CaMus^2 – Collaborative Music Performance with Mobile Camera Phones

Michael Rohs
Deutsche Telekom Laboratories
TU Berlin
Ernst-Reuter-Platz 7
10587 Berlin, Germany
michael.rohs@telekom.de

Georg Essl
Deutsche Telekom Laboratories
TU Berlin
Ernst-Reuter-Platz 7
10587 Berlin, Germany
georg.essl@telekom.de

ABSTRACT

CaMus^2 is a multi-user multi-phone extension of the CaMus system. Mobile camera phones use their cameras to track position, rotation, height, and other parameters over a marker sheet to allow interactive performance of music. Multiple camera phones can use the same or separate marker sheets and send their interaction parameters via Bluetooth to a computer where the sensor information is converted to MIDI format to allow control of a wide range of sound generation and manipulation hardware and software. The semantics of the mapping of MIDI message to performance parameters of the camera interactions are fed back into the visualization on the camera phone.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*input devices and strategies, interaction styles, screen design*; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems—*Audio input/output*; H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces—*collaborative computing, synchronous interaction*

General Terms

Design, Human Factors

Keywords

Mobile phones, camera phones, music performance, collaboration, camera-based interaction, small displays, spatially aware displays

1. INTRODUCTION

Mobile technology is a common place commodity and often these devices have vast potential for both sound generation and for new modes of interaction. The role of mobile phones as entertainment devices becomes increasingly important, as reflected in better audio codecs for high-quality music reproduction and higher storage capacities. The widespread availability of mobile phones enables the ad-hoc collaboration of non-expert music performers. Yet there are few examples of attempts to turn mobile devices into interactive music performance devices.

CaMus is an implementation of an interactive paradigm for mobile camera phones which allows interactive performance and flexible mapping to arbitrary MIDI capable sound software and hardware [12]. A camera phone is tracked over a marker sheet and serves as a spatially aware display [4] into a virtual interaction space. The marker sheet provides a fixed reference frame for the virtual space and the movable device display is used as a dynamic peephole [9]. Multiple sounding elements, including instruments and sound effects, can be freely moved in this space. The position and orientation in the physical space controls parameters of the sounding elements. The actual sound generation takes place on a nearby PC that is connected via Bluetooth. The basic mapping of position and orientation parameters to music is predefined with off-the-shelf MIDI sound synthesis software.

In this paper we describe how the basic CaMus system was extended to allow simultaneous and collaborative performance of multiple phones. We call the collaborative version *CaMus*^{2}. Multiple camera phones connect to a Bluetooth enabled PC that then maps camera interaction parameters to MIDI format controls, which can be easily mapped to a wide variety of MIDI-enabled software and hardware. In addition CaMus^2 allows the mobile phones to be informed about the semantics of the mapping of interaction parameters to sound generation and modification parameters and can now share, modify and display this information. On the technical side we have enabled the use of dynamic digital zoom to increase the range of the height parameter and extended the grid surface area.

Mobile technology for music performance has been used in various ways. Most of these have so far been with an emphasis on playback [1, 2, 8, 16, 17, 18], and mobile music interactions with a strong input component are yet in their infancy [7]. An example of an interaction paradigm based on mobile technology is GpsTunes [14] where walking navigation is supported by variation in musical playback. It in turn is closely related to the Sonic City project, which however does not use a mobile device for sensing [6]. "miniMIXA" from SSEYO (www.sseyo.com) is a music mixer for mobile devices. However, it does not use camera-based input.

ACE'07, June 13–15, 2007, Salzburg, Austria.

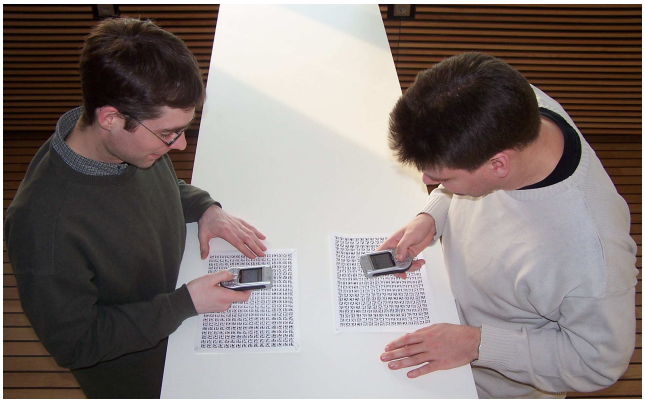

Figure 1: Setup of a collaborative CaMus performance.

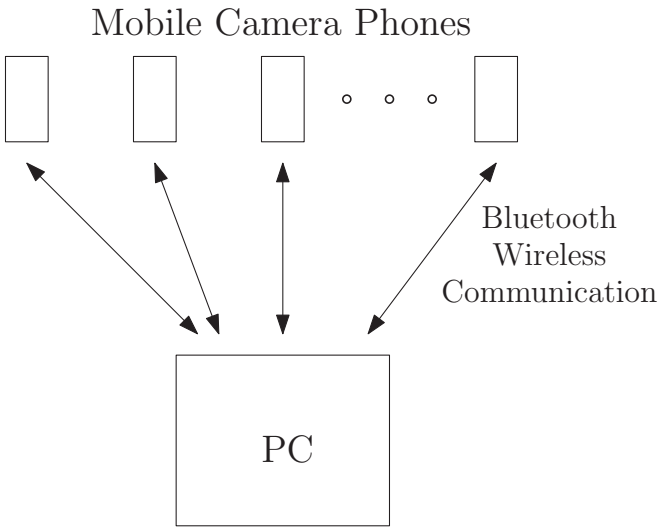

Figure 2: Bluetooth wireless network between mobile camera phones running the marker recognition software and the personal computer running MIDI based sound synthesis software.

Mobile performance is however a strong emerging community [5] where new frameworks are starting to emerge [15]. A recent example of mobile technology being center of a very interactive piece was the Pocket Gamelan technology where mobile devices were suspended on cords [13].

2. MAKING CAMUS COLLABORATIVE

Musical instruments often are played together, making the music playing experience a collaborative one. The joint playing can either be supported by common musical scores or knowledge of a musical piece, or mediated by interpersonal communication in a chamber music setting or a conductor in a large ensemble setting. A number of new musical instruments have been designed with the primary concern of collaboration in mind, see [10, chapter 2.7] for a review.

Mobile technology often comes with inherent communication methods and collaborative music making can use these methods to enable or support performing together. The original CaMus system could in principle be played together, but there was no inherent support for joint activity and the infrastructure was strictly separate. In the original CaMus system the virtual planes of interaction in a sense constitute a score. Yet when two such systems are played together there is no inherent support for joint structure of the score or playing different aspects of a synchronized score.

The goal of making CaMus collaborative was to structurally support joint performance through mobile interactions. We implemented two features in order to achieve this: (1) a networking protocol that allows to change, share and join CaMus performance information over the wireless networking capability of the mobile camera phones and (2) allow the exchange of semantic information between participants in the networked setup to give each participant meaningful information about the context and the dynamics of the music performance.

Bluetooth (www.bluetooth.org/spec) allows multiple simultaneous connections to an RFCOMM server. In order to use this to make CaMus collaborative, we extended the receiver software on the personal computer from the original architecture [12] to allow multiple serial Bluetooth connections to be used. The computer opens and registers the serial devices as needed upon startup in the local SDP database. The mobile camera phones then get to select which of the found serial channels are to be used for communication of this particular phone with the computer. Each phone retains the channel as identifier for itself within the network to communicate settings with the PC and other devices. All connections currently go through the PC (see Figure 2) but direct communication between mobile camera phones through the same serial Bluetooth protocol is planned.

The interaction parameters received from the various camera phones via the serial Bluetooth channels are then in turn converted into MIDI channels and can be used by any MIDI-enabled software and hardware. Parameters can be mapped to the same or separate sounds or effects and hence performers can jointly manipulate one sound or contribute different sounds or effects through the same architecture.

The PC usually is used to support semantic exchange with the mobile phones. As the mapping of interaction parameters to MIDI data and then sounding results is ultimately placed on the PC, the PC also has to serve the mobile camera phones with this information to provide appropriate visual feedback to the user. The PC provides this data before a performance starts to each participant in the network. Furthermore mobile camera phones can send changes in semantics back to the PC to allow editing of this information and to allow sharing it with other participants in the network. The protocol is in principle set up to allow direct communication between any participant in the network in preparation for performance pieces that solely rely on the mobile devices for all aspects of the performance.

The protocol is bidirectional. Each participant can send or request information and no participant takes in principle a privileged role. Dedicated roles are however possible, as for example a PC can serve as a shared host for semantic information. Each communication starts with an opcode that identifies the function of the exchange. Upon connection a new participant informs the other end of a connection link of its identity using the CONNECT opcode. Thereafter either side of a connection can send performance data via SEND_MOVEMENT and SEND_PARAMETERS opcodes. Using the REQUEST_SEMANTICS opcode any

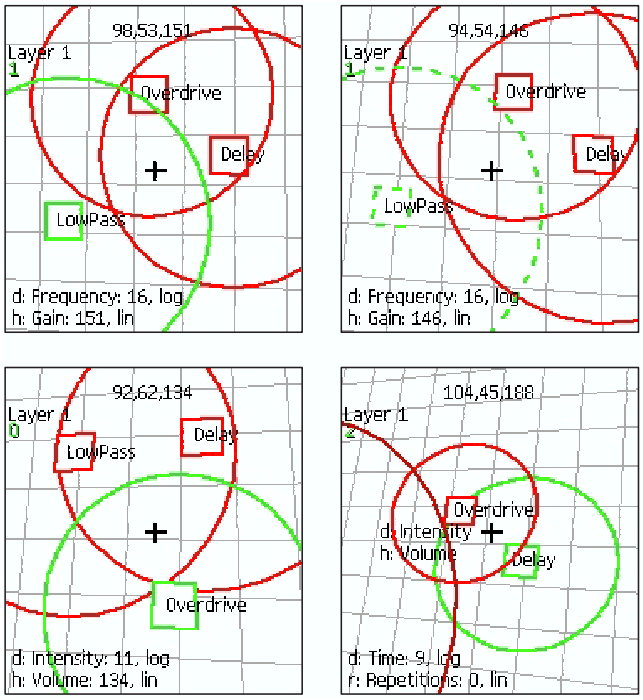

Figure 3: Three different filters (low pass, delay, and overdrive) with their semantics are located at different places in the workspace.

participant can request semantic information of other participants in the network, to which either SEND_SEMANTICS or NO_SEMANTICS is returned, depending whether the respondent has the information available. On the fly editing of sounds or effects can be shared in similar fashion. For example DELETE_TARGET informs a peer in the network that a target was removed from the performance space.

3. SOUND SPACE VISUALIZATION ON SMALL DISPLAYS

Visualizing interactions on a mobile device is challenging due to the limits of the available display area. In the original CaMus system we implemented a number of visualization strategies to help guide interactions in a virtual plane using virtually placed targets and circular arcs called halos to indicate their range of influence [12]. In experiments we showed the general efficiency in comparable mobile navigation tasks [11] but it also indicated needs for extensions. Specifically additional semantic information is needed to make the mapping of interactions with the camera phone to sounding events more accessible to the performer. Furthermore the experiments showed that the physical limits of the vertical interaction impacted the performance of users in interactions. To this end we extended the technology to increase the vertical interaction range.

3.1 Visualization of Target Semantics

In order to improve the visualization on the camera phone display we introduce semantic information and a communication protocol to share this information between participants in the Bluetooth network. The semantic information carries the mapping of interaction parameters from the tar-

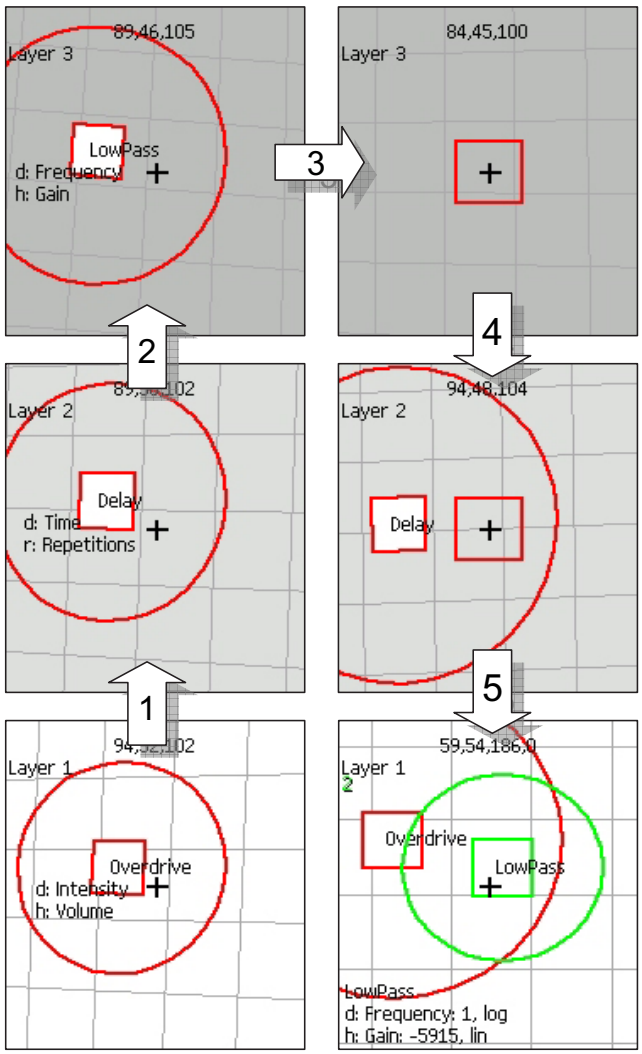

Figure 4: The workspace consists of three layers: Navigating through layers is done by vertical movement gestures. Targets can be dragged between layers.

get detection system to sounding sources and sound manipulation parameters.

The visual targets of the original CaMus system now will display a textual description of the function of the target. For example a target may represent a low-pass filter (see Figure 3). In this case the relevant semantic information can be presented inside the target square by the visualization.

In addition for each target, semantic information of the control parameters related to the target can be exchanged. These control parameters are [12] x and y distance from the target, Euclidean distance d from the target, height h over the target sheet, rotation angle α, and tilt angles θ_x and θ_y relative to the x- and y-axis, respectively. Each of these can be given a semantic name. For example distance d can be named "Volume". Furthermore, the raw input data can be mapped to a new value range. For example height may be between 40 and 160 but the control parameters used by the MIDI software are 0-127. Through the exchange the mobile phone is informed what the mapping ranges are, and if the

mapping is linear or logarithmic. This information can be requested for any participant from the network. Currently the personal computer serves to host and provide this information, but the protocol in use is designed to allow exchange and modification of this data by all participants in the network.

Rendering a complex workspace can put a lot of computing load on the mobile device. However, there are initiatives for standardizing scalable graphical user interfaces for mobile devices, for example *SVG Tiny* (www.w3.org/TR/SVG-Mobile) and *OpenGL ES* (www.khronos.org/opengles). If graphics coprocessors are capable of rendering complex scenes, the load on the main processor can be reduced. With the workspaces we displayed (see Figure 3), consisting of a couple of targets, some text and the background grid, the devices we used had no problems.

3.2 Interaction on Multiple Layers

We found that it is beneficial to divide the workspace into multiple layers that are stacked upon each other. Each layer can have its own targets. Only the targets of the current layer are manipulated in an interaction. The settings of the other layers remain unchanged. This allows users to group conceptually related items on the same layer and to interact with them independently, without influencing other targets at the same time. For example, instruments could be moved to one layer and sound effects to another.

Users can navigate to the next higher layer by making a quick up-flick gesture with the phone and to a lower layer via the same gesture in the opposite direction. Objects can be dragged to other layers by selecting them with the joystick button, keeping the button pressed and doing the flick-gesture. A sequence through the three layers is illustrated in Figure 4. Initially, one target is located in each layer. The user is initially located at layer 1 (left bottom screenshot). With two flick-up gestures (arrows 1, 2) layers 2 and 3 are reached in sequence. The user now selects the target (arrow 3), drags it down to layers 2 and 1 (arrows 4, 5), and finally releases it at layer 1.

4. EXTENDED GRID TRACKING WITH DYNAMIC DIGITAL ZOOM

Camera phones are tracked over a grid of visual markers that provides a fixed frame of reference for the virtual workspace within which the user interacts. The approach is an extension to the one described in [12]. The absolute position and orientation of the device within the physical space above the grid can be tracked with low latency and high precision. While moving within the space the device display is continuously updated in real time and virtual objects seem to have a fixed position in physical space. The graphics is rendered perspectively, as if looking through the device onto the background surface (see Figure 5). The device acts as a window into the virtual workspace. This setup is known as a *spatially aware display* [4, 19]. The space can be shared by multiple phones or multiple spaces with different semantics can be set up. Interaction takes place single-handed, as is in line with current mobile phone interaction styles.

4.1 Extended Tracking Surface

The grid defines a global coordinate system, with one unit corresponding to a single black-and-white cell (*ccu* = *code coordinate unit*). Each marker has a size of 6×6 ccu, spaced in a regular array of 8×8 ccu. Each marker stores its own index position within the grid. The markers in the top row (left-to-right) have index positions (0,0), (0,1), etc. From the index position (i, j) the grid coordinates of the marker corners are computed (see Figure 5a): left-top $(8i, 8j)$, right-top $(8i + 5, 8j)$, right-bottom $(8i + 5, 8j + 5)$, left-bottom $(8i, 8j + 5)$. Each marker has a raw capacity of 16 bits. With 2 parity bits this leaves 2×7 bits to store index coordinates. The maximum grid size is thus 128×128 markers or 1024×1024 ccu. Suitable printing sizes are 1.5 mm to 2.0 mm per black-and-white cell, which yields a maximum grid area of 1.54 m to 2.05 m.

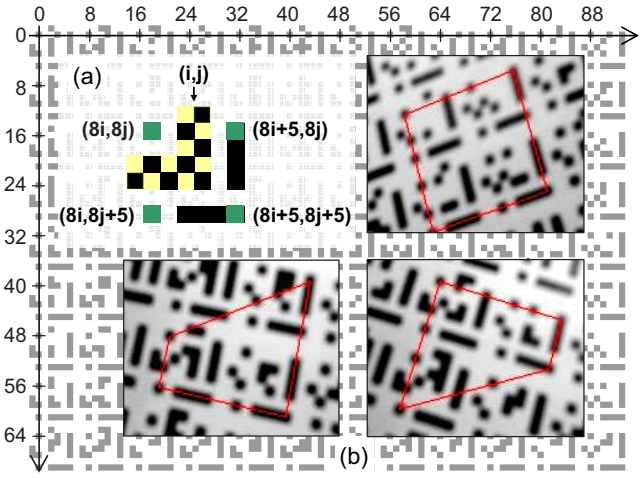

Figure 5: The grid defines a global coordinate system for absolute positioning. (a) Each marker stores its index position in the grid (i,j) from which the grid coordinates of the corners are computed. (b) The homography is computed from the largest possible area.

The size of individual markers and their spacing in the grid is such that uniform position and orientation detection during movement are ensured. The markers are small enough that at any focus point on the surface, at least a single marker is completely contained in the camera view. The grid also provides the basis for perspective rendering of the workspace.

A perspective mapping (planar homography) between the camera image coordinate system and the grid coordinate system requires at least four points whose coordinates are known in both coordinate systems. The corners of the markers serve for this purpose, since we know their position in the camera image and, via the index position of the marker, their position in the grid. However, since the markers appear very small in the camera image, basing the perspective mapping on the features of a single marker would yield a very unstable rendering. We thus use features of potentially different markers that are closest to the image corners, in order to base the perspective mapping on the largest possible area. The red lines in Figure 5b show on which points the mapping is based in different situations.

4.2 Extended Tracking Distance

In the original implementation, the vertical tracking range (the distance of the camera lens to the grid) was limited to

between 2 and 10 cm. This proved insufficient for effective interactions along the z-dimension. In the current extension we use the digital zoom feature that is present in many camera phones to substantially extend the vertical tracking range.

In contrast to optical zoom, which changes the optical characteristics of the lens system and thus influences image formation before the light reaches the optical sensor, digital zoom manipulates an image after it has been acquired by the sensor. Digital zoom increases the apparent focal length at which an image was taken by cropping an area at the image center with the same aspect ratio as the original image. The cropped area is rescaled to the original dimensions by interpolation. No optical resolution is gained in this process, but digital zoom is done by the camera before any compression and does not have to be done by the main processor of the device, it essentially gives high-quality rescaling for free.

The Symbian camera API allows to set the digital zoom level between 0 and some device-dependent maximum value. In an experiment we kept the physical distance to an object in the camera view constant, continuously changed the digital zoom level, and measured the size at which the object appeared in the camera view ($size_{zoomed}$). We found that $\text{size}_{\text{zoomed}} = \text{size}_{\text{unzoomed}} \times e^{k \times \text{level}}$, or equivalently $\text{distance}_{\text{zoomed}} = \text{distance}_{\text{unzoomed}} \times e^{-k \times \text{level}}$. For Nokia the 6630 (6× digital zoom) $k = 0.0347$ ($R^2 = 0.9983$), for the Nokia N70 (20× digital zoom) $k = 0.0386$ ($R^2 = 0.9992$), and for the Nokia N80 (20× digital zoom) $k = 0.0345$ ($R^2 = 0.9974$).

During grid tracking, digital zoom is continuously adjusted, such that markers appear in a size that is best suited for detection. The algorithm tries to keep the size at which markers appear in the camera image constant. If no markers are detected in a camera frame, a different zoom level is chosen. The algorithm is complicated by the fact that changes to the zoom level via the camera API do not result in immediate changes in the next camera frame. Instead, the new digital zoom setting becomes valid only 2 to 5 frames after the adjustment is made. Therefore, the algorithm chooses the setting that is most likely to yield smooth distance changes. The details of the algorithm are given below. The frame index (*frame*) is a sequence number for frames from the camera. The zoomed distance (*distZoom*) is provided by the marker recognition system, if at least one marker is detected in the camera image. It depends on the current digital zoom setting.

```
global state:

int level;
int levelNew, levelNewFrame;
int d, dFrame;
int dPrev, dFramePrev;


void AdjustDigitalZoom(int frame, int distZoom)
{
  if (no markers found) {

    if (more than N frames without markers) {

      // try different zoom level
      levelNew = (level + Range / 4) mod Range;

      // new zoom level not immediately valid
      camera->SetDigitalZoom(levelNew);

      // guaranteed to be valid 7 frames from now
      levelNewFrame = frame + 7;

    }

  } else {

    distNoZoom = distZoom / exp(-k level);

    if (level ≠ levelNew) {

      // in transition to new level
      dNoZoomNew = distZoom / exp(-k levelNew);

      // extrapolate distance based on old values
      dExtrap = d+(d-dPrev)/(dFrame-dFramePrev);
      deltaOld = abs(dExtrap - distNoZoom);
      deltaNew = abs(dExtrap - dNoZoomNew);

      // choose new one if it is closer
      if (deltaNew < deltaOld or
          frame ≥ levelNewFrame)
      {
        distNoZoom = dNoZoomNew;
        level = levelNew;
        if (levelNewFrame > frame) {
          // define how long to keep new setting
          levelNewFrame = frame + 1;
        }
      }
    }

    // remember previous distance value
    dPrev = d;
    dFramePrev = dFrame;

    // compute the new unzoomed distance value
    d = (3 distNoZoom + dPrev) / 4;
    dFrame = frame;

    if (frame ≥ levelNewFrame) {

      // distZoom valid
      distNoZoom = distZoom / exp(-k level);

      // set zoom level to ideal value
      levelNew = -1/k ln(dZoomIdeal/distNoZoom);

      // new zoom level not immediately valid
      camera->SetDigitalZoom(levelNew);

      // guaranteed to be valid 7 frames from now
      levelNewFrame = frame + 7;

    }
  }
}
```

4.3 Current State of the Implementation

The CaMus^2 system has been implemented for Symbian S60 devices. On a Nokia 6630 or a Nokia N70, markers are recognized in view finder mode with a frame size of 176×144 pixels at a rate of 10-15 frames per second, depending on the complexity of the rendered scene. With the dynamic digital zoom method the vertical recognition range for a grid with a cell size of 1.5 mm increases from 10 cm to 30 cm for a Nokia 6630 and to 50 cm for a Nokia N80. Even though we use fixed focus cameras the method works reliably over the full distance range, since the marker recognition algorithm is tolerant to out of focus (blurred) images. At larger distances the perspective mapping of the grid gets more unstable, which could be mitigated by filtering and smoothing the perspective mapping.

Overall, the marker grid provides a fixed reference frame for visual tracking in a 3-D physical interaction space of 150 × 150 × 30 cm. Multiple devices can be tracked with low latency and high precision. Since each device determines its

position on its own, the method is scalable to a large number of devices. Moreover, the approach runs on off-the-shelf camera phones and does not require additional hardware.

5. CONCLUSIONS

In this paper we presented the extension of the CaMus system for performance of music with mobile camera phones for multiple users. Multiple camera phones communicate with a personal computer through a Bluetooth network sending performance parameters derived from a visual tracking system. This information is then mapped to MIDI to connect to arbitrary MIDI-enabled sound software or hardware.

In order to enhance the contextual information for the performer, we have added semantic information that can be shared by all participants in the network and displayed on the mobile camera phone's visualization. Different aspects of the performance can be placed on multiple separate layers. This way performers can separate sounding functions from effect functions and can choose to share a common performance space or separate it as needed for a given performance context.

Technologically we have extended the range of the height allowable by the system through the use of dynamic digital zoom to improve the possible performance. This is in part result of an interface study which showed that the confinement of the vertical movement was a limiting factor [11].

As immediate future work we plan to implement the sound synthesis and manipulation engine completely on the mobile device itself and hence remove the need for a personal computer to serve as the sound source. This has already been prepared by the portation of the sound synthesis toolkit STK to Symbian OS [3]. As the mobile device itself defines the mapping semantics of interaction parameters to sounding results, we plan to implement editing of this information on the phone itself. At the same time, the communication will be extended to allow direct exchanges between multiple mobile devices in the absence of a personal computer. Through this network semantics information is planned to make sharable among all users of the CaMus^2 system and allows each phone to visualize the performance context of all participants interactively and while this context is edited on the fly.

6. REFERENCES

[1] F. Behrendt. *Handymusik. Klangkunst und 'mobile devices'*. Epos, 2005. Available online at: `www.epos.uos.de/music/templates/buch.php?id=57` (retrieved on April 1, 2007).

[2] W. Carter and L. S. Liu. Location33: A mobile musical. In *NIME '05: Proceedings of the 2005 Conference on New Interfaces for Musical Expression*, pages 176–179, May 2005.

[3] G. Essl and M. Rohs. Mobile STK for Symbian OS. In *Proc. International Computer Music Conference*, pages 278–281, New Orleans, November 2006.

[4] G. W. Fitzmaurice. Situated information spaces and spatially aware palmtop computers. *Commun. ACM*, 36(7):39–49, 1993.

[5] L. Gaye, L. E. Holmquist, F. Behrendt, and A. Tanaka. Mobile music technology: Report on an emerging community. In *NIME '06: Proceedings of the 2006 conference on New interfaces for musical expression*, pages 22–25, June 2006.

[6] L. Gaye, R. Mazé, and L. E. Holmquist. Sonic City: The urban environment as a musical interface. In *NIME '03: Proc. 2003 conference on New interfaces for musical expression*, pages 109–115, May 2003.

[7] M. Kaltenbrunner. Interactive music for mobile digital music players. In *Inspirational Idea for the International Computer Music Conference (ICMC)*, Barcelona, Spain, Sept. 2005.

[8] G. Levin. Dialtones - a telesymphony. `www.flong.com/telesymphony`, September 2 2001. Retrieved on April 1, 2007.

[9] S. Mehra, P. Werkhoven, and M. Worring. Navigating on handheld displays: Dynamic versus static peephole navigation. *ACM Trans. Comput.-Hum. Interact.*, 13(4):448–457, 2006.

[10] E. R. Miranda and M. M. Wanderley. *New Digital Musical Instruments: Control and Interaction Beyond the Keyboard*. A-R Editions, 2006.

[11] M. Rohs and G. Essl. Which one is better? – information navigation techniques for spatially aware handheld displays. In *ICMI '06: Proceedings of the 8th International Conference on Multimodal Interfaces*, pages 100–107, Nov. 2006.

[12] M. Rohs, G. Essl, and M. Roth. CaMus: Live music performance using camera phones and visual grid tracking. In *Proceedings of the 6th International Conference on New Instruments for Musical Expression (NIME)*, pages 31–36, June 2006.

[13] G. Schiemer and M. Havryliv. Pocket Gamelan: Tuneable trajectories for flying sources in Mandala 3 and Mandala 4. In *NIME '06: Proceedings of the 2006 conference on New interfaces for musical expression*, pages 37–42, June 2006.

[14] S. Strachan, P. Eslambolchilar, R. Murray-Smith, S. Hughes, and S. O'Modhrain. GpsTunes: Controlling navigation via audio feedback. In *MobileHCI '05: Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, pages 275–278, Sept. 2005.

[15] A. Tanaka and P. Gemeinboeck. A framework for spatial interaction in locative media. In *NIME '06: Proceedings of the 2006 conference on New interfaces for musical expression*, pages 26–30, June 2006.

[16] N. Warren, M. Jones, S. Jones, and D. Bainbridge. Navigation via continuously adapted music. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1849–1852, Apr. 2005.

[17] D. Waters. Mobile music challenges 'iPod age'. BBC News Online, Monday, 7 March 2005. Available online at `news.bbc.co.uk/1/hi/technology/4315481.stm` (retrieved on April 1, 2007).

[18] T. Yamauchi and I. Toru. Mobile user-interface for music (poster). In *Proceedings of the Mobile Music Workshop*, Vancouver, Canada, May 25 2005.

[19] K.-P. Yee. Peephole displays: Pen interaction on spatially aware handheld computers. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1–8, Apr. 2003.