Tracking Cooking tasks using RFID

CS 7470 Final Project Report

Rahul Nair, Osman Ullah

While brainstorming about the various projects that we could do for the CS 7470 B-Mobile and Ubiquitous computing class we came across an interesting project that was conducted in the Aware Home [1]. The Cooks Collage project by Tran et al. [2][3] investigated people's response to interruptions in daily life and how they dealt with them. The Cooks Collage augments peoples tendency to place information "in the world" rather than "in the head" [4] by using a series of snapshots of the users past actions as a memory aid to help the use smoothly resume a task. However, since current vision technology is not good enough to do activity recognition or context detection with a high degree of accuracy, the researchers settled on using the wizard of oz technique to take snapshots and compose the collage.

We wish to implement a true sensor based technology to track user activity on the kitchen counter and replace the human "wizard of oz" operator in the Cooks Collage. While the initial objective of our system is to merely take photographs from appropriate cameras whenever a user makes an important action the scope of the project can be expanded to tracking activities that a user does in their daily lives.

Project Justification

A huge area of research in ubiquitous computing is the determination of user's context and using that information to tailor services to help the user perform specific tasks. The lack of really good techniques to do this context detection has led to several Wizard of Oz studies which use imaginary sensors to capture data and then use this information to drive context sensitive services and applications. We wish to provide a real technological solution to the Cooks Collage study which currently uses imaginary vision based sensors to track user's action while they are cooking, and display snapshots of the various actions so that the users can quickly recall what steps they have already performed. Thus the two main goal of our system will be a) to sense the user's activity and b) to know which camera angle to use to best describe the action.

While this initial system is specifically made to work in this environment it can be expanded to track users actions in various other context such as inventorying the food in the refrigerator by knowing what the user is adding and removing every time they open the door. Other possibilities include deducing the user's activity level to find out when distractions and notifications will do the least harm. In all we feel that this project truly embraces the ideals of Mark Weiser's vision of ubiquitous computing [5] where technology makes users lives better.

Background Research and Work

In order to track the user's actions while cooking we did a lot of background research on the possible sensing technologies that we could use. While computer vision was an obvious contender we felt that it was not a robust enough technology to be reliably used in the kitchen setting. The most promising technology we found was Radio Frequency Identification (RFID). Not only was it fast and robust, but also the tags are cheap enough to be actually fitted onto everyday cooking utensils. New 13.56 MHz RFID technology also has the ability to detect multiple tags in a single read using anti collision algorithms.

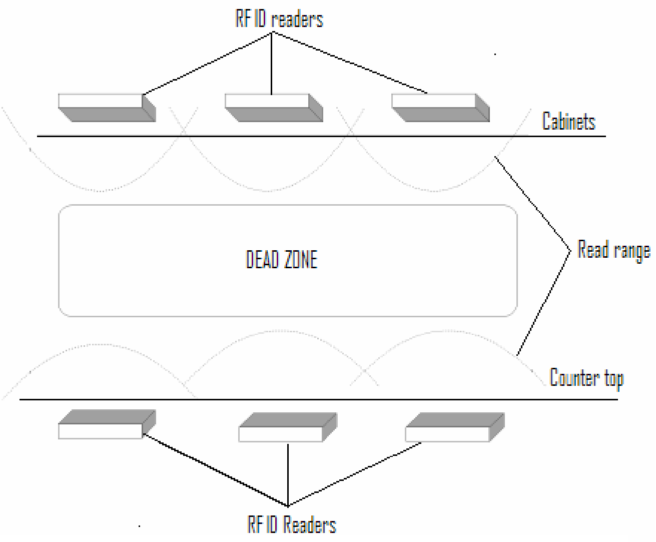

Fig 1: The short range of the RFID readers embedded in the environment creates a dead zone where the system cannot sense the user's activities

Our initial plan was to place multiple readers in the environment and tag all objects as well as the user's hands. We would be able to map the location of each reader to an area in 3D space. By knowing the location of the tags in 3D space, we would be able to deduce what the user was doing in the environment. After further research, however, we found that this solution would not be as feasible as initially thought. RFID read range is limited to about 6 inches. Therefore, embedding the readers in the counter top and the cooking area ceiling would create a "dead zone" in the center of the cooking area, where the user is most likely to be doing most of the work. In addition, the cost of obtaining and embedding the readers into the counter was also prohibitively high. We then decided that using a single RFID reader mounted on the user's wrist and tagging the objects in the environment might provide us with the information about which items the user is interacting with. This was a simple and cheap solution and had already been done before with low frequency RFID technology [6]. The RFID solution we decided to use was the Texas Instruments S6350 mid range RFID reader which has an integrated antenna and RS232 interface. We intended to build/purchase a smaller antenna and attach it to the reader with a long cable. This would allow the lightweight and small antenna to be mounted on the user's hand, while the heavier and more bulky reader could be mounted in a more unobtrusive location such as on the user's belt.

While this system would give us reliable data on which objects the user was interacting with, it would not give us information about what the user was doing with these objects. This is because the reader is mounted on the user's hands. We do not know where the tagged objects are in 3D space, so we cannot find out where the hands are located. In order to get this information we need another sensing system. We felt that the best way to do this would be to us an accelerometer mounted on the users wrist to find the motions of the users hand. Once calibrated to a certain origin, we should be able to use digital signal processing on the incoming sensor stream and find out the location and movement direction of the user's wrist. In order to do this we requested the use of an HP Badge4 [7] which consists of an Intel Strong Arm processor running a Linux derivative and interfaced to several different sensors including a 3-axis accelerometer. The Badge4 also

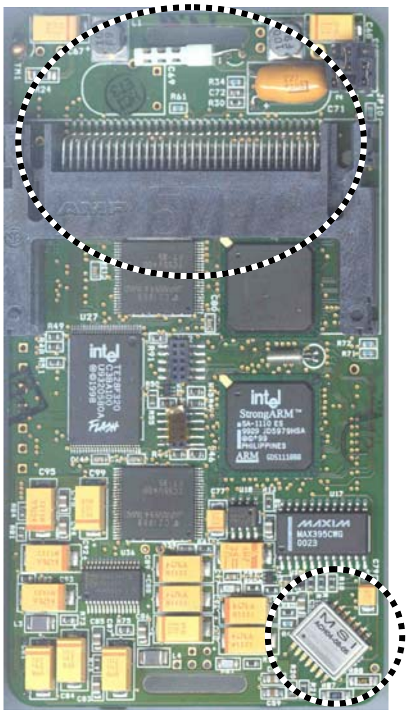

Fig 2: HP Badge 4 with 3 axis accelerometer and PCMCIA slot

has PCMCIA interface to which a wireless LAN card can be attached for wireless connectivity, and also has an RS232 connection to allow communication with the RFID reader. Therefore our idea was to mount the Badge4 on the user's wrist, and connect it to the reader. We could process the accelerometer data on-board, and send out the reader information as well as the gestures that were being performed by the user.

There were some issues which prevented us from bringing this plan to fruition. Primarily, we were told that the HP Badge4 prototype available did not contain an accelerometer, and the ETA for the model with the accelerometer was indefinite. Because the accelerometer was the primary reason for acquiring the Badge4, we decided not to use it.

The next sensing technology we investigated was the using the ultrasound tracking system available on a Nintendo PowerGlove [8]. The PowerGlove is a simple video game accessory marketed in mid 80’s. This device tracks the user's hand position in 3d space as well as the motion of the user's fingers. The reasoning behind choosing this hardware was the low cost ($30) of the device and its ability to tell if the users hand is open or closed. Using a conductive ink strip embedded in the fingers of the glove the glove uses the resistance of the strip to calculate finger joint position. The glove uses ultrasonic detection for 3D positioning and has a resolution of ¼ inch at 30 feet. The glove can give details such as X, Y and Z axis coordinates along with tilt, roll and yaw of the hand in relation to the reader.

There are already projects and how-to's on the internet about how to attach the PowerGlove to the PC, and source code to read this information was freely available. This made it possible for us to implement the PowerGlove in our project despite the time constraints of finishing by the end of the semester.

Fig 3: Nintendo PowerGlove

Once we obtained a PowerGlove, we set about converting the proprietary Nintendo 9-pin interface into a format that would be easily usable by a computer. Research led us to two options: 1) using the a serial port via a Nintendo-to-serial converter which was built by the ACM for use with VR systems, or 2) physically modifying the PowerGlove interface and manually wiring it for use with a PC parallel port [9]. The ACM converter was not only expensive but hard to find. The work required to modify the PowerGlove interface was not very advanced so we decided to go down that route. However, after we wired the PowerGlove to the parallel port interface we found that we were unable to actually send or receive any data from the glove. We are unable to determine the cause of this failure because we bought the PowerGlove used off an auction site on the internet. Due to time constraints and lack of a proper troubleshooting platform (I.e. a Nintendo), we decided that it would not be a good idea to pursue the PowerGlove route any further.

Upon abandoning the PowerGlove, we decided to go ahead with building our final system using a hand motion emulator. The purpose of the accelerometer and PowerGlove was to basically track what the user's hand is doing. Therefore, we felt that we could assume such a system is available to our software, and just create an emulator to provide our system with this information. In reality our system would be replaced by a sensing system that reports position and movement of the hand and sends events to the system saying what actions are being performed by the hand, i.e. tilt, shake, scoop, etc…

System Architecture

Our demo system is meant to be a basis for a test system. We did not have the hand-mounted antenna nor did we have the hand tracking system. The movement of the hand is simulated with an emulator, and picking up objects is simulated by waving tags in front of the reader. We have two applications. There is the RFID interface which reads from the reader and does all the logic, and then there is a PowerGlove emulator which communicates with the RFID interface using UDP.

There are two primary components that make up our system; objects and actions. Objects are representations of the physical things in the environment. There is one object for each individual item in the environment. Each object has associated with it a name, an exclusive set of tags, and a list of actions that can be performed with that object. Actions are gestures that are performed by the user such as pouring, shaking, stirring, etc.

Interacting with Objects

When the user picks up an object, this object becomes the active object. The active object is detected in the following sequence. When a user closes his/her hand, the system records all objects being read at that time. This can be imagined as the user's hand being amongst a collection of utensils for example. Then as these items are no longer read, the system removes them from the list until only one remains, as would happen when the user carried one utensil out of a pile. This remaining object is termed the active object. The object is no longer active in one of two ways: 1) the user opens his/her hand; 2) the active object is no longer detected for a certain timeout period. This helps avoid problems of the object not being read for one or two reads for some reason or another.

Once an active object is detected, actions can be performed with the object. As stated earlier, each object has specific events assigned to it. When the system perceives that the user is performing an action, the action is queried against the list of actions for the currently active object. If the action is assigned to the object, a record of that action being performed is shown on the screen. The location information from the hand tracking system can be used to determine which camera is best suited for taking a picture of the action.

Adding Objects to the System

Adding objects to the system should be very easy for the user, since the user is not expected to have a degree in databases. In order to add an object to the system, the user will theoretically just press a button in the application and then run his/her hand over the object a few times to capture all tags on the object. The tags will have unique information for that model of object encoded in them. This can be done very easily from the factory. There are two possible methods of using this information. The first method is that there is a central database, and the tags are encoded with a specific ID. This ID can be queried against the database to find the name and actions performed by that object. The drawback of this system is it requires vendors to register in a database, which makes the system almost unfeasible. The second solution is just to encode a URL to the object information in a standardized format and store it on the tag. Upon reading the tag, the system would go out to the specific URL and retrieve the information about the object. This would allow each manufacturer to easily keep their information up to date and would also remove the reliance on a central infrastructure for keeping information.

Since our system is just a demo of the tracking capabilities, we did not implement a database system for storing object information. Instead when the user adds an object in the system they specify a name and a list of valid actions for the object. The object is then scanned and all the tags are recorded.

Software Structure

The RFID interface is made up of three primary classes. There is the primary application, an RFID Reader class and an RFID Objects class. The RFID Reader class is responsible for communications with the reader. It is started in its own thread and is constantly polling the reader for tags. Every time the reader responds, an event is fired which carries a collection of tags. When no tags are present, this collection is empty. This allows the system architecture for the reader to change and be transparent to the application. The RFID Object class provides the infrastructure for reading the database of objects and querying objects based on the tag number. This allows for architecture changes to be transparent to the system. Our current system just uses a comma and colon delimited file to store the object information.

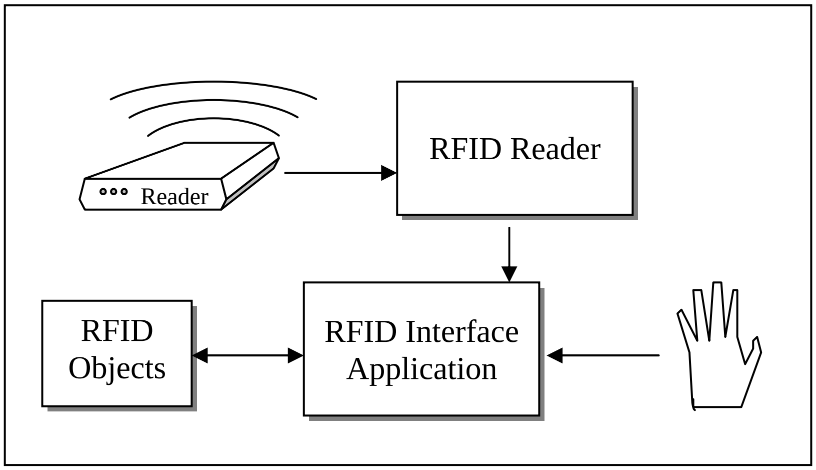

Fig. 4. The structure of the RFID Interface Application

The primary application handles all the incoming information from the power glove simulator and the reader, and provides an output to the user. It then takes this information and performs all necessary logic needed to determine what the user is doing, and outputs it to the display. It also provides the client for adding tags to the system.

PowerGlove Emulator

The PowerGlove emulator is a very simple application. It allows us to send the action "events" to the RFID Interface application through UDP. When we want to send an event we just click the button. The system also sends a constant stream of location information through UDP. Currently this is simulated by sending the cursor position as the location data. This is the same thing that would happen if there was an external system tracking the user's hands and doing gesture recognition on them.

Future Work

Future work can move in the direction of getting an actual working prototype installed. This would include getting a hand tracking system installed and creating an antenna that is mountable on the hand. Ideally, this system should track both hands and not just one. Before actually being used, further work needs to be done in evaluating the health issues involved with mounting a radio antenna on the user's hand for extended periods of time.

References

[1] Kidd, Cory D., Robert J. Orr, Gregory D. Abowd, Christopher G. Atkeson, Irfan A. Essa, Blair MacIntyre, Elizabeth Mynatt, Thad E. Starner and Wendy Newstetter. (1999) "The Aware Home: A Living Laboratory for Ubiquitous Computing Research"

[2] Tran, Q., Mynatt, E. (2003). "What Was I Cooking? Towards Deja Vu Displays of Everyday Memory."

[3] Tran, Q., Mynatt, E. (2002). "Cook's collage: Two exploratory designs."

[4] Hutchins, E., (1995). "Cognition in the Wild"

[5] Weiser, M., (1991). "The Computer for the Twenty-First Century"

[6] Schmidt, A., Gellersen, H., Merz, C, (2000). "Enabling implicit human computer interaction: A wearable RFID-tag reader."

[7] http://www.hpl.hp.com/personal/Mat_Hans/research/badge4/

[8] http://www.angelfire.com/ok2/stepinto/PowerGlovePage.html

[9]http://www.ee.ucr.edu/~dtan/ee128%20PowerGlove%20Project/PowerGlove%20Schematics/SimplePowerGloveInterface.gif