Scalable Recognition and Tracking for Mobile Augmented Reality

Jaewon Ha[*] KAIST
Kyusung Cho[†] KAIST
Francisco A. Rojas[♯] KAIST
H. S. Yang[‡] KAIST

ABSTRACT

For an augmented reality application to be realistic, exact tracking of target objects is essential. However, recent mobile augmented reality applications such as location or recognition-based applications lack realism in augmentation due to inexact tracking methods. Vision-based tracking is capable of being exact and robust, but in a mobile augmented reality system, the number of objects that can be augmented is far limited. In this paper, we propose a new framework that overcomes the limitations of the previous works. First, our framework is scalable to the number of objects being augmented. Second, our framework provides an improved and realistic augmentation by adopting a real-time accurate visual tracking method. To the best of our knowledge, there has been no system proposed successfully integrating both of these properties. To achieve scalability, the bag of visual words based recognition module with a large database runs on a remote server and the mobile phone tracks and augments the target object by itself. The server and mobile phone is connected by conventional Wi-Fi. Including network latency, our implementation takes 0.2 seconds for initiating an AR service on a 10,000 object database, which is acceptable for a real-world augmented reality application.

Keywords

Mobile augmented reality, mobile phones, scalable recognition, visual detection and tracking.

1. Introduction

Aided by the increasing popularity of smart phones, mobile augmented reality (AR) has received much more attention than ever before. For high quality AR, it is important to know the pose of the target object in every frame, and so tracking is essential. Early AR researches assumed enough computational power for fully utilizing heavy vision algorithms. These previous works succeeded in showing acceptable tracking performance for AR. However, these vision algorithms are not available in mobile devices due to their low resources and computational power. For this reason, not until recently did real-world interactive mobile AR applications become available. However, developers for these particular AR applications chose to work on different domains such as recognition-based AR, or they chose to avoid directly augmenting the information on the real object as shown in location-based AR applications. On the one hand, these approaches can be evaluated as a good strategy since they roughly overcome the technical limitations and successfully provide an AR service. But on the other hand, they didn't provide a realistic AR experience, which is what we are ultimately looking for.

Fortunately, computational limitations on mobile devices have being reduced significantly recently. Embedded processors in the mobile phones have become much more powerful than previously. By redesigning or tuning PC-based vision algorithms for the mobile device environment, state-of-the art works on natural features such as SIFT and Ferns have been successively ported to the mobile platforms. The detection/tracking methods suited for mobile phones have been proposed and verified [Wagner et al. 2008].

Some of the works adapting these new technologies have been released recently. [Wagner et al. 2008; Wagner et al. 2009] provides an elaborated level of AR adapting these finely adjusted vision algorithms. But still the authors only showed an augmenting capability for a small set of objects. To the best of our knowledge, there has not been any mobile augmented reality system proposed which is scalable in regard to the number of augmented objects.

In this paper, we propose a new mobile augmented reality framework that provides an elaborated level of tracking by adopting recent mobile vision technologies and that is also scalable to the number of objects being augmented. We achieve these two goals via a scalable recognition module on the server side and by performing tracking on the mobile side. The information needed is sent over conventional Wi-Fi networks. In our implementation, a cold start of the AR service takes about 0.2 seconds, which is acceptable for a real-time application. As for the bag of visual words based scalable recognition module used on server side, it is already verified to have stable performance up to 1 million objects according to previous researches [Nister et al. 2008; Chum et al. 2007].

The rest of the paper is organized as follows. In section 2, a quick overview of the overall framework is provided. In section 3, the server side scalable recognition module is described. In section 4, the mobile side detection and tracking algorithm is explained with detailed parameters. In section 5, the optimization we did for the mobile device programming is summarized. In section 6, experiments focused on a time interval is given. Lastly, the conclusion is provided in section 7.

[*] e-mail : hjw@paradise.kaist.ac.kr
[†] e-mail : qtboy@paradise.kaist.ac.kr
[♯] e-mail : francis@paradise.kaist.ac.kr
[‡] e-mail : hsyang@paradise.kaist.ac.kr

2. Framework

The framework is configured into two parts: the mobile phone and the server computer, both of which are connected through the network. A visual stream comes from the mobile phone camera and the user is required to select a specific region of interest on the mobile phone screen. After the user selects the region, the image information on that region is sent to the server and the recognition process is preceded on the server side. After the server finds a match, it sends back the corresponding tracking information including the initial homography, keypoint locations, description vectors, and the information to be augmented on. With the tracking information received, the mobile phone runs the detection process until it finds the target object. After succeeding in detection, the tracking is carried on. The overall framework is depicted in Figure 1.

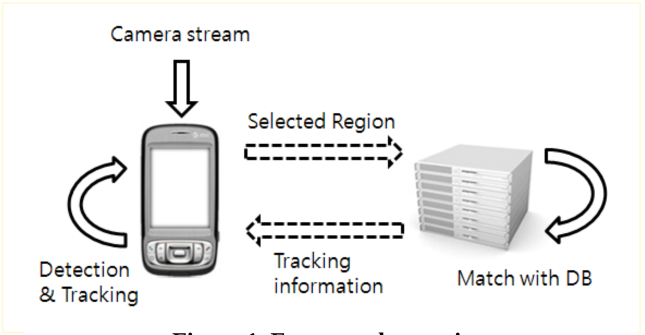

Figure 1. Framework overview

3. Server Side: Scalable Recognition

The bag of visual words scheme is used for scalable recognition. Among its variation, we adopted the vocabulary tree [Nister et al. 2008]. A major step in the bag of visual words is the quantization step which extracts representative data points called visual words from a large quantity of description vectors; this is an elaborate retrieval method that is accurate and fast. To extract the descriptor image features, SIFT [Lowe 2004] is used.

3.1 Quantization

The resulting description vectors from thousands of images are tremendously large. Therefore, storing and searching the entire vectors is not a viable option. To handle the large data efficiently, a clustering method is usually adopted. With a proper clustering method, the large database is summarized with a small number of clusters, and the representative data points of the clusters (usually the mean) are named visual words.

Once the visual words are identified, the images in the database are summarized with the obtained visual words. An actual keypoint description on the image is replaced with a visual word for which the feature belongs. This process drastically reduces the size of data and eventually the search space in the retrieval process. For the vocabulary tree, the hierarchical clustering method is used. This method clusters data points using K-means clustering, and it does K-means clustering again on K result clusters from the previous clustering. This process is repeated for a specified number of iteration steps. Therefore in hierarchical clustering, K is the branch factor of the tree, and it needs another parameter for the number of iterations corresponding to the depth of the tree. For a vocabulary tree with K=6 and depth=6, 300k descriptors from 1000 images are summarized to 47k leaf nodes.

As K-means clustering is sensible for an initial seed, it needs additional steps to distribute seeds uniformly and for verifying that the seeds were proper. In our implementation, we used the mean and the standard deviation of the cluster, and initial seeds are distributed centered on the mean and the standard deviation scale. The distance between seeds is also required to be farther than a certain threshold. As a verifying step, we checked that the number of data points assigned to the cluster was more than zero.

3.2 Retrieval

Scoring The appropriate weighting of tree nodes reduces the search space of tree nodes and this directly improves searching time and accuracy of the retrieval process. Tree nodes are weighted according to the traditional text retrieval TF-IDF (Term-Frequency Inverse Document Frequency) scheme. For a visual word node i, its weight is defined as

$$n_i = k_i w_i,$$

where k_i is the number of features that visited the node, and w_i is the weight of the node. The weight is defined as

$$w_i = \ln \frac{N}{N_i},$$

where N_i is the number of images that has at least one visual word at node i, and N is the total number of images. This weighting rule clearly reflects the TF-IDF scheme. k_i corresponds to the term frequency (TF) by being the number of visual words that occurred on an input image and w_i corresponds to the inverse document frequency (IDF) by being the inverse of the number of occurrence of the visual word in each image in the overall database. By multiplying the IDF to the TF, a vocabulary occurs frequently on the overall database and is treated insignificant by being assigned small weights.

An image is described by the path down its entire description vectors by the vocabulary tree and the concatenating leaf node weightings following the above weighting rule. Then the similarity between the query image q and the database image d is calculated by the normalized difference:

$$s(q,d) = \left\| \frac{q}{\|q\|} - \frac{d}{\|d\|} \right\|.$$

For an efficient implementation, the inverted files at each leaf node store the term frequency of the vocabularies for each database image and pre-calculate the node weight from the large test dataset. For a query image, only those database images linked to an inverted file at the visited node are compared by the following equation

$$\|q-d\|_2^2 = 2 - 2 \sum_{i|q_i \neq 0, d_i \neq 0} q_i d_i,$$

where q and d are normalized leaf node weightings for the query and the database image. Detailed proofs are given in [Nister et al. 2008].

Geometric matching As a post-processing step, PROSAC is proceeded to refine the result and find an exact matched region. The top score of three images is selected as a candidate and the one image that has the most inliers is chosen to be the retrieval image. Iteration steps for PROSAC is limited under 300 times for fast execution.

4. Mobile Side: Detection and Tracking

Once receiving the keypoint locations and the corresponding descriptions, the mobile side runs the detection and tracking by itself. The detection is carried on first, and if the detection result is fine, tracking is carried on repeatedly. For every frame after the tracking ends, the quality of the tracking is evaluated, and if its result is bad, detection is carried on again.

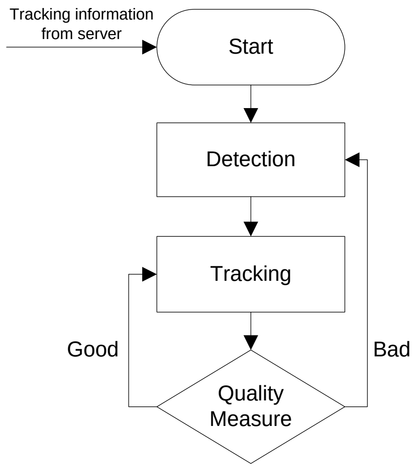

Figure 2. Detection and Tracking

4.1 Detection

Concerning the computational power and memory consumption of mobile phones, the Modified SIFT [Wagner et al. 2008] is selected as the mobile phone side detector. Modified-Fern was also a candidate, but as its resulting descriptor is rather large to be sent through the network, it is discarded for the sake of the overall fast response of the framework. In Mobile SIFT, the computationally intensive Difference of Gaussian and the Local min/max search are replaced with the FAST detector on an image pyramid. The image pyramid at level 5 with a scale ratio of 1 over root 2 is built and keypoints from each pyramid are extracted and projected onto the original image size. The maximum number of keypoints is limited up to 250 points to maintain computation speed. For a larger scale variation, the input image is also formed at image pyramid level 3 with half the scale. And this half scale is efficiently implemented with a subsample.

For the descriptor, rather than the usual 128 dimension descriptor, a configuration of 36 dimensions of 4 orientation bin and 3x3 sub region descriptor is used. In Lowe's original experiment, the usual 128 dimensional setting showed a 50% matching rate for 40,000 database keypoints and a 3x3 4 bin histogram setting showed 43%, and this is a quite good result for a 1/4 computation.

After putative matching, the outlier removal is carried on by 300 PROSAC iteration steps. If the number of inliers for the homography exceeds a certain threshold, the homography is used in initiating the next step of the tracking procedure.

4.2 Tracking

Initiated with a homography from the detection phase, the purpose of the tracking is find the 6DOF pose of the target objects. To be robust and fast, the Coarse-to-fine matching [Klein and Murray 2007] is used. In the Coarse step, keyframe-based tracking is used. In the fine step, frame-to-frame tracking is used. The keyframe-based tracking is free from drifting but suffers from jittering whereas the frame-to-frame tracking is free from jittering but suffers from drifting. Using both together, these deficiencies of each method are complemented. To be more robust to a blur effect, the coarse step is carried on 160x120 projected keypoints. Working the coarse step at a higher level of the image gives a robust tracking capability on fast camera motions.

Given previous keypoint locations from the detection/previous tracking result, the actual matching is done using a Normalized Cross Correlation (NCC) of an 8x8 patch. Forming a circle with a certain radius centered at the current keypoint, all the pixels inside the circle at the previous frame is searched. Patches need to approximate an affine warp and this is done like in [Klein and Murray 2007]. Given the matching result, a robust estimator is proceeded to estimate an accurate 6DOF pose. Estimation of the pose is obtained by minimizing a Tukey biweight objective function [Hartley and Zisserman 2003] of the re-projection error, iteratively. If the matching ratio of the fine step exceeded 0.5, it is evaluated as good quality and the next frame tracking is initiated. If not, the detection is prepared for the next frame. A detailed procedure with parameters is given in Figure 3.

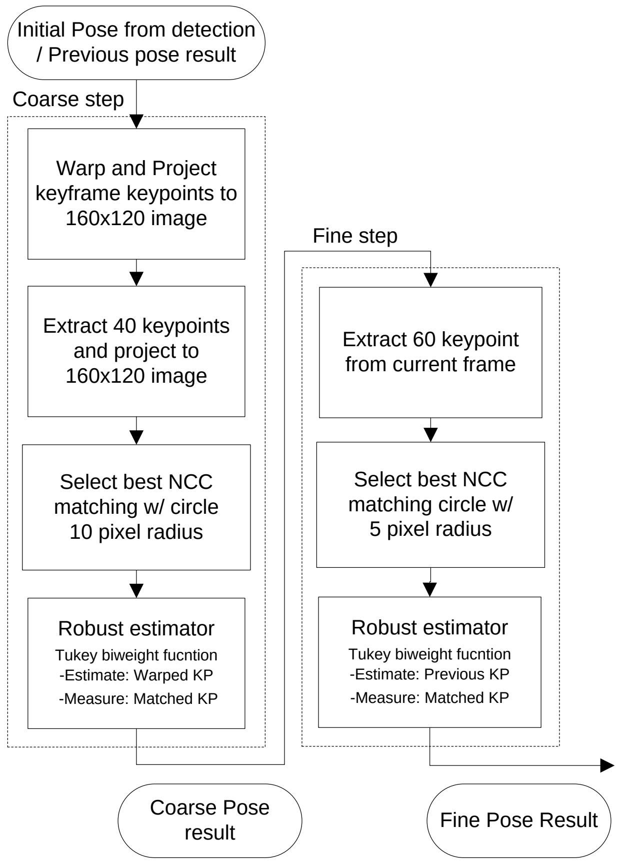
Initial Pose from detection / Previous pose result
Coarse step
Warp and Project keyframe keypoints to 160x120 image
Extract 40 keypoints and project to 160x120 image
Select best NCC matching w/ circle 10 pixel radius
Robust estimator
Tukey biweight fucntion
-Estimate: Warped KP
-Measure: Matched KP
Fine step
Extract 60 keypoint from current frame
Select best NCC matching circle w/ 5 pixel radius
Robust estimator
Tukey biweight fucntion
-Estimate: Previous KP
-Measure: Matched KP
Coarse Pose result
Fine Pose Result

Figure 3. Tracking procedure

5. Optimization

Fixed point conversion Normally mobile phones usually don't support the floating point processing unit (FPU). In our configuration, floating point processing is emulated by software and is usually much slower than integer processing. Therefore, all the floating point computations are replaced with fixed point computations by pre-multiplying and using an integer data structure. For our implementation, the three digits after the decimal position were fine to use.

Lookup tables For repeated, deterministic, and bounded output operations, these results are pre-calculated and stored in lookup tables for fast computation. In our implementation, the calculation of the image warping positions in small patches, the orientation, and the magnitude calculation of SIFT descriptors are boosted with lookup tables.

Single Instruction and Multiple Data (SIMD) Recent mobile processors support SIMD operations. With SIMD operations, a further degree of exploiting parallelism is possible. In tracking, 8x8 patch similarity calculations are boosted by SIMD operations, and as a result the overall process of tracking time is reduced by half.

6. Experiment

An experiment is carried out on an Android Nexus one with a 1GHz snapdragon processor. The camera has an image size of 320x240 pixels, gets frames at around 20Hz depending on the illumination conditions, and its network connection is IEE 802.1b 100Mbps. The database has pictures of 10k music CD covers and 10k corresponding music video scenes; it is used for correctly augmenting the music video scenes on the CD case covers.

6.1 Initial start time interval

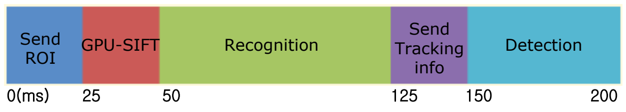

Figure 4. Time band of initial start time

The initial start time interval, including the network send and receive time composition, is shown in Figure 4. The time spent to send a region of interest (ROI) through the network was 25 ms (±10 ms); recognition on a 10k database was up to 100 ms including the GPU version of the SIFT description. Tracking information receiving time was 25 ms (±10 ms) and detection time was 50 ms (±15 ms). The pure network overhead was 50 ms (±20 ms), which is almost equal to the time taken processing detection for one frame. Though the experiment was carried out with no network traffic, the send and receive data size is up to 80Kb and only happens during user initialization; the influence from an external network condition is not expected to affect the system much.

6.2 Overall performance time measure

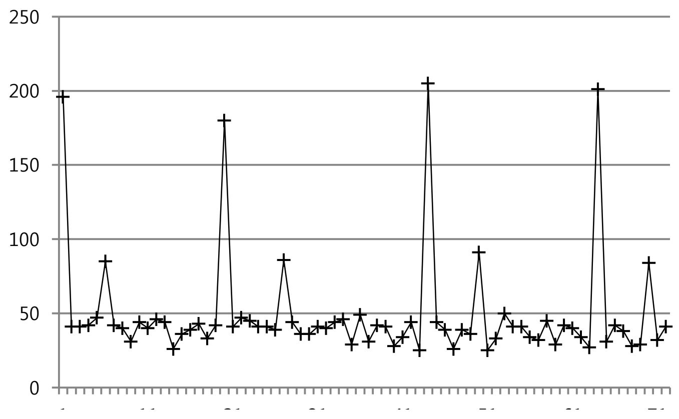

Figure 5. Overall performance time

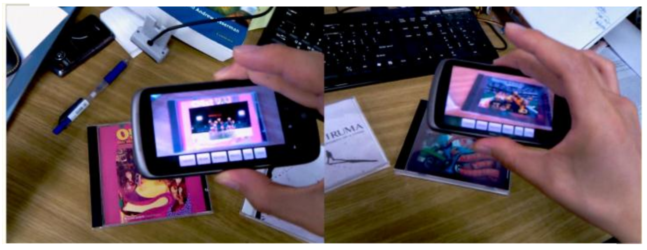

Figure 6. Augmentation of a music video on a CD cover

The overall time spent on a real use experiment is depicted on Figure 5. The horizontal axis is the number of frames and the vertical axis is the time spent on that frame in ms. For every reinitiallization of the AR service, the time for processing the frame is increased nearly 200 ms. As reinitialization by the user does not happen frequently, the latency from reinitialization would not much hurt the overall real-time performance of the system. Trailing peaks near 80 ms is due to internal garbage collection of the Android OS. Except for reinitialization, the average processing frame rate for 71 frames was 23.9Hz, which is faster than the mobile camera capture speed of 20Hz.

7. Conclusion

In this paper, we proposed a new mobile AR framework that provides an elaborate level of AR by natural feature tracking and that is also scalable to the number of objects to be augmented. We successfully integrated this scalable recognition technology based on a bag of visual words and the natural feature tracking on mobile phones through conventional Wi-Fi networks and proved that its performance is acceptable to real-world applications. In our research we remove the limitations on mobile AR coming from inexact tracking and make scalability possible. We expect that more high quality AR applications are created based on the presented techniques.

8. ACKNOWLEDGMENTS

This research is jointly supported by the Ubiquitous Computing and Network (UCN) Project and the Knowledge and Economy Frontier R&D Program of the Ministry of Knowledge Economy (MKE) in Korea as a result of the subproject UCN 10C2-J2-11T" and ICC research grants funded by KAIST in Korea.

9. REFERENCES

CHUM, O., PHILBIN J., SIVIC J., ISARD M., ZISSERMAN A. 2007, Total Recall - Automatic Query Expansion with a Generative Feature Model for Object Retrieval *IEEE International Conference on Computer Vision*, (Rio de Janeiro, Brazil, October 14-20, 2007)

HARTLEY, R., ZISSERMAN, A. 2003, A. Multiple View Geometry, and 2nd ed., Cambridge University Press, pp. 616-622.

KLEIN, G., MURRAY, D. 2007. Parallel Tracking and Mapping for Small AR Workspaces *6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. (Nara, Japan 13 - 16 November)

LOWE, D. 2004. Distinctive Image Features from Scale-Invariant Keypoints *International Journal of Computer Vision,* vol 60, no 2, pp 91 – 110

NISTER, D., STEWENIUS, H. 2008, Scalable recognition with vocabulary tree. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (New-York, USA, June. 17-22, 2006)

WAGNER, D., REITMAYR, G., MULLONI, A, DRUMMOND, T., SCHMALSTIEG, D. 2008. Real-Time Detection and Tracking for Augmented Reality on Mobile Phones. *IEEE Transactions on Visualization and Computer graphics*, vol 16, no. 3, pp 355-368.

WAGNER, D., SCHMALSTIEG, D., BISCHOF, H. 2009. Multiple Target Detection and Tracking with Guaranteed Framerates on Mobile Phones. *IEEE International Symposium on Mixed and Augmented Reality* (Florida, USA, October 19-22, 2009)