CPSC 483: Senior Design

Critical Design Review

March 5, 2003

Wireless Robot Control with a Personal Digital Assistant

Ta-Chang Chao

Quang Dang

Matt Dean

Brady Wied

Introduction

The purpose of this Critical Design Review is to present and analyze both the current and projected states of our project. For each aspect of the project we have taken a step back and asked three questions: Where are we, where should we be, and where do we go from here? We are confident that we will successfully reach our goal in the near future, but to facilitate proper development and make sure that we are on the right track we need to analyze our progress so far.

The content of this design review is organized according the project's various modules, or subprojects. These are: Knowledge Development and Training, Porting ARIA to the Pocket PC, ARIA DLL Production, User Interface Development, Pocket PC to Robot Communication, and the Robot Application. We then present potential problems and sources of delay, our updated schedule, and final remarks concerning the project.

Subproject Progress and Planning

Knowledge Development and Training

During the course of this project, each of the team members has been faced with the challenge of learning a new development paradigm. This includes learning several programming languages and APIs along with the features and limitations of each. Most of our team's time to this point has been devoted to learning these new languages and techniques The main areas that we have had to educate ourselves on are outline below.

Embedded Visual C++ (eVC++)

Embedded Visual C++ (eVC++) is a subset of Microsoft's Visual C++ that is specifically designed for mobile devices running Windows CE including the Pocket PC device that we are using. The major limitations of eVC++ are that many of the dynamic libraries available on desktop machines are unavailable for Windows CE and that several common features of C++ such as RTTI and throwable errors are not implemented in the current version of eVC++ (version 3.0). These limitations were a significant problem during the porting of ARIA and will be discussed in more detail in the section describing the ARIA port.

Pocket PC Emulator

The Pocket PC Emulator has proven to be an invaluable tool during our development process as it allows each of us to write software for the iPAQ without having to use the actual hardware, but the emulator has several nuances that we have each had to become acquainted with.

Microsoft Foundation Classes (MFC)

The Microsoft Foundation Classes (MFC) is an API that is often used to develop windows-based applications for both PCs and Pocket PCs. We initially envisioned this as the main foundation for our user interface development, so each of the team members has spent some time learning how to program graphics using MFC. After further research our team has decided to use a different technology called GapiDraw (detailed below) for our user interface, but the knowledge gained about MFC will still be valuable to the team members.

ActivMedia Robotics Interface for Applications (ARIA)

ActivMedia Robotics Interface for Applications (ARIA) is a development API written in C++ that the robotics team at ActiveMedia, the makers of our robot, has provided for us. This API was created to facilitate object-oriented development of robot control applications and provides a large amount of functionality that our team will take advantage of during the development of our application. Each member of the team has spent time becoming familiar with the architecture and functionality of ARIA, which is quite extensive.

GapiDraw

GapiDraw is an efficient graphics library that provides DirectDraw-like development tools for both PC and Pocket PC environments. While the main focus of GapiDraw has traditionally centered around game interface development, our team has decided that the functionality that it provides will help us produce a user-friendly interface for our robot control application. We decided to use GapiDraw because of its ease of use, its computationally efficient implementation, the ability to easily design full-screen applications, the small size of the required library, and the fact that interfaces developed

on a PC can easily be converted to the Pocket PC environment. This decision was made within the past week and each team member is rapidly learning how to use the GapiDraw library.

Bluetooth communications protocol

We have also learned more about the Bluetooth communications protocol that our team will use to connect our robot application running on the Pocket PC to the hardware on the Amigo robot. The Bluetooth specification consists of over a thousand pages of information, and our team spent a large amount of time researching the capabilities of Bluetooth devices. Through this research we learned that we should be able to emulate a serial connection using two Bluetooth devices, one of which is embedded in our Pocket PC.

As mentioned previously, the majority of our team's time has been spent on this important step of research and experimentation to learn how to develop a robotics application for the Pocket PC. Each member of the team has been self-motivated and proactive during this learning process, even purchasing books on Windows CE and Bluetooth. The next step that we need to take as a team is to coordinate our knowledge and make sure that everyone understands the fundamental concepts. This aspect of the project is very important and will continue throughout our development process.

Porting ARIA to the Pocket PC

Our team realized at a very early stage of this project that ARIA provided a large amount of functionality that we would otherwise need to implement in order to control the robot. While the benefits of ARIA were obvious, it had not yet been ported to a Pocket PC. This posed a significant challenge because ARIA uses approximately 90 source files and 85 header files. As mentioned earlier, many common features of C++ were not implemented in the subset supplied by eVC++. Our team was also required to find alternate implementations of libraries used by ARIA, including the standard template library (STL) and Winsock.

During the porting process we made every effort to document our changes both within the code and in a separate document. We also tried to limit the number of changes that we had to make in order to retain full functionality. The changes that we have made to the source code will be made publicly available according to the ARIA licensing agreement, so one of the goals of the port was to maintain readability and extensibility.

Our team has successfully compiled and built ARIA for the Pocket PC, and several tests have been conducted using the robot simulator and the Pocket PC emulator to ensure that the port works. We have not created a comprehensive set of tests to make sure that there are not any problems with the port and do not currently plan on doing so, but we will analyze specific problems as they arise and create tests to ensure that any corrections we make will continue to operate properly.

ARIA DLL Production

One of the goals of our project is to provide a common system and process for developing robot applications on the Pocket PC. To facilitate this, our team has decided to convert the ARIA library that we ported into a dynamically-linked library that can be used by more than one application at a time. The DLL will also remove the need to recompile the library every time an application built on top of ARIA is compiled. ARIA has already marked the portions of the code that should be exposed by the DLL, so our main goal now is to produce a DLL specific to the Pocket PC. Once this DLL is created we will also need to incorporate it into our robot control application. We estimate that this process will be complete within a week or two (mid-March).

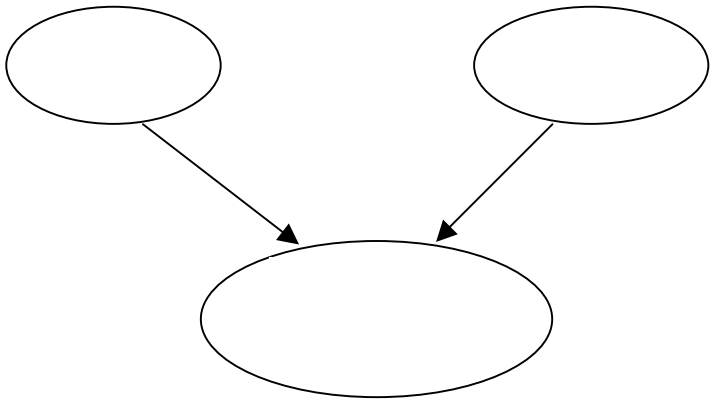


Figure 1

Basic composition of an application using the ARIA DLL

User Interface Development

The user interface for our robot control application is one of the most important aspects of the project. The success of many software projects is often ultimately decided by whether users are able to easily and intuitively communicate with the application using the graphical user interface (GUI). The GUI that we envision for our application will include three basic custom-designed components: a sonar display, an on-screen joystick, and an interactive map. The plans and status of each of these components is detailed below.

Sonar Display

The sonar display is a component that will display the current status of each of the sonar sensors on the robot. It will show a small graphic of the shape of the robot with lines extending from this graphic along the same direction as the physical sonar sensors and having a length proportional to the current strength of the sonar signal. The display will have 3 different colors indicating the proximity of the nearest object that the sonar is detecting: red will represent a near object, yellow will represent a medium-distance object, and green will represent either a distant object or none at all. The development process of the sonar display consists of three steps. First, we will create a bar graph indicating the current strength of each sensor. Then, these lines will be rotated and translated to match the position and direction of the physical sensors. Finally, the lines will be converted into color-coded triangles indicating the range of each sensor. Our

team has recently gained the knowledge necessary to create this display and hope to have all three stages completed by the end of March.

On-Screen Joystick

The ability to control the motion of the robot using a graphical component is an important feature of our GUI design. We currently envision an on-screen joystick as the primary control interface. Our team has researched several joystick designs online and the basic design that we are currently developing uses the position of a dot around a circular area to indicate the current joystick position (see **Figure 2**). Tapping the stylus on the Pocket PC

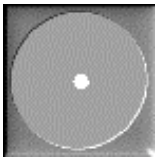


Figure 2

Basic joystick design

and moving it around the screen will change the position of the dot, and double-tapping the screen will lock the dot in place. This dot can later be modified to reflect a more visually-pleasing design. We hope to have a working product by mid-March.

Interactive Map

In order to create a functional robotics application, a good mapping tool and display will be needed. Since ActivMedia has already established a map file format and has tools available to edit those maps, we chose that format. A class is being designed to read in these map files and create a map on the Pocket PC. This map can be interacted with at a higher level in the actual robot application. **Figure 4** shows a basic class diagram for the map that is being designed. The functions that the class will provide are not listed, but this does give a good indication of what the class will keep track of. All of our map drawings are represented in lines, so that is what the map is based on. This class will be fully functional within the next couple weeks so that we can begin to integrate our applications. An example map is shown below in **Figure 3**.

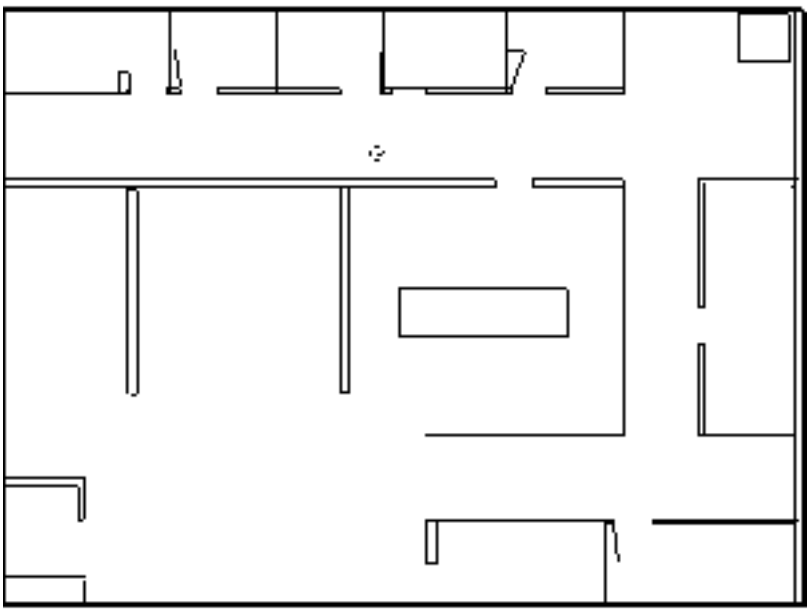


Figure 3

Sample ActivMedia Map

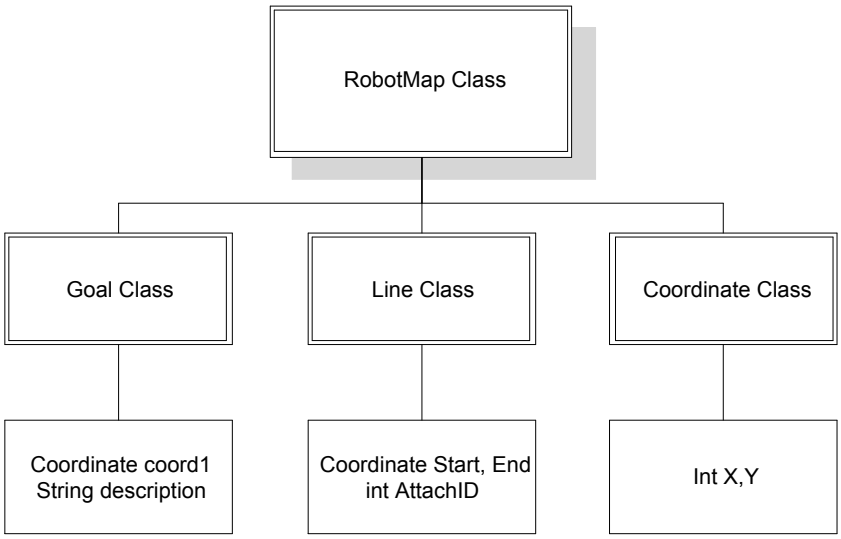


Figure 4

Interactive map class diagram

Work has also begun on using the ActivMedia map files to display maps of locations on the Pocket PC. A sample file (office.wld) provided with ARIA was examined to see what the format was, and how easy it would be to recreate and analyze the format. It turned out that it was a fairly simple text file that we could easily read and do something

with. The format of the file is described as follows. Items in parentheses represent where data would actually be placed. X and Y refer to row/column coordinates in units of millimeters.

```
width (distance in mm)
height (distance in mm)
OriginPad X Y
position X Y

Start Line X1 Y1 X2 Y2
AttachID n
      X1 Y1 X2 Y2
End

Start Goal X Y "Description"
End
```

The line portion of the code can be repeated as much as needed in order to provide all of the line segments necessary to draw the location. The `Start Goal` command allows textual names to be associated with certain coordinate locations. `Width` and `height` define the size of the entire area; `position` defines the initial location of the robot. N represents a sequential integer that identifies that line segment. A C++ class will be developed that can take in a map file and then draw the map using a Pocket PC API. By using this format, we are maintaining compatibility with the ActivMedia tools that will create and interact with these maps.

Component Integration

Once the individual components have been created, they will be combined into a single interface. This will allow the user to control the motion of the robot while viewing its current state. One goal of this combined interface is to allow easy integration and swapping of future components. This portion of the project requires the implementation of the other elements of the interface and will therefore probably not be complete until the end of March.

Establishing Pocket PC to Robot Communication

Another important subproject is establishing communication between the robot and the Pocket PC. One of the main goals of the project is for this communication to be wireless and we will be using Bluetooth to enable this communication. We have not yet received either the robot or the Bluetooth-to-serial adapter that will connect to the robot, and we are unable to develop the wireless communication portion of the project until they arrive. One thing that we have done successfully, however, is establish interaction between the robot simulator and the Pocket PC using a wired TCP/IP connection. This is an important step because ARIA was designed to allow easy conversion between communication types. We are confident that the transition from wired to wireless interaction will be fairly easy and quick, but this is certainly an unknown and could easily cause problems and delays. We will also need to design hardware to power the Bluetooth-to-serial adapter using the robot's power supply connected to a voltage regulator.

Developing the Robot Application

While the previous components are certainly important and complicated pieces of the project, they are not very useful unless they are utilized by a robot control application. Our goal is for the Pocket PC to serve as the "brain" of the robot, and to prove our design we plan to create a high-level application that takes advantage of all of the modules that we are implementing. The specifics of this application are still being discussed, but we currently intend to allow the user to move the robot using the on-screen joystick and have the robot indicate where it thinks it currently is on the interactive map. The user can then correct the robot by providing the actual position by pointing to it on the interactive map. We also would like the user to be able to point to a location on the map and have the robot travel to that location. The application will also incorporate obstacle avoidance and safety features.

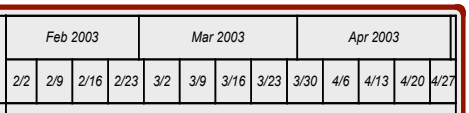
Potential Problems

The main "unknowns" of the project currently stem from the fact that we have not received several essential hardware components, namely the robot and the Bluetooth-to-

serial adapter. We are hoping that the Pocket PC and the robot will be able to establish a virtual serial connection and are confident that it will according to our research, but we will not know for sure until we actually receive the hardware and can test it.

Updated Project Schedule

ID	Task Name	Start	Finish	Duration
1	Order Robots / Parts	2/3/2003	2/28/2003	4w
2	Begin PDA User Interface Development	2/3/2003	3/14/2003	6w
3	Begin map specification and algorithm	2/3/2003	3/14/2003	6w
4	Establish Robot Control with a PC	3/3/2003	3/21/2003	3w
5	Establish basic robot communication with PDA	3/3/2003	3/21/2003	3w
6	Assemble PDA User Interface and Application	3/7/2003	3/25/2003	2.6w
7	Integrate Control and Communication to form PDA API	3/7/2003	4/1/2003	3.6w
8	Integrate API and User Application	3/10/2003	4/4/2003	4w
9	Test	4/15/2003	5/1/2003	2.6w



Conclusion

In conclusion, our team's progress is currently ahead of schedule appears to be headed on the right track. We are continuing to work hard to make sure that the project is successful and on-schedule, and are looking forward to receiving the robot and Bluetooth adapter so that we can test our software on hardware rather than software emulators. Each member of the team has made valuable contributions to the project and we are hoping to coordinate our efforts even more as we begin to bring together the various modules of the project.