Indoor Localization Using Camera Phones

Nishkam Ravi, Pravin Shankar, Andrew Frankel,
Ahmed Elgammal and Liviu Iftode
Department of Computer Science, Rutgers University, Piscataway, NJ 08854
{nravi, spravin, afrankel, elgammal, iftode}@cs.rutgers.edu

Abstract

Indoor localization has long been a goal of pervasive computing research. In this paper, we explore the possibility of determining user's location based on the camera images received from a smart phone. In our system, the smart phone is worn by the user as a pendant and images are periodically captured and transmitted over GPRS to a web server. The web server returns the location of the user by comparing the received images with images stored in a database. We tested our system inside the Computer Science department building. Preliminary results show that user's location can be determined correctly with more than 80% probability of success. As opposed to earlier solutions for indoor localization, this approach does not have any infrastructure requirements. The only cost is that of building an image database.

1 Introduction

Most of the early systems for indoor localization [21, 5, 3, 11] focussed mainly on location accuracy and involved the use of custom hardware. This implied heavy deployment costs and labour requirements. Of late, the focus has shifted to minimizing infrastructure requirements without compromising substantially on accuracy [10, 16, 4]. The reason is well understood: since location information only serves as a parameter to location-based services, the cost of deploying localization systems should be a minute fraction of the total cost of provisioning location-based services.

Camera-equipped mobile phones are being put to many uses as an interesting study indicates [8]. In this paper, we explore the possibility of determining user's location indoors based on what the camera-phone "sees". The camera-phone is worn by the user as a pendant(Figure 1), which captures images periodically and sends them to a web server over GPRS. The web server has a database of images with their corresponding location. Upon receiving an image, the web server compares it with stored images, and based on the match, estimates user's location. We accomplish this with off-the-shelf image matching algorithms, by tailoring them for our purpose. We improve location accuracy by using an algorithm that takes into account the trajectory of the user. We built an image database for the Computer Science building with nearly ten pictures per "corner" to account for real-life issues such as varying heights of the users, different angles that may correspond to the same image, etc. Our experimental results indicate that room-level accuracy can be achieved with more than 90% probability, and meter-level accuracy can be achieved with more than 80% probability. The error can be further reduced by using more sophisticated image matching algorithms, which is not the subject of this paper. The key advantage of using this approach is that it does not require any infrastructure. Neither custom hardware, nor wireless access points are required. Physical objects do not have to be "tagged" and users do not have to carry any device apart from what they already do: a mobile phone. The only cost involved is that of building an image database.

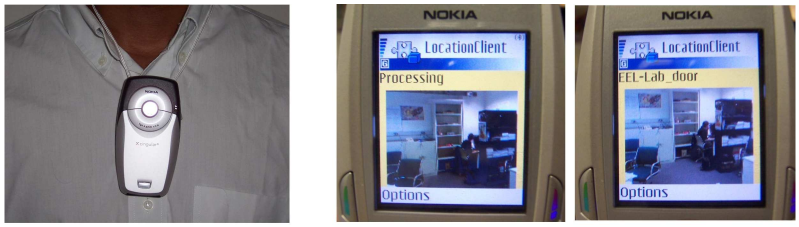

Figure 1: Left: User wearing the phone as a pendant, Right: Snapshots of the client running on the phone

2 Approach, Issues and Solutions

2.1 Location Determination

When the web server receives a query image, it compares it with the images stored in the database. Every image that matches with the query image is assigned a weight which reflects the degree of similarity between the two images. By image comparison, we imply feature comparison. We use three off-the-shelf algorithms for image comparison: Color Histograms [19], Wavelet Decomposition [6] and Shape Matching [7]. Each algorithm assigns a weight to the image. The total weight of an image is calculated as a linear combination of the weights assigned by each algorithm. If the weight of the best match is less than a certain threshold value, the query image is discarded. This is necessary to prevent wrong location updates from being sent to the client.

Once the weight of the images in the database with respect to the query image are known, the *Naive approach* is to return the location of the image that matches the query image with maximum weight. Instead, we use a *History-based approach*. In this approach, the web server keeps track of the trajectory of the user. Based on the past locations of the user, a better estimate of the current location can be derived. When the server receives a query image, it looks at the last *n-1* query images. The current location of the user is the one that maximizes the probability of *seeing* the n query images in the shortest period of time. For implementing this, we use a Viterbi-like algorithm [18]. For details refer to [12].

2.2 Database Creation

For good results, it is important to build an extensive database of images. We wrote a Java client (createDB) that runs on the phone and records video as the database creator walks around, simultaneously sending the images to the web server. The web server extracts features from the received images on the fly and stores the features in a database. The image may or may not be stored. The images/features in the database are tagged with location, manually afterwards. The process of tagging images with location can be partialy automated, by using a speech recognition interface on the phone, so that the database creator can tag images by announcing her location while pictures are taken.

2.3 Energy Optimization

Energy consumption on the phone is primarily determined by two factors: frequency of sending query images and size of the image. We found that: so long as the resolution of images in the database is the same as the resolution of query images, performance is unaffected. We worked

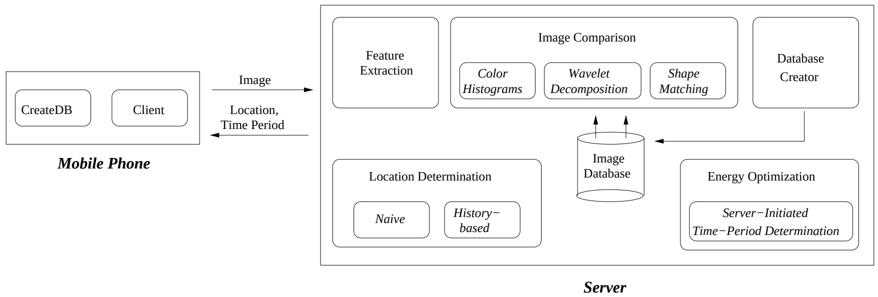

Figure 2: System components

with low-resolution images of size 5KB and could save substantially on energy consumption. For reducing the frequency of sending query images without affecting the frequency of receiving location updates, we implemented a *server-initiated* location query approach. In this approach, when the server responds with location, it piggybacks the time period after which the phone should send the next query image. The value of the time period is determined by the current speed of the user and his distance from the next *critical point*. Critical points are those where the user can change directions (e.g intersections of corridors). The server continously sends back location updates by *guessing* the location of the user based on their speed and last location. For details refer to [12]. Note that this approach is applicable to only corridors and such, and is based on the assumption that the user does not abruptly change his speed and direction.

2.4 Privacy

We have not yet built any privacy mechanisms into our system, but the following mechanism can be easily incorporated. Instead of sending one query image, the client would send multiple images, where one image would correspond to user's actual location while others would be arbitrary. The web server would send back location corresponding to every image, and the client would pick the one corresponding to his actual location. This would make it hard for the web server to discover the user's location. Sending extra images, however, would increase the energy consumption on the phone. This can be worked around by using a trusted agent as an intermediary between the phone and the location server. The phone would send the query image to the trusted agent which would insert dummy images and send them to the server.

Figure 2 shows the various components of our system. *Client* and *CreateDB* were implemented in Java using MMAPI [1] which is supported on Symbian OS phones. We used the Nokia 6620 mobile phone. All server side components were implemented in C++ for performance reasons.

3 Experiments and Preliminary Results

We created an extensive image database for the third floor of the Computer Science Department building, which includes 16 rooms, staircase, a bridge and the corridors. The goal of the experiments was to test the feasibility of our approach. We therefore sought the answers to the following two questions : (1) How successful is our approach in achieving room-level accuracy?, and (2) How successful is our approach in estimating the orientation and location of the user anywhere in the building?

Figure 3: Low-resolution pictures of different corners

Figure 4: Query image matches with Image 4

We conducted three main experiments. The experiments were carried out on two users with five trials per user. In the first experiment, the user would wear the phone as a pendant and enter the room as the camera took a picture. The image database would comprise of only pictures corresponding to user standing at the door of a room and facing inside, with 10 images per door. This experiment was conducted to find out the probability of success for room-level accuracy. History-based approach does not apply to this scenario. In the second experiment, the image database comprised of only inside-room pictures of all rooms. The experiment was conducted with the user wearing the phone as a pendant and walking around inside rooms, sending a query image every 4 seconds. Figure 3 shows inside-room pictures of a few rooms. Figure 4 shows example of a query image that matches best with Image 4 in Figure 3. Most of the rooms in the Computer Science building are around 4mx7m. We report user's location to quarter-room level accuracy (North-East, North-West, South-East, South-West), and report one of the 4 orientations (facing North, facing South, facing East, facing West). The database consists of 10 images per corner to account for varying user heights and angles, and 16 corners per room (4 corners per quarter corresponding to the 4 orientations). The third experiment was conducted with the complete database, as the user walked around on the 3rd floor. This includes corridors, staircase and the bridge in addition to rooms. We call this *corner-level* accuracy.

Table 1 shows the results for the three experiments. The results correspond to the ten trials. From the results, it can be concluded that for applications that require only room-level accuracy, the Naive approach would suffice. Corner-level accuracy corresponds to the most general case: the user walking inside rooms, across rooms and inside corridors. History-based approach clearly shows improved performance for this case.

We worked with low-resolution JPEG images of size 5 KB. Our experiments with high-resolution images (128 KB in size) resulted in approximately the same success rate as low-resolution images. By working with low-resolution images we could save substantially on energy as well as response time. Table 2 shows the average response time (i.e latency) for our experiments, as well as the average energy consumption on the phone for sending a query image and receiving one location update. In the idle mode, the phone consumes around 4mJ per second.

What is currently missing from our experiments is a thorough user study which will be helpful in evaluating the usability of the system as well as the effectiveness of the *server-initiated* energy optimization approach. The constants used in the experiments, such as 10 images per corner, or n=5

Table 1: Probability of Success for the Three Experiments

Approaches	Naive	History-Based
Room-level accuracy	93%	N/A
Quarter-room-level accuracy	83%	94%
Corner-level accuracy	50%	80%

Table 2: Energy Consumption and Response Time

Image Size	Avg. Response Time	Avg. Energy Consumption
5KB Image	720 msec	630mJ
128KB Image	4100 msec	3600mJ

for the history-based location determination approach were arrived at after careful experimentation and tuning. Due to lack of space we omit the description of experiments done to arrive at these constants.

4 Related Work

A number of indoor positioning systems have been built. Most of these require special hardware such as ultrasound transmitters/receivers [5, 11], IR badges [21], microphones [16], PCs [4] or use wireless access points to determine location [3, 10]. Microsoft's EasyLiving [9] project uses cameras installed in rooms to track humans using vision techniques.

Work is being done in using camera phones as interaction devices by tagging physical objects with visual codes and using vision techniques to extract and interpret the information stored in these visual codes [13, 15, 17]. Localization could also be possibly achieved with this method. However, physical objects would have to be tagged. Sarvas et al [14] had demonstrated the use of camera phones for getting meta-information about physical objects using a human-in-the-loop approach. They demonstrated their approach for getting meta-information about outdoor landmarks. We demonstrate the feasibility of using camera phones for indoor localization and without user involvement. Some augmented reality systems employ vision techniques for augmenting physical objects with meta-information [2, 20]. The goals as well as the approach are quite different. We use image matching algorithms, while AR community uses object recognition algorithms for getting information about objects in sight. Determining similarity between two images is easier than recognizing objects inside an image.

5 Conclusions

In this paper we showed that it is feasible to achieve indoor localization using just camera phones and off-the-shelf image comparison algorithms. We were able to attain room-level accuracy with more than 90% chance of success and corner-level accuracy with more than 80% chance of success, using a history-based location determination approach. The latency of receiving location updates over a GPRS connection is less than a second, which would be even lower for a 3G connection. We discovered that the image comparison algorithms remain unaffacted if resolution of the images in the database is the same as the resolution of the query image. By using low resolution images, we were able to reduce energy consumption and response time significantly.

Remaining Challenges: The scalability of the approach is limited by two factors: (i) varying lighting conditions, and (ii) presence of moving objects such as human beings, especially in crowded buildings such as museums. We are currently working on these challenges.

References

[1] Mobile Media API(MMAPI), http://java.sun.com/products/mmapi/index.jsp.

[2] Augmented Reality, http://www.augmented-reality.org/.

[3] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM (2)*, 2000.

[4] G. Borriello, A. Liu, T. Offer, C. Palistrant, and R. Sharp. Walrus: wireless acoustic location with room-level resolution using ultrasound. In *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, 2005.

[5] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Mobile Computing and Networking*, 1999.

[6] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast multiresolution image querying. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 1995.

[7] T. Kato, T. Kurita, N. Otsu, and K. Hirata. A sketch retrieval method for full color image database. In *In Proceedings of International Conference on Pattern Recognition*, 1992.

[8] T. Kindberg, M. Spasojevic, R. Fleck, and A. Sellen. The ubiquitous camera: An in-depth study of camera phone use. *IEEE Pervasive Computing*, 4(2), April-June 2005.

[9] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. Multi-camera multi-person tracking for easyliving. In *VS '00: Proceedings of the Third IEEE International Workshop on Visual Surveillance (VS'2000)*, 2000.

[10] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit. Place lab: Device positioning using radio beacons in the wild. In *Proceedings of the Third International Conference on Pervasive Computing*, 2005.

[11] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, 2000.

[12] N. Ravi, P. Shankar, A. Frankel, A. Elgammal, and L. Iftode. Indoor localization using camera phones. Technical report, Department of Computer Science, Rutgers University, October 2005.

[13] M. Rohs and P. Zweifel. A conceptual framework for camera phone-based interaction techniques. In *Proceedings of the Third International Conference on Pervasive Computing*, 2005.

[14] R. Sarvas, E. Herrarte, A. Wilhelm, and M. Davis. Metadata creation system for mobile images. In *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, 2004.

[15] D. Scott, R. Sharp, A. Madhavapeddy, and E. Upton. Using visual tags to bypass bluetooth device discovery. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(1), 2005.

[16] J. Scott and B. Dragovic. Audio location: Accurate low-cost location sensing. In *Proceedings of the Third International Conference on Pervasive Computing*, 2005.

[17] E. Toye, R. Sharp, A. Madhavapeddy, and D. Scott. Using smart phones to access site-specific services. *IEEE Pervasive Computing*, 4(2), 2005.

[18] A. J. Viterbi. Error ounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–269, 1967.

[19] R. B. W. Niblack and et al. The qbic project: querying images by content using color, texture and shape. In *In Proceedings of SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[20] D. Wagner, T. Pintaric, F. Ledermann, and D. Schmalstieg. Towards massively multi-user augmente reality on handheld devices. In *Proceedings of the Third International Conference on Pervasive Computing*, 2005.

[21] R. Want, A. Hopper, V. Falco, and J. Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10, 1992.