Optical Motion Detection as Interaction Technology for Mobile Phones

Optical Motion Detection as Interaction Technology for Mobile Phones

Physical interaction with a mobile device is a crucial design step in mobile device hardware and software development. Especially mobile phones have small keys and mass market phones are usually equipped with a sub-QVGA-screen. Therefore the possibilities for concepts of interaction with the device are limited to some extent. There are a few established interaction techniques that have settled troughout the mobile device market. Namely these are Keypads, Touchscreens and Voice Recognition.

Classification of Input Techniques

There are a few main parameters that define the usability of those interaction techniques. The **reaction time** between the user input and the response on output devices such as the display is a very crucial parameter. Any visual response on the output device after a few 100 ms is not interpreted as a direct reaction to the user, but as a separate event. The **quantity** of actions a user is able to perform using a specific input technique defines the speed at which he can interact with the device. The **intuitivity** of an input method strongly affects the usability and user acceptance.

Pros and Cons of Common Input Techniques

The **keypad** is the most often used interaction channel on mobile phones, many devices are accessible throuch keypresses only. The Metaphor of pressing keys is from a hardware and software point of view well developed and standardized nowadays. Keys offer a very short reaction time but mostly lack intuitivity. If the concept of the key usage is consistent and once learnt by the user the input quantity is very high. **Voice recognition** is a very intuitive input technique, but today algorithms lacks fast reaction time, input quantity and also reliability. **Touch screens** can be a very intuitive way of interacting with a screen based mobile device. The reaction time is comparable with key input but the input quantity strongly depends on the UI design. The reaction time on devices nowadays is comparable with key input. Though being a pretty neet input technique touch screens only make it into high end phones, because of their high production costs comparedto key input. Each of these techniques has mentionable advantages. Key interaction offers a good reaction time. Voice recognition is very intuitive to use. Touch Screens combine a fair amount of intuitivity and interan quantity.

A concept of a new intuitive Interaction Technology

In the times where computers were evolving people had to learn the physical language of the devices they built. Nowadays people are used to more intuitive devices, that can understand them, at least to some extent. Filling the last gap between human and machine would require the most intuitive way to interact with devices by copying interaction metaphors we use in everyday life. Using motion is one of the most natural ways to interact with our environment. If a device could sense and interpret the motions of the user this would be a very intuitive interaction method. Since there are not many mobile phones equipped with sensors being capable of doing that, one exception is the acceleration sensor on the Nokia 5500 for example, we have to use the standard hardware of the device. There are hardly any mobile phones purchasable that do not come with a built-in camera, we couold use for this purpose. The key of this concept is the user moving the mobile device which results in a moving video stream of the camera. The direction and velocity of the movement of the device can be calculated with a suitable algorithm and voilà ... our device can sense user motion and use them as an interaction technique. The obvious advantage over the other interaction methods is the high intuitivity, but also the feature of the high resolution of the input actions in contrast to the binary keypresses (pressed or not) for example.

The Projection Shift Analysis algorithm

We developed an algorithm, that qualifies for detecting motion information in an image sequence using a very low amount of CPU time and memory resources. In the first step the image is projected onto its x- and y-axis. That means all luminance values of each row is summed up in the vertical and each column in a horizontal projection. If the scene within an image sequence is moved vertically or horizontally the corresponding projection buffers will shift equally. For example an image sequence containing a horizontal panning shot from the left to the right will result in a horizontal projection buffer whose values will shift to the left over time. Thus our approach searches for the best matching shift between two projection buffers of two successive images. To estimate the best match we introduce a value called badness factor that is similar to a correlation factor and characterises the badness of a specific shift. A lower value indicates a higher probability for this shift to be correct. The algorithm calculates the badness factor for every reasonable shift through summing up all squared differences of the values of the compared projection buffers. The shift value used to calculate the smallest normalised badness factor is assumed to be the correct relative motion between the two compared images. Here is an example of the source code that shows the simplicity of this algorithm.

The following code lines contains the algorithm that calculates the best fitting shift of (e.g. the motion between) two projection buffers (either horizontal or vertical) of two successive images.

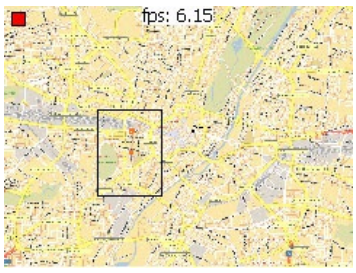
 (C) Copyright Nokia 2013. All rights reserved.

```
int CompareAccBuffers(unsigned long* acc_buffer_1, unsigned long* acc_buffer_2, int length, int max_shift) {
 int compare_length;
 unsigned long *buffer1, *buffer2;
 int i, shift, badness_delta, best_index=0;
 unsigned long badness_factor, best_badness_factor=0xFFFFFFFF;

 for (shift=-max_shift; shift<=max_shift; ++shift) {
  if (shift<0) {
   buffer1=acc_buffer_1-shift;
   buffer2=acc_buffer_2;
   compare_length=length+shift;
  }
  else {
   buffer1=acc_buffer_1;
   buffer2=acc_buffer_2+shift;
   compare_length=length-shift;
  }

  badness_factor=0;
  for (i=0; i<compare_length; ++i) {
   badness_delta=(*buffer2)-(*buffer1);

    // linear coorelation
   //badness_factor+=(badness_delta<0)?(-badness_delta):badness_delta;
    // square coorelation
   badness_factor+=(badness_delta*badness_delta)>>9; // shifting down the value avoids exeeding the max value of the badness_factor
   ++buffer1;
   ++buffer2;
  }

  badness_factor/=compare_length;

  if (best_badness_factor>badness_factor) {
   best_badness_factor=badness_factor;
   best_index=shift;
  }

  ++temp_shift_buffer;
 }

 return best_index;
}
```

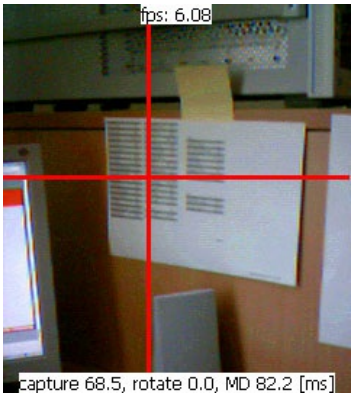
Application Case Studies

To test the concept of sensing motion detection through the camera we implemented a highly optimized algorithm called Projection Shift Analysis, which used very low memory and CPU resources. On top of that we built various applications to test the concept with common metaphors which require interaction:

Map Navigator is an application that allows the user to view and scroll a map on the screen. As simple as the scrolling problem might sound, the greater was the impact on the user experience to view different parts of the map by just moving or turning the device. All of the test persons instantly knew how to use the application, partly without even knowing what we wanted them to to with our application.

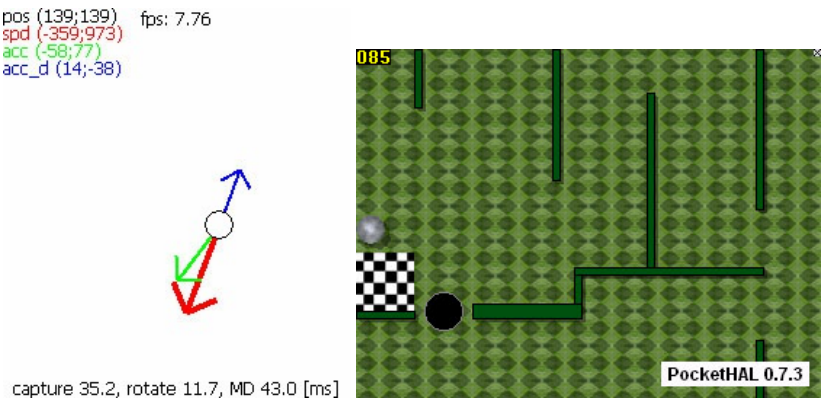


With **CameraCursor** we wanted to test if our concept is capable of controlling a cursor to support a fair interaction speed. The user is able to control a cross cursor in our application, again by just moving the mobile phone. By pressing and holding the appropriate button the cursor draws lines while moving on the screen.



 (C) Copyright Nokia 2013. All rights reserved.

Labyrinth Game is a remake of the good old wooden labyrinth game in which you can control a marble by tilting the play field. The nature of the game is very similar to the motion concept we developed.



TheBiggerPicture: There are already some tools and cameras that can combine several static images to one big panorame image for example, but this does not happen at "shoot"-time but afterwards. Using the motion detection the application can stitch together the video stream of the camera at run-time to generate "the bigger picture". By just moving the mobile phone into the desired direction you can do a custom size image. The usage of the application is perceived as a painting the screen with the video content by only moving the phone.

Resumée

The four described applications worked very well with the presented motion detection interaction concept. They also showed, that the concept offers fair amounts of all the above mentioned properties (reaction time, interaction quantity and especially intuitivity)

With the optimized algorithm it would be instantly possible to integrate this kind of input method to any device where the developer has access to the camera. The next step of the motion detection concept would be to recognize motion gestures on basis of the captured device motion, which is very similar to mouse gestures, that are used in the desktop version of the opera browser for example. The user can draw a virtual gestures in the air with his device, which executes a predefined command. In the last years the motion detection and gesture recognition metaphor has been used in some proprietary applications, but never made it to the mass market. Except for the Samsung SPH-S4000 and SCH-S400, which has an acceleration sensor built in, which is used for gesture detection on the devices.

The technology is there, but the mass market business cases are still not ready yet to incorporate the motion detection concept as a key form of interaction method. But providing it as an alternate form of interacting with a device along with other conventional forms could be a start for this concept.

References

[1] Paper submitted to the Permid 2005 in Munich: http://www.medien.ifi.lmu.de/permid2005/pdf/StephanDrab_Permid2005.pdf

[2] Presentation at the Permid 2005 in Munich: http://www.medien.ifi.lmu.de/permid2005/pdf/Presentation_StephanDrab_Permid2005.pdf

 (C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.