Shenglin Yang, Ingrid Verbauwhede
UCLA
Last change: 7/29/2002

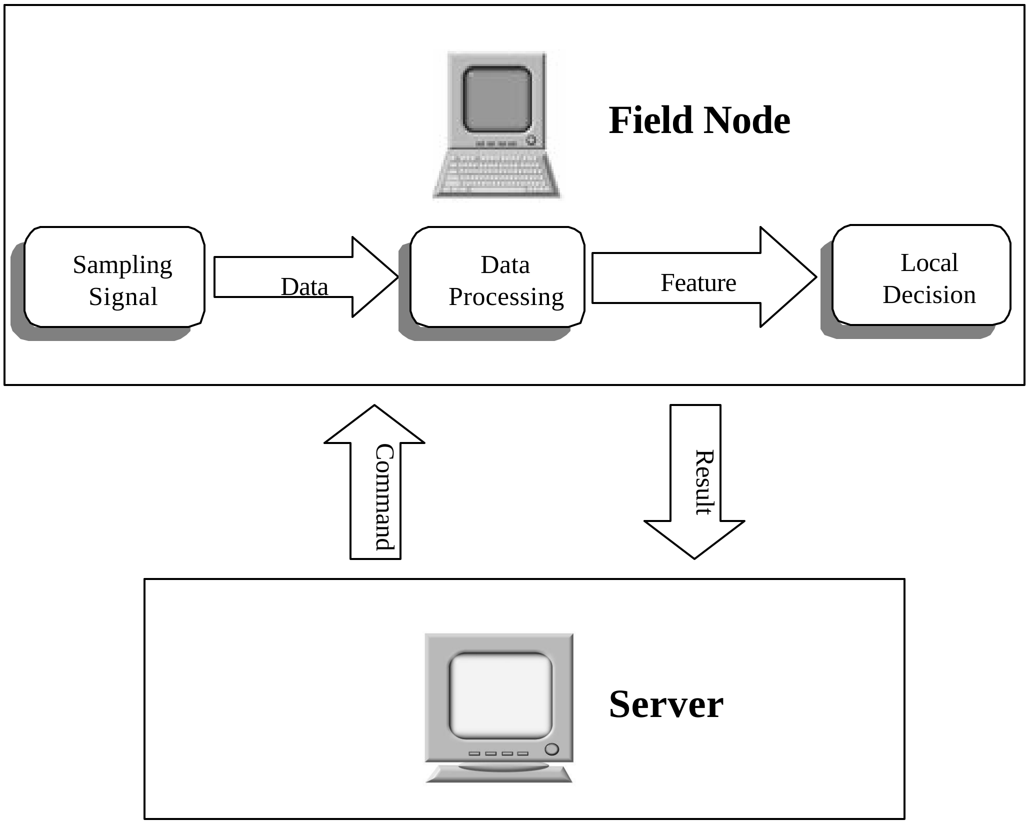
Poly-sensing environment
Summary

Goal

The goal of this project is to set up a sensor network, which can sense the environment and response to it according to the observation. In the network, the field nodes need to capture many types of sensory information such as sound, light, temperature, image, etc from the environment. The raw data is first processed locally. Then the cooperation between the sensor nodes and the server will decide on the actions. To make more accurate decisions, the local results are sent to the server. The server makes use of these results and makes the final decision. Depending on the final decision, the server can act either in the virtual space or give feedback to the real space.

Work That Has Been Done

Till now the system can do the acoustic signal sampling, data processing and result communication as shown in the following figure:



So far, our work has focused on speech processing in a one-Server one-Client LAN. Following is a detailed description of each part.

1. Sampling Signal:

This part is done in the field node. The program opens the soundcard. When the voice volume reaches the pre-set threshold, the field node begins to sample the signal. The program can control the duration, the sample rate, the channel number as well as the input format. After getting data, the field node begins to process them to get some useful information.

2. Data Processing:

The main purpose of this part is doing limit word speech recognition.

First of all, we need to set up a HMM (Hidden Markov Module). This should be done before the system is put into use. The feature used in this project is a coefficient called MFCC (Mel Scale Filter Ceptrum Coefficient). When we hear a sound, the sound actually goes through our ear. According to the research, our ear acts as a nonlinear filter bank. Furthermore, the shapes of the filter bank are almost the same for every person who has normal hearing. Researchers get the parameter of this kind of filter bank by experiments and then make the computer do what exactly our ears do when receiving a sound. MFCC includes this kind of parameter. So it is chosen as speech feature in our project. After getting the MFCC feature matrix of the train data, they are put in an HMM and do the training. After several iterations (five iterations are used in the project), we get the final module for recognition.

When doing the real-time speech recognition, we convert the sampled data into a MFCC feature matrix, which is ready for the next step: recognizing.

3. Local Decision:

In the case of our project, the local decision is doing speech recognition to figure out which command is given to the field node. After getting the MFCC feature matrix of the input data, we put it into the Hidden Markov Module which we get in advance, then search in the module to decide which command the recorded data most probably is. This result is store in a file together with a score indicating how similar the input signal is.

After getting the local decision, we need to report to the server what the field node gets. This is the next step: Communication.

4. Communication:

Socket is used for communication programming. TCP/IP protocol is used to send/receive the message between server and client. When local decision is made in field node, the server can receive it through socket at once. The communication between the field node and the server is two-direction. Server can also give instruction to field node to tell what kind of information is needed.

5. Test:

This system is tested using four short sentences: “open the door”, “close the door”, “turn on the light”, “turn off the light”. 10 samples per sentence are used for training (this number maybe a little smaller, but it works. We can improve the accuracy of this system by increasing this number). Then we can speak one of these four commands around the field node and get corresponding information at the server side.

Work to do

Next step, we need to do the following:

1. Implement the field node in iPAQ running Linux. By now, a Linux desktop is use as field node. In real sensor network, the sensor node needs to be small, low power and easy to be re-configured. So iPAQ is chosen because that it has all these advantages and it is easy to use.

2. Add more nodes to the system. Till now, the system is a one-server/one-client system, which only deals with the speech signal. We need to add more field nodes, which can deal with more kinds of signal (such as light, pressure, temperature, image, etc.) and do some collaborate processing to get more reliable result.

3. Considering processing-communication trade-off. To save the energy used for transferring the local result, it is better to get higher-level result in the field-nodes. But on the other hand, this means more processing needed to be done locally, which requires more memory and CPU source. Doing trade-off between local processing and result communicating is important for making a low-cost system, which offer low-power consumption.

4. Setup the server part to make use of the final result. Depending on the information gathered from the field nodes, server makes final decision about what happens around. Then it can do corresponding action in the virtual space or give some feedback to the real space.