. Cohen's kappa of 0.8.

Results

For our initial examination of the 5-image dataset, answers had an average length of 53.7 characters (median 33, $\sigma = 53.0$), with a minimum of 3 characters and a maximum of 217. The crowd's answers yielded 37 valid answers, 5 partial answers, and 8 invalid; 13 were minimal; an average of 2.69 objects described (median 2, $\sigma = 2.18$); an average of 1.4 detailed descriptions (median 0, $\sigma = 2.28$); and an average of 0.46 pieces of spatial information (median 0, $\sigma = 1.13$).

Iterative Responses

Our most complex scene (the outdoor image, Figure 5(e)) had a minimum length of 3 characters (*"car"*), a maximum length of 44 characters, and an average of 21.4 characters (median 23.5, $\sigma = 15.1$). We used this image and these descriptions to explore increasing the descriptive level of the answers.

In initial responses, workers tended towards shorter, more generic initial answers due to the complexity of describing any more detailed version of the image. In this study, we seeded workers with the 10 initial responses about Figure 5(e), then had workers iteratively improve the last worker's answer (with questions like *"Can you improve the description of this image?"*). Each of the 10 initial responses was iterated on 3 times, for a total of 30 additional data points.

In a pilot version of this tool, we showed workers the prior results from other workers and asked them to write a more descriptive version of the answer. Unfortunately, because workers had to take the extra step of copying or rewriting the initial

answer, the resulting responses were often even more brief than the original, or simply did not build off of the prior content. For our final version of the iterative response tool, we pre-populated the prior answer into the response area so that workers could more easily expand on the existing responses.

Results

We found that when responses were pre-populated into the answer field, workers provided strictly more detail in each iteration. In terms of length, there was a monotonic increase in the length of the descriptions (Figure 6). Interestingly, while the relative length of each sequential iteration of a description was correlated with the length of the prior step (average pairwise $R^2 = 0.74$, $\sigma = .096$), the final length after 3 additional iterations was not meaningfully correlated with the initial length ($R^2 = .099$).

For our outdoor scene, the initial answers yielded 9 valid answers and 1 invalid; 3 minimal answers; an average of 1.95 objects described (median 1.5, $\sigma = 1.46$); an average of 0.3 detailed descriptions (median 0, $\sigma = .95$); and an average of 0.3 pieces of spatial information (median 0, $\sigma = .48$). Interestingly, despite being our most complex image, this scene resulted in a much lower number of objects described and less detailed descriptions than the rest of the set. These differences were not significant, but the reduction in the number of descriptive details was near-significant ($p = .088$).

After 3 iterations on the initial answer, all 10 responses were rated as valid; there were no minimal answers; an average of 8.3 distinct objects described (median 8.5, $\sigma = 2.71$); an average of 6.3 detailed descriptions (median 6, $\sigma = 3.02$); and an average of 6.5 pieces of spatial information (median 6.5, $\sigma = 2.12$). This indicates a significant increase in the number of objects described ($p < .001$), details provided ($p < .001$), and spatial information provided ($p < .0001$) (Table 1). The reduction in the number of minimal answers was also near-significant ($p = .081$), but the sample size was too small to confirm this with higher confidence.

Figure 6. Response length increases as answers are sequentially iterated on and converges after the third iteration. An example path is shown.

	Objects	Details	Spatial cues
Before	1.95	0.3	0.3
After	8.3	6.3	6.5

Table 1. Means of response codings before and after 3 iterations.

Limitations of Iteration

Our results show that iteration is an effective way to more consistently get reliable answers in terms of validity and avoiding minimalist answers (which is backed up by the crowdsourcing literature on such workflows), but is also a good way to increase the number of details and amount of spatial layout information provided by workers. Unfortunately, because iteration on these responses cannot be parallelized, there is a necessary decrease in response speed. To reach the point of any of our final results, we would have had to recruit a total of four consecutive workers, meaning the response would take roughly $4\times$ as long to return.

In addition to time costs, using an iterative workflow to expand the details of a scene can result in lengthy descriptions that are unwieldy for users to listen to. In our test, the longest answer was 708 characters. Tracking this much information from a text description can put extra cognitive strain on a user trying to understand what is in a scene and how it is laid out. Furthermore, the ability for users to scan quickly through objects in a space is severely limited if presented in a manner where they must listen to audio sequentially.

In practice, the workload provided to each worker in an iterative approach can also be hard to predict - initial workers may be provided with a simple image (such as Figure 5(a)), and not have many objects to describe, or a complex image (such as Figure 5(e)) and have to write a detailed description of many objects. Additionally, each subsequent workers' workload depends on what work has been done - if a previous worker did not describe the image well, the subsequent worker may have a lot more work to complete than if the previous worker had done a good job. This inequality in workload makes fairly compensating workers difficult, and may lead to frustration from workers if they cannot accurately estimate the hourly rate they will make from a task [17].

Parallelized Responses

Based on the results from our evaluation of the iterative approach, we wanted to find a way to elicit more complete, detailed, and spatially informative answers from workers, while also more fairly compensating them for the work they contribute. To do so we implemented a parallelizable approach that simultaneously collects multiple answers from workers, with each answer concretely grounded to be associated with a specific region of the image.

As shown in Figure 7, when crowd workers first begin a task, they select a region of the image that they will provide a label for. The goal is to have this be a region that contains an important object. Workers are left to select this region on their own, just as they would be left to choose what level of description to give in a typical VizWiz task. Previously labeled regions are also shown to the current worker to prevent

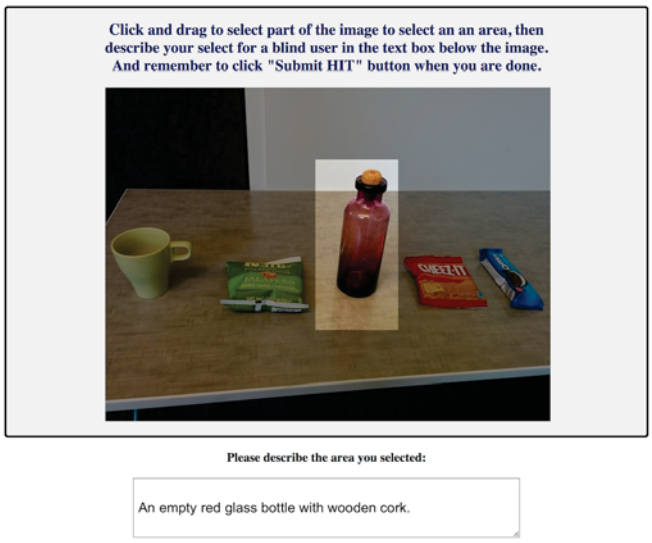

Figure 7. The parallelized crowd worker interface of RegionSpeak. Workers select a key region of the image with a bounding box, then provide a label for that region. Each region and label is aimed at helping to answer the user's question, which is provided just as in VizWiz.

duplications. The worker interface uses a Javascript plug-in, JCrop[4], to support region selection.

Our selection process is similar to LabelMe [18], which asked crowd workers to carefully select objects and areas in an image by outlining them. LabelMe's goal was to train a computer vision system to recognize objects in a scene – as such, its boundary precision needed to be much higher than to get general layout information. Our parallelized approach uses simple rectangles to simplify the task for workers.

The rectangular regions that we get from workers can be averaged and combined by clustering the rectangles and using approaches such as the Generalized Intersection Over Maximum metric. This metric was initially introduced for use in ShareCam [20], a collaboratively controlled webcam that let multiple viewers select their preferred viewing area and then found an optimal region based on that input. However, in practice, high levels of redundancy are often not possible or not practical due to cost and latency constraints. VizWiz also successfully uses an unfiltered response model, where users are able to evaluate the plausibility of answers themselves and determine if they should wait for additional responses.

Evaluation

We ran our evaluation of the parallized approach on all five images collected before (Figure 5). For each of the five images, we collected region tags and descriptions from five unique workers in one session. We repeated each session five times to obtain five different sets of responses, for a total of 125 data points. We coded the same five features as before (validity, minimalism, number of objects identified, number of details given, and number of spatial cues provided). For number of objects identified and number of spatial cues, bounding boxes were counted as both object identifiers (assuming they contained a valid label for a subsumed portion of the image). Additional object and details could be identified within the tag as well (for instance, a label for a section of a menu may identify multiple entrées).

We also added one additional feature, bound tightness, to determine how well workers did when selecting the appropriate region for a given label. We redundantly coded all 125 marked segments with two coders. There was a strong inter-rater agreement on a subset of 25 images (Cohen's kappa .74).

Results

The combined answer lengths for each of the 25 sessions were higher than in the iterative approach, with an average of 305.2 characters (median 149, $\sigma = 282.3$), a minimum of 50 and maximum of 935. Overall, these tasks resulted in no minimal answers; an average of 5.0 distinct items marked per session (median 5, $\sigma = 1.81$); an average of 4.8 descriptive details (median 6, $\sigma = 2.07$); and an average of 4.6 spatial cues (median 5, $\sigma = 1.63$). Additionally, 72% of the 125 segments marked by workers were rated as being a "tight bound" on the object they were framing, 18% were considered a "loose bound", and just 10% (12 marking) was rated as incorrect.

However, because the validity of tags was marked per-image, as would be the case with a single description from a worker in our baseline labeling example, just 44% of our images were rated as containing a completely valid label set, with the remaining 56% being rated partially correct. None of the label sets were entirely wrong. This highlights an important aspect of aggregating answers from the crowd: by using aggregated answers, it is more likely the *some* error is introduced, but the chances of an answer containing *entirely* errors falls similarly. In our case, "partially correct" ratings were almost always small errors in one or two labels.

Using parallelization also resulted in more evenly distributed workload for the crowd workers completing the tasks. We analyzed the length of the 125 individual image descriptions from the paralellized workflow, and compared them to the descriptions from the iterative workflow using Levenshtein edit distance[5]. The initial response's distance was the length of the answer, and each subsequent response's distance was the number of edits made. A homogeneity of variance test revealed that the standard deviation of parallelized responses ($\sigma = 70.19$) was signficantly less than the standard devation of iterative responses ($\sigma = 106.13$), with a Levene's statistic of $W(1, 163) = 11.867, p = .001$. By giving workers small portions of the image to focus on, their workloads were more fair and resulted in less variance in response length.

While individual answers in a parallelized session might have lower validity than iterative answers, the speed at which responses were collected is much improved since each object descripiton requires only one worker, and descriptions can be collected in parallel. Additionally, asking users to select a bounding box containing the object they are describing provides spatial information that can be used separately.

[4]http://deepliquid.com/content/Jcrop.html

[5]The Levenshtein edit distance is the number of character additions, deletions, or modifications in a piece of text.

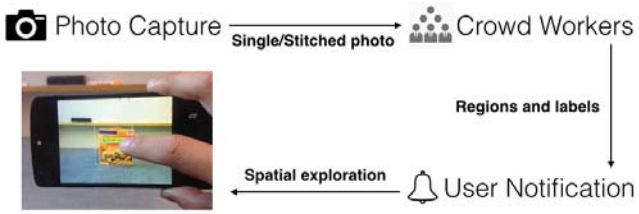

Figure 8. Interaction flow of a common RegionSpeak dialog turn.

REGIONSPEAK

By combining the image stitching algorithm and the parallelized object description presented in previous sections, we are able to create a tool which allows for quick but thorough visual descriptions of complex images. We applied these methods to create RegionSpeak, a mobile application that allows blind users to compose panoramic images to accompany a question, and then perform spatial exploration of the space.

Workers are given the panoramic images and asked to describe objects in them as described above. When the answers are forwarded back to users, they can explore the the space by sliding their finger over the regions and having the system speak the label. By doing this, users can get a better sense of where objects in a scene are located relative to one another. This also reduces the search space that users have to process and preserves spatial information, which has the potential to reduce cognitive load [12]. Since this interaction is similar to established screen reader behavior on touch screen mobile devices, which users are already familiar with, it is easier to learn and more acceptable for blind people [11].

Implementation and User Interface

A dialog turn of RegionSpeak interaction has three steps (Figure 8). First, the user begin by opening the RegionSpeak app and either taking a single photo or a stitched image with the RegionSpeak stitching interface. The user then sets the phone down and waits for responses from crowd workers. When the first response comes back, RegionSpeak notifies the user to launch RegionSpeak which opens up the real-time camera view port and starts aligning regions marked by workers in the view port. This functionality is supported by an existing real-time object tracking algorithm (OpenTLD [9]).

When a region labeled by the crowd is recognized, it is added as an overlay on the camera view port and tracked in real time as the camera is being re-framed. A region can be re-detected if it was lost and reappears, this way RegionSpeak can keep the interaction light weight by allowing users to leave and resume. Given the single object limitation of the OpenTLD algorithm, if there are multiple regions returned, we use the largest region as our tracking target and apply the transformation of primary target to other region overlays. To help ensure that regions are not subject to occlusion by other regions, we filter smaller regions to the top layer.

In addition to projecting overlays on real time view port, we also give users the option to explore multiple regions and labels in the initial static frame they captured in the very beginning of each dialog turn. The advantage of exploring in live camera view port is that users can find the direction of objects marked by crowd by pointing at the corresponding region. Audio and haptic feedback are both provided, when the user moves a finger into a labeled area the phone vibrates and reads out the description provided by web workers. This kind of single-finger scan interaction in exploratory interface has been studied in prior projects and proven successful [4, 11].

Timing and Live Trial

Following our main trials, we then ran another trial to see how a "real" run would look. We captured a stitched image of a long workbench that contained multiple tools (Figure 2), then recruited workers to mark regions. While workers were intended to be recruited in parallel, we did not explicitly accelerate recruitment, and if workers joined the task after others had finished, they were able to see the marked regions.

For this live trial, we also measured the time it took workers to mark regions and to provide descriptions. The time workers took to complete the task ranged from 0:38 seconds to 1:58 minutes. On average, the task completion time was 1:05 minutes. For perspective, this is on the same order of the speed at which VizWiz gets responses during live usage ($\sim$133.3s for the first answer [3, 6]).

Regarding the quality of the ten responses collected in the live trial, the total length was 566 characters; no description of regions was minimal or invalid; 11 distinct objects were described with 11 descriptive details and 12 spatial cues provided; 7 of the bounding boxes drawn by web workers were tight while the other 3 were loose and none was incorrect.

Limitations of RegionSpeak

While, unlike iterative response generation, RegionSpeak is able to elicit answers from workers in parallel, there is the chance that workers will provide overlapping or redundant answers (as we saw cases of in our experiments). This is especially problematic in situations where there may be one or a small set of very salient objects in a scene (for example, a bookshelf with a television on it may attract many workers' attention to label the television first and ignore other objects on the shelf or the shelf itself).

While this is typically not harmful and did not happen frequently in our experiments as RegionSpeak shows previous labels, it does waste worker effort and complicates responses. Although RegionSpeak shows workers the entire set of objects that others have labeled up to that point, this approach does not guarantee prevention of duplication unless tasks are run in series. However, as with the iterative answer case, this leads to significantly increased latency. A more sophisticated way to avoid these issues within the context of a synchronous task is to maintain a shared state of each task so that workers can see what others are annotating in real time. Future versions of RegionSpeak will include this functionality.

DISCUSSION

Our results suggest that the responses we get from RegionSpeak are much more closely aligned with our intended outcomes: we get shorter responses that contain more details and information about the spatial layout of objects, and can be run in parallel to avoid high latency. Workers generally