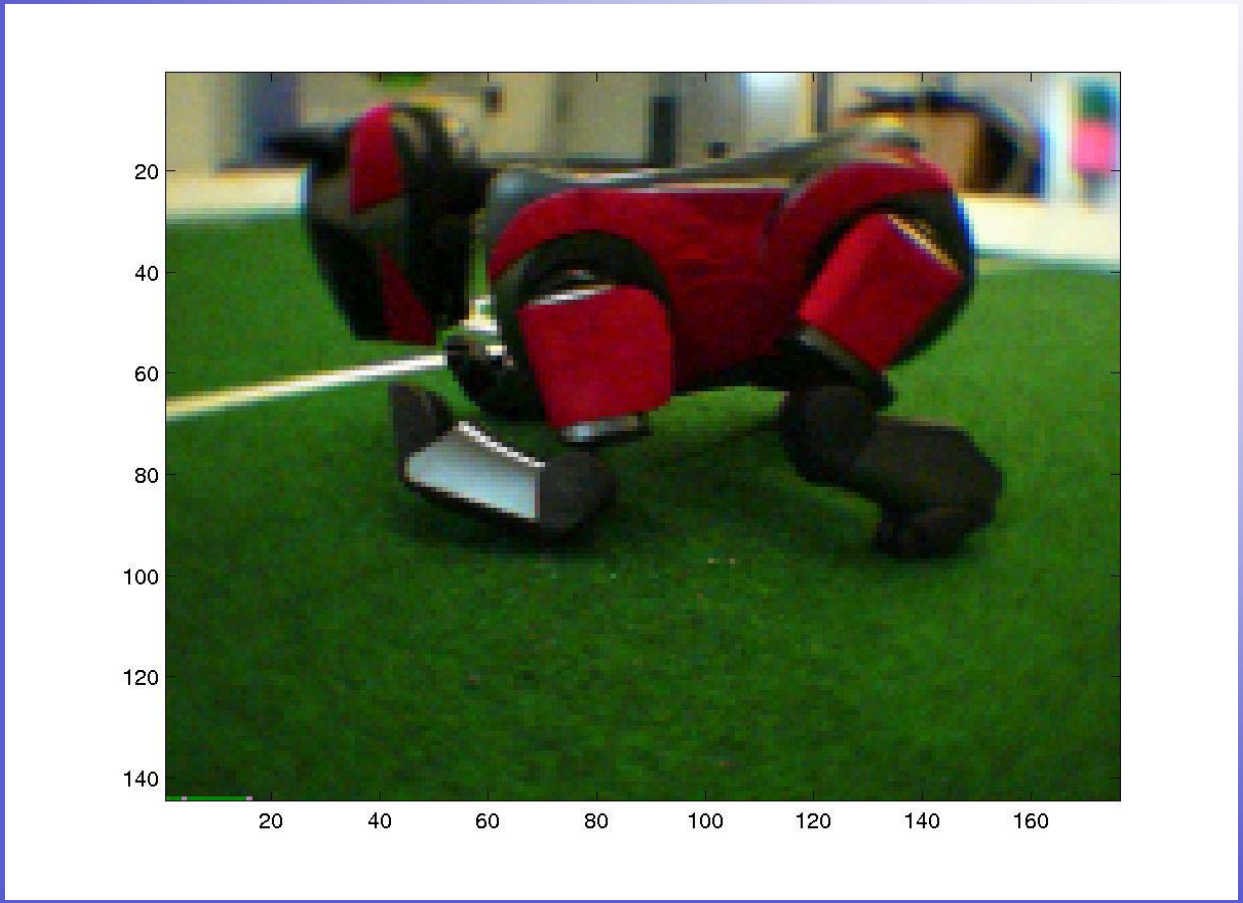
Aibo Motion Calibration and Control/Monitoring System

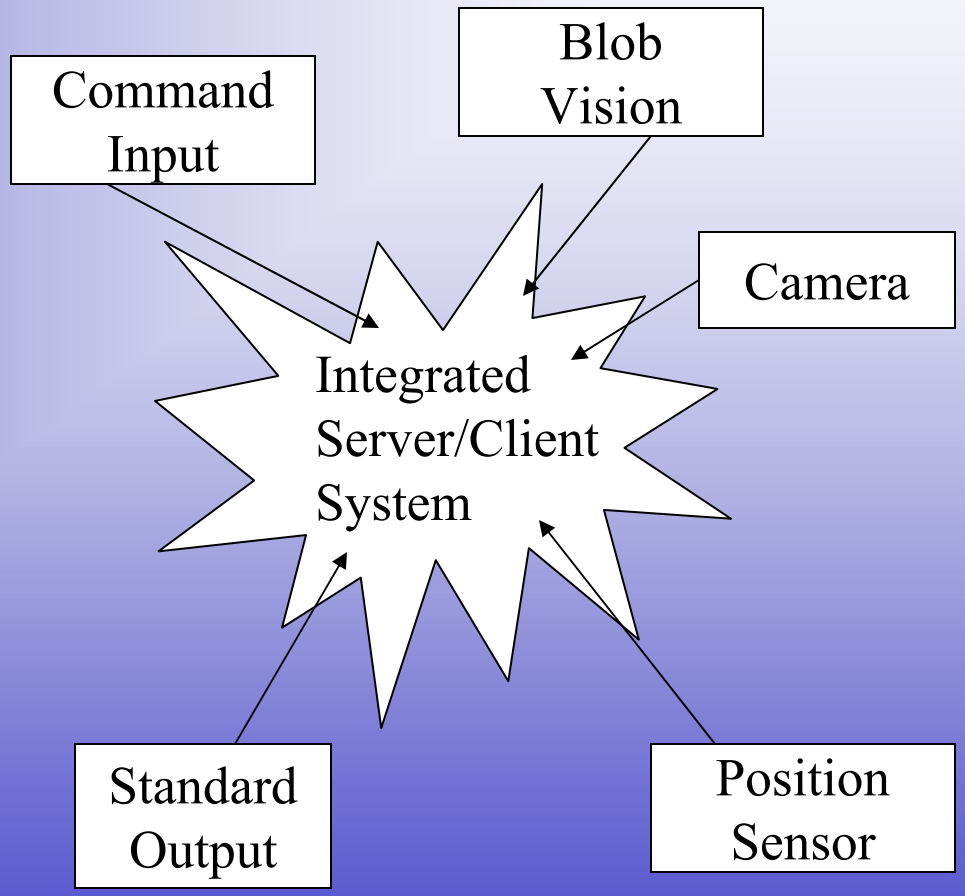


Brian Corwin

Dr. Daniel Lee- Advisor

Purpose

- Robotic devices, like the Sony dogs, often have many types of sensor outputs.
- Want to integrate all of these information sources, from multiple dogs into one server/client system for expediting debugging and development.
- Why? You can better understand what the dog is doing if you can “see” the world as it does.
- Want to get the sensor information in real-time.



Example Use

- You use a group of Sony Aibo's to look for a bomb.
- They output the view from their cameras, their positions, and data from their olfactory sensors.
- This data is compiled and output by a server. A client program uses this information to find the bomb (by processing the input data) and control the dog in real-time.

Step One: Positioning System

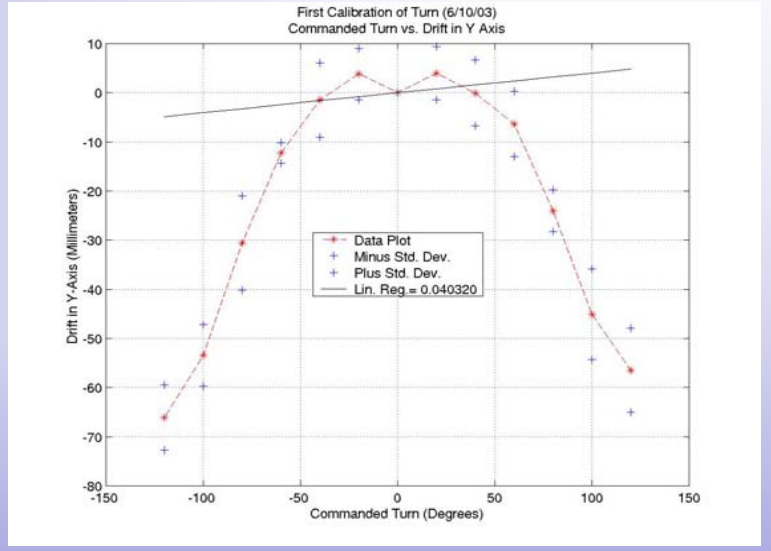
Motion Calibration

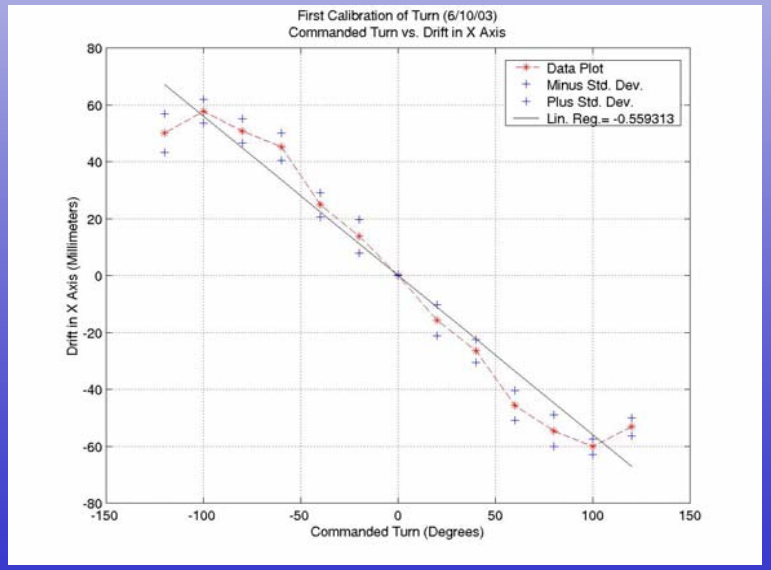
- Problem: Want to create a system that can give real-time positioning information on the dogs.
- Solution: Use Nest of Birds electromagnetic sensor device. Provides 3-D positional and orientation data to a PC via the Serial Port.
- Created a C++ program to run the NOB and Matlab scripts, to use this program to calibrate the dog's walk.

Step One: Positioning System

Motion Calibration Results

- Found the calibration factors and walk errors for forward/backward and left/right walks as well as left/right turns.
- Most importantly found that the dog was not turning around its head, but actually turning around its shoulders.
- Had problems with limited NOB range and magnetic interference from dog's motors.





Calibration factors:

X displacement = -0.6946 /+0.7398

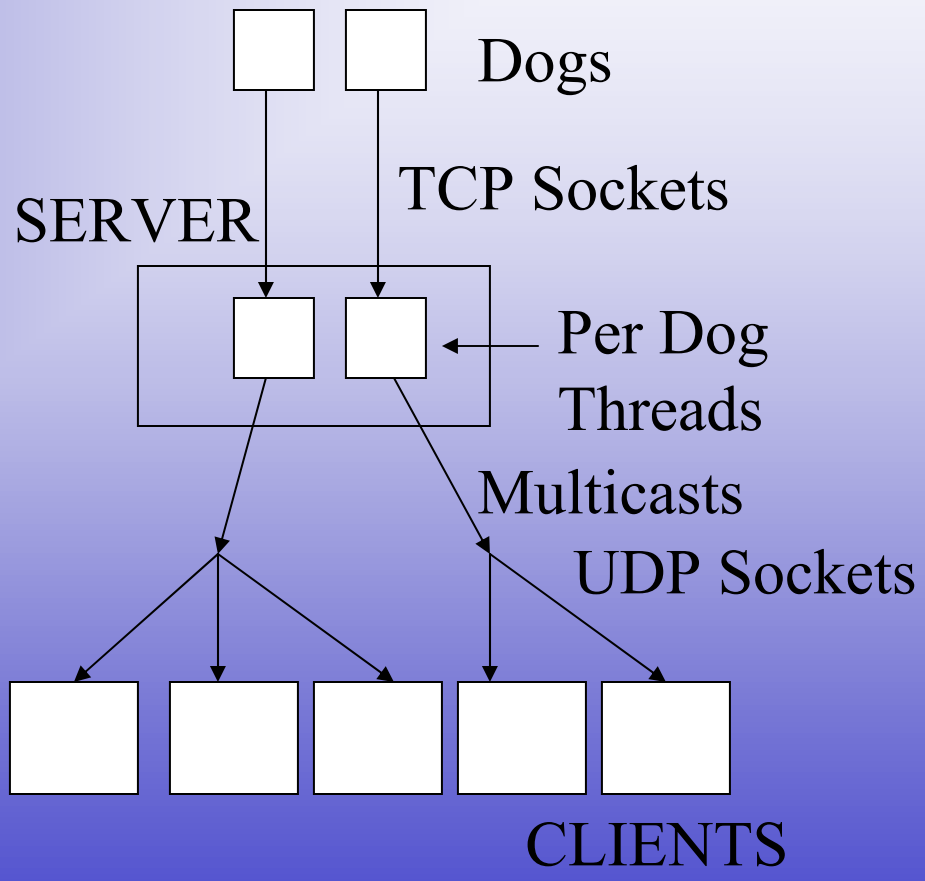
Y displacement = -0.7111/+0.8404

Θ displacement = -0.8987/+0.9249

Step Two: Servers

Design

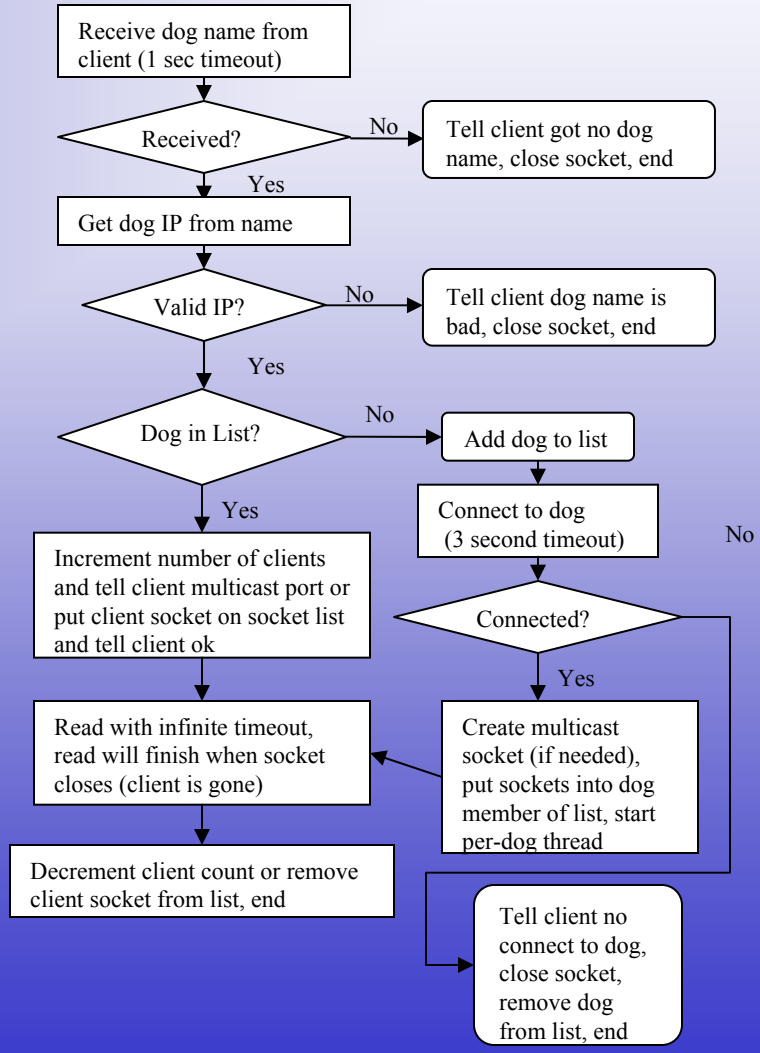
- Server is combination of four C++ programs for Linux: blob vision, camera, input, and output.
- Data flows from client to server to dog or vice versa.
- Each server has different requirements and design. The camera server needs speed, not reliability. The input server is the opposite.
- Camera and blob vision servers use multicasts for greater efficiency. Output server sends data for dog to each client one by one, by going through list of unicast TCP sockets to each client.



Step Two: Servers

Implementation

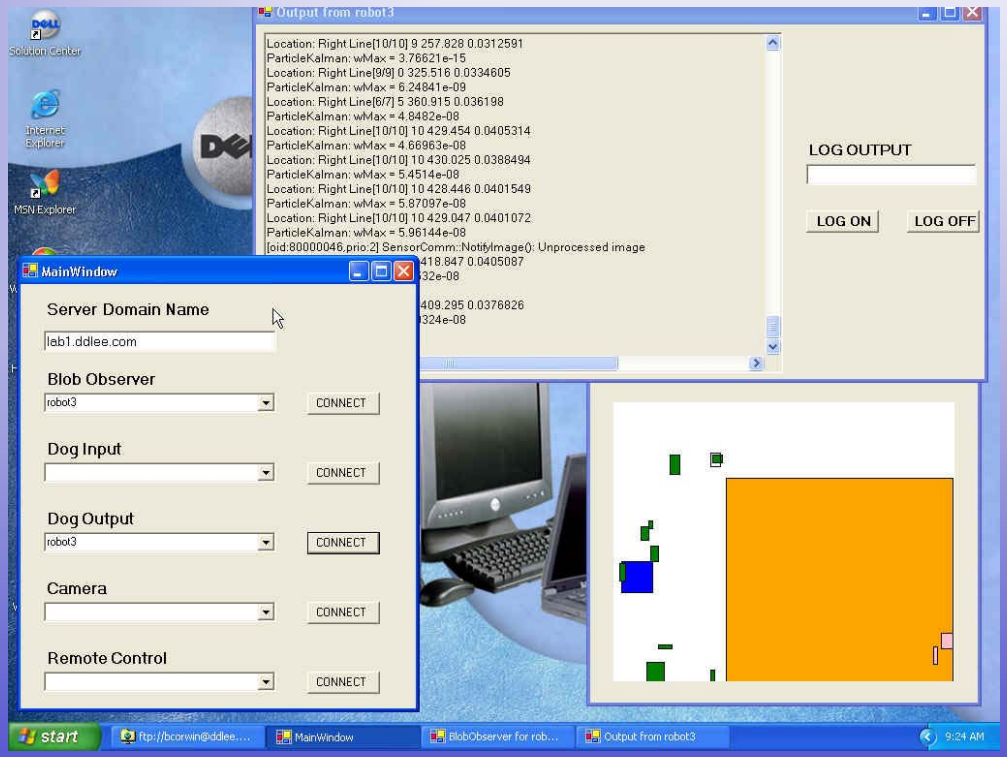
- Wrappers for socket system calls were written to handle errors.
- Each kind of socket was implemented as an object; all socket types had the same simple interface: socket created, read from, written to, and destroyed.
- Each server is multithreaded (one thread per-client and one thread per-dog), to increase efficiency.
- Each server has a list of the dogs in use; list elements holds any data for that dog's use.



Step Three: GUI Clients

Windows .NET Client

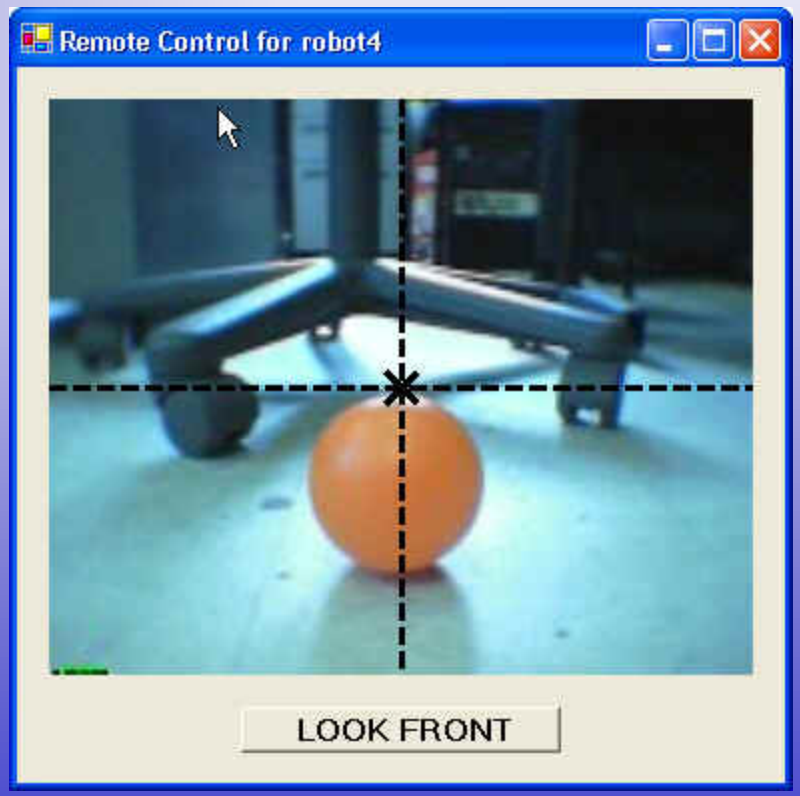
- Lastly we need create integrated client applications with rich GUI's.
- Looking to have clients on Windows, Linux, and PalmOS.
- The first client made is for Microsoft's new .NET framework (written in C#).
- The client can create windows for connections to all four different servers as well as special window type: the dog remote control.



Step Three: GUI Clients

Remote Control

- Remote Control Connects to the Input and Camera Servers
- Has a camera window, which if clicked will command the dog to turn its head to look at the clicked spot on the image.
- Binds keys to forward, backward, left, etc. motion commands.
- Allows for user to be able to navigate dog around a room.
- First example of more complex, integrated client program for this monitoring/control system.



Future Work

- Upgrade the Nest of Birds system with the Extended Range Transmitter.
- Investigate other alternative positioning systems.
- Integrate some positioning system with the other servers.
- Create more advanced client programs that integrate different sensory outputs in interesting ways (ex. color map creator, bomb locater, and localization teacher).