Master Project Report
Sonic Gallery

Ha Tran

January 5, 2007

Contents

1 Introduction

This project is inspired by an aesthetic idea in which instrumental sounds are mapped onto objects and contours. We implemented this idea within a building, creating a "smart gallery" where paintings and other artworks can interact with visitors, to experiment the effect of music on visitors.
Our goal is to develop a system for indoor sensing of conditions of proximity, movement, speed... in order to create a sense of interaction between humans and objects. Originated from the sensing system that works with outdoor sensing using the Global Positioning System (GPS), we want to implement a system that can also be used indoor that utilizes the power of sensor networks for the purpose of localization and data streaming. The final goal is to create a system that gives the users the aesthetic appreciation of the contour and the environment by the sound of music.
In order to deliver the desired system, we want to implement the idea on a smaller scale. In particular, we want to create a framework that can be used within the Rhode Island School of Design called Sonic Gallery.

2 Application description

The Sonic Gallery:
This hypothetical gallery will be completely indoor. There are about 10 - 15 paintings in this gallery. Visitors to this gallery will be automatically tracked by their locations with respect to the paintings. When the visitors come to a proximity of a painting, they will hear a sound of music depending on the painting. This effect is created to give the visitors of the gallery the sense of the space they are in.

3 Design

There are two subsystems of the application. The tracking and locating system is used to track the visitors of the gallery while the other sound system takes care of playing the pieces of musics depending on the location of the visitors.

3.1 SonicTrack - Indoor localization

While the GPS system provides a great way to track objects outdoor, the GPS signal cannot be traced from inside buildings. Therefore, we need to find a way to track an object inside a building. There are many ways to track a person indoor. Initially, we use Cricket[1], a sensor system with a sensor that can receive radio and sonic signals and other beacons broadcasting such signals. That approach seems to work but such a system was too costly to implement in a gallery. As a result, in our application, we used stereo calibrated networked cameras to track the position of a visitor

[1] MIT's indoor location system http://cricket.csail.mit.edu/

inside the gallery.
There have been published works for tracking people indoor with cameras[2]. The technique we use is similar: identifying visitors from 2 different cameras and triangulating the point to get the coordinates. However, instead of tracking the whole moving body of a person, we limit our search to a small portion of the body to make stereo triangulation accurate.
For tracking people using such cameras, we'll make the visitors to the gallery wear a light emission device (LED) with different colors. Then, when the LED point becomes visible for any pair of calibrated networked cameras, the coordinate of the visitor can be calculated. This coordinate is the coordinate of the visitor with one of the cameras. Depending on the location of the camera and its orientation, we can translate this coordinate to the actual coordinate of the visitor inside the gallery.

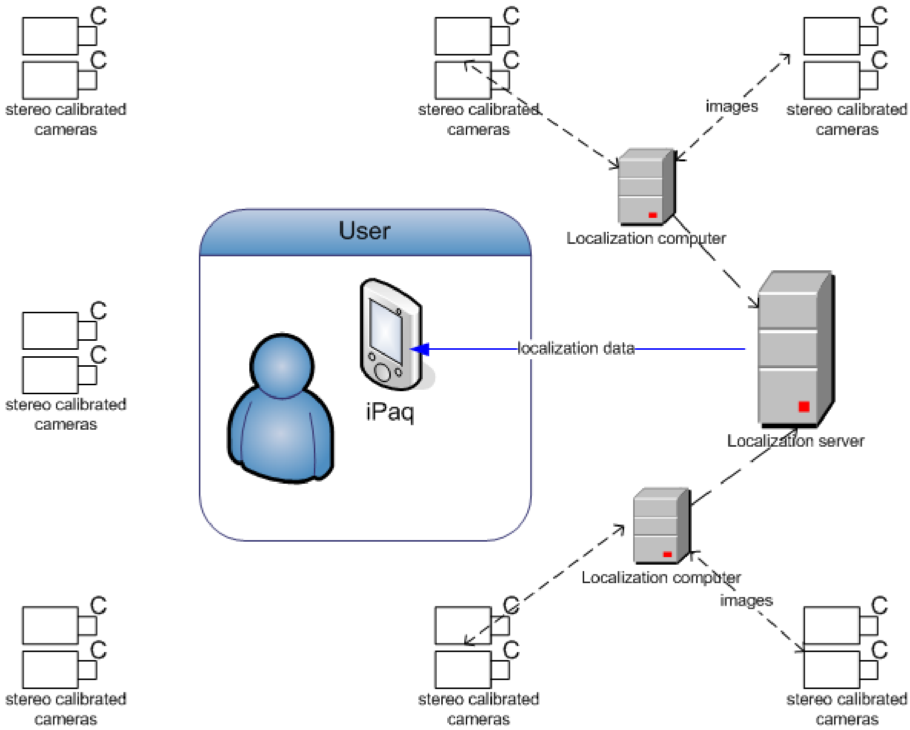
3.2 Client sound system

In order for the visitors to listen to different pieces of music based on their locations, there is a Pocket PC handed to each visitor of the gallery. The Pocket PC is the device for receiving the localization data and playing the corresponding pieces of music.
The Sonic Gallery is designed to be composed of artworks and conditions. This gives the gallery the robustness to adapt to many kinds of conditions depending on different kinds of paintings. It also gives the application the ability to add virtual paintings to the gallery. Moreover, the position history of visitors is stored so that their movement can also be tracked.
This system keeps communicating to a server to request the localization information of the visitors. Based on the information given, it will decide to play certain pieces of music. Different musical pieces can be played simultaneously with different volumes, giving the users the 3D effect of the sounds and the sensibility of the space they are in.

4 Architecture

The following diagram illustrates the major components of the project:

[2] http://www.research.ibm.com/peoplevision/3DTrack.html



The client sound system and the tracking system are connected through the localization server. As a result, if we use a different technology for indoor localization, the rest of the system remains unchanged. This is helpful for future development.

5 Implementation

5.1 Cameras calibration

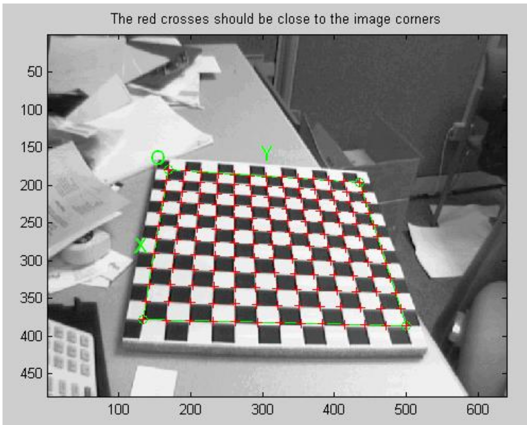
For every camera, we did internal calibration of the camera. Cameras close to each other and pointing parallely to the same direction are paired and called stereo cameras. Then, for each pair of such cameras, we did stereo calibration with them. Single and stereo calibrations are done with the help of the MATLAB Calibration Toolbox by Caltech[3].

5.1.1 Internal calibrations

Internal calibrations are done to measure the focal points, the skew factor, the distortion level, . . . etc. of each camera. This is done by taking a set of pictures of different angles of a chessboard (see figure below). The Calibration Toolbox then would assist in doing corner extraction and cali-

[3]Calibration Toolbox for Matlab http://www.vision.caltech.edu/bouguetj/calib_doc

brating the camera.

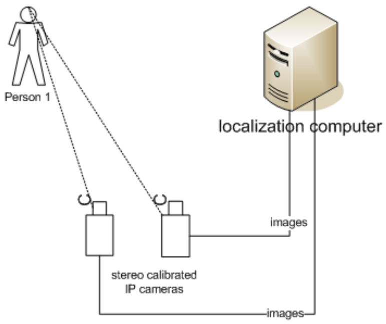


5.1.2 Stereo calibrations

With two cameras, each of which has been internally calibrated, we can do stereo calibration with them. This is done by taking a set of images of the chessboards visible to both cameras. In our application, we use the same set of images used for internal calibration. The calibration toolbox then can calculate the relative position of one camera with respect to the other.

5.2 Automatic Tracking System

By populating the space of the gallery with calibrated cameras, a visitor can be tracked as long as the tracking point is visible to a pair of calibrated cameras. In our application, we used a tennis ball with a clearly distinguished color and a limited visible area for both cameras. By doing so, the tracking poing will be limited to a relatively small area, making tracking more accurate.
Initially for each camera, we took the picture of the background, which is the empty room first. When a visitor comes, we can easily distinguish the body by doing a subtraction with the background, which in turn reduces significantly the search area for the tennis ball and helps improve the performance. When the same tracking point is identified from a pair of stereo calibrated cameras, triangulation is done to produce the localization information of that point (see figure below).
Since MatLab has a very good framework for handling images from networked camera, this system was written in MatLab.



Ideally, a separate computer is used for a pair of cameras to maximize performance. When the localization data is computed, it is then sent to a server storing the pool of coordinates of different visitors.

5.3 Sound system

This is the client side of the application where the users actually hear the music associated with the paintings. We use Microsoft Visual Studio .NET version 1.1 for Pocket PC to develop this application. We use FMod Sound System[4], an open source .NET system to play pieces of music. This system is very robust and extendable since it also supports 3D sounds for programming Pocket PC games. Therefore, this system can potentially support galleries with dynamic sound conditions. In addition, the Pocket PC application also allows the users to see where they are in the gallery and the information of the paintings.

6 System deployment

6.1 Placing the cameras

When deploying Sonic Gallery to an actual gallery, we need to place pairs of calibrated cameras in a way such that they cover the whole area of the gallery. The localization computers then can get the images from these cameras to compute the location of visitors. In our demo application, we only use one computer for computing and serving the locations to the clients.

6.2 Defining the gallery

For every different gallery, the client application needs to be aware of the environment of the gallery. We created a simple way of defining the galleries with paintings and the conditions of pieces of music that can be heard in accordance with the paintings. Defining a gallery can be done by creating an XML file with simple syntax. For example, the one we created has the content as follows:

[4] It can be found at http://www.fmod.org/

```
<?xml version="1.0" encoding="utf-8" ?>
<gallery name="Gallery one">
<artworks>
  <artwork name="painting one">
    <imagefile>\Storage Card\sonic\a.jpg</imagefile>
    <position x="5.0" y="12.0" z="10.0"/>
  </artwork>
  <artwork name="painting two">
    <imagefile>\Storage Card\sonic\b.jpg</imagefile>
    <position x="5.0" y="20.0" z="10.0"/>
  </artwork>
  <artwork name="painting three">
    <imagefile>\Storage Card\sonic\c.jpg</imagefile>
    <position x="5.0" y="30.0" z="10.0"/>
  </artwork>
</artworks>
<conditions>
  <condition type="IsWithinDistance" artwork="painting one">
    <distance>7</distance>
    <soundfile>\Storage Card\music\mozartotj.mp3</soundfile>
  </condition>
  <condition type="IsWithinDistance" artwork="painting two">
    <distance>7</distance>
    <soundfile>\Storage Card\music\mozarthorn.mp3</soundfile>
  </condition>
  <condition type="IsWithinDistance" artwork="painting three">
    <distance>10</distance>
    <soundfile>\Storage Card\music\mozartflute.mp3</soundfile>
  </condition>
</conditions>
</gallery>
```


7 Known problems and future work

7.1 Failure of tracking

When the object is not identified by the cameras in the gallery, there is a chance that the object is not present there any more or the cameras failed to spot the object. When the latter happens, we need a good prediction system to make the system more user friendly.

7.2 Scalability

This system scales perfectly in terms of the number of objects. However, there are only a limited number of colors that can be used to assign for the increasing number of objects, which is a bottle neck of this application. A different technology for localization like Cricket can be more useful.

7.3 Limitation of Pocket PC's compact framework

At this point, we are using Microsoft .NET Compact Framework to develop the Sonic Sound System on Pocket PCs. This framework is a compact version of the Microsoft .NET framework; and consequently, it has less functionalities than the original one. One of the disadvantages of this framework is the lack of support on multithreading. Handling threads in a Pocket PC application is known to be quite manual. Threads need to be handled more carefully. The application we built is multithreaded; nevertheless, there are some problems when a connection with the server is dropped. These need to be improved.

7.4 Other future work

In addition to making paintings interact with visitors, we want visitors to feel the interaction with one another through sonic effects. Also, as the project currently plays sounds based on the position of the visitor, we want to make the sound effect more dynamic with movements of the visitors in the gallery.

8 Appendices

8.1 Appendix A: Key components of Sonic Gallery's Pocket PC Client Sound System

There are three main packages for the Sonic Gallery project: FModSoundSystem, GalleryLib and MainPPC.

8.1.1 FModSoundSystem

This package is simply the open source package used by the application to play sounds on Pocket PC. The sound architecture of FMod allows multiple pieces of music to be played simultaneously by assigning different channels to different music files.

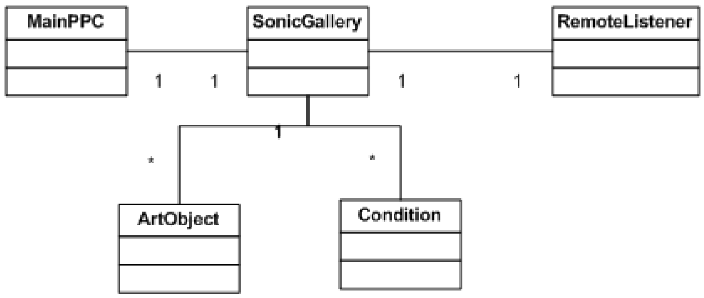
8.1.2 GalleryLib

This package contains the important classes used by Sonic Gallery: the class of the gallery object, the class of art objects and conditions ... The essential class in this package is SonicGallery. This is the class of the gallery object, which contains the art objects and the conditions of playing sounds. SonicGallery has a RemoteListener object, which is used to listen to new event from the localization server. An event is a notification of a new position of the visitor. When an event is received, all the conditions are checked to see which one is satisfied and a corresponding resolution for it is performed.

The second most important class in this package is the Condition class. In this application, Condition is designed abstractly enough to be extendable for more complex conditions. Derived from Condition is the Condition_WithinDistance, which is the condition we use for checking if a visitor is within a distance with a painting. By adding more conditions this way, the model is robust for creating other conditions such as "within a distance from the wall ..."

8.1.3 MainPPC

This package contains the Graphic User Interface (GUI) classes, which are what the users actually see and interact on the Pocket PC's screen. The main class in this package is MainPPC, which is the main form showing the positions of the paintings and the position of the specific visitor holding the Pocket PC. When the application is run, the object MainPPC is created. In addition, there are 3 more threads running: one for checking the changes in the latest visitor position and updating it on the screen, one for occasionally requesting the position of the visitor and the last one for the gallery object to be run, checking the conditions and playing sounds. The relationships between the classes are shown in the figure below:


MainPPC
SonicGallery
RemoteListener
ArtObject
Condition
1
1
1
1
1
*
*

8.2 Appendix B: How to deploy the Client Sound System onto a Pocket PC

In this project, we use Microsoft .NET Framework 1.1 for compiling the sources, Microsoft .NET Visual Studio 2003 as an IDE and Pocket PC model HP iPAQ 2003.
These are the steps to deploy the application onto the pocket PC:

- Connect the Pocket PC to a PC computer running Windows XP with either the serial port or the USB port.
- Open the source codes of the client Sonic Gallery application with Microsoft .NET Visual Studio 2003.
- Open the solution SonicGallery and build it.
- Run the application; it will ask whether you should deploy it on a Pocket PC emulator or an actual Pocket PC. Choose to deploy it on the Pocket PC.

After this, when the folder MainPPC is created in Program Files on the Pocket PC, copy the dll file fmodce.dll to this folder and redeploy the project again. It should work by then.
Be sure to copy the right version of fmodce.dll to the Pocket PC. Our dll file works with our Pocket PC models. For more details, check http://www.fmod.org/