New Video Applications on Mobile Communication Devices

Olli Silvén\textsuperscript{a}, Jari Hannusela\textsuperscript{a}, Jani Boutellier\textsuperscript{a}, Miguel Bordallo-López\textsuperscript{a}, Markus Turttinen\textsuperscript{b}, Matti Niskanen\textsuperscript{b}, Marius Tico\textsuperscript{c} and Markku Vehviläinen\textsuperscript{c}

\textsuperscript{a}Machine Vision Group, University of Oulu, Oulu, Finland
\textsuperscript{b}Visidon Ltd, Oulu, Finland
\textsuperscript{c}Nokia Research Center, Tampere, Finland

ABSTRACT

The video applications on mobile communication devices have usually been designed for content creation, access, and replay. For instance, many recent mobile devices replicate the functionalities of portable video cameras and video recorders, and digital TV receivers. These are all demanding uses, but nothing new from the consumer point of view. However, many of the current devices have two cameras built in, one for capturing high resolution images, and the other for lower, typically VGA (640x480 pixels) resolution video telephony. We employ video to enable new applications and describe four actual solutions implemented on mobile communication devices. The first one is a real-time motion based user interface that can be used for browsing large images or documents such as maps on small screens. The motion information is extracted from the image sequence captured by the camera. The second solution is a real-time panorama builder, while the third one assembles document panoramas, both from individual video frames. The fourth solution is a real-time face and eye detector that can be used with auto-focusing and red eye reduction techniques. It also provides another type of foundation for motion based user interfaces as knowledge of presence and motion of a human faces in the view of the camera can be a powerful application enabler.

Keywords: User interaction, camera, motion analysis, panorama construction, face detection

1. INTRODUCTION

Modern mobile communication devices are becoming attractive platforms for multimedia applications as their display and imaging capabilities are improving together with the computational resources. Many of the devices have two built-in cameras, one for high resolution still and video imaging, and the other for obtaining lower, e.g. VGA resolution (640x480 pixels) frames. Table 1 points out the versatility of user interfaces of handheld devices in comparison to lap top computers.

To illustrate the current typical designs we see two modern cellular phone designs in Fig. 1 below, both with two cameras. The flip phone has a high resolution camera on the cover, on the same side with a display. However, it is intended to be operated with the lid open, exposing a higher resolution display and an additional camera to the user. The monoblock design is similar to digital still cameras with the high resolution display and cameras on opposite sides. It is obvious that with the display side cameras the designers have aimed at hand held video telephony, while at the same time satisfying the needs for occasional photography, video capture, and playback.

The usability of mobile communications devices in portable imaging applications is on par with lap-top computers despite the order of magnitude disparity between the computing power budgets. The sizes and semi-dedicated user interfaces of the hand-held devices are significant benefits over the general purpose personal computer technology based platforms, despite their apparent versatility. On the other hand, even the most recent mobile communication devices have not used their multimedia and computing resources in a novel manner, but are merely replicating the functionalities already provided by other portable devices, such as digital still and video cameras. Also the popularity of lap top PCs as portable DVD players, and as a means to access multimedia content via public WiFi networks, have clearly influenced the hand held application designs. Consequently, most of the hand held devices rely on keypad-and-pointer user interfaces, while their applications rely on content
Table 1. Characteristics of typical laptop computers and recent handheld mobile devices.

<table>
<thead>
<tr>
<th></th>
<th>Lap top computer</th>
<th>Handheld device</th>
<th>Typical ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Still image resolutions</td>
<td>up to 1 Mpixel</td>
<td>up to 352x288–1944x2582</td>
<td>0.33x</td>
</tr>
<tr>
<td>Number of displays</td>
<td>1</td>
<td>2</td>
<td>0.5x</td>
</tr>
<tr>
<td>Number of cameras</td>
<td>0–1</td>
<td>1–2</td>
<td>0.5x</td>
</tr>
<tr>
<td>Video resolution</td>
<td>720x576/25Hz</td>
<td>640x480/30Hz</td>
<td>1x</td>
</tr>
<tr>
<td>Display size (inches)</td>
<td>12–15</td>
<td>2–4</td>
<td>5x (area 20x)</td>
</tr>
<tr>
<td>Processor clock (GHz)</td>
<td>1–3</td>
<td>0.3–0.5</td>
<td>10x</td>
</tr>
<tr>
<td>Display resolution (pixels)</td>
<td>1024x768–1600x1200</td>
<td>176x208–800x352</td>
<td>15x</td>
</tr>
<tr>
<td>Processor DRAM (MB)</td>
<td>256–2044</td>
<td>64–256</td>
<td>16x</td>
</tr>
</tbody>
</table>

As more and more applications are being crammed into hand held devices, their limited keypads and small displays are becoming too overloaded, potentially confusing the user who needs to learn to use each individual application. Based on the personal experiences of most people, increasing the number of buttons as with remote control units is not the best solution from the usability point of view. The full keyboard, touchpad or mouse, and higher resolution displays of laptop PCs appear to give them clear benefits as platforms for multiple simultaneous applications. However, the size of the hand held devices is an under-exploited asset as well as their multiple cameras. Properly combined these characteristics can be used for novel user interfaces and applications that may not make much sense on laptop computers.

In this paper we show how image sequences captured by the cameras of mobile communications devices can be provided via the Internet or broadcast services such as DVB-H, to supplement locally stored music, movies, and maps. Although the users can create content and stream it to the network for redistribution, and make video calls, these uses are not very common.

![Figure 1. Two mobile communications device with two cameras (Nokia 6290 and Nokia N73).](image-url)
be used for new self intuitive applications and user interface concepts. The key ideas rest on the utilization of the hand held nature of the equipment and the user being in the field of view of a camera. Four actual implementations are described, all running on multimedia capable cellular phone platforms.

The first solution is a real-time motion based user interface that can be used for browsing large images or documents such as maps on small screens. The motion information of the device itself, face, or the hand of the user is extracted from the image sequence. The second solution is a real-time panorama builder, while the third one assembles document panoramas, both from individual video frames based on the motion information. The fourth solution is a real-time face and eye detector that can be used with auto-focusing and red eye reduction techniques, essentially providing a basis for user interfaces that are aware of the presence human faces and the direction of the gaze. When combined, face or limb, and motion information can be powerful application enablers and may change the expectations on how hand held devices are supposed to be aware of the user and react to his actions.

In the following we initially describe the typical platform characteristics of the mobile communication devices, and then proceed into the video based user interface and application solutions. The limitations and the potentials of the realizations are analysed against the current state-of-the-art. Finally, the desirable future platform developments are considered from the camera based user interface and application point of view.

2. MOBILE COMMUNICATIONS DEVICE PLATFORMS

A typical top level hardware organization of a mobile communications device with multimedia capability is shown in Fig. 2. Two main interconnects are used for transfers between system units to that are partitioned to avoid bottlenecks. The Interconnect 2 provides for high transfer bandwidths between the cameras, memories, and the video and image processing unit that in turn has high speed local buses between its internal subsystems. Interconnect 1 is the system bus that interfaces the essential units with the master application processor.

![Figure 2. Organization of portable multimedia device.](image)

We notice that the application processor has rapid access to the data produced by the camera, so the design is not a simple replacement for a camcorder. Instead, potential for software based image and video applications has been engineered into the system architecture. Transfer resources- and power, can be conserved if the camera images need not to be shown. In contrast, camcorder mode is rather transfer intensive as it requires real-time encoding and display functions. Video calls are even more complicated as they require the simultaneous operation of video encoder, display, and decoder together with both uplink and downlink streaming via the baseband unit.

The power budgets of the mobile devices are designed and optimized based on the worst case use cases. Video applications in their various forms are among the most demanding ones as the users tend to demand long, at least 3-4 hour active use times from small devices without connecting to the electrical mains network for recharging the battery. As the capacity of the batteries depend on the discharge current in a non-linear manner, relatively small cuts in power consumption can significantly extend the battery life. Consequently, the manufacturers are tempted to employ hardware accelerators for video due to their power efficiency. For the builders of alternative video based applications this can be a blessing, as well as the availability of graphics processors.

Table 2 presents power breakdowns of three devices in video playback mode to illustrate the impacts of design philosophies. The PDA device can be characterized as a scaled down personal computer. The early 3G
mobile phone is mostly a communications device with an application processor, while the future device adds hardware based video and a GPU to achieve a 3h battery life that is often considered critical in entertainment use. In both cellular devices the modem and the RF consume a major part of the power budget. Although it is tempting to expect improvements, the future air interfaces are unlikely use less power as the increasing data rates and multiple radio protocols are adding to their complexity. Also the miniaturization of RF-components may actually decrease their power efficiency.

Table 2. Power consumption breakdown examples of pocket sized devices.

<table>
<thead>
<tr>
<th>System component</th>
<th>Power consumption (mW)</th>
<th>3G phone in video streaming mode</th>
<th>PDA device in MPEG-4 playback</th>
<th>Expected future mobile devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application processor and memories</td>
<td>600</td>
<td>833</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Display, audio, keyboard and backlights (UI)</td>
<td>1000</td>
<td>2441</td>
<td>400</td>
<td></td>
</tr>
<tr>
<td>Misc. memories</td>
<td>200</td>
<td>754</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>RF and cellular modem</td>
<td>1200</td>
<td>N/A</td>
<td>1200</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>3000</td>
<td>4028</td>
<td>1800</td>
<td></td>
</tr>
<tr>
<td>Battery capacity mAh/usage time</td>
<td>1000mA/1h</td>
<td>N/A</td>
<td>1500mA/3h</td>
<td></td>
</tr>
</tbody>
</table>

So far a key to improved power efficiency has been in augmenting software solutions with hardware or Application Specific Instruction-set Processor (ASIP) support for computing intensive tasks. On the other hand, enabling the applications to access the services of these subsystems requires system software architectures and interfaces that hide the actual nature of the implementations, and the differences between the platforms of product generations. These benefits are not achieved without overheads, but they can be optimized for the typical expected uses such as streaming video playback. However, any novel uses of the system resources are likely to encounter the full interface costs.

Table 3 presents the estimated power costs of using the camera for user interface purposes in a mobile device. Three options are considered: first, the only computing resource is the application processor that needs to run at full speed, second, the system hardware resources such as the GPU and parts of the video codec can be re-used to conserve power, while the third option is a conventional keypad and display user interface. Obviously, with proper system design the cost of a camera based user interface is reasonable.

Table 3. Estimated power costs of using the camera for user interface purposes.

<table>
<thead>
<tr>
<th></th>
<th>Software based camera UI [mW]</th>
<th>Hardware for camera UI [mW]</th>
<th>Conventional UI [mW]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application processor and memories</td>
<td>600</td>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td>Display, audio, keypad, backlights</td>
<td>400</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>Camera (VGA)</td>
<td>50</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>Misc. memories</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Total</td>
<td>1150</td>
<td>750</td>
<td>650</td>
</tr>
</tbody>
</table>

As the usability of a user interface critically rests on its latency. This is most obvious with computer games in which many perceive joystick action-to-display delays exceeding about 100-150 milliseconds disturbing, but this applies even to key press-to-sound or display. If we employ a camera as a user interface component, its integration time will add to the latency, as well as the image analysis computing. If we sample the scene at 15 frames/s rate, our base latency is 67 ms. Assuming that the integration time is 33 ms, the information in
the pixels read from the camera are on an average 17 ms old. Consequently, with computing and display/audio latencies achieving a total below 100-150 ms is a challenge.

The table 4 summarizes the latency budget for two frame rates, 15 and 30 frames/s that are typical for mobile devices. The integration time of the camera is assumed to be 33 ms at both rates. If the computing is done in a pipeline that contains more than a single processor, the image analysis may effectively be longer than the base latency. Interestingly, at the lower frame rate less time is available, on the other hand, the camera operated at a higher rate demands more power.

<table>
<thead>
<tr>
<th>Component</th>
<th>Rate (frames/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base latency (ms)</td>
<td>67</td>
</tr>
<tr>
<td>50 % camera integration time (ms)</td>
<td>17</td>
</tr>
<tr>
<td>Display latency (ms)</td>
<td>20</td>
</tr>
<tr>
<td>Image analysis max (ms)</td>
<td>46</td>
</tr>
<tr>
<td>Total max (ms)</td>
<td>150</td>
</tr>
</tbody>
</table>

An obstacle to camera based user interfaces into is the turn-on time that is not only dependent on the power-up delay of the camera, but is mostly caused by software. The current multimedia frameworks intended for use on mobile platforms have substantial latencies, when the resources are reconfigured for the applications. For instance, Rintalouma et al. found that the Symbian MMF (MultiMedia Framework) consumed approximately 60000 processor cycles for accessing a device driver from the application layer. The OpenMAX is claimed to be a lighter weight interface to use streaming multimedia hardware and software components. However, it has not yet been intended for UI purposes. For that purpose we may find some models on how to proceed from OpenGL ES that and has been designed with an eye on game applications. It is a highly optimized graphics system designed for accelerators used on embedded and mobile devices.

In addition to user interfaces, the applications described next provide insight into the computing needs and characteristics of camera based user interfaces. If cameras become standard UI components in mobile devices, energy efficiency requires that the bulk of the computing is carried out using hardware acceleration. These resources can an outgrowth of the current graphics or codec solutions, or both.

### 3. MOTION BASED USER INTERFACE

The motion based user interface enables a new flexible way to interact with mobile phones. With this interface, the user can operate the phone through a series of hand movements whilst holding the device. During these movements the motion is extracted from the image sequence captured by the camera. As an application example, the solution has been implemented on Nokia N series mobile phones allowing the user to browse large image documents on small screens as shown in Fig. 3.

In the application, only a small part of the high resolution image is visible at a time (See Fig. 3 b) and the measured motion information is used as a control input (See Fig. 3 a). For instance, the lateral movement upwards scrolls the focus towards the upper part of the display, and back and forth motion is interpreted as zooming in and out. The rotation component is utilized to change the orientation of the display. In practise, the user can also tilt the device in order to navigate over the display, which is more natural and convenient way to control the device. A typical usecase example is illustrated in Fig. 4. There user browses the large image on the small screen of the mobile device by moving the device in his hand.

We estimate the ego-motion of the device while the user operates the phone by determining the parametric model, which approximates the dominant global motion between two images in the sequence captured by camera. Our approach utilises the feature based motion analysis where a sparse set of blocks are first selected from one image and then their displacements are determined. In order to improve accuracy of the motion information, an uncertainty of these features is also analysed.
Figure 3. Motion based user interface estimates the motion of the device relative to the user enabling also zooming functionalities (a). It can be used, for example, to browse large image documents on the screen (b).

The main steps of motion estimation are presented in Fig. 5. For details, please see the paper of Hannuksela et al. Blocks in the top left present the selected image regions to be used. Lines in the top right image illustrate the block displacement estimates, $d$, and ellipses show the related uncertainties. The bottom left image shows the trusted features that are used for parametric model fitting. In this case, the ellipses illustrate the weight that a particular displacement estimate has in the fitting. By combining the feature selection with uncertainty information we obtain very robust motion estimate of the sequence. This information can be directly used to estimate the motion of the device in the user’s hand.

We have implemented our method using only fixed-point arithmetic due to the lack of floating-point unit in most of current devices. The use of integer operations in the inner loops guarantees the high real-time performance of the method. Our solution can also take advantage from the hardware acceleration used with other video processing applications. Acceleration hardware are designed to support the block-based and pixel-level processing tasks that are not efficiently handled by the CPU architecture. Typically such hardware contains highly optimised motion estimation instructions on blocks from 16x16 to 4x4 pixels which are also usual sizes for blocks in our method.
4. PANORAMA BUILDER

The panorama building solution analyses the frames in the video for motion and moving objects, quantifies the quality of each frame, and stitches up to 360 degree panoramas from the best available images. We have developed a method where the devices are able to stitch images in real time obtaining a result image that is growing with the frame acquisition. Three examples are shown in Fig. 6.

![Image of panorama builder results](image)

Figure 6. Efficient panorama builder stitches high quality images even if there are moving objects in the scene.

The panorama capturing procedure is illustrated in Fig. 7. In order to get a final panorama image, the user focuses the camera to the desired starting point of the mosaic. The camera starts turning around up to 360 degrees and a sequence of images starts to be captured. Each image is then individually processed to estimate the shift and rotation. The blurriness of each picture is measured and moving objects are identified. Based on the quality of each individual frame, a selection process takes place. The idea of selection is to consider only
good quality frames for creating the best possible output. The selection process is shown in Fig. 8. Each frame is either accepted or discarded. For every selected frame, if a moving object is present and it fits the sub image, the image is blended drawing a seam that is outside the boundaries the object. If only a partial object is present, the part of the frame without the object is the one blended.

Figure 7. During the panorama capturing the user first focuses the device to the desired direction and then turns around in order to create panorama of the view.

Figure 8. Automatic frame selection based on blur and motion estimation assures that only the best quality frames are considered in the panorama construction.

Image registration relies on the method of Vandewalle at al.\textsuperscript{7} that offers shift and rotation estimation being robust against blur. Only a fixed square template on the central part of each frame, where the image quality is better, is used. This square is downscaled by a factor of two and filtered to allow faster performance, interpolating then the results of the registration estimation.

The amount of motion blur in the frame is computed with summed derivatives.\textsuperscript{8} The method estimates the image’s sharpness by summing together the derivatives of each row and each column of the overlapping part. Blur
calculation produces one single number that expresses the amount of high-frequency detail in the image. The value is sensible if it is used to compare images: if a certain image \( I_a \) acquires a higher result than image \( I_b \), it means that \( I_a \) has more high-frequency detail than image \( I_b \) (implying that both images depict approximately the same scene). Usually this means that \( I_a \) is sharper than image \( I_b \), but in some occasions the difference in the image content distorts the result.

To perform motion detection, the difference between the current frame and the previous frame is computed. The result is a two-dimensional matrix that covers the overlapping area of the two frames. Then, this matrix is low-pass filtered to remove noise and is thresholded against a fixed value to produce a binary motion map. If the binary image contains a sufficient amount of pixels that are classified as motion, the dimensions of the assumed moving object are determined statistically. First, the centerpoint of the object is approximated by computing the average coordinates of all moving pixels. Second, the standard deviation of coordinates is used to approximate the dimensions of the object.

Frame selection is done based on the score of the blur measurements and the motion detection. Among the set of images that present an overlap with the previous blended frame, only the best frame is selected, while the others are discarded. The frame blending is done with feathering method,\(^9\) where a linear function gradually merges one frame to the next by changing the frames’ weight.

The application has been implemented using only fixed-point arithmetic that guarantees real time performance on most of devices. The solution can also take advantage of the acceleration hardware by using the graphic processing unit if it is present. The mean processing time for a Nokia N95 (ARM 11) is about 8 frames per second (125ms/frame), with 320x240 frames. Other configurations on the frame resolution (from 160x120 to 640x480) can be chosen and no theoretical limits are present in the final size.

5. DOCUMENT PANORAMA BUILDER

Document panorama builder is essentially a camera based scanner as shown in Fig. 9. Instead of using devices such as flatbed scanners, the users can capture a high quality images with their mobile phones. Mobile cameras enable portable and non-contact image capture of any kinds of documents. Although they cannot replace flatbed scanners, they are more suitable for several scanning tasks in less constrained situations.

![Figure 9. Mobile device can be used as a camera based document scanner.](image)

We have developed a method\(^{10}\) where the device interactively guides the user to move the device over, for example, a newspaper page in a manner that a high quality image can be assembled from individual video frames. During online scanning, motion determined from low-resolution image sequences is used to control the interaction process. As a result, good high-resolution images of the document page can be captured for stitching. Images with coarse alignment information are used to construct a mosaic automatically using a feature based alignment method.

In the first stage, partial images of the document are captured with the help of user interaction. The basic idea is to apply online camera motion estimation to the mobile phone to assist the user in the image scanning
process. The user starts the scanning by taking a high resolution image of some part of the document (see Fig. 10 a). Then, the user is asked to move the device to the next location. The scanning direction is not restricted. One possible way is to use zig-zag style scanning path shown in Fig. 10 a. The camera motion is estimated during movement and the user is informed when the suitable overlap between images is achieved (for example 25 %). When the device motion is small enough, a new high resolution image is taken. The movement should be stopped because otherwise images are blurred.

In order to measure device motion the same principle as the one used in Sec. 3 is utilized. These estimates are then used for computing cumulative displacement estimates. The requirement here is that the error in this estimate does not become too high, so that sufficient overlap between stored images is guaranteed.

![Figure 10. An example use case is illustrated for building a large map image document. (a) The user makes zig-zag style scanning (b) mosaic obtained.](image)

After online image capturing, the partial high resolution images of the document page can be stitched together. The automatic mosaicing is based on robust estimator (RANSAC\textsuperscript{11}) with a feature point detector (SIFT\textsuperscript{12}). Also, graph based global alignment and bundle adjustment steps are performed in order to minimize image registration errors and to further improve quality. Finally, warped images are blended to the mosaic using simple Gaussian weighting. Fig. 10 b) illustrates the mosaic (1397x1099 pixels) constructed for an A4 document page from eight VGA images. Instead of using raw VGA frames as input, the device must be switched to the high resolution mode which has relatively large latency and is a platform limitation.

6. FACE AND EYE DETECTOR

Our object detection approach is based on efficient gray scale invariant texture features and boosting.\textsuperscript{13,14} The AdaBoost algorithm is used to learn the most discriminative combination of the weak texture features to separate the desired object category from the background.\textsuperscript{15} The learning is made off-line using the set of labelled training images for building a classification tree. In our case the training images are either faces or eyes. In the detection phase, the image pyramid is constructed. At each pyramid stage, the input image patches with the known template size are classified with the trained tree classifier. The outputs are the object coordinates and the level of detection confidence. This information can be directly used by face based approaches, such as auto focusing or color enhancement as illustrated in Fig. 11. From the technical point of view, faces and eyes are also important feature sources: the camera directed towards the user (as in the models shown in Fig. 1) is usually adjusted so that the field of view contains the face region very well. This has great advantage in various HCI solutions.
The algorithm was implemented in the Symbian OS to detect and track maximum of five objects (faces) in the images. The demo software processes the QVGA view finder images with the high frame rate and the detection performance is very good also in demanding illumination conditions. The minimum face size was set to $20 \times 20$ pixels. We run simulation experiments with the RealView Development Suite using the 180MHz ARM9 CPU and the mean processing time with various image sequences is 68ms/frame (about 15 fps). The face detection Symbian demo running on Nokia N95 (ARM11) runs even faster (approximately 26-30 fps). This is fast enough for real-time auto focusing or white balancing applications, for example.

Fig. 12 shows some example detection results. Input QVGA frames are captured by Nokia N95. The algorithm is robust in varying illumination conditions and detects faces of different sizes and poses very efficiently.

In addition to face based auto focusing or auto white balancing solutions, robust face detection provides numerous other applications on mobile platforms. For example, the knowledge of presence and motion of a human faces in the view of the camera can be a powerful application enabler or provide tools for creating new innovative mobile HCI solutions, such as auto-rotation. One example application is shown in Fig. 13 where the face detection is combined with the motion based UI. The user can give motion gesture command to switch the face in the screen.

In addition to the red-eye removal, efficient face and eye detection has also other applications in mobile devices. One example is the gaze recognition, where the relative position of the user face and the gaze direction with respect to the device are estimated. In this area, some very innovative HCI solutions can be expected in the near future.
7. VIDEO APPLICATIONS INFRA-SUPPORT

The development of the demonstration applications has contributed to the identification of features that in mobile platforms would benefit alternative video applications and camera based user interfaces. Both software and hardware developments are needed to remove the latency and computing bottlenecks.

First, it should be possible to use two cameras at the same tie or quickly alternate between cameras as image sources. The motivation for this capability is a practical one: sunlight, lamps, or reflections may saturate one of the cameras, so the trivial automatic adaptation method is to switch to another image source, although may be a more power consuming high resolution device. However, the current mobile devices have single camera interfaces, and alternating between cameras requires reconfiguration that may take hundreds of milliseconds.

Second, a stand-by mode for the cameras should be exist, perhaps initiated by the handling of the device recognized by built-in accelerometers, to reduce the start-up latency of the vision based user interfaces. In the stand-by mode the camera could capture images, say, at the rate of a frame per second, adjusting to the ambient illumination. The cold start power-up latencies of the camera hardware modules alone are around 100ms. At least two images are needed to determine the first motion estimates even if no gain correction is needed to bring the image information into the useful dynamic region. These plain hardware dependent delays in total amount to 150-200ms, but would be only 50-100ms from stand-by.

Third, the data formats of the camera and GPU/display units should be compatible as for a number of image processing functions, such as interpolations and warps, it is desirable to use the GPU as a hardware accelerator. The OpenGL interface is highly efficient, but the necessary format changes result in needless copying of data, resulting in reduced energy efficiency, increased computational burden, and latency.

8. SUMMARY

In the described new video applications the cameras of mobile devices are employed as motion and feature sensors in user interface and imaging solutions. Augmenting the information provided by accelerometers and touchscreens in a complementary manner, there is no doubt on the potential of the cameras. In fact, with future mobile devices they may for most of the time be used for other purposes than for taking pictures for the human consumption.

However, the cameras on mobile device platforms are a rather recent add-on, primarily intended for capturing still and video frames, while the computing and display resources have been optimized for video playback.
a result, the energy efficiency of other uses is inferior. This may not essentially limit the practical utility of the presented applications, but the camera can be used as an "always-on" sensor.

On the other hand, the power consumption of a VGA camera can be pushed to around 1mA/frame/s. Coupled with optimized computing solutions the stand-by (1 frame/s) needs of a camera based user interface could run at 2-4mA. This is in the same range with the stand-by currents of cellular phones.

REFERENCES