Mobile Visual Computing

(Invited Paper)

Kari Pulli, Wei-Chao Chen, Natasha Gelfand, Radek Grzeszczuk,
Marius Tico, Ramakrishna Vedantham, Xianglin Wang, Yingen Xiong
Nokia Research Center, Palo Alto, CA, USA
firstname.lastname@nokia.com

Abstract

Smart phones are becoming visual computing powerhouses. Using sensors such as camera, GPS, and others, the device can provide a new user interface to the real world, augmenting user's view of the world with additional information and controls. Combining computation with image capture allows new kind of photography that can be more expressive than is possible to obtain with a traditional camera. New APIs allow harnessing more computation power of a smart phone to visual processing than what one can obtain from just a CPU.

1. Introduction

The modern smart phone is a compact visual computing powerhouse. It has a capable CPU, and an increasing number of mobile phones include a graphics processor (GPU). They sport a high-quality color display, up to screen resolutions of 640×480 and beyond. Most high-end phones have a camera that can take megapixel images and at least VGA resolution video. There are separate co-processors or DSPs for image processing and encoding and decoding. Sensors such as GPS, electronic compass, and accelerometers help in determining where the device is and what it points at, and fast data connection allows connection to large databases.

We cover two major application areas, followed by emerging technologies that allow faster visual processing on mobile devices. We call the first area augmented reality. There various sensors, especially the camera, are used to make sense of the world around the user, and the output mechanisms, especially the display, are used to provide user with information about objects and locations around her.

Camera phones are also beginning to replace dedicated digital cameras. However, sometimes the nature of multipurpose device and the requirement of compact size mean compromises in camera elements such as sensor size and lens optics. By taking several images and combining the images using the general computational capacity of a smart phone, one can improve the quality of the images. The improvements could relate to overcoming some sensor limitations, or create new kinds of images that cannot be captured with a traditional camera. This application area is often known as computational photography.

Finally, we describe how the various processing elements available in smart phones can be harnessed for visual computation. The CPU is readily available, and the second generation of mobile GPUs provides programmable vertex and pixel shaders that can be used for image processing. Yet other processing elements such as DSPs are typically not readily available for application programmers; OpenCL allows easier programming and allocation of computation to CPUs, DSPs, and GPUs.

2. Augmented Reality

Augmented Reality (AR) means experiencing the real world and augmenting the experience, often by adding images of virtual objects or textual annotations over the scene. AR can provide a fundamentally better user experience on a mobile system than is possible on desktop. The first task for an AR system is to recognize nearby objects, for example via visual recognition using camera images. If you want to augment the images in real time, you have to also track the view and objects. Neither of these tasks is feasible unless you have a model of the world around you.

2.1. Object recognition

We have built an outdoors augmented reality system for mobile phones that matches camera-phone images against a large database of location-tagged images using a robust image retrieval algorithm [11], see Fig. 1. Matching is performed against a database of highly relevant features, which is continuously updated to reflect changes in the environment. We achieve fast updates and scalability by pruning irrelevant features based on proximity to the user.

Transmission and storage of robust local descriptors are of critical importance in the context of mobile distributed camera networks and large indexing problems. We have proposed a framework for computing a low bit-rate feature that represents gradient histograms as tree structures which can be efficiently compressed [3]. Distances between descriptors can be efficiently computed in their compressed representation, eliminating the need for decoding.

Figure 1. Our outdoors augmented reality system augments the viewfinder with information about the objects it recognizes in the image taken with a phone camera.

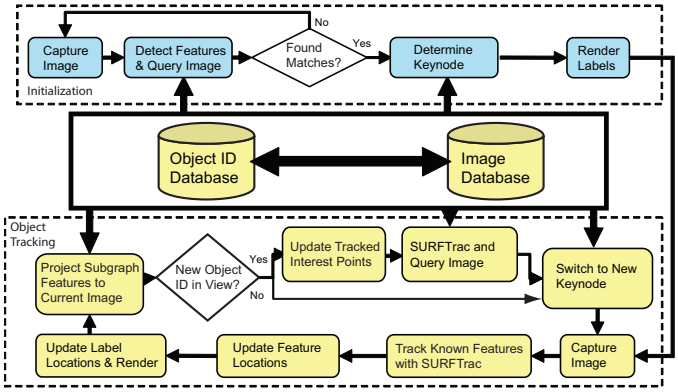

Figure 2. Real-time tracking and recognition pipeline.

2.2. Tracking

For continuous overlaying of text or graphics in AR applications, one can simply repeat the same object recognition step for every input image. However, tracking is much cheaper than repeatedly trying to recognize objects in each frame. Our viewfinder alignment algorithm [1] can track the camera motion in real time on a mobile device, which makes it possible to interleave object recognition and tracking for real-time augmentation. One can also accelerate tracking by using dedicated video encoding hardware and extracting motion vectors [12].

In order to compensate for the longer latency in object recognition, it becomes necessary to pipeline tracking together with object recognition. Load-balancing these two tasks becomes a tricky system design issue. Alternatively, we have developed an algorithm called SURFTrac [10] that tracks with the same features used in object recognition. This means that tracking and object recognition can be performed continuously to achieve natural load balance between the tasks. The SURFTrac pipeline is illustrated in Fig. 2.

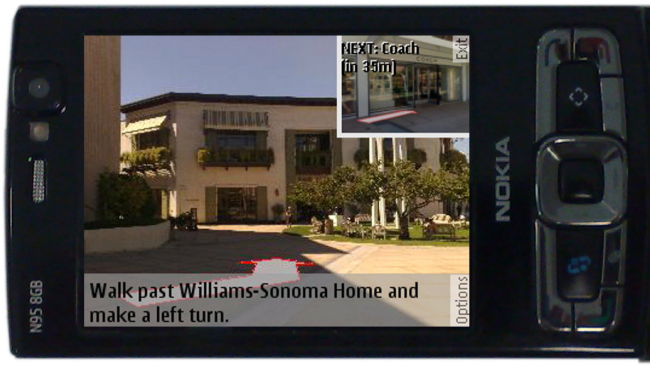

Figure 3. A sample view of a generated instruction. The direction to take is displayed prominently, and a preview of the next step is shown on top corner.

2.3. Scene modeling

Having a 3D model of an environment and being able to register the pose of images taken by a mobile phone user against that model creates new opportunities for richer and more immersive augmented reality applications. In [5], we use computer vision techniques to compute camera poses of image collections of landmarks and register them to the world using GPS information extracted from the image tags. Computed camera poses allow us to augment the images with navigational arrows that fit the environment. We also utilize an image matching pipeline based on robust local descriptors to give users of the system the ability to capture an image and receive navigational instructions overlaid on their current context.

3. Computational Photography

Camera phones are convenient computational photography platforms as they include an increasingly high quality camera together with a general purpose computing device. Here we introduce two computational photography applications, high dynamic range imaging and panorama capture.

3.1. High Dynamic Range Imaging

Dynamic range of a scene is defined as the ratio between the minimum and maximum brightness values present in that scene. Many common scenes have dynamic range that exceeds the maximum range of brightness values that can be recorded by an image sensor, resulting in loss of detail in the shadows, clipped highlights, or both. HDR imaging is a computational technique that combines several regular images taken at different exposures into a single image with expanded dynamic range that better reflects the brightness variations in the scene.

Figure 4. First row: An exposure stack taken at a sculpture garden. Second row, from left to right: single best exposure has areas that are either too bright or too dark, notice the clipped highlights on the building facade and loss of detail in trees and shadows; standard HDR restores those details, but creates ghosts of walking people; consistent HDR image generated by our algorithm.

The basic approach to HDR imaging assumes that nothing in the scene moves, so all images can be used to calculate a radiance value for each pixel. However, many scenes have moving targets, such as people, and it is almost impossible to take an image sequence of a tree in outdoors without any motion of its branches or leaves. Gallo *et al.* [4] developed a method that detects image locations that have changed, selects a suitable anchor view based on which the final, consistent image is created, and uses as much data from other images as possible to calculate accurate radiance values at each pixel (see Fig. 4). In this way, no user interaction is needed to generate an artifact-free high dynamic range image, making it possible for the camera to produce images that more faithfully represent the captured scene, even given the dynamic range limitations of current camera sensors and displays.

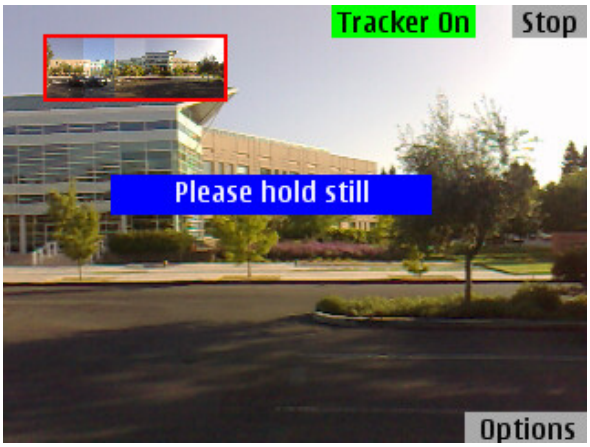

Figure 5. Panorama is automatically captured and constructed as user pans the camera around the scene.

3.2. Capturing panoramas

We have developed a complete mobile panorama application that captures and stitches multiple high resolution images in order to create an image with a larger field of view. We track the camera motion and capture high-resolution images at appropriate times, register the images, map and resample them into spherical coordinates, and blend them into a panorama.

Camera motion is roughly estimated by tracking consecutive viewfinder frames captured at a high frame rate. Our alignment algorithm [1] creates compact summaries of the frames that allow rapid tracking of the camera motion. The high resolution images used to build the final panorama are captured automatically as user pans the scene with the camera. As the camera captures an image, the user is instructed to stop moving to avoid motion blur (see Fig. 5).

Once the high resolution images are captured, they are registered and blended into the final panorama. An unlimited angle of view is enabled by mapping the captured images onto a sphere. We also register images in spherical coordinates. Image-based registration is adopted at the coarse levels of an image pyramid where the features are less reliable, but feature-based matching is employed at the finer levels of the pyramid. This registration approach is robust to both moving objects in the scene and significant illumination differences between images. A seamless image stitching is finally achieved by labeling (selecting which input images

contribute to which areas of the output image [7]) and Poisson blending (taking the gradients of the input images and solving a consistent output image [8]).

4. Standards for Mobile Visual Computing

Images contain lot of pixels, and processing them requires lot of computation power. New APIs, and the hardware they allow application programmers to access, enable faster execution of image processing applications. Here we address mobile 3D graphics standards, and a new standard, OpenCL, for high performance computing even on mobile devices.

4.1. Mobile Graphics APIs

Mobile graphics APIs such as OpenGL ES and M3G [9] enable new types of applications on mobile devices [2]. The obvious ones include faster user interfaces with more eye candy, interactive games, and browsing maps and web pages. However, they are useful also for image processing tasks. For example image warping can be accelerated easily 10-fold with OpenGL ES 1.1 fixed functionality graphics pipeline, even when accounting for the data transfer overhead. Even if the device does not have special graphics hardware, the highly assembly-optimized software graphics engines can typically process the pixels faster than regular image processing C-code. The second generation of graphics hardware and the matching APIs (OpenGL ES 2.0 and M3G 2.0) allow more flexible pixel processing using programmable shaders, increasing the number of algorithms acceleratable via graphics hardware.

4.2. OpenCL

High-end camera phones have much more processing power than just the CPU, but usually the other processing units are not easily accessible by the application programmers. Their units may be dedicated to some particular process, such as the phone modem, voice processing, or video encoding, but those units can be idle when visual processing is needed. Even if the additional units were available for the programmer, they may be difficult to program, each supporting a different instruction set. OpenCL, Open Computing Language, [6] is a new standard that hides the hardware differences under a unified abstraction layer. GPGPU (general processing on graphics processing units) has utilized the GPU instruction sets for other tasks such as image processing. OpenCL generalizes the GPGPU concept as it not only hides the details on which GPU the number crunching takes place, but also whether the processing is distributed to GPUs, CPUs, or even DSPs, providing a unified programming model for all these execution units. OpenCL will make it easier to harness all the power of mobile devices for visual computing.

5. Conclusion

For augmented reality, smart phones now provide computation power, cameras and other sensors, data connectivity, and good displays, and most importantly, are ubiquitous and easy to use when actively moving. The key tasks of near-real-time image recognition and real-time tracking have been demonstrated on smart phones, and modeling the real world to support them in large scale is becoming feasible. Camera phones are also ideal computational photography platforms, providing easy programming access to researchers, and more versatile I/O capabilities than traditional digital cameras. New standards such as OpenGL ES and OpenCL make the varied computing hardware on mobile phones more accessible to application developers.

References

[1] A. Adams, N. Gelfand, and K. Pulli. Viewfinder alignment. *Computer Graphics Forum*, 27(2):597–606, 2008.

[2] T. Capin, K. Pulli, and T. Akenine-Möller. The state of the art in mobile graphics research. *IEEE Computer Graphics and Applications*, Jul-Aug 2008.

[3] V. Chandrasekhar, G. Takacs, D. Chen, S. S. Tsai, R. Grzeszczuk, and B. Girod. CHoG: Compressed Histogram of Gradients: A Low Bit-Rate Feature Descriptor. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR09)*, 2009.

[4] O. Gallo, N. Gelfand, W.-C. Chen, M. Tico, and K. Pulli. Artifact-free high dynamic range imaging. In *IEEE Int. Conf. on Computational Photography (ICCP09)*, 2009.

[5] H. Hile, R. Grzeszczuk, A. Liu, R. Vedantham, J. Košecka, and G. Borriello. Landmark-Based Pedestrian Navigation with Enhanced Spatial Reasoning. In *7th International Conference on Pervasive Computing*. Springer, 2009.

[6] Khronos Group. The OpenCL Specification, Version 1.0, 2008. Available from `www.khronos.org/registry/cl/specs/opencl-1.0.33.pdf`.

[7] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick. Graphcut Textures: Image and Video Synthesis Using Graph Cuts. *SIGGRAPH*, 2003.

[8] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Transactions on Graphics (SIGGRAPH '03)*, pages 313–318, 2003.

[9] K. Pulli, T. Aarnio, V. Miettinen, K. Roimela, and J. Vaarala. *"Mobile 3D Graphics with OpenGL ES and M3G"*. Morgan Kauffman, 2007.

[10] D.-N. Ta, W.-C. Chen, N. Gelfand, and K. Pulli. SURFTrac: Efficient Tracking and Continuous Object Recognition using Local Feature Descriptors. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR09)*, 2009.

[11] G. Takacs, V. Chandrasekhar, N. Gelfand, Y. Xiong, W.-C. Chen, T. Bismpigiannis, R. Grzeszczuk, K. Pulli, and B. Girod. Outdoors Augmented Reality on Mobile Phone using Loxel-Based Visual Feature Organization. In *MIR '08: Proceeding of the 1st ACM International Conference on Multimedia Information Retrieval*, pages 427–434, 2008.

[12] G. Takacs, V. Chandrasekhar, B. Girod, and R. Grzeszczuk. Feature Tracking for Mobile Augmented Reality Using Video Coder Motion Vectors. In *ISMAR '07: Proceedings of the Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007.