Point-of-Capture Archiving and Editing of Personal Experiences from a Mobile Device

Chon-in Wu[a], Chao-ming (James) Teng[b], Yi-chao Chen[a], Tung-yun Lin[a], Hao-hua Chu[a], Jane Yun-jen Hsu[a]

[a] Department of Computer Science and Information Engineering & Graduate Institute of Networking Multimedia, National Taiwan University
Email: {r92079, b89902066, b90209014, hchu, yjhsu} @csie.ntu.edu.tw

[b] MIT Media Lab
Email: JTeng@media.mit.edu

ABSTRACT

Personal experience computing is an emerging research area in computing support for capturing, archiving, editing, and utilizing of digital personal experiences. This paper presents our design, implementation, and evaluation of a mobile authoring tool called *mProducer* that enables everyday users to effectively and efficiently perform archiving and editing *at or immediately after the point-of-capture* of digital personal experiences from their camera-equipped mobile devices. This point-of-capture capability is crucial to enable *immediate sharing* of digital personal experiences anytime, anywhere. For example, we have seen everyday people who used handheld camcorders to capture and report their personal, eye-witnessed experiences during the September 11 event. With mProducer, they would be able to perform editing immediately after the point of capture, and then share these news-worthy, time-sensitive digital experiences on the Internet. To address the challenges in both user interface constraints and limited system resources on a mobile device, mProducer provides the following innovative system techniques and UI designs. (1) *Keyframe-based editing UI* enables everyday users to easily and efficiently edit recorded digital experiences from a mobile device using only key frames with the *storyboard metaphor*. (2) *Storage constrained uploading (SCU) algorithm* archives continuous multimedia data by uploading them to remote storage servers at the point-of-capture, so that it alleviates the problem of limited storage on a mobile device. (3) *Sensor-assisted automated editing* uses data from a GPS receiver and a tilt sensor attached to a mobile device to facilitate two manual editing steps at the point-of-capture: removal of blurry frames from hand-induced camera shaking, and content search via location-based content management. We have conducted user studies to evaluate mProducer. Results from the user studies have shown that mProducer scores high in user satisfaction in editing experience, editing quality, task performance time, ease-of-use, and ease-of-learning.

1. INTRODUCTION

Recent advances in multimedia hardware manufacturing technologies have led to a vibrant consumer market of affordable camera-equipped mobile devices such as: smart phones, PDAs, digital cameras, and handheld camcorders. For many of us, these devices have become ubiquitous as parts of our everyday inseparable items. Their great market success can be attributed to convenience brought by the combination of a digital camera and a communication radio within one small mobile device. This convenient combination is ideal for everyday people to explore its content capturing capability. Its benefits also include the communication capability to share and distribute these recorded personal experiences on the Internet, and the ability to record everyday personal experiences, such as what we see, where we go, whom we meet, etc. Everyday people can become *content producers* of our own personal experiences, not restricted to the traditional role of mostly consumers of mass media content. The ideas that "everyone can be a content producer" and "everyone has a content-producing mobile device" are expected to bring a fundamental change in the type of future digital contents, and how future digital contents will be produced and consumed. These ideas have been demonstrated with the recent popularity of web blogging [2]. As shown in Table 1, content consumers can access digital contents produced not only from professional news and media reporters, but from ordinary everyday people. At the same time, ordinary people who have different skill levels in digital content production can produce and contribute their own personal experiences using these simple, mobile, ubiquitous, content-capturing, and communication devices.

Table 1. Traditional Content vs. Personal Experience Content

	Types	Producers	Devices & Editing Tools
Traditional content	Mass media contents	Professional content providers	PCs & professional content producing tools
Personal experience content	Personal experience contents	Everyday people	Smart phones & mProducer

Accompanying this vision of digital personal experience is an emerging research area called *personal experience computing* [10] [19]. This area is about computing support for recording, storing, sharing, and re-visiting of their personal or group experiences. We can categorize this research area into four major phases of digital content production and utilization as shown in Figure 1. (1) The personal experiences are first captured as digital content using cameras on mobile devices. (2) The digital content is then stored and archived on mobile or remote server storage. (3) The raw digital content is then retrieved from storage for editing. (4) The edited digital content is then utilized in various applications such as: sharing with friends and family, revisiting past memories, etc.

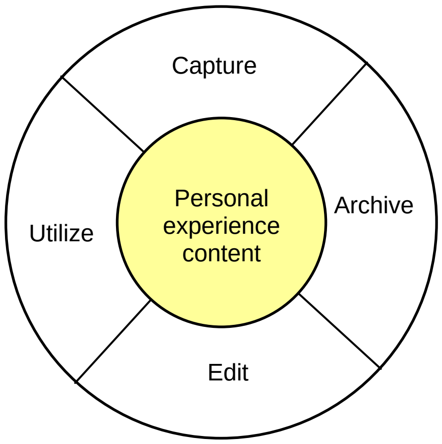


Figure 1. Content Lifecycle for Personal Experience Content

The mProducer addresses the first three phases of digital content production – capturing, archiving, and editing. The objective of mProducer is to realize so called *point-of-capture* archiving and editing from a mobile device. Point-of-capture means that as soon as personal experiences are captured from mobile devices, users can archive and edit them from the same mobile devices that are used for capturing. This allows users to publish or share the edited digital contents *immediately* after the point of capture. In comparison to the existing PC-based editing approach, this mobile editing functionality can eliminate the *production delay* between content capturing and publishing. In the existing PC-based editing approach, the raw digital content has to be transferred from a mobile device to a PC for archiving purposes and then edited using software such as Apple iMovie^{TM} or Microsoft MovieMaker^{TM}. In general, most users do not carry PCs in the mobile environment; thus, content archival and editing are usually delayed until users return home or to their offices. This could possibly be many hours after the time of content capturing. This production delay can reduce the value of personal experiences that are *time sensitive*, meaning that audience interest in the content decreases with the time delay. For example, there were many eye-witness reports of the September 11 tragedy captured by everyday people using their digital cameras or camcorders. These first-hour on-site reports carried more value to audiences than those news reports that ranged from a few hours to a few days old. In order to reduce the time it takes for the content recorded by producers to be edited and finally reach the audience, *point-of-capture mobile editing* is required. An additional benefit of point-of-capture mobile editing is that it allows a user to operate a single device during the entire content production lifecycle (capturing, editing, and publishing), therefore, saving user effort in transferring content between devices. Furthermore, wireless technologies (3G or Wireless LAN) with MMS (Multimedia Messaging Services) can make it easy for people to share their digital experiences from mobile devices immediately after editing. However, to our best knowledge, we have not seen any point-of-capture mobile editing tools on smart phones or PDAs. Without them, users have to either share all captured contents without any editing (such as removing unwanted or blurry frames) or carry a relatively heavy (in comparison to mobile phones) laptop computer to accomplish the editing of the captured content. These two options are unsatisfactory and inconvenient to users. This motivates us to create mProducer, a point-of-capture mobile editing tool.

In order to meet the objective of enabling everyday people to produce their personal digital experience content at the point-of-capture from their mobile devices, the design of mProducer considers the following mobile challenges:

- *Specialized Editing User Interface:* Small screen space, inconvenient input methods, limited mobile user attention, and average consumers with little computing experience require a different interaction model and user interface design, where simplicity, ease-of-use, and ease-of-learning are as important as the final quality of edited content.
- *Limited Storage for Archiving:* Mobile devices have very limited storage space that restricts the length of recordings that a user can capture.
- *Limited Computing Resource:* Most image/video processing techniques for media editing are computationally intensive and demand the high computational power of PCs. They are beyond the limited computing resources on a mobile device.

Our contributions in mProducer include the following innovative solutions to address the challenges described above:

(1) *Mobile editing user interface:* The major component of the mProducer design for the editing UI is the *keyframe-based editing*. A key frame is defined as a video frame that best represents a shot or scene, i.e., a user can get a good understanding of what a shot is about by viewing its key frames[14]. Our user studies in section 6 have shown that casual, everyday users can edit video clips using only key frames, and at the same time, produce satisfactory editing quality for the purpose of sharing personal experiences. In addition, the results of user studies (described in section 6) have shown that keyframe-based editing allows casual users to perform basic editing functions (merging or deleting of video content, text annotation, etc.) on average twice more efficient than traditional frame-by-frame editing. Another component in the mProducer UI design is the *Location-based Content Management*. When mProducer is used for recording personal experiences at multiple locations (e.g., a trip covering multiple sightseeing destinations), we have found through interviews with users that they can better mentally organize these experiences based on recording locations rather than recording times. To match users' *location-based mental model*, a simple, intuitive, map-based content management interface is designed to enable easy navigation and browsing of media clips. A GPS receiver on a mobile device is then used to record location meta-data for each recording captured by a user.

(2) *Storage Constrained Uploading (SCU):* We leverage keyframe-only editing to solve the challenge of limited mobile storage. When a mobile device is running out of storage space, we have designed a *storage constrained uploading* (SCU) algorithm to selectively upload non-key frames to a remote storage server. Uploading allows more contents to be captured on a mobile device [23].Given that users can edit video clips using only key frames, uploading and removing non-key frames from the mobile device during content capturing will not affect the quality of keyframe-only editing described in (1).

(3) *Sensor-assisted Automated Editing:* Traditional PC-based video editing tools rely on image-based processing methods to extract context meta-data, which is in turn being used to semi-automate raw content editing so that the amount of user effort can be reduced. Examples of these techniques include object recognition, location determination [21], and shaking artifacts removal [25]. Since these techniques are generally too computationally intensive to run on a resource-poor mobile device, we incorporate sensors attached to the device that can automatically achieve a similar result with a relatively small computing cost. In Section 4, we describe how to use (1) a GPS receiver and (2) the tilt sensor to extract context information without soliciting the already scarce computational capability a mobile device has.

The rest of this paper is organized as follows. Section 2 describes related work in personal experience computing. Section 3 defines the design requirements for mProducer and its overall design architecture. Section 4 explains how the GPS receiver and tilt-sensor are used in sensor-assisted automated editing. Section 5 presents the storage constraint uploading (SCU) algorithm. Section 6 shows the editing user interfaces. Section 7 describes user studies that evaluate mProducer. Section 8 draws our conclusion and shows our future work.

2. RELATED WORK

We have divided related work into four main categories corresponding to the capturing, archiving, editing, and utilizing phases in the content production lifecycle.

In the capturing phase, many research activities have focused on *context-aware media capturing*, which proposes techniques for intelligent, early metadata acquisition at the time of content capture, rather than a later, complex content analysis [3]. In general, these techniques follow these steps: (1) deploy a variety of sensors at the point-of-capture, (2) interpret sensor data into different context meta-data for the content, and (3) utilize context data for different applications. Life log agent [1] is a system that can capture life-log videos and audio from a wearable camera. At the point-of-capture, the life-log videos and audios are automatically annotated with context meta-data from wearable sensors including: a GPS receiver, an accelerometer sensor, a gyro sensor, and a brain-wave analyzer. Sensor data are interpreted into context meta-data of the form: who, what, where, when. These context meta-data annotations are used as index keys in a context-based video retrieval system, allowing a user to input queries in the form of the 4Ws and retrieve matching video and audio segments. Kern et al. in [13] focuses on automatically annotating meeting recordings for easier retrieval. To accomplish this, sensor data (context information) are captured from body-worn acceleration sensors, audio capturing sensors, and location sensors. Then, sensor data are used to derive the user's activities, such as sitting, walking, standing, or shaking hands. Furthermore, the system can infer the user's interruptability in his/her environment. Sumi et al., [22] describes a system that utilizes multiple wearable and environmental sensors to help scientists to analyze and learn human social protocols. These sensors include the ID tags, LEDs, IR trackers, and cameras. These sensors record a person's position context along with audio/video data. In addition, it can also determine one's gaze, which shows where the person's attention is focused on. Based on this context information, the system interprets and summarizes people's social interaction patterns. MMM system [20] can automate content metadata creation using available context information on a camera phones, such as location and time. When a photograph is taken at a location, the system can re-use shared metadata from previous photographs taken at that location. This approach requires a centralized repository that stores the shared meta-data information for photographs at various locations. In our work, we also adapt the context-aware media capturing approach. The mProducer uses a tilt sensor to measure the level of hand-induced camera shaking and automatically remove blurry images resulting from excessive amount of shaking. In addition, we use the location information to provide a map-based media content presentation, which in turn makes it simple to search for content on small displays.

The second phase of the content production lifecycle is archiving. Archiving deals with how to store the captured digital content. We have found that most of camera phones suffer from very limited storage space. For examples, the Toshiba T-08 TM comes with only 8 MB of mobile storage and allows users to record merely three minutes of video clips at five frames per second. Although Nokia 7610 TM and Sony Ericsson K700i TM are equipped with high-resolution mega-pixel digital cameras, they are severely restricted by their small storage capacity. In our work, we provide the storage constrained uploading algorithm to upload frames to free up limited mobile storage space.

The third phase is editing phase. This phase is about providing a user interface for users to edit their contents. Hitchcock [11] is a PC tool that uses key frames to speed up editing of home videos. The tool displays key frames in piles (based on color similarity of key frames) for selection, and a storyboard to drag-and-drop key frames (shots) according to the sequence of shots the user wants. Since mProducer runs on a PDA with a much smaller display, the idea of presenting shots in piles was not a workable solution. In addition, it is not possible to have both the key frame presentation area and a storyboard on a small mobile screen at the same time. On the mobile device, Jokela [12] presents an overview of the key opportunities and challenges in developing tools for authoring multimedia content in mobile environments. However, no solutions were provided. Lara et al. in [15] describes a collaborative tool that allows mobile authors to collaboratively download and edit content with different fidelity. They address the replica inconsistency problem occurring when revisions at different fidelities are merged. mProducer for this phase, we found key-frame based editing, and storyboard presentation are more suitable for average users and mobile environment. Also, Map-based content organization is easier for users to find the content he wants to edit because map gives more information about content.

The final phase is by far the most popular phase. There are many of ongoing research activities on how to utilize captured content. The major topics are: (1) how users may revisit those experiences themselves, (2) how to enhance social interactions by sharing experiences with others, and (3) how to share people's personal experiences. For (1), Alberto Frigo in [8] described an experiment consisting of continually photographing a participant's right hand. The result of this experiment was the creation of an autobiography. Audio-based memory aid [24] describes a wearable memory aid with the goal of helping to alleviate some everyday memory problems by creating a searchable, personal archive of personal experiences. The memory aid was designed with capturing and retrieval functions based on a person's memory. For (2), the borderland wearable computer [18] allows users to communicate with each other. Their tool allows for remote participants to provide a single user with information in order to assist multiple people at a distance. This tool is also useful for health-care and fire-fighting situations. For (3), Flipper [6] from HP is a simple and automated sharing tool that pushes new photographs captured by a user to his/her friends and family members on his/her buddy list. The goal of Flipper is to support a social presence, so a user can keep up with the lives of his/her friends and family members through a simple and automated photograph sharing tool. WatchME [16] is also a sharing interface used only for close friends and family. It provides three layers of information that allows different degrees of communication: (a) awareness, (b) thinking of you, and (c) message exchange. Our next step is focused on the utilizing phase to design the sharing tool for helping along face-to-face communication.

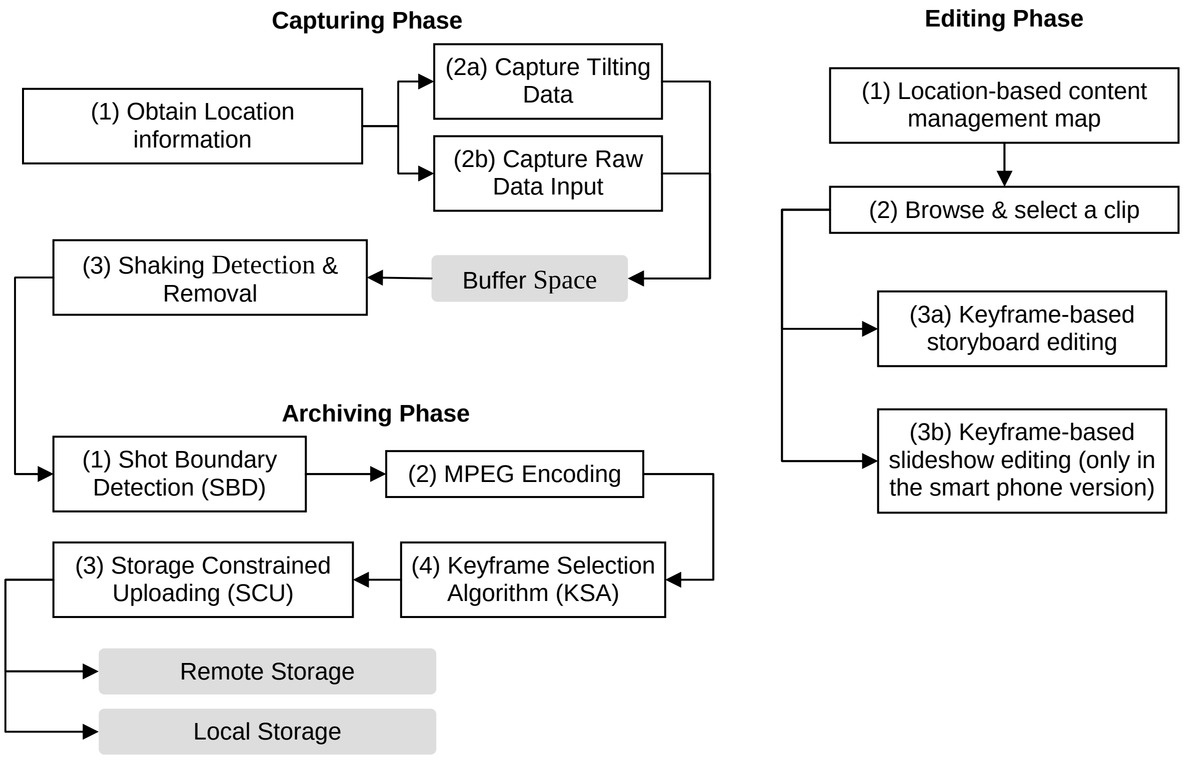

Capturing Phase
(1) Obtain Location information
(2a) Capture Tilting Data
(2b) Capture Raw Data Input
Buffer Space
(3) Shaking Detection & Removal
Archiving Phase
(1) Shot Boundary Detection (SBD)
(2) MPEG Encoding
(4) Keyframe Selection Algorithm (KSA)
(3) Storage Constrained Uploading (SCU)
Remote Storage
Local Storage
Editing Phase
(1) Location-based content management map
(2) Browse & select a clip
(3a) Keyframe-based storyboard editing
(3b) Keyframe-based slideshow editing (only in the smart phone version)

Figure 2. The Capturing, Archiving, and Editing Phases in mProducer

3. DESIGN

The design of mProducer is based on the results of our pilot user study to help understand and derive the requirements for mProducer on two of the major mobile device platforms: PDAs and smart phones. We provide a short summary on the results of the pilot user study below. The details about the experimental setup and the full results of the pilot user study are discussed in section 6.1.

- Users prefer to think of the organization of contents in terms of the location, not the time of capturing. Therefore, map-based content organization is better than time-based content organization.
- Users prefer a keyframe-only editing interface because it provides an easy learning curve for both PDAs and smart phones. Also, in terms of user satisfaction, they rank it best in editing experience, task completion time, and ease-of-use.
- Users have found that a major category of frames that they want to remove during editing is blurry frames. Amateur video capturing can produce frequent blurry frames that are caused by hand-induced camera shaking.

Based on the requirements defined above, we have come up with the design of mProducer. This design is implemented on two hardware platforms (HP iPAQ 5550 PDA and Nokia 7610 smart phone) with peripheral attachments shown in Figure 5. The PDA has built-in WiFi and Bluetooth modules for network connectivity, a digital camera, a GPS receiver, and a tilt sensor.

The smart phone has built-in Bluetooth and GPRS for network connectivity, a Bluetooth GPS receiver, and a Bluetooth tilt sensor.

The design of mProducer can be described using the flow chart shown in Figure 2, which consists of the capturing phase, archiving phase, and editing phase. Typical usage of mProducer involves repeating patterns of capturing multiple media clips along with their context information, continuously archiving media clips on a networked storage server (which frees up spaces in the mobile storage), and editing them. We will explain these three phases in detail below.

The Capturing Phase: At the start of a new media recording, mProducer queries the GPS receiver to obtain the location of the new recording. The second step involves two tasks executing concurrently. The first task captures streams of raw video and audio, whereas the second task records a second stream of camera tilting angles. The media stream and the tilt angle stream are stored in a buffer, and then combined to automate the detection and removal of blurred frames.

The Archiving Phase: The archiving phase starts by applying the shot boundary detection (SBD) algorithm[1] to separate a video clip into disjoint shots[2] or scenes. After the raw video frames are compressed by a MPEG encoder [17], a key frame selection algorithm (KSA) [26] is used to identify a representative key frame for each shot. Then, meta-data are annotated to each video frame, including: (1) whether it is a key frame or not, (2) its MPEG frame type (I, P, or B), and (3) its byte size. When the mobile storage runs out of space, an offloading algorithm called SCU assigns a *frame priority* to each frame based on its meta-data values. This frame priority dictates the offloading order of the frame to the server. The SCU algorithm is described in detail in section 5.

The Editing Phase: When a user wants to search for media clips for editing, a location-based content management screen is displayed that organizes media clips based on their GPS capture locations. The user starts editing by first selecting a point on a map which represents recordings made there. When the user clicks on a map point, a list of media clips that were recorded at that map location is displayed to the user. Then the user chooses a media clip among the list to edit. Figure 3 shows three screen shots of the editing user interface on the PDA. The left screen shot is a map of the location-based content management. Dots on the map represent locations of prior content recording(s), and the value within each dot represents the number of video clips recorded at that map location. Users can use the map interface (zoom in/out and move) to locate and search for previous media recordings. After the user clicks on a map dot, the middle screen shot called the material pool appears with a list of media clips previously captured at that dot location. When the user selects a media clip from the material pool time, the recording time and duration of that media clip are displayed. Then a keyframe-based slideshow UI is provided to the user to preview the selected clip and decide if it is the correct one for editing. The reason for using the keyframe-based slideshow metaphor is because of the positive results of a user study discussed in section 6, which demonstrates that it is preferred by most users. After the preview, a keyframe-based storyboard UI is displayed to the user for editing, and then the user can remove unwanted frames from the chosen clip. The keyframe-based storyboard UI is shown on the right screen shot of Figure 3. The reason for using the keyframe-based storyboard metaphor is also because of positive results in the user study, showing that it is most preferred by users.

The smart phone version of mProducer UI is shown in Figure 4. It is similar to its PDA version with some customizations to fit the UI into a smaller screen size and for a keypad input. The screen shot (1) is the location-based content management UI. Red points on the map indicate some recording(s) have been made at those map locations. The value on each red map point corresponds to the numeric hotkey used to select a map point. After the user clicks on a numeric hotkey from the keypad, the screen shot (2) is displayed showing a material pool of previous media recording(s) at the selected map point. The main difference between the smart phone and PDA UIs is in the editing UI. We have found from a user study (described in section 6) that users prefer a *hybrid* of storyboard and slideshow interfaces on a smart phone. Therefore, we provide both storyboard and slideshow editing interfaces on smart phones, and a user can freely switch between them. The storyboard and slideshow interfaces are shown on screen shots (3) and (4) respectively.

[1] We implemented the SBD algorithm based on color histograms described in [9]. Note that the SBD algorithm is not the focus of our work. We chose the color histogram-based SBD because of its ease for rapid prototyping. More sophisticated SBD algorithms [4] can surely enhance the key frame selection quality; which in turn improves the editing satisfaction of keyframe-based content editing.

[2] A shot is defined as one or more frames generated and recorded contiguously and represents continuous action in time and space [7].

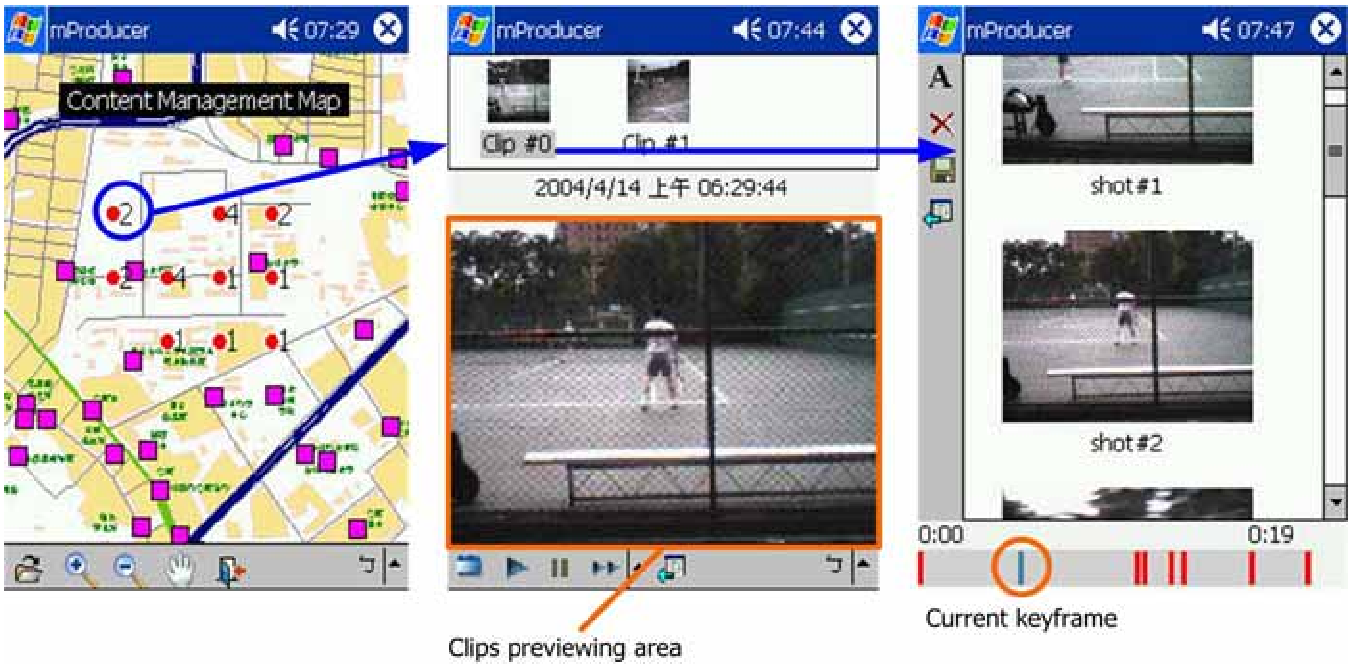


Figure 3: mProducer UI screen shots on a PDA

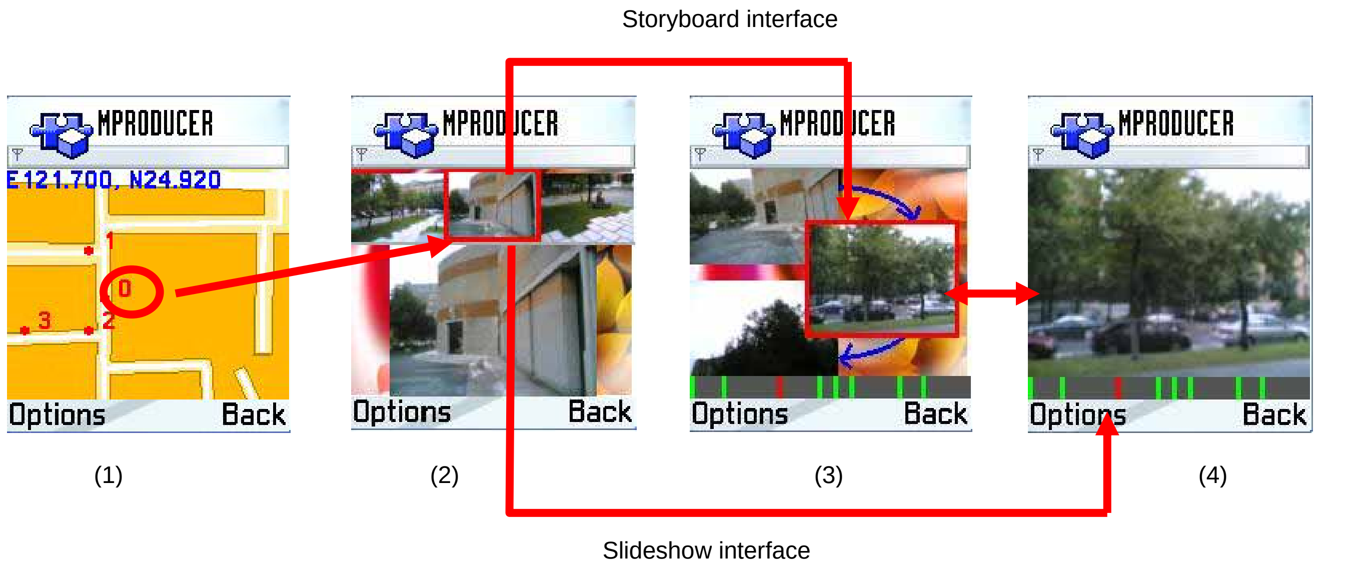


Figure 4. mProducer UI screen shots on a smart phone

4. SENSOR-ASSISTED VIDEO PROCESSING TECHNIQUES

From the interview with the participants in the pilot study, we also found two user requirements on a mobile editing tool. The first requirement is that as the number of content recordings increase, the organization of content needs to be in terms of the recording locations rather than in terms of the recording times, because of the reported user preference for this. There were many occasions where the users had to remove many blurry frames, thus, the second requirement is that there needs to be a mechanism for automatic removal of blurry frames caused by hand-induced camera shaking.

Solutions to address these two requirements can be found in many traditional image processing and content analysis techniques [26][21] that at the time of content production, extract metadata context information such as: location, objects, amount of shaking, and lighting levels, etc. Due to the limited computing capability on a mobile device, those computationally intensive image/video analysis techniques are not adequate for mobile device authoring. We believe in the *context-aware media capturing* approach where sensors are deployed at the point-of-capture to assist the capture and inference of a variety of context metadata. This sensor-assisted approach, in general, requires less computation; therefore, it is ideal for a resource-poor mobile device.

To meet the user requirements, mProducer incorporates two sensors to automatically create contextual metadata at the point of capture: (1) a global positioning system (GPS) receiver which detects location meta-data, and (2) a tilt sensor which measures the amount of camera shaking. The location meta-data for each content recording is used in the location-based content management, so that a user can easily navigate the map to locate a previously recorded content. The camera shaking measurements are used to detect the excessive amount of camera shaking, which results in blurry, unusable frames to be cut automatically. Figure 5 shows the hardware components of the prototyped PDA system together with a GPS receiver and a tilt sensor.

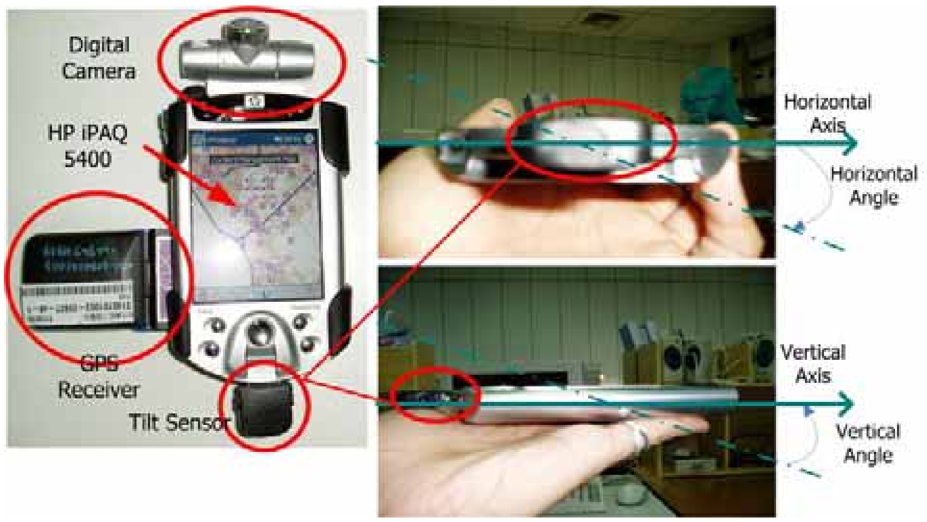


Figure 5: The HP iPAQ 5550 with camera, GPS receiver, and a Tilt sensor

GPS Receiver: It is the GPS-CF card from CHIPCOM Electronics. Each time a user records a video clip, mProducer will probe the GPS receiver for current location information. Then, this clip will be annotated with location information. Location information of each clip is used to construct the location-based content management map (described in Section 3). For the smart phone version of mProducer, we use the Bluetooth-GPS receiver from Pretec Electronics Corporation.

Tilt Sensor: It is the TiltControl CF card made by ECER Technology shown in Figure 5. It contains an accelerometer that measures the horizontal and vertical tilt of the device. Changes in the tilt are used to compute the magnitude of camera shaking and predict its impact on video quality. The tilt sensor measures both the direction and the magnitude of tilt angles. We elaborate on how to use tilt sensor for camera shaking detection in the following subsection.

Experiment to Identify Camera Shaking Pattern

We use a tilt sensor to measure the level of camera shaking and automate the process of shaking artifact detection and removal. This is an ideal alternative to computationally intensive video analysis on a resource-poor mobile device. To determine the signature of camera shaking, we have conducted an experiment to distinguish between excessive amount of shaking (e.g., resulting from putting the device in a pocket during walking) from moderate shaking that comes naturally with unstable hands when walking while filming. Our experiment is described below.

Data Acquisition: The TiltCONTROL sensor monitors the vertical and horizontal tilt of the device throughout the experiment. A series of readings are recorded and analyzed to determine if camera shaking occurs. The sample rate of tilt sensing is set to be

200 milliseconds. The standard deviation of the changes in the device angles is computed for each sliding window of the most recent 10 readings.

Shaking Detection: Device shaking can be detected when changes in a device's tilt angles create oscillations between two opposite directions. The intensity of shaking can be measured by calculating the rate of change in device tilt angles and the oscillation rate. Walking while holding the device will create oscillations of smaller magnitude (see the middle graph of Figure 6; X-axis represents time, Y-axis represents the magnitude of change of degree per unit time). Walking with the device in a pocket will also create oscillations, but of a larger magnitude (see the right graph of Figure 6). For the experimental setup, we measured three activities for each participant: (1) holding the mobile device while sitting or standing still for 2 minutes (collecting 591 samples), (2) holding the mobile device while walking for 2 minutes (collecting 591 samples), and (3) putting the PDA in a pocket or a bag while walking for 2 minutes (collecting 591 samples).

Result: Based on empirical data shown in Table 2, we have determined two conditions for excessive shaking: (1) the standard deviation of the tilt angles is larger than 20°(degrees) , calculated by 89.9% of actual shaking frames (externally observed) having higher standard deviation values than this threshold value, and (2) the frequency of oscillations in both directions exceeds 1.5 oscillations per second, again, calculated by 76.5% of actual shaking frames having higher value than this threshold value. In Figure 6, we depict a partial result of one participant's experiment. We can see from this figure, under the normal case, that the standard deviation is small, and the vibration is moderate. Walking introduces constant vibration, but the standard deviation is below 20°. When shaking, we can see that the standard deviation is high and the vibration is frequent. This pattern helps the system to detect camera shaking with a simple computation of standard deviation, this demonstrates how sensor measurements may assist in processing video content using simple computation.

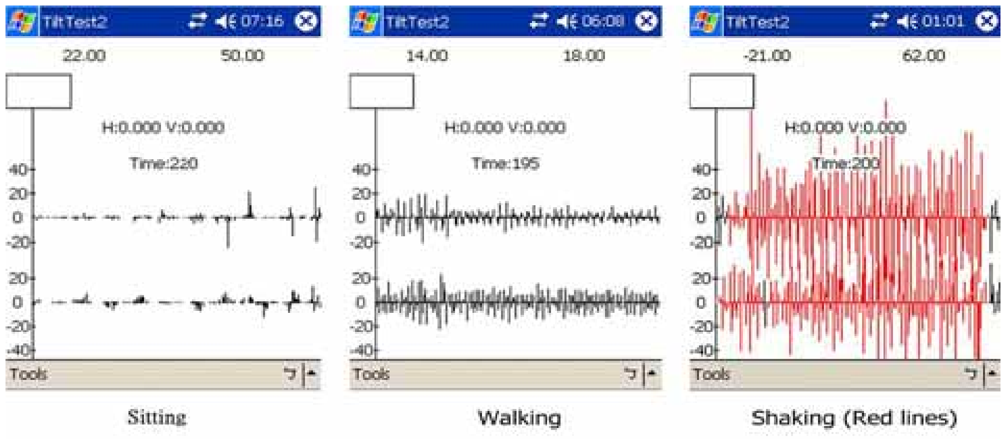


Figure 6. Measured Oscillation Magnitudes for Three Activities

Table 2. Oscillation Measurements for Three Activities of Sitting, Walking, and Pocketing

Activities	Standard deviation on tilt angle degree changes		Frequency of oscillations (per second)	
	Horizontal	Vertical	Horizontal	Vertical
Sitting	2.62	3.00	1.36	0.76
Walking	5.27	7.13	1.89	1.97
Pocketing	64.72	75.96	1.73	1.85

5. STORAGE CONSTRAINED UPLOADING (SCU)

An uploading algorithm makes the following two decisions: (1) when to upload, and (2) what portion of contents to upload. We design the SCU so that it can make good decisions to minimize the network communication (including both the uploading and downloading) in both phases of authoring. We describe the SCU algorithm by how it makes these two decisions.

SCU will not upload contents until the current storage space is full. The benefit is that we can avoid uploading frames that will later be cut by a user. SCU chooses frames for uploading based on an observation that there is a difference in quality requirements between personal experience authoring tools targeting the average consumers, and so-called mass media content authoring tools targeting professional content providers. We believe that there is no need to provide a mobile personal experience authoring tool that can produce professional quality content. In other words, fine-grained editing (e.g., frame-by-frame) used in a PC-based authoring tool for professional quality content is in fact unsuitable for a mobile authoring tool, because they require both a significant amount of user efforts and high resolution screens.

We define *editing granularity* as the level of detail that an authoring tool allows a user to edit. Take MPEG video editing as an example, its' finest granularity is frame-by-frame editing, where a user can preview and choose any arbitrary frames for cutting, adding text, etc. A coarser granularity is I-frames, where a user can preview and edit I frames only. This observation leads to the discovery of the fact that for the average user, a portion of the frames can be uploaded without degrading the editing experience. For example, in MPEG video editing, if the average user only requires I-frame editing granularity, uploading non-I-frames does not affect the user editing process and experience.

The SCU algorithm is based on a mapping between types of frames and priorities for uploading. In the above example, I-frames have higher priority than non-I-frames when it comes to uploading. In our current work, we design the SCU algorithm to prioritize frames into three levels based on their frame types:

Table 3. Frame Priorities Mappings to Frame Types

Frame Priorities	Frame Types
Level 1 (Low)	Non I and Keyframes
Level 2 (Medium)	I-frames
Level 3 (High)	Keyframes

We adopt the technique of key frame selection from the field of video summarization and set the key frames as the highest priority because key frames are still images that best represent the content of a video sequence [5]. As a result, key frames are never uploaded in order to guarantee a minimal keyframe editing granularity.

The SCU algorithm is also based on a concept called *storage granularity*, which is about the types of frames that mobile storage can accommodate during the capturing phase:

Table 4. Storage Granularities Mappings to Frame Types

Storage Granularities	Frames to Store
High	All frames
Medium	I and Keyframes
Low	Keyframes

Initially, a mobile storage is empty, so the SCU algorithm will store all types of frames in the mobile storage. The mobile storage is said to be at high storage granularity when it can accommodate all types of frames. As a mobile user captures new frames, mobile storage will eventually run out of free space at the current storage granularity. When a newly captured frame causes an overflow in mobile storage, the SCU algorithm will need to drop down a level to the medium storage granularity. From this point on, it will store only new I/keyframes, and upload all new non I/keyframes to the storage server. At the same time, it will also gradually upload existing non I/keyframes to the remote storage because they have a lower priority level than what is allowed by the medium storage granularity. By uploading existing frames, it will create free space in the mobile storage for new I/keyframes. Note that the editing granularity cannot exceed the storage granularity. For example, to support I frame editing granularity requires I-frame or above storage granularity.

5.1 ALGORITHM

The SCU algorithm preserves two properties when uploading frames to remote storage. They are (1) fairness to all clips, and (2) gradual uploading of frames. If a mobile storage contains multiple clips, the SCU algorithm should try to maintain fairness. This means equal storage granularity among all the clips currently in the mobile storage. This fairness property can ensure that mProducer tool can provide equal, consistent editing granularity for different clips in the mobile storage. When the SCU algorithm drops down one level of storage granularity (e.g., from high level to medium level), the uploading of frames should be done gradually and on an as needed basis, i.e., it does not upload all the non I/keyframes at once to remote storage. The reason for gradual uploading is to avoid unnecessary uploading of frames that will later be cut by users.

The example in Figure 7 illustrates how the SCU algorithm works. The example is explained as follows: the mobile storage is currently full and contains the entire clip *#1* and clip #2 that is still being captured. The block G^i_j is the *j*-th group-of-pictures (GOP) of clip *#i*. Assume when a new frame comes into the mobile storage, the SCU algorithm will upload all the frames except the I-frame and key frame (if any) of the GOP that is marked by the marker (clip *#1* in this case) to the storage server. This uploading frees up a block of space for new frames. This marker will then move to the next clip's (clip *#2*) foremost un-cleared GOP, where SCU will upload the non-I/keyframes in this GOP in the next round. In order to achieve fairness among clips, the SCU algorithm uploads groups of frames marked by the marker that moves in a round-robin fashion among all clips currently in mobile storage.

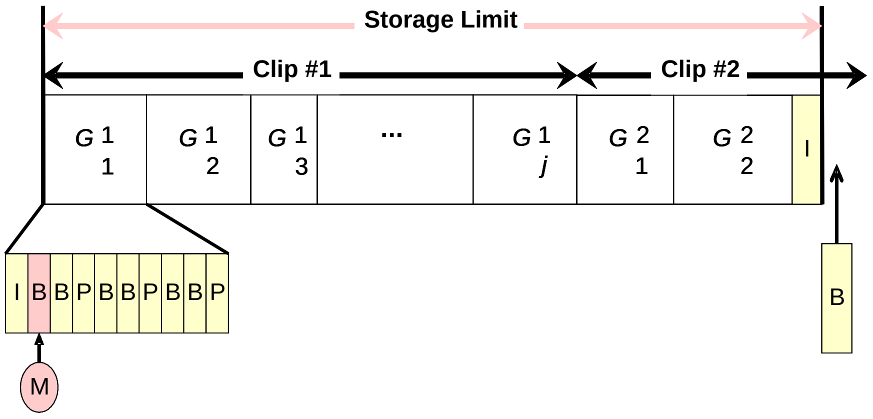


Figure 7. The storage view for illustrating SCU - Case I

Suppose that all non-I/keyframes are uploaded to a storage server. The SCU algorithm will then upload I frames. Figure 8 illustrates the state of a mobile storage at the *medium storage granularity*, where all the non-I/keyframes are uploaded to a storage server to make space for I/keyframes. I^1_1 to I^1_j and K^1_1 to K^1_p are I-frames and key frames of clip *#1* respectively. Consider that a new frame is generated. If it is not an I or a key frame, it will be uploaded right away. If it is an I or key frame, the SCU algorithm will drop to the *low storage granularity*, and it will start to upload I-frames to a storage server. Figure 8 shows an advancing marker which points to the next frame that will be uploaded.

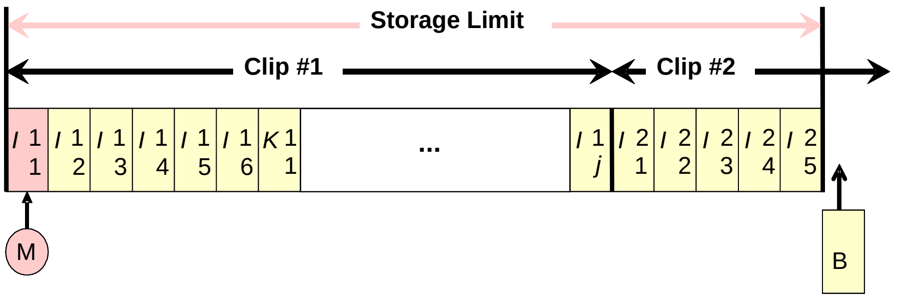


Figure 8. The storage view to illustrate SCU – Case II

We have designed an *uploading list* that computes the order frames will be uploaded to the mobile storage. It first sorts frames based on the frame priority and then applies a round-robin algorithm over clips. With this list, mProducer can simply look up the head of the list to get the frames for uploading. For example, in Figure 7, the 9 B and P frames of G^1_1 will be placed at the positions 1 to 9 on the uploading list with B and P frames of G^2_1 being placed at positions 10 to 18 on the list, and so forth.

The main body of the SCU algorithm is shown below. We denote the reserved space for mProducer in the storage as *Z*, the size of total frames in the storage is *T*, the *i*-th frame of clip *#j* as f^j_i, its size as $^Sf^j_i$, the newly coming frame as f^N_{new}, and *N* is the number of clips in the mobile storage.

In the current work, mProducer does not allow storage granularity to fall below the keyframe level (i.e., the mobile storage must store all key frames). Therefore, there exists a limitation on the size of multimedia content that a user can capture at any given time. The reason for this size limitation is that mProducer does not want to upload key frames to the storage server and then download them again during the editing phase. When this limit is reached, mProducer will inform its user to stop capturing new data and to start editing clips.

```
Algorithm 1 The basic SCU algorithm
Require: A new coming frame $f^N_{new}$ (size, type)
Ensure: Frames to upload to storage server or save $f^N_{new}$
1: if $S_{f^N_{new}} + T > Z$ then
2:     upload the frames in the order of the "Uploading List"
       until $S_{f^N_{new}} + T < Z$;
3:     adjust the "Uploading List" accordingly
4: end if
5: if $f^N_{new}$ is not uploaded then
6:     save $f^N_{new}$ and adjust the "Uploading List"
7: end if
```

5.2 Variants of SCU Algorithm

There are many possible variants to the SCU algorithm. We can use different priority metrics for incoming frames, which affects the ordering of frames in the uploading list. The priority metric can be based on the time of capturing (e.g., the later time has the higher priority), the size or fidelity of each piece of content (e.g., the higher fidelity has the higher priority), or the hierarchy of content established by video indexing [21].

6. USER INTERFACE

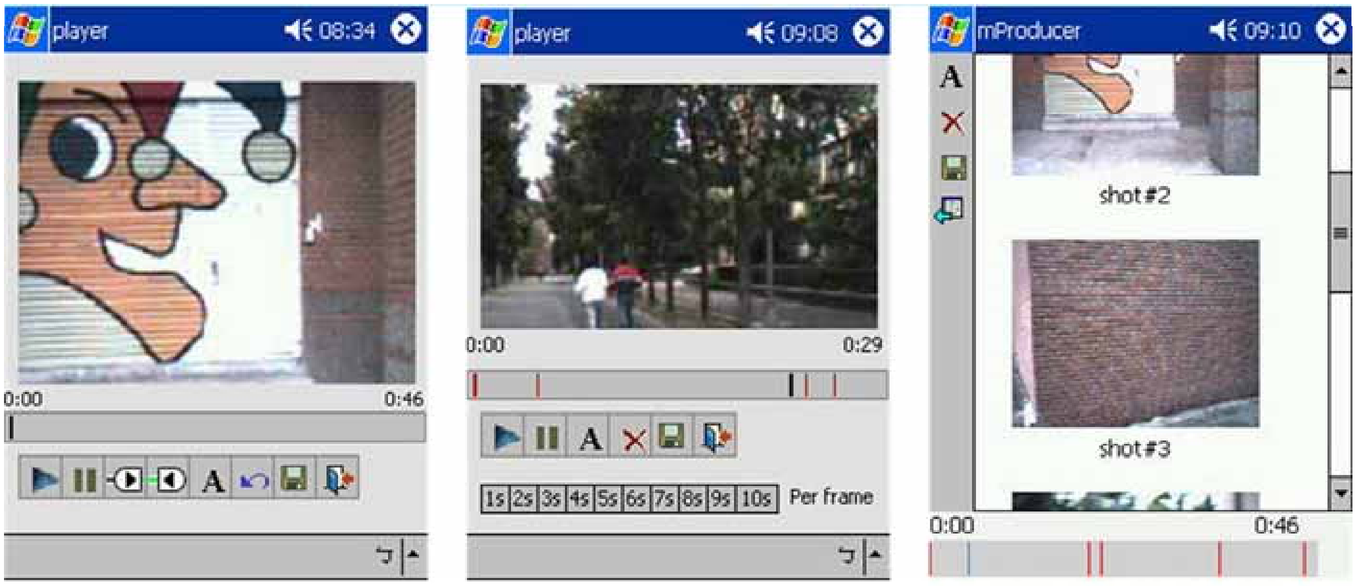
The design of a mobile user interface needs to consider small screen size, inconvenient input methods, limited user attention, and limited user computing experience. Existing video editing interfaces designed for desktop computers (such as Cyberlink's Power Director™, Ulead's Video Studio™, etc.) are all designed using the frame-by-frame editing method. In the frame-by-frame video editing, a user browses through the entire video clip frame-by-frame, and then selects mark-in and mark-out points as starting and ending points to extract the desired portion(s) of video. The Hitchcock [11] tool has pointed out that a major problem of this frame-by-frame editing approach is that selecting good mark-in and mark-out points is a time-consuming, manual process for users on PCs. This problem, as shown in our user studies, simply becomes worse on a mobile device with a small screen, inconvenient input methods, and limited user attention. As a result, we need to consider an alternative UI design, called keyframe-based editing, for authoring user interfaces on a mobile device. To illustrate that keyframe-based editing has better usability than frame-by-frame editing for a mobile device, we have built a keyframe-based and frame-by-frame editing UIs on the PDAs and smart phones (shown in Figures 9), and then performed a user study to compare their usability. The user study and its result are discussed below.

6.1 User Study on Mobile Editing UIs

The user study consists of testing the following three proposed user interfaces shown in Figure 9 and Figure 10.

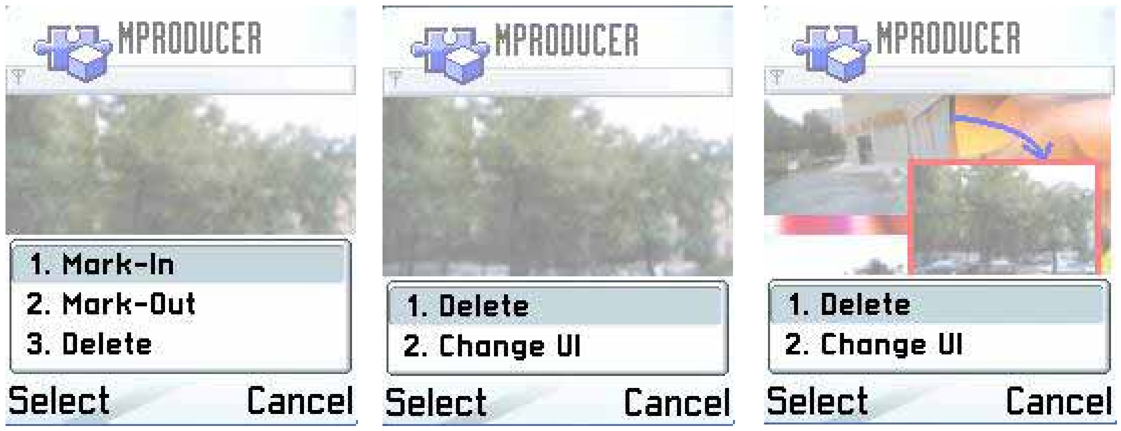
- **(UI-A):** *Frame-by-frame editing with a video player:* the video clip is played back frame-by-frame to a user and the user selects the mark-in/mark-out points to extract desired portion of the video. This is a scaled down version of conventional desktop editing interface.
- **(UI-B):** *Keyframe-only editing with a slideshow player:* only the key frames of the video clip are played back to a user. The user can control the time interval between two key frames. Rather than selecting mark-in/mark-out points, a user can delete the unwanted shot by simply pushing a delete button when its key frame is shown.
- **(UI-C):** *Keyframe-only editing with a storyboard player:* a storyboard-like interface displays a collection of key frames based on the order of the shots' recording times. A user can delete a shot by simply pushing the delete button.

This goal of the user study is to understand the tradeoff between the effectiveness (quality on the editing results) and efficiency (task completion time) for the above three editing UIs on mobile devices. In addition, this user study investigates user satisfaction with these three editing UIs. For example, the frame-by-frame editing provides a user with the finest editing control on marking the precise boundary of wanted video clip, but choosing desirable mark-in/mark-out points at this level of editing granularity can be time-consuming and inconvenient on a mobile device, thus leading to less efficiency and less user satisfaction. On the other hand, keyframe-based editing offers a coarse editing control, but it requires less user effort and allows higher efficiency and user satisfaction in the mobile environment. Below we describe the procedure and results of our user study on PDAs and smart phones.



(a) frame-by-frame playback (b) keyframe-based slideshow (c) keyframe-based storyboard

Figure 9: Screen Shots for the Three Editing User Interfaces on PDAs



(a) Frame-by-frame Playback (b) Keyframe-based Slideshow (c) Keyframe-based Storyboard

Figure 10: Screen shots for three editing user interfaces on smart phones

Independent variables: The three editing interfaces detailed above.

Dependent variables: Task performance measures the amount of time to complete editing tasks using a selected editing interface. Subjective satisfaction ranks the interfaces in terms of overall editing experience, the user's perception of the quality of editing, ease of use, and ease of learning.

Participants for PDA version: We randomly chose eleven participants (eight males and three females) on campus for this user study. Their ages range from 20 to 41 years, with a mean of 24. Three of them (all male) have had previous experiences in using a PDA. Five of them (four male and one female) have had previous experiences in using PC video editing tools. None of them had previous experience in using mobile video editing tools. All participants have used smart phones.

Participants for smart phone version: We randomly chose 6 participants (three males and three females) on campus for this user study. Their ages range from 16 to 31 years, with a mean of 23. Three of them (one male and two female) have had previous experiences in using PC video editing tools. One of them had previous experience in using mobile video editing tools. All participants have had previous experiences using smart phones.

Procedures: Participants were briefed on the goal and the procedure of the user study. We demonstrated how to capture videos using the PDA or smart phone and how to edit using each of the three interfaces. The evaluation consisted of three sessions:

1. Each participant was asked to record a total of 6 minutes of video containing three 2-minute clips. Examples of content captured included scene-recordings, self-introductions of people in a group, and specific events.
2. The participants were asked to edit three clips, each using one of the three different editing interfaces. In this case the editing task involved only removing unwanted content from the raw video clips. We measured the length of time it took to complete each editing task for each participant. Note that the assignment between clips and editing interfaces were randomly chosen to reduce the first clip bias.
3. Each participant filled out a questionnaire with demographic information including age, sex, and experience with video editing tools. The questionnaire also asked each participant to rate each of the three editing interfaces using the four characteristics defined in Table 5.

Table 5. Questions for Interviewing in the Pilot Study

#	Questions (Rank each UI from 1 ~ 10 for Q1 to Q3)
1	Perceived quality of editing
2	Ease-of-use
3	Ease-of-learning
4	Overall editing experience

Results in task completion time on PDAs: We recorded the time each participant took to complete editing a two-minute video clip for each of the three interfaces using PDAs. The results are shown in the left graph of Figure 11. The mean task completion time for each UI is: (UI-A) 4 minutes and 32 seconds, (UI-B) 3 minutes & 58 seconds, and (UI-C) 2 minutes and 48 seconds. Ten out of the eleven participants completed the editing task fastest using (UI-C). All participants finished editing sooner using (UI-B) in comparison to (UI-A). The result shows that users can perform editing tasks *more* efficiently using a keyframe-only editing interface. In addition, the keyframe-only storyboard editing interface provided the best task completion time.

Based on our interviews with participants, they reported that the storyboard UI helped them by enabling them to see several key frames at the same time. They could quickly identify which frames or shots they did not like and remove them. Some participants also mentioned that their problem with frame-by-frame editing was that it required uninterrupted, focused attention on the screen. Finding exactly which frames to set as mark-in/mark-out is also difficult because it puts a heavy mental-load on the users.

Results in task completion time on smart phones: We recorded the time each participant took to complete editing a two-minute video clip for each of the three interfaces using smart phones. The results are shown in the right graph of Figure 11. The mean task completion time for each UI is: (UI-A) 9 minutes and 40 seconds, (UI-B) 6 minutes and 2 seconds, and (UI-C) 5 minutes and 19 seconds. Five out of the six participants completed the editing task fastest using (UI-C). In comparison, participants took more time to complete the same task using the same UI on the smart phones than on the PDAs. This extra time on the smart phone is reasonable given that it is more difficult for users to perform editing on the smaller phone display.

Results in subjective satisfaction on PDAs: Participants answered the questions listed in Table 5. Their responses to the first three questions are shown in left side of Figure 12. The results show that users rated keyframe-only storyboard editing as producing superior editing quality. Our explanation is that when using frame-by-frame editing, casual users are not willing to spend time to find good mark-in and mark-out boundary points for unwanted content. Because of this, they find our SBD algorithm can find better boundary points for both wanted and unwanted shots. The results also showed that users rated keyframe-only storyboard editing to have the best ease-of-use and best ease-of-learning. We were told that the advantages of the keyframe-only storyboard interface were that: (1) it allows users to quickly move among shots, which is useful during editing, and (2) it allows users to quickly delete unwanted shots by a single-click on the key frames corresponding to these shots.

The results for overall experiences in the three editing interfaces showed that UI-C (key frame + storyboard) was consistently selected as most satisfying from all participants (100%), and 64% (seven) of the participants found UI-B to be more satisfying to use than UI-A.

Results in subjective satisfaction on smart phones: Participants answered the questions listed in Table 5. Their responses to the first three questions are shown in the right side of Figure 12. The results show that users rated keyframe-only storyboard editing as producing superior editing quality. With the more constrained UI on the smart phone, the frame-by-frame editing quality becomes even worse than on the PDA.

The results for overall experiences in the three editing interfaces also showed that UI-C (key frame + storyboard) was consistently selected as most satisfying from four of the six participants (67%), and five of the six participants (83%) found UI-B to be more satisfying to use than UI-A.

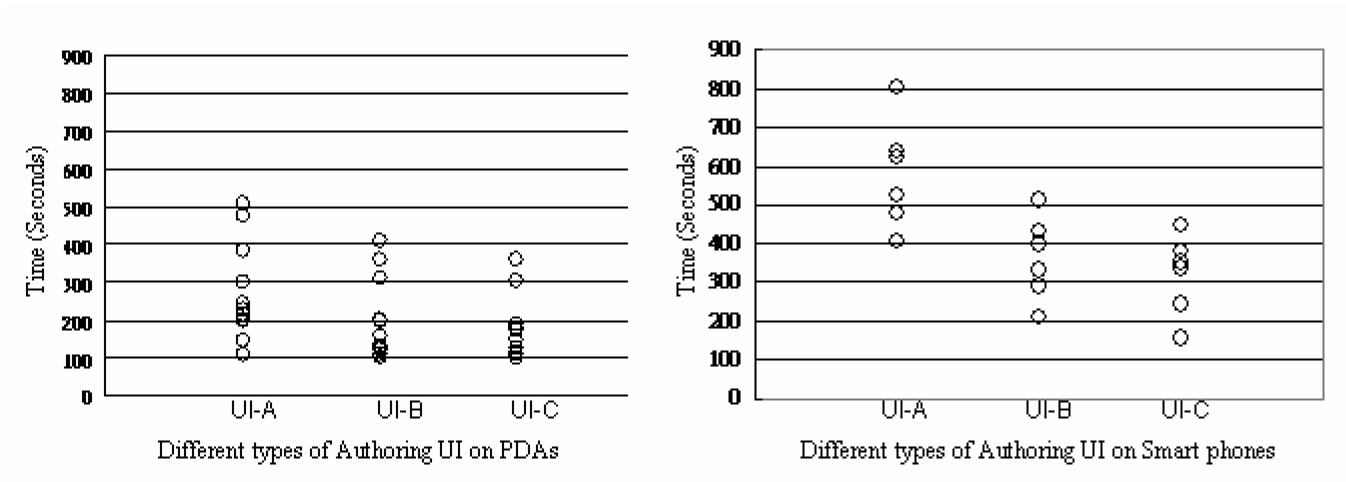


Figure 11: Task Completion Time

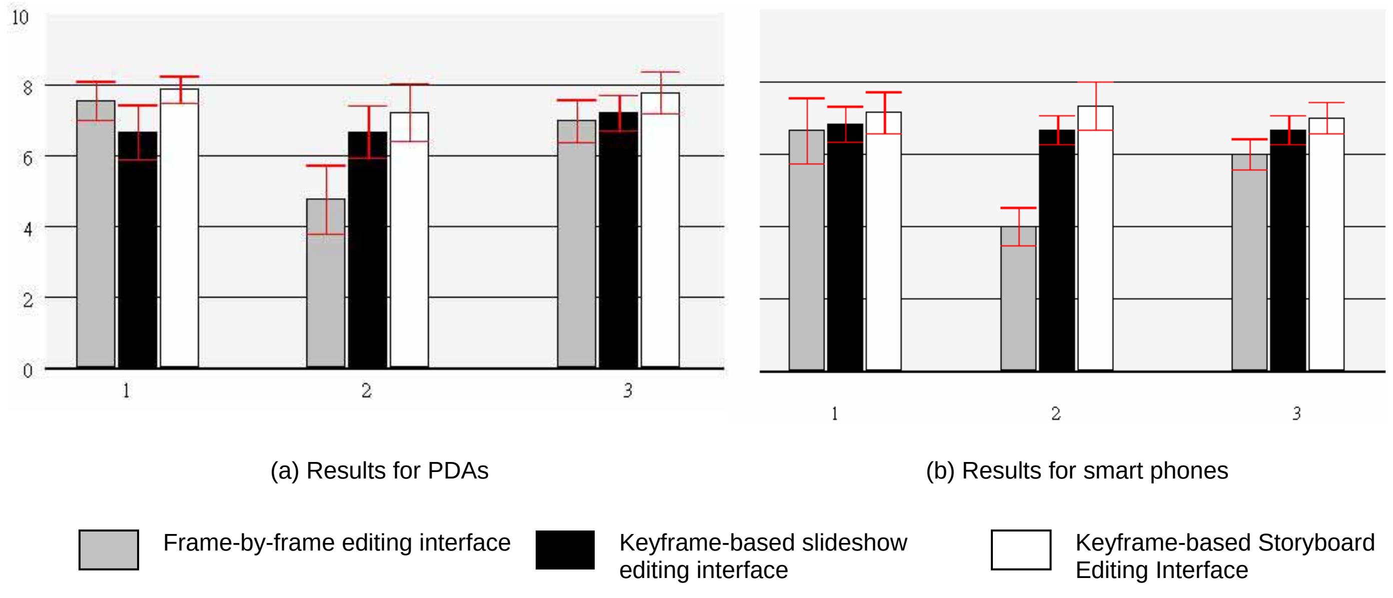


Figure 12: Response to questions in Table 5. X-axis represents the three questions. Y-axis represents their scores.

7. USER STUDY OF MPRODUCER

We conducted user studies to evaluate the overall experience of using mProducer with the location-based content management interface and keyframe-only editing interface on both PDAs and smart phones.

7.1 User Study on PDAs

Participants: We observed seven participants using mProducer to record video clips. Five were male and two were female. The ages of users varied from 21 to 33 years old, with the average being 23.8 years. Three have had previous experiences using PDAs, while all of them have used smart phones. Three have had previous experiences with desktop PC video editing tools. One of them had previous experience with a mobile device's video editing tool. All were chosen randomly on campus.

Procedure: Each participant was provided with mProducer running on an HP iPAQ 5550 mounted with a GPS receiver and a digital camera.

1. Participants were briefed on the goal and the procedure of the user study. We demonstrated how to capture and edit video using the PDAs.
2. Participants were asked to shoot any type of footage they wanted. They were encouraged to walk around campus, and record what they found interesting. We asked them to record about 10 minutes of footage with any number of clips.
3. Participants used the editing component of mProducer immediately on the content they had produced. They were asked to edit two clips chosen randomly from the pool of clips they had recorded. During the editing sessions, participants were asked to "think aloud" in order to let us know their intentions and the cognitive process of using mProducer.
4. After the editing session, participants were asked to fill out a questionnaire and discuss their overall experiences using mProducer. The questionnaire included questions about demographic information, participants' previous experiences with mobile devices and video editing tools, their impression of the mProducer tool (before and after using it), their experiences of navigating among different clips and editing the two clips they chose, and any other improvements they thought we could make.

Result in Overall Experience: In general, the participants' feedback was very positive. One of the participants described mProducer as *"a pretty cool tool to use."* Another participant said that *"the keyframe-only storyboard is very helpful for me to delete content that I do not like. Editing tools on desktop PCs should incorporate this feature too!" "Map based content management is very informative for choosing which clip to edit,"* said the other.

All participants said that editing with a keyframe-only storyboard interface was fast and easy. Some of the participants mentioned that the slideshow interface was better for getting a rough idea about the clip, while the storyboard interface was better for editing. Therefore, they suggested that the UI give the users the option to switch between these two interfaces. One participant suggested that we allow for location tracking of indoor recordings where the GPS receiver does not work. Some participants said that the content management map sometimes responds slowly.

7.2 User Study on Smart phones

Participants: We observed twelve participants using mProducer to record video clips. Ten were male and two were female. The ages of users varied from 22 to 30 years old, with the average being 23.8 years. Seven have had previous experiences using PDAs, while all of them have used smart phones. Eight have had previous experiences with desktop PC video editing tools. Three of them have had previous experience with a mobile device's video editing tool. All were chosen randomly on campus.

Procedure: Each participant was provided with mProducer running on Nokia 7610 with a built-in digital camera and a Bluetooth GPS. The rest of the procedure is similar to the user study on PDA.

Result in Overall Experience: Most of the feedbacks were very positive and similar to the results of the user study on PDA. However, there are some differences given that smart phones have more constrained screen sizes and input methods than PDAs. This leads to different requirements on the mProducer UI design on smart phones. We describe these differences as follows. Firstly, in the PDA version of mProducer, users prefer the storyboard editing interface because they can see several key frames at the same time. The storyboard interface helps them to quickly identify which shots to keep and which to delete. In the smart phone version, some participants also mentioned this advantage in storyboard interface. However, some participants found that by squeezing several key frames on the already small phone screen, each key frame image simply became too small for comfortable viewing. Unlike the PDA version, there is no clear winner between storyboard and slideshow interfaces for smart phones. Among twelve participants, when they wanted to browse key frames, eight of them preferred the storyboard interface. When they wanted to remove consecutive shots, they switched to the slideshow interface. Three participants used storyboard interface only, and one participant used slideshow interface only.

Secondly, sometimes users want to jump to the middle of the clip or jump over 4 or 5 key frames to quickly reach some key frames. On a PDA, users can do this with a scrollbar and a touch screen. However, on a smart phone, users can only input through a keypad. Some participants suggested that we could set a few hot keys to achieve this. Some participants also said that without an icon on screen to click like on a PDA, they must open the "option" menu and then select the command by pressing small buttons on the smart phone. It is inconvenient. Therefore, they would like to see an intuitive and simple manipulate interface, i.e. designing a set of hot keys, to reduce the number of button presses.

8. CONCLUSION AND FUTURE WORK

We describe our design, implementation, and evaluation of a mobile authoring tool called mProducer that enables everyday users to capture and edit their personal experiences at the point of capture from a mobile device. The mProducer can transform our everyday camera-equipped, mobile devices from simply content capturing devices to content producing devices. The unique aspect of mProducer is that it enables immediate point-of-capture editing and archiving from a mobile device, so that users can quickly distribute time-sensitive digital personal experiences.

To realize this mobile authoring tool, mProducer addresses the challenges of both user interface constraints and limited mobile system resources. For the mobile UI, we have designed the keyframe-based editing user interface. We have demonstrated that keyframe-based editing outperforms traditional frame-by-frame editing in terms of task completion time, ease-of-use, ease-of-learning, and editing quality. To address the problem of limited mobile storage, we have designed the storage constrained uploading (SCU) algorithm, which uploads large, continuous multimedia content to remote storage servers. To address the challenge of limited computing resources, we have designed sensor-assisted automated editing which incorporates a tilt sensor on a mobile device to automate the detection and removal of blurry frames resulting from excessive amount of shaking. Also incorporated is a GPS receiver to derive recording locations and enable easy, intuitive navigation using a map-based content management interface. Based on our user studies, the results have shown that users are satisfied with mProducer and that they have found it to be both easy and fun to use on a mobile device.

For future work, we would like to develop applications on top of personal experience content. One application area of interest is storytelling. Storytelling provides us with an effective and entertaining way to share interesting experiences with people in a social setting. Such social settings can arise when we want to become acquainted with new friends, or try to keep in touch with old friends and family members. Traditionally, storytelling is based on voice or gesture language to present a story to audiences. We believe that this traditional storytelling can and will be greatly enhanced with digital media technology. During a digital storytelling, the storyteller can retrieve the needed experience recordings from his/her personal experience repository and play/show them to story listeners on a digital media display device. Everyday storytelling will become media rich. For story tellers, the story presentation will no longer be confined to simply the voice and sign language, but enhanced with vivid multimedia content. For story listeners, they can better enjoy the stories by actually seeing and hearing these digital personal experiences presented in video, audio, and photos.

We are interested in the privacy and security aspects of personal experience content. Since the captured personal experiences may intentionally or unintentionally include other people and their activities, releasing and sharing these personal experiences, without consent from these people, can be viewed as a violation of their privacy. We are looking at sharing and ownership policies that incorporate privacy concerns from those who are captured in our personal experience content, as well as image processing techniques to remove them from our personal experience content.

9. REFERENCES

[1] Kiyoharu Aizawa, Tetsuro Hori, Shinya Kawasaki, and Takayuki Ishikawa, "Capture and Efficient Retrieval of Life Log," *In Pervasive 2004 Workshop on Memory and Sharing of Experiences*, 2004.

[2] Blog Website, http://blogdex.net/

[3] Susanne Boll, Dick Bulterman, Tat-Seng Chua, Marc Davis, Ramesh Jain, Rainer Lienhart, Svetha Venkatesh, "Between Context-Aware Media Capture and Multimedia Content Analysis-Where do We Find the Promised Land ? " *Proc. of Multimedia conf*, 2004

[4] Paul Browne, Alan F Smeaton, Noel Murphy, Noel O'Connor, Seán Marlow, and Catherine Berrut, "Evaluating and Combining Digital Video Shot Boundary Detection Algorithms," *Proc. of Irish Machine Vision and Image Processing Conference*, 2000

[5] Juan Casares, A. Chris Long, Brad A. Myers, Rishi Bhatnagar, Scott M. Stevens, Laura Dabbish, Dan Yocum, and Albert Corbett, "Simplifying Video Editing Using Metadata," *Proc. of Designing Interactive Systems (DIS)*, 2002

[6] Scott Counts, and Eric Fellheimer, "Supporting Social Presence through Lightweight Photo Sharing On and Off the Desktop," *ACM CHI*, 2004.

[7] Glorianna Davenport, Thomas Aguirre Smith, and Nstalio Pincever, "Cinematic primitives for multimedia," *IEEE Computer Graphics and Applications*, pages 67–74, July 1991.

[8] Alberto Frigo, "Storing Indexing and Retrieving My Autobiography," *In Pervasive 2004 Workshop on Memory and Sharing of Experiences*, 2004.

[9] U. Gargi and R. Kasturi, “An evaluation of color histogram based methods in video indexing,” *International Workshop on Image Databases and Multimedia Search*, 1996.

[10] Jim Gemmell, Gordon Bell, Roger Lueder, Steven Drucker and Curtis Wong, “MyLifeBits: Fulfilling the Memex Vision,” *ACM Int’l Conf. on Multimedia*, 2002.

[11] Andreas Girgensohn, John Boreczky, Patrick Chiu, John Doherty, Jonathan Foote, Gene Golovchinsky, Shingo Uchihashi, and Lynn Wilcox, “A Semi-automatic Approach to Home Video Editing,” *ACM UIST*, 2000.

[12] Tero Jokela, “Authoring tools for mobile multimedia content,” *IEEE Int’l Conf. on Multimedia and Expo (ICME)*, pages 637–640, 2003.

[13] Nicky Kern, Bernt Schiele, Holger Junker, Paul Lukowicz, Gerhard Tr¨oster, and Albrecht Schmidt, “Context Annotation for a Live Life Recording,” *In Pervasive 2004 Workshop on Memory and Sharing of Experiences*, 2004.

[14] Anita Kumlodi, Gary Marchionini, “Key frame preview techniques for video browsing,” *ACM International Conference on Digital Libraries*, pages 118–125, 1998.

[15] Eyal de Lara, Rajnish Kumar, Dan S. Wallach, and Willy Zwaenepoel, “Collaboration and multimedia authoring on mobile devices,” *Int’l Conf. on Mobile Systems, Applications and Services (MobiSys)*, 2003.

[16] N. Marmasse, C. Schmandt, D. Spectre, “WatchMe: Communication and Awareness between Member of a Closely-Knit Group,” *Proc. of Ubicomp 2004,* 2004.

[17] MPEG Industry Forum, http://www.m4if.org

[18] Marcus Nilsson, Mikael Drugge, Peter Parnes, “Sharing Experience and Knowledge with Wearable Computers,” *In Pervasive 2004 Workshop on Memory and Sharing of Experiences*, 2004.

[19] Pervasive 2004 Workshop on Memory and Sharing of Experiences.

[20] Risto Sarvas, Erick Herrarte, Anita Wilhelm, and Marc Davis, “Metadata creation system for mobile images,” *Int’l Conf. on Mobile Systems, Applications and Services (MobiSys)*, June 2004.

[21] Cees G.M. Snoek, and Marcel Worring, “Multimodal video indexing: a review of the state-of-the-art,” *Multimedia Tools and Applications*, 2004 (In Press).

[22] Yasuyuki Sumi, Sadanori Ito, Tetsuya Matsuguchi, Sidney Fels, and Kenji Mase, **“**Collaborative capturing and interpretation of experiences**,”** *Proc. of Advances in Pervasive Computing*, 2004.

[23] Chao-Ming Teng, Hao-hua Chu, and Chon-In Wu, “mProducer: Authoring Multimedia Personal Experiences on Mobile Phones,” *IEEE Int’l Conf. on Multimedia and Expo (ICME 2004)*, 2004.

[24] S. Vemuri, C. Schmandt, W. Bender, S. Tellex, B. Lassey, “An Audio-Based Personal Memory Aid,” *Proc. of Ubicomp 2004,* 2004.

[25] Wei-Qi Yan and Mohan S Kankanhalli, “Detection and removal of lighting & shaking artifacts in home videos,” *ACM international conference on Multimedia*, pages 107–116, Dec 2002.

[26] Hong-Jiang Zhang, Chien-Yong Low and J.-H. Wu, “Video parsing, retrieval and browsing: An integrated and content-based solution,” *ACM international conference on Multimedia*, 1995.