4 Sensor-rich robots driven by real-time brain circuit algorithms

Andrew Felch and Richard Granger

4.1 Introduction

The analysis of particular telencephalic systems has led to derivation of algorithmic statements of their operation, which have grown to include communicating systems from sensory to motor and back. Like the brain circuits from which they are derived, these algorithms (e.g. Granger, 2006) perform and learn from experience. Their perception and action capabilities are often initially tested in simulated environments, which are more controllable and repeatable than robot tests, but it is widely recognized that even the most carefully devised simulated environments typically fail to transfer well to real-world settings.

Robot testing raises the specter of engineering requirements and programming minutiae, as well as sheer cost, and lack of standardization of robot platforms. For brain-derived learning systems, the primary desideratum of a robot is not that it have advanced pinpoint motor control, nor extensive scripted or preprogrammed behaviors. Rather, if the goal is to study how the robot can acquire new knowledge via actions, sensing results of actions, and incremental learning over time, as children do, then relatively simple motor capabilities will suffice when combined with high-acuity sensors (sight, sound, touch) and powerful onboard processors.

The Brainbot platform is an open-source, sensor-rich robot, designed to enable testing of brain-derived perceptual, motor, and learning algorithms in real-world settings. The system is intended to provide an inexpensive yet highly trainable vehicle to broaden the availability of interactive robots for research. The platform is capable of only relatively simple motor tasks, but contains extensive sensors (visual, auditory, tactile), intended to correspond to crucial basic enabling characteristics for long-term real-world learning. Humans (and animals) missing sensors and limbs can nonetheless function exceedingly well in the world as long as they have intact brains; analogously, Brainbot has reasonable, limited motor function and all necessary sensors to enable it to function at a highly adaptive level: that is, prioritizing sensorimotor learning over unnecessarily complex dexterity.

Brainbots are being tested with brain-circuit algorithms for hierarchical unsupervised and reinforcement learning, to explore perceptual, action, and language learning

Neuromorphic and Brain-Based Robots, eds. Jeffrey L. Krichmar and Hiroaki Wagatsuma. Published by Cambridge University Press.

capabilities in real-world settings. We describe details of the platform and of the driving algorithms, and give examples of the current and planned abilities of the Brainbot system.

4.1.1 Background

Over the last several decades the capabilities of robot hardware have improved substantially; it can be argued that hardware is no longer the key bottleneck to achieving the grand challenge of cognitive robots. Remotely operated robots can now demonstrate useful and economically valuable tasks, and significant expectations have been placed on their autonomous capabilities (HR, 2000; DOD, 2005; Thrun *et al*., 2006). The algorithms and computational architecture necessary to provide such autonomy, however, have not been derived and it is widely agreed that this presents the greatest barrier to achieving the promise of these robots (USN, 2004). Despite advances in artificial intelligence (AI), robots cannot yet see, hear, navigate, or manipulate objects without rigid restrictions on the types of objects, phrases, or locations they are presented with.

There are, of course, no actual specifications describing the processes of recognition, learning, recall, imitation, etc.; rather, these abilities are all defined solely by reference to observation of corresponding abilities in humans (and other animals). For instance, speech recognition performed by automated telephone operators is a widely deployed industrial application, which nonetheless fails a large percentage of the time. This failure rate might well have been the best possible performance for this task; the only reason for believing that much better performance can be achieved is that humans achieve it. Nothing tells us what minimal mechanisms or components will suffice to attain human-level performance; it is not known, for instance, whether natural language learnability requires at least one complementary sensory input (sight, sound, touch), or other supporting capabilities. This observation generalizes whenever we attempt to extend simplified systems to more complex tasks. Algorithms that work on toy problems (e.g. software simulations of robots) often fail to scale to more difficult environments such as those encountered by autonomous robots in the real world. Thus, algorithms for broad robotic use may have to be designed and tested on platforms that are sufficiently similar to the final robot, yet robots with the sensing, motor, and onboard processing power for advanced development are typically quite expensive (see Table 4.1). The Brainbot platform (Figures 4.1, 4.2) was fashioned to provide research laboratories with the option of affording one or more such robots to support the development of brain algorithms and other AI and machine learning methods.

Extant robot platforms (Table 4.1) feature a wide variety of capabilities and costs. The Surveyor Corporation's SRV-1 provides significant functionality such as mobility, wireless connectivity, and a digital camera at low cost. The iCub and MDS have high levels of dexterity, impressive standard features (though relatively fixed sensor options) at high cost. The Pioneer robot comes with few standard features but is quite extendable by the user and some options are available from the manufacturer.

In this context, Brainbot is sensor rich, easily extensible, and carries the most substantial onboard computer available on any extant platform. Brainbot leverages her

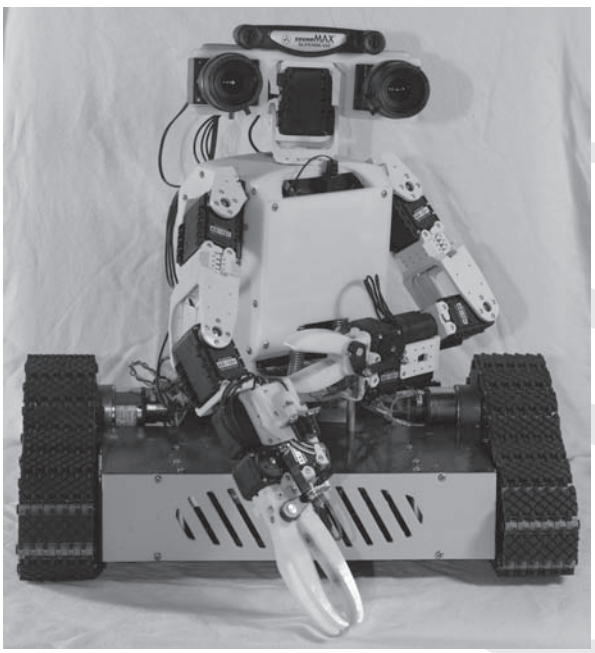

Figure 4.1 The design integrates off-the-shelf components with a minimum of customized pieces. The custom pieces include 27 printable plastic parts, 10 CNC machined Delrin pieces, 7 small printed circuit boards (PCBs), and 2 small machined aluminum pieces (see Figure 4.2).

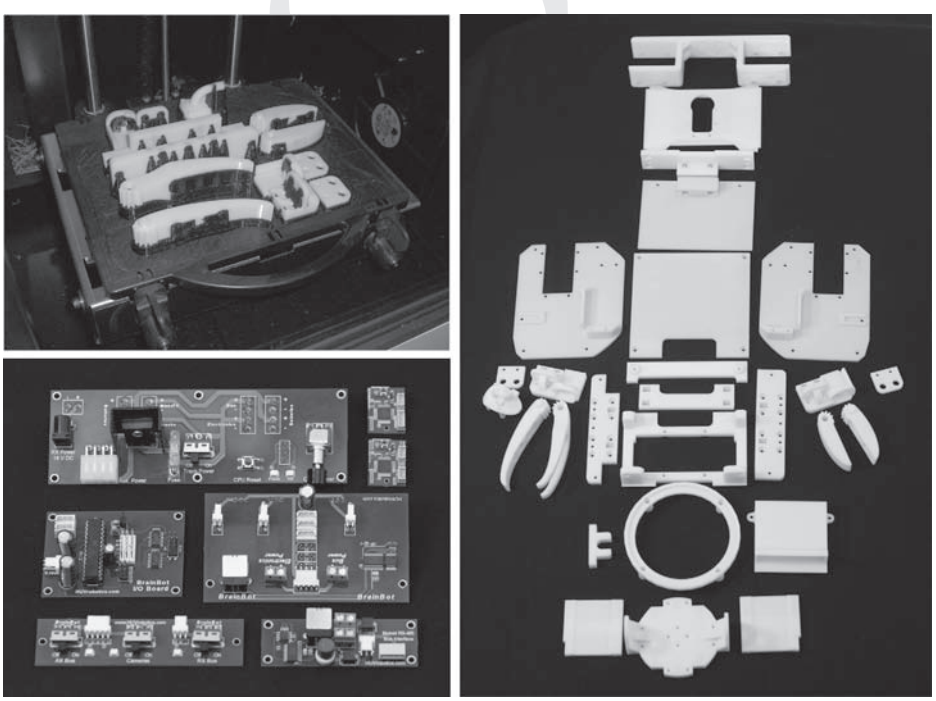

Figure 4.2 Gripper pieces approaching completion in the 3D printer (upper left), a complete layout of the printed plastic pieces (right) and custom printed circuit boards (PCBs, lower left) are shown. The power board, miniature IO boards (top), standard IO board, AX-12 bus board (middle), power switch board, and RX-64 bus board (bottom) can be seen.

Table 4.1. Comparison of existing robot platforms

Robot name	SRV-1	Pioneer P3-DX	Scout	Nao	Brainbot	MDS	iCub
Company	Survey or Corp.	Mobile Robots	Dr Robot	Aldebaran	Robozyme	MIT, Xitome Design, *et al.*	Consortium
Price	$495	$4200	$8750	$15,600	$28,000	~$200,000 (est.)	~$300,000 (est.)
Processor	Blackfin 500 mHz;32 MB RAM;802.11g	PC/104(e.g. 1 GHz);USB	microcontroller;low-speed; 802.11g (912 kbps)	AMD Geode 500 mHz;256 MB RAM;802.11g	Core 2 Quad 2.83GHz;4 GB RAM;10 USB;802.11n	PC/1041 GHz; USB	Celeron dua l2.13 GHz; 2 GB RAM; USB
Speech recognition	none	none	none	none	onboard	unknown if onboard	unknown if onboard
Speech synthesis	none	none	none	none	onboard AT&T Voices	unknown if onboard	unknown if onboard
Cameras	basic camera	optional	basic wireless camera	2 CMOS cameras	Prosilica GC-650c; up to 300 fps ROI	Stereo CCD	Point Grey DragonFly2
Expandable vision	not expandable	many options	second camera	not expandable	Prosilica GC, Tyzx Point Grey, etc.	not expandable	not expandable
Vision software	some onboard processing	some onboard vision process-ing, requires PC/104	not onboard	some onboard processing	RoboRealm software real-time onboard feature extraction; custom vision programmable	some onboard processing	some onboard processing
Manipulators	none	one 7-DOF arm	two 6-DOF arms	two 5-DOF arms; two 1-DOF grippers	two 4-DOF arms;two 1-DOF grippers; interchangeable grippers	two 7-DOF arms, 10 lb payload; two grippers, 4 digits each	two 7-DOF arms; two 9-DOF grippers
Manipulation sensors	none	none	touch pressure available	position; force	position; force; touch pressure	position; force; touch pressure	position; force; touch pressure
Sensors	sonar	many options;laser scanner, com-pass, tilt, etc.	microphone, IR, sonar	microphone;5 axis IMU, sonar	any USB, GigE, or Atmel sensor;Sonar, Lidar, GPS, 9-axis IMU, compass, etc.	laser range-finder; balance sensors	microphone; IMU

The list is not exhaustive. Note that the Willow Garage PR2, carrying strong arms (4 lb/1.8 kg+), rich and extensible sensors similar to Brainbot, and multiple quad-core processors, was not available for purchase at time of publication. Pricing has been estimated to be more than $300 000 depending on sales' contri-bution to future R&D and pricing against similar robots.

computing and sensing capacity to integrate off-the-shelf voice, audition, and speech "middleware" with user-programmed intelligent real-time algorithms. The intent is to enable programmers to leverage the existence of existing support systems (visual feature extraction, speech recognition, speech production, motoric balance, etc.) to focus on advanced algorithms design and testing.

4.2 Platform

Brainbot has an open design with all source code and documentation available under an open source license including hardware designs and virtual Brainbot models for the widely used Webots simulation system. Intelligent algorithms that are written to control Brainbot's behavior have no open source requirements and can be closed source (private) or open sourced by their author, regardless of the commercial or research purposes for those algorithms, enabling the system to be used as a testbed for further research.

For actuation, 13 Robotis AX-12 servos provide 11 degrees of freedom (DOF) with 16 kg-cm holding torque, and a twelfth DOF providing tilt to the waist with double strength. The waist is also supported with springs, which were added to increase Brainbot's ability to lean over an object on the ground, pick it up, and lean back with the object in hand. Two Robotis RX-64 servos are embedded in the chest of Brainbot and provide each arm with a strong shoulder capable of 64 kg-cm holding torque. All of these servos include load, position, temperature, and voltage sensors. The servos are commanded, and their sensors polled, over the AX-12 bus and the RX-64 bus, which are each capable of 1 mbps transfer speeds. Interface to the servos is currently rate limited to commanding and polling all servos together at approximately 30 Hz due to the interfacing of Brainbot's computer to these buses through a USB adapter with approximately 1 ms latency. An alternative design utilizing one or more embedded processors to command and poll the servos directly over the AX-12 and RX-64 serial busses, and to present these servos to Brainbot's onboard computer as a whole, has been investigated and this design is believed to be capable of achieving 500 Hz to 1 kHz; this project may be pursued in the future. In sum, the 14 DOF are allocated as two grippers with 1-DOF, two wrists with 1-DOF, two elbows with 1-DOF, and two shoulders with 2-DOF each, in addition to one waist with 2-DOF, and one neck with 2-DOF each.

The standard Brainbot configuration uses a tracked base for locomotion (Figures 4.1, 4.3). Initial versions of Brainbot had bipedal locomotion; this, however, necessitated lower overall weight, which in turn limited the onboard processing power that could be carried. In addition, the legs were underpowered using the AX-12 servos, and an upgrade to RX-64 was not expected to raise maximum speed to human walking speed. In contrast, the Brainbot tracked base is able to travel at approximately human walking speed while still allowing objects on the floor to be reached and manipulated. Future research goals may necessitate a legged Brainbot, and the platform remains backward compatible with the legged design. It is worth noting that a leg design comprising the stronger EX-106 Robotis servos has been researched and is believed to

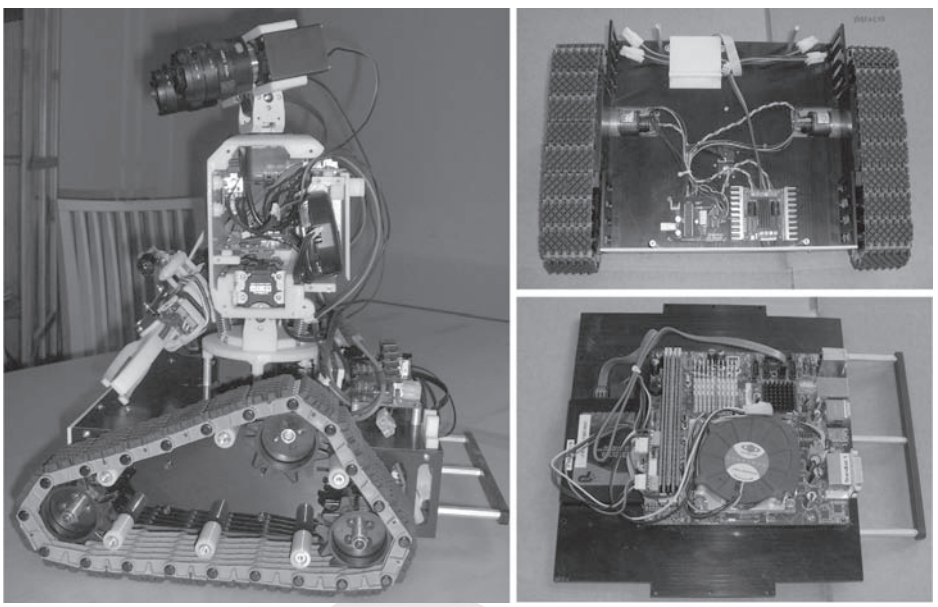

Figure 4.3 Internal chest and track views (left), and top and bottom base plates are shown. The chest houses three AX-12 servos, two RX-64 servos, an 18V voltage regulator, as well as the AX-12 and RX-64 USB bus adapters and the power switch board. The track motors are mounted to the track modules, and the rotation/distance encoders, IO board and motor driver are mounted to the top plate, which is turned over for viewing (upper right). The PC motherboard and SSD drive are mounted to the bottom plate (bottom right).

be capable of carrying the standard quad-core Brainbot computer system and sensor complement while achieving speeds similar to human walking. These servos, which contain the same position and load sensing of all Robotis servos, allow perceptual bipedal algorithms to accurately estimate the force that is placed on a joint, and are complemented by these servos' capacity for high-frequency sensing and commanding, further improving the types of experiments that can be run when Brainbot is used as bipedal balance and locomotion research platform.

Two Prosilica GC-series cameras can be used at the top of the neck, which can be synchronized to within a few microseconds for stereo vision algorithms. (A single camera can be used in cases where stereo vision is not of interest.) The cameras can operate with any C-mount or CS-mount lens. Alternatively, Point Grey, Tyzx or other cameras can be mounted in this position using GigE, USB, or Firewire (using a low-profile PCI Express Firewire card). The Prosilica GC650C gives Brainbot 659 × 493 resolution at 90 frames per second (fps) and provides uncompressed image data in RGB-24, Bayer-16, and some other user-selectable formats. Binning (resolution reduction) and regions of interest (ROI) can be specified, and when a 100 × 100 ROI is used, for example, the frame rate increases to 300 fps. This increased frame rate can greatly simplify real-time tracking algorithms by reducing the number of possible locations to search for an object, which increases quadratically with the inverse of the frame rate.

The Computar H2Z0414C-MP 4–8 mm f/1.4 varifocal lens has been selected for its ability to support wide angle (84.5° diagonal), zoom (2×, 44.1° diagonal), wide maximum aperture for reduced motion blur in low light (f/1.4), megapixel sensor resolutions, light weight (100 g), small size (50 mm × 40 mm), and a focal range suitable for visual inspection of items held by grippers. In particular, research that set out to find lenses for the GC650C that supported variable focus, low weight and a close minimum focal range was fortunate to find this Computar lens available, as the alternatives that were found were significantly less suitable (e.g. not sufficiently small, or lacking close focus). When stereo vision sensing is unnecessary, Brainbot has been designed to carry a camera blank and spare lens in the opposite robot eye position to maintain balance and to retain a matched lens pair for potential follow-on stereo vision research. (The single camera design was inspired by the human ability to function exceedingly well with only one eye.) An alternative purpose for a second camera is to function as a fovea, by applying zoom on the second camera lens (either by adjusting the Computar lens to 8 mm, or with a higher zoom lens such as Fujinon's fixed-zoom lenses). This allows detailed visual features to be extracted that would otherwise (without zoom) require a 25 megapixel or greater camera.

Speech recognition is a difficult task for computers and several software vendors exist that claim various accuracy levels. In the case of English speech recognition, after considerable training it is quite possible for existing off-the-shelf software to achieve extremely good recognition accuracy on a limited vocabulary for most common accents. Given the human ability to understand speech at a range of distances it is perhaps unintuitive that computers perform much more poorly at speech recognition when a microphone is displaced at even a conversational distance of 1 meter, but the presence of noise substantially reduces the performance of most current speech processing systems. The typical solution is to place a headset on the speaker, thus providing a microphone at very close range. This solution also works with Brainbot; however, in real-world situations with multiple speakers and/or multiple robots it may be unsatisfactory. A larger, heavier, and typically more expensive alternative is to use an array microphone, which creates a virtual microphone beam aimed at the speaker using multiple built-in microphones and internal computation. Such a solution was judged to be too large and heavy to fit on board Brainbot. Thus, to achieve reasonable performance at conversational distances with off-the-shelf software, Brainbot integrates the Andrea Electronics SuperBeam SoundMax array microphone which achieves reasonable performance at very low cost by offloading the microphone array processing to the connected personal computer. This is an example of one of the multiple features that would be more expensive or not possible without the substantial processing power of the onboard Intel Core-2-Quad.

The Brainbot grippers were designed to manipulate many types of everyday objects. Initially a single gripper design was used for both grippers; however, a difficulty arose in that grippers optimized for pinching small items such as pencils were incapable of grasping relatively larger objects such as light bulbs or soda cans. This problem was exacerbated in initial designs by weak AX-12 based waist and shoulders, often resulting in overheating of shoulder servos and stalled reset of the waist servo. By

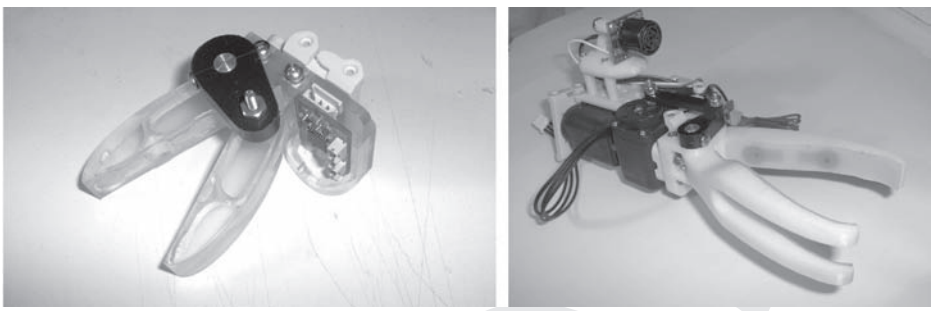

Figure 4.4 The pincher (left) and claw gripper (right) are shown with sonar and laser mounted (right).

strengthening the shoulders to be four times stronger and reinforcing the waist with springs and a servo pair with double strength, the grippers were enabled to lift and hold heavier objects, providing the impetus for a larger gripper that traded away abilities with small objects for large ones, coupled with a gripper able to manipulate smaller objects (Figure 4.4).

The grippers are fabricated by a 3D plastic printer and some aspects are not able to be fabricated without a dissolving system that removes support structure. For example, the pressure sensor wiring pathways internal to the gripper structure, as can be seen in the gripper fabricated in transparent plastic, are inaccessible to tools. Various types of pressure sensors can be placed on the surface of the gripper such as a combination of small circular sensors and a series of pressure-sensing strips that cover the entire interior surface. A layer of polyurethane is then placed over the sensors to provide stickiness. Each gripper is coupled with a miniature input–output (IO) circuit board that receives input from the pressure sensors. The miniature IO board also interfaces with other analog or digital sensors and outputs, such as the MaxBotix EZ0 sonar (interchangeable with EZ1–EZ4 to control volume characteristics), and a programmatically controlled laser pointer. The movement and force sensing of the gripper is provided by an AX-12, and the grippers are interchangeable by interfacing to the standard Robotis brackets for structural support and providing control and sensing through the mini IO board or by plugging directly into the AX-12 bus. It is also possible to mount a self-contained pressure sensor array to a gripper by routing the interface wire, typically USB, from the PC motherboard to the gripper.

Power and data is transferred between the electronic components of Brainbot over many pathways (Figure 4.5). Either wall power or battery power can be supplied to the power board, which provides power either directly or indirectly to all Brainbot systems. We have used a 17-amp 12-volt AC–DC converter for wall power, as well as NiMH 12V batteries with 200 watt-hours of capacity, which achieves 1–2 hours of run time. An important design goal was hot-swappability of power so that intelligent algorithms could be tested "in the wild" all day with a minimum of interference, and to facilitate transfer to/from the work bench. In this configuration, approximately three to four battery kits, allowing two to three to be simultaneously charging, has been sufficient to perform all-day testing without the use of wall power. The power board provides 12V

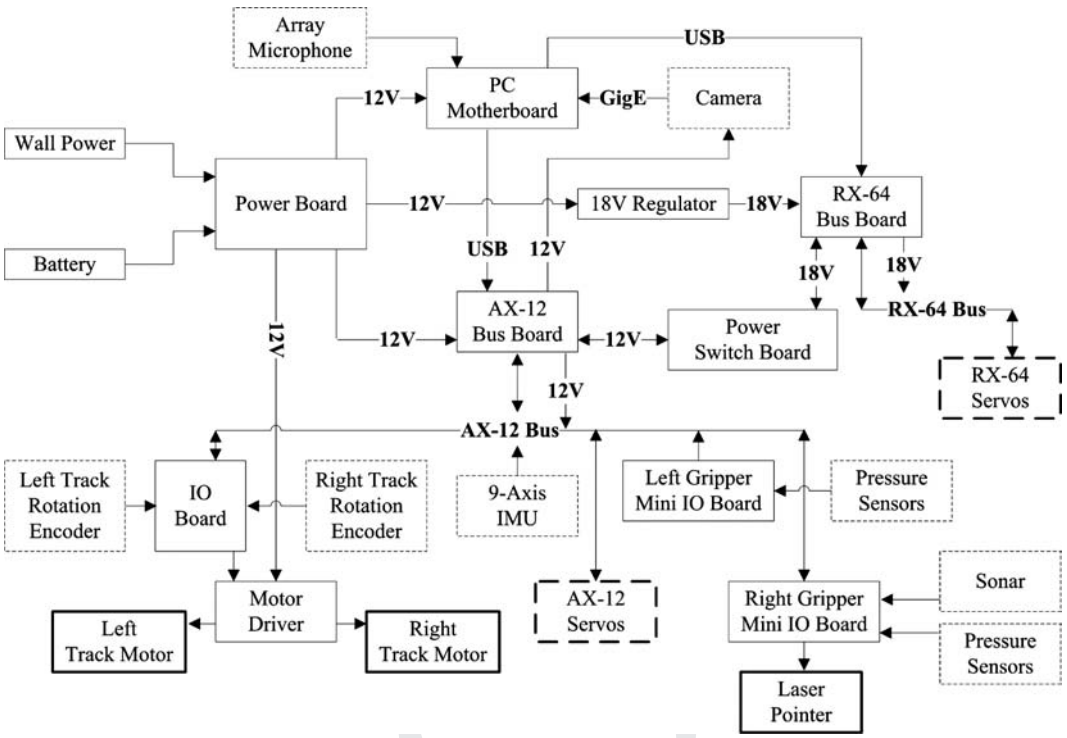

Figure 4.5 Brainbot electronics power and data flow. Bold boxes indicate outputs; dashed boxes indicate sensors.

supplies to the PC motherboard through a 150-watt DC–DC power supply. The power board also supplies 12V to the motor driver that powers the track motors. The RX-64 bus, which provides 18V power to all connected bus devices, receives power from the power board through an 18V step-up regulator. The AX-12 bus provides 12V power to all connected devices and receives 12V power from the power board. An additional 12V supply passes through the AX-12 bus board, which carries an interchangeable array of voltage regulators, such as two AnyVolt Micros from Dimension Engineering and one chassis-mounted AnyVolt-3 in order to power sensitive electronics such as cameras or a laser scanner. The RX-64 and AX-12 bus boards route their power supplies through a power switch board mounted to the back of the chest of Brainbot. In addition, the track motor driver power supply and the PC motherboard have power switches mounted on the power board.

The PC motherboard interfaces with the AX-12 and RX-64 bus boards via USB, through which all servos and bus-based sensors and devices (e.g. track motors, pressure sensors, sonar, laser pointer) can be communicated with. The microphone array and cameras interface directly with the PC motherboard, as is fitting for these higher-bandwidth sensors. The motherboard itself is mounted in the base of the chassis, which is slightly larger than a 19-inch half-rack space, with jacks presented in the back of the Brainbot chassis that include six USB, two gigabit Ethernet, audio, DVI, and eSATA

ports. Internally, four additional USB ports and three SATA II ports are available, and the PCI-Express slot, typically providing the second gigabit Ethernet port, can be repurposed to allow connectivity to other interfaces such as Firewire. Behind the waist, the top of the chassis base serves as a mounting location for USB devices such as 802.11n.

4.2.1 Intelligent algorithm programming interface

BrainTalk provides a simple interface of ASCII over TCP/IP socket to allow any programming language on any operating system to access all of Brainbot's sensors, actuators, and output devices (Figure 4.6). The socket interface also allows algorithms to be executed onboard or externally on a desktop or supercomputer without requiring changes to the software. A difficulty arises in this type of interface, which is that the ASCII over TCP/IP interface is not the most efficient communication mechanism for the large amounts of sensor data collected by the camera or microphone, which can total hundreds of megabits per second in their uncompressed binary form. Two solutions have been implemented on Brainbot to simplify this issue. The first is that the microphone and speaker have speech-to-text voice recognition and text-to-speech voice production built into the BrainTalk interface. This allows programmers to retrieve text that has been spoken to Brainbot as ASCII text and avoids forcing programmers to integrate their own voice recognition to achieve interactivity. Similarly, ASCII text can be sent to Brainbot over the BrainTalk socket, which then uses the built-in text-to-speech software to speak this text over the onboard amplified speaker. The second solution integrates RoboRealm with the onboard camera(s). RoboRealm then provides vision options, such as feature extraction, with multiple interfacing options including TCP/IP, thus preserving the generic operating system and programming language capability of Brainbot. BrainTalk also interfaces with RoboRealm to simplify access to visual feature extraction data for the programmer if desired. Alternatively, the GigE

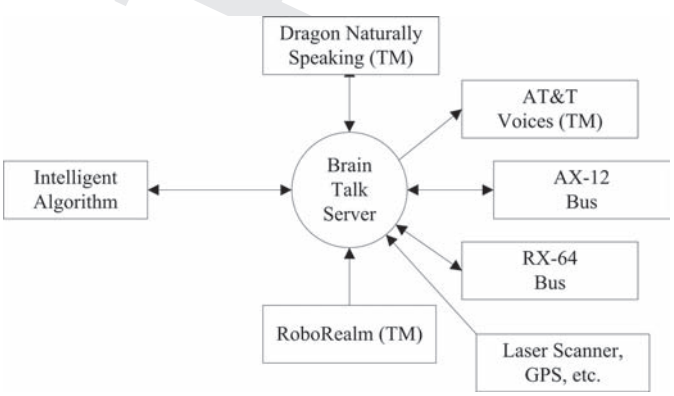

Figure 4.6 The BrainTalk server provides an interface to Brainbot hardware that is optimized for ease of programming.

camera protocol can be used to directly interface with the cameras, or Prosilica's free multiplatform SDK software can be used.

Because BrainTalk is ASCII over TCP/IP, a user can interact with the sensors, motors, etc. over a simple Telnet session. In this way it is possible for users to immediately get up and running with Brainbot, even before selecting a programming language. For example, a Telnet session might proceed in the way shown in Figure 4.7; server responses are indented, comments prefixed with "#".

4.2.2 Brainbot's interactive building blocks

Although Moore's Law has delivered computers with a million times more performance than they had 40 years ago, predictions of a future in which intelligent robots interact with humans in the real world have not yet come to pass. Implementing such creations is further complicated by the limited energy supply that can be carried on board robots. This has traditionally led mobile robots to be designed with low-power embedded computers that are not capable of executing the basic building blocks of intelligent interactive robots in real time (Table 4.2).

To support these building blocks, Brainbot's onboard computer utilizes an Intel Q9550S 65-watt (max) processor containing four Core 2 processor cores running at 2.83 Ghz, 4 GB of memory, and a solid-state hard drive (SSD) for storage. This system allows Brainbot to make at least one processor core available at all times for execution of intelligent algorithms. This arrangement also frees the intelligent algorithms from the low level details such as examining each camera pixel, microphone sample, or auditory frequency, and also frees programmers from having to interact with the low level API interfaces for microphones, speakers, and cameras.

4.3 Application example: a vision algorithm

Recently developed power-efficient algorithms and architectures for real-time visual processing (Felch and Granger, 2009) are being adapted to run with improved performance on hardware suitable for integration on board Brainbot. The vision-processing system extracts features from images and/or image sequences and processes these using a combination of bottom-up (features to abstractions) and top-down (abstractions to features) mechanisms to represent and recognize objects in the images (for background, see Rodriguez *et al*., 2004; Granger, 2006; Felch and Granger, 2009). Here we describe the algorithm as it is being developed for use and testing on the Brainbot platform.

4.3.1 Parallel computing for improved performance of mobile brain algorithms

Parallel processing has become increasingly common, and typical desktop computers now come standard with four processor cores. Although a sufficient number of transistors have been available to implement multiple cores in a processor for many

```
Speaker.say = "Hello, my name is Brainbot, and I am here to learn."
        Ok    #Brainbot says the above text
RightGripperSensor.laser = 1       #Activates laser
        Ok
{Camera.laserSpotX ; Camera.laserSpotY}
        { 632; 413}    #laser spot is believed to appear in upper right corner of camera view
RightWrist.val = 800    # Rotate servo at right wrist to position 800.  Limits are 0-1023, 300 degrees of motion.
Dragon.buffer
        "I am talking to Brainbot and using telnet to retrieve the text she has heard"
Dragon.clearBuffer
{ io.quadA, io.quadB } #retrieve current odometry counts for the tracks, left and right respectively
        { 0; 0 }
{ io.leftSpeed=255; io.rightSpeed=255 }    #Tell motors to drive full speed forward
        Ok
 { io.quadA, io.quadB } #retrieve current odometry counts for the tracks, left and right respectively
        { 926321; 927123 }    #We have moved about 100-inches forward (approximately 9,265 ticks per inch)
{ io.leftSpeed=0; io.rightSpeed=0 }    #Tell motors to drive full speed backward
        Ok
{ io.leftSpeed=127; io.rightSpeed=127 }    #Tell motors to stop
        Ok
Compass.heading    #Which direction are we facing?
        137    #South East (0 = North, 90 = East, 180 = South, 270 = West)
{ Gps.longitude; Gps.latitude }    #Get GPS position.
        { 43.7040 N, 72.2823 W }    #We're at the corner of Wheelock St. and Park St., Hanover, New Hampshire
{ LeftWrist.val = 327; LeftElbow.val = 432; HeadPitch.val = 300 }    #Move servos to designated positions
        Ok
{ LeftWrist.val; LeftElbow; HeadPitch.val }
        {325; 436; 301}    #At desired position with a small error, typical of moving and sensing in the real world
```

Figure 4.7 This telnet session demonstrates the BrainTalk server's ability to access speech input/output, laser pointing and camera sighting, servo movement and sensing, track movement and sensing, and GPS and compass sensing.

Table 4.2. Examples of current state-of-the-art software building blocks for intelligent interactive robots

Ability	Example software	Requisite processing capacity for real time[a]	Requisite memory capacity
Speech recognition	Dragon naturally speaking	1 core	512 MB
Speech production	AT&T voices	1 core	256 MB
Visual feature extraction	RoboRealm	1 core	64 MB

[a] Assuming multigigahertz processor cores.

years, manufacturers of personal computer processors historically pursued better performance for single cores. In fact the transition to multicores was forced upon processor manufacturers when additional improvements to single core designs resulted in malfunctioning processors that consumed too much power and created too much heat to be effectively cooled.

By improving the power efficiency of processors, much more performance can be delivered under a given power envelope. (Brains may well achieve their combination of high computational capacity and relatively low power requirements via the massive parallelism intrinsic to brain circuit design, incorporating billions of processing elements with distributed memory.)

Moore's Gap refers to the difference between system hardware improvements versus system performance improvements. While Moore accurately predicted that hardware components of computer systems would grow exponentially, nonetheless this growth has not resulted in commensurate performance speed-up, measured as the ability of the hardware to carry out software tasks correspondingly faster. The gap arises from the difference between processor speed on one hand, versus the mapping of software instructions onto those processors on the other.

The phenomenon of Moore's Gap is consistent with, and indeed was in part predicted by Amdahl's Law, which can find the maximum expected improvement to a system as a function of an increased number of processors. If S is the fraction of a calculation that is inherently sequential (i.e, cannot be effectively parallelized), then $(1 - S)$ is the fraction that can be parallelized, and Amdahl's Law states that the maximum speed-up that can be achieved using N processors is:

$$\frac{1}{S + \frac{(1-S)}{N}}$$

In the limit, as N gets large, the maximum speed-up tends toward $1/S$. In practice, price to performance ratios fall rapidly as N is increased: that is, once the quantity $(1 - S)/N$ becomes small relative to S. This implies differences in kind between tasks that can be effectively parallelized ($S \ll 1$) versus those that cannot ($S \approx 1$). Current architecture designs are tailored to $S \approx 1$ tasks such as large calculations, and perform notably poorly on $S \ll 1$ tasks.

Typical approaches to this problem are aimed at trying to modify tasks to be more parallelizable. For instance, recently Asanovic *et al*. (2006) have identified multiple classes of numerical methods that can be shown to scale well in parallel and yet are general methods with broad applicability to multiple domains. These methods are known to include many statistical learning, artificial neural net, and brain-like systems.

4.3.2 BrainBot as a brain-derived vision algorithm and hardware development platform

The vision system extracts features from images or image sequences, which is a preliminary step common to many vision systems. For example, two types of features that can be extracted are corner features and line segment features. The system is able to work with them interchangeably: that is, using relationships of corners to corners, corners to line segments, line segments to corners, and line segments to line segments. For explanatory purposes we will limit the discussion to corners; the system extends beyond this in a straightforward way.

The system uses a special data structure (Figure 4.8) to hold information about the types of corner configurations or "partial-constellations," which we will term atomic relations or atoms. An atom describes two or more subfeatures and their expected spatial relationship in the image plane. The data structures are used to identify substructures in an image that have been previously associated with objects that can be recognized. For example, an atom might relate the four corners of a windshield to each other so that windshields can be recognized and, once recognized, can cause the system to search for other features found on automobiles.

Each atom has a "center of gravity" (CoG), which is a point in the middle of the constellation to which all of the subfeatures relate. Each subfeature has an identity such as

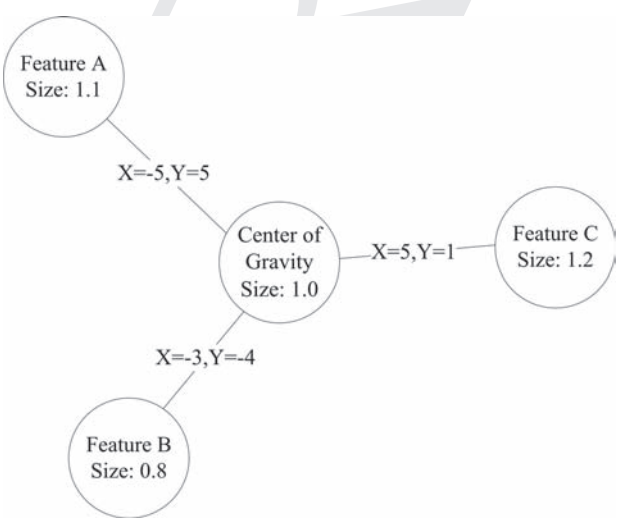

Figure 4.8

"feature A," "feature B," or "feature C," and an X,Y offset of the subfeature to the CoG given a certain size of the atom or subfeature. The process of deriving atom centers and sizes from subfeatures is called the hierarchical bottom-up process or HBU. The process of deriving subfeature x,y locations and sizes from an atom's CoG and size is called the hierarchical top-down process or HTD.

More rigorously, for a given feature detected in an image F_i with type T_{Fi} and size S_{Fi} an atom data structure A_j with one subfeature of type T_{Fi} having relational offset $\left(\bar{X}_{A_{j.T_{F_i}}}.\bar{Y}_{A_{j.T_{F_i}}}\right)$ and relational size $\bar{S}_{A_{j.T_{F_i}}}$ can derive its center of gravity location $\left(\tilde{X}_{F_i,A_j},\tilde{Y}_{F_i,A_j}\right)$ using Equations (4.1) and (4.2) and center of gravity size $\tilde{S}_{F_{i.A_j}}$ using Equation (4.3):

$$\tilde{Y}_{F_i,A_j} = Y_{F_i} - \left(\bar{Y}_{A_{j.T_{F_i}}} * \frac{S_{F_i}}{\bar{S}_{A_{j.T_{F_i}}}}\right), \tag{4.1}$$

$$\tilde{Y}_{F_i,A_j} = Y_{F_i} - \left(\bar{Y}_{A_{j.T_{F_i}}} * \frac{S_{F_i}}{\bar{S}_{A_{j.T_{F_i}}}}\right), \tag{4.2}$$

$$S_{F_i,A_j} = \frac{S_{F_i}}{\bar{S}_{A_{j.T_{F_i}}}}. \tag{4.3}$$

The HBU process derives the expected location and size for each relevant atom's CoG. Equations (4.1) to (4.3) are calculated for all incoming features and relevant atoms during the HBU process. The HTD process, by contrast, derives for each atom's CoG instance the expected locations and sizes of the other subfeatures within that atom. Equations (4.1) to (4.3) are solved for subfeature location/size derivation given a CoG location/size in Equations (4.4) to (4.6) respectively:

$$X_{F_i} = \tilde{X}_{F_i,A_j} + \left(\bar{X}_{A_j.T_{F_i}} * S_{F_i,A_j}\right), \tag{4.4}$$

$$Y_{F_i} = \tilde{Y}_{F_i,A_j} + \left(\bar{Y}_{A_j.T_{F_i}} * S_{F_i,A_j}\right), \tag{4.5}$$

$$S_{F_i} = S_{F_i,A_j} * \bar{S}_{A_j.T_{F_i}}. \tag{4.6}$$

Once expected locations for subfeatures have been calculated, the image can be compared with these expectations, and image features of the same type T_{Fi} can be measured for distance from the expected subfeature location. The minimum distance to the closest matching feature (same type) is summed with the other minimum distances to determine an overall match score for the atom A_j Figures 4.9–4.14 give an example of calculating the overall expectation-deviation score (EDS).

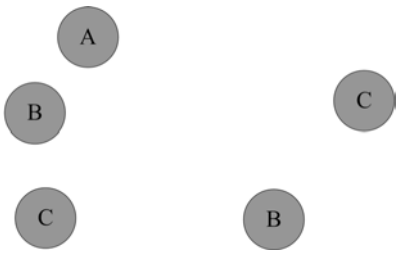

Figure 4.9

Figure 4.10

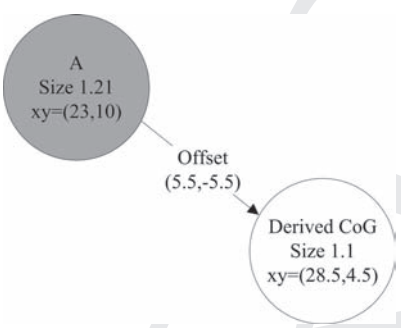

Figure 4.11

Corner features of types A, B, and C have been extracted from an image and are arranged as shown in Figure 4.9. Figure 4.10 shows the details of one of the feature instances, namely the A-type feature instance. Using the data structure of Figure 4.8 and Equations (4.1–4.3) the location and size of the CoG are derived as shown in Figure 4.11.

In Figure 4.12 the derived CoG location and size are used with Equations (4.4–4.6) to determine the location and size of the constituent type-B subfeature. In Figure 4.13 the expected location of subfeature B is compared with type-B features found in the image. Note that any image features not of similar size to the expected size (e.g. within 20%) will not be considered for the minimum-distance calculation. In this example two type-B image features are of a size similar to the expected size and their distances to the expected location are calculated as 1.8 and 4.3. Thus the minimum distance is 1.8 for the type-B constituent subfeature.

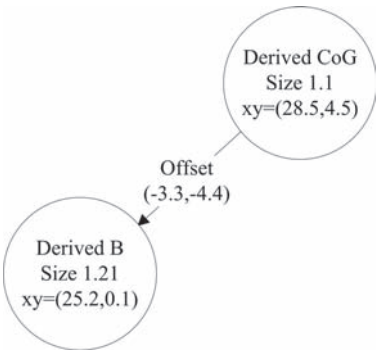

Figure 4.12

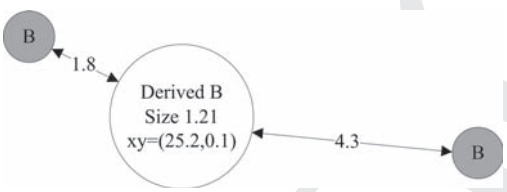

Figure 4.13

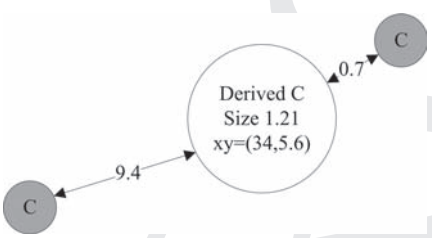

Figure 4.14

Figure 4.14 shows the minimum distance calculation for the type-C subfeature, calculated in a similar fashion to that of the type-B subfeature (Figures 4.12, 4.13). The minimum distance for the type-C subfeature is 0.7. Thus the overall expectation-deviation score, the sum of the minimum distances, is 0.7 + 1.8 = 2.5.

The process of searching for all the implied atom instances in an input image is shown in Figure 4.15. When an input arrives (1201) the atom data structures that use the input feature's type in a subfeature are iterated through (1202) until finished (1203). It is also possible that the input feature is not a subfeature (1204) but is in fact the CoG of a specific atom (passed from a HTD process), in which case only the relevant atom data structure is retrieved. In the case that the input is a subfeature (not a CoG) the size (1206) and location (1207) are derived for the CoG using Equations (4.1–4.3), an example of which was shown in Figure 4.14. Next, the subfeatures of the atom are iterated through (1205). For each subfeature (1208) the expected size and location is

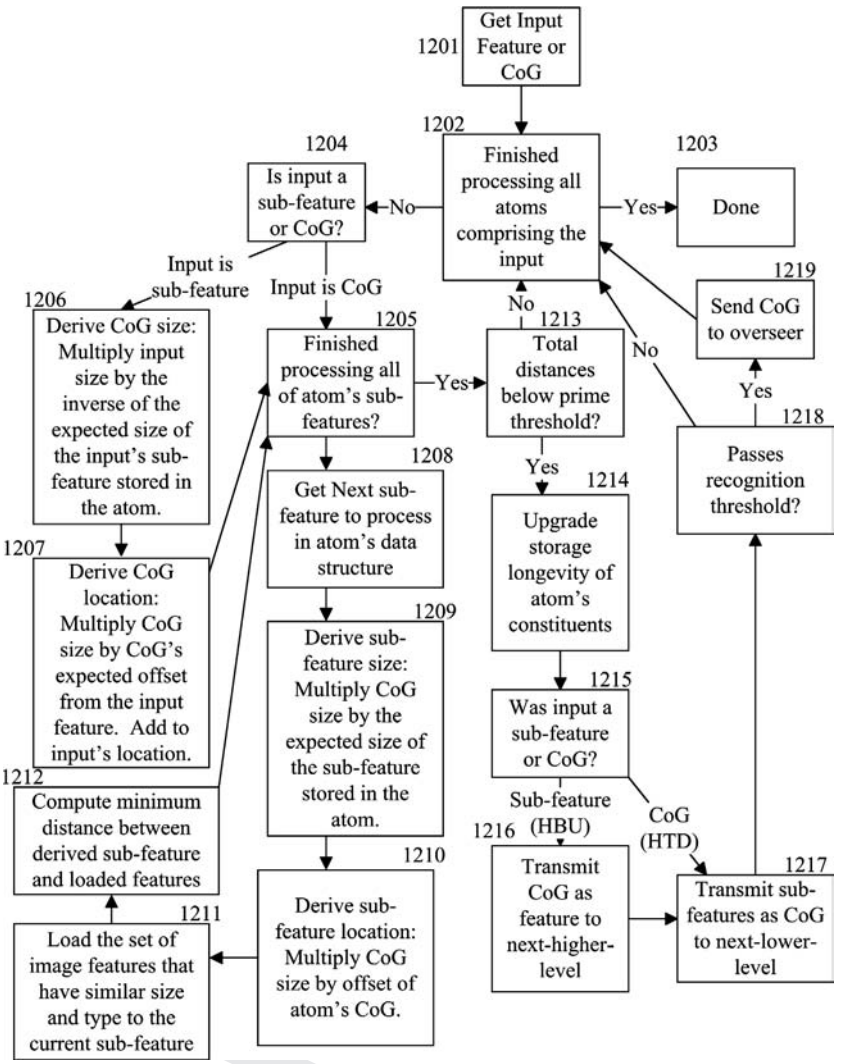

Figure 4.15 Flow diagram of node message processing.

derived using Equations (4.4–4.6), an example of which was shown in Figure 4.12. Next, the relevant image features (of the same type) that are within a certain maximum distance and which have a size within some percentage of the derived subfeature's size are processed (1211). The distance from the image feature's location to the current sub-feature's expected location is measured and the minimum distance (1212) is summed to the EDS running total for the current atom.

Once complete, the EDS is compared with a threshold (1213) and if the EDS is too high then processing the current atom has completed and the next atom (if not finished, 1203) is moved onto (1204). If the EDS is below the threshold, then the image features that achieved minimum distances are "refreshed" in memory so that when future

image frames arrive the old features will still be accessible. Image features that are not refreshed by this process are removed from memory after input of the next or just a few new frames, and thus will no longer arrive in sets processed in step 1211. Note that if DRAM is used to hold the image features in memory then it may be possible to use DRAM's natural data leakage "forgetting" mechanism to implement this erasure.

If the EDS qualified for refresh, then additional signals are also sent depending on whether the input feature was a subfeature or CoG (1215). In either case the refreshed subfeatures are transmitted as CoGs to lower level processes (HTD). If the input was a subfeature (sent from an HBU process) then the CoG identified by the current atom and the size/locations derived in steps 1206 and 1207 is sent to higher-level processes (HBU) as a subfeature. Finally, if the EDS is below the "recognition threshold" (1218) then the CoG is transmitted to an overseer process (1219). The overseer process determines what object is being recognized by the input CoG and can act on this information such as navigating a robot to further investigate the object, to direct a grasping action, or to navigate around the object.

In summary, the system forms a hierarchy from the atoms found in an image. When initially detected features (corners) are found in a configuration expected by a particular atom, the atom acts as a "detected feature" to the next-higher level of processing. The next-higher level of processing performs exactly as if its input features are corners, but in fact they are atoms that identify constellations of corners. The atom data structures used in the next-higher level of processing describe relationships not between corners, but between atoms identified at the next-lower level of processing. This procedure allows identification of higher and higher feature levels, with each processing level called a layer. Once a hierarchy is formed in one of the higher levels, and the constituent atoms and corners have matched the input image well, a signal is sent to an overseer process indicating that a very high level feature has been identified. The overseer maintains a list that associates high-level atoms with object names, so that the overseer can determine what object has been identified based on the highest-level atom in the hierarchy.

A prototype of the above vision algorithm was tested on a class of difficult data (Figure 4.16) and performance was shown to closely match the current best system at the task (Carmichael, 2003; Felch *et al*., 2007); however, these tests took several days to compute. Research is being conducted in order to develop new computer hardware architectures that utilize the intrinsic parallelism of the algorithm to greatly improve performance while satisfying the low power requirements of mobile robots. Prior research into a related algorithm has shown performance-per-watt improvements on the order of 1000 × (Furlong *et al*., 2007).

4.4 Task design and customization

4.4.1 Voodoo control

During development of intelligent algorithms, issues arise that must first be identified as an intelligence issue or a system issue. For example, the robot may repeatedly fail

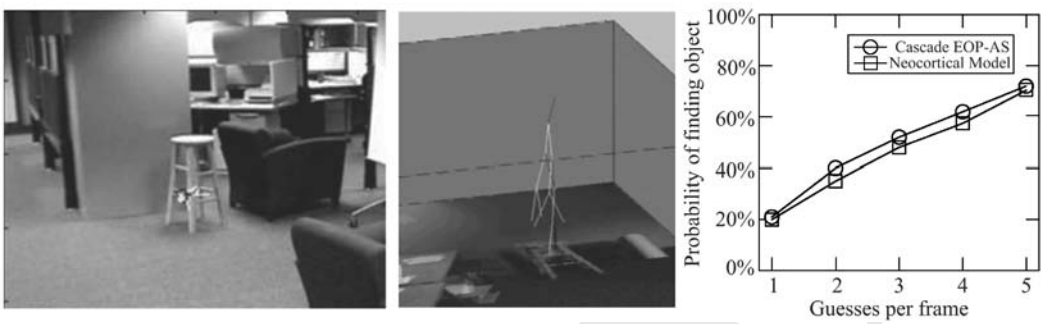

Figure 4.16 A representative wiry object (sitting stool) is recognized by the vision algorithm prototype (left). A 3D visualization of the hierarchy of abstract features is constructed through HBU and HTD processes (middle). Performance closely matches the best extant system at recognizing these objects.

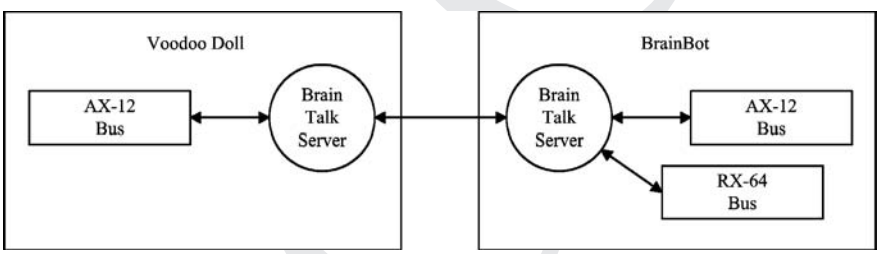

Figure 4.17 The hardware/software configuration of Voodoo control.

at a grasping task, and the issue may arise from a lack of dexterity or a lack of intelligence. An interesting aspect of the Brainbot platform is that the robot can be remotely controlled by a human using a controller, called the Voodoo bot, fashioned with the same shape and degrees of freedom as the Brainbot robot without the base component (Figures 4.17, 4.18). By first testing with Voodoo, it is possible to verify that a task is in fact possible for a given Brainbot configuration, and this is true in both the real and virtual world. Using a monitor also allows a first-person viewpoint that can help in identifying the degree of visual difficulty in a task.

Voodoo control has proven to be extremely useful in a number of ways, the most valuable of which in our experience has been its ability to weed out hardware and software bugs very early in development, before any advanced algorithms need to be written. Furthermore, a joystick has been integrated to allow driving of the tracks by wireless so that a large outdoor area can be tested for issues, such as for wireless interference. Voodoo was also very useful for testing and debugging the Brainbot virtual environment before the virtual model had been fully developed.

The Voodoo bot is constructed of AX-12 servos on all joints, and a laptop or netbook connects to the AX-12 bus using an AX-12 bus board (the bus board is powered by a local battery such as a 9.6V NiMH). Handles are attached at the head and gripper joints

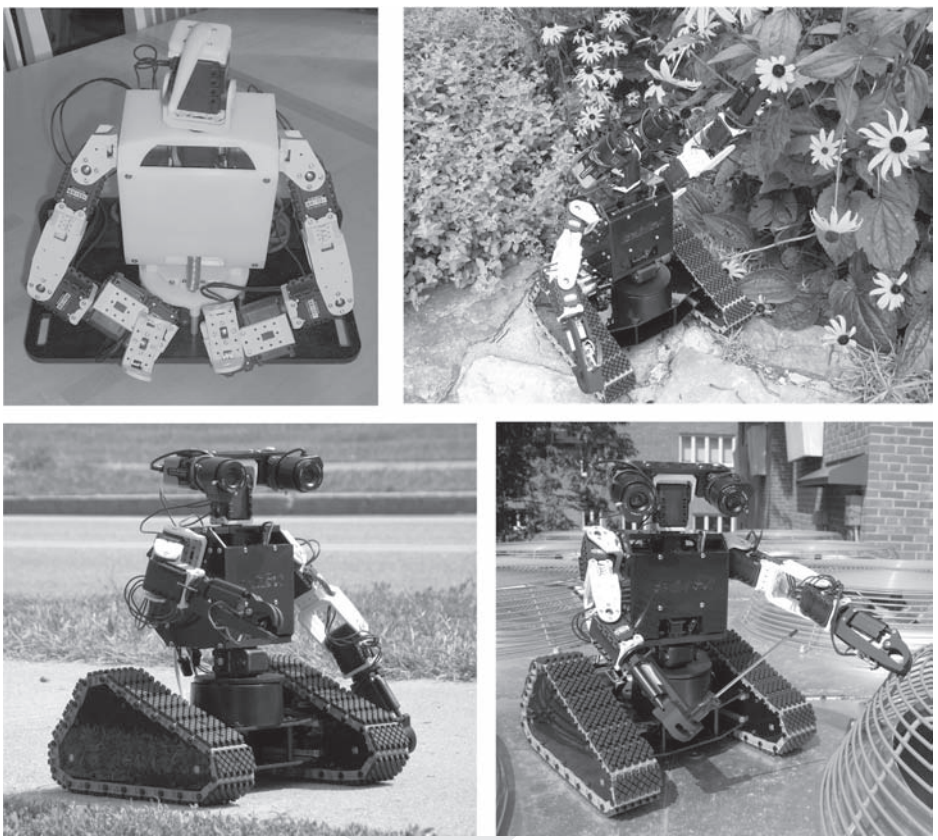

Figure 4.18 An experimental Brainbot design is tested through Voodoo control (Voodoo controller in upper left) for hardware capabilities at various tasks.

of the Voodoo bot to improve control, and the entire system is made wireless by using a shoulder strap to suspend the Voodoo bot in front of the user and placing the laptop in a backpack. A BrainTalk server runs on the laptop and frequently polls the position of the servos, which are then sent as position commands to the Brainbot robot designated in a configuration file which may be connected through a wireless network such as an ad-hoc 802.11n network connection.

4.4.2 Social cognition

Some problems can be better tackled by a group of robots rather than a single robot. Understanding the design principles that better enable robots to work together is an increasingly important research area. Multiagent systems have many properties that make the design of intelligent agents more difficult. These robots can have different sensor complements, morphologies, and locomotion, each more or less well suited for certain steps in the task at hand. The Brainbot platform has been studied for its

suitability to multiagent tasks in which a team of robots collaborates to efficiently work together toward a common goal at the behest of a human or another robot. The result is that Brainbot appears quite amenable to multiagent tasks and may be especially suited for a difficult type of multiagent task in which a heterogeneous group of robots must work together in the changing and uncertain real world. The highly configurable nature of the Brainbot platform supports the creation of a diverse group of agents, each with different capabilities. For example, the extensibility of Brainbot allows each agent to be outfitted with different sensors such as a team that includes one member with a laser range scanner, one with stereo vision, and one with an infrared camera. The Brainbot platform provides all sensor data through TCP/IP sockets which, coupled with multiple USB wireless modules such as 802.11n, yields a high bandwidth mesh network that allows multiple robots to see the problem using a collection of sensors that no single robot is directly outfitted with.

Similarly, Brainbot's morphology can be easily changed through the reconfiguration of the arms, neck, or torso using the erector-set-like Robotis pieces that have been designed for this purpose, or by modifying the CAD files of the printer parts to enable 3D printing of new custom parts such as new grippers (Figure 4.19). The daisy-chain connection of new servos avoids excessive wiring issues (which plagued small robot servos prior to the AX-12), and the plastic Brainbot chassis is easily drilled/tapped to allow screw holes for attachment of additional appendages or instruments. Various Robotis servos can be added such as the AX-12 and RX-64 (shoulders) standard types, as well as the RX-28 and stronger EX-106 servos. Combined, this provides for configurations with joints ranging from 16 kg-cm to 106 kg-cm holding torque. Wheeled and legged locomotion arrangements have also been successfully tested in the laboratory in both virtual and real-world settings, allowing for truly diverse studies of social cognition using teams of Brainbots (Figure 4.20).

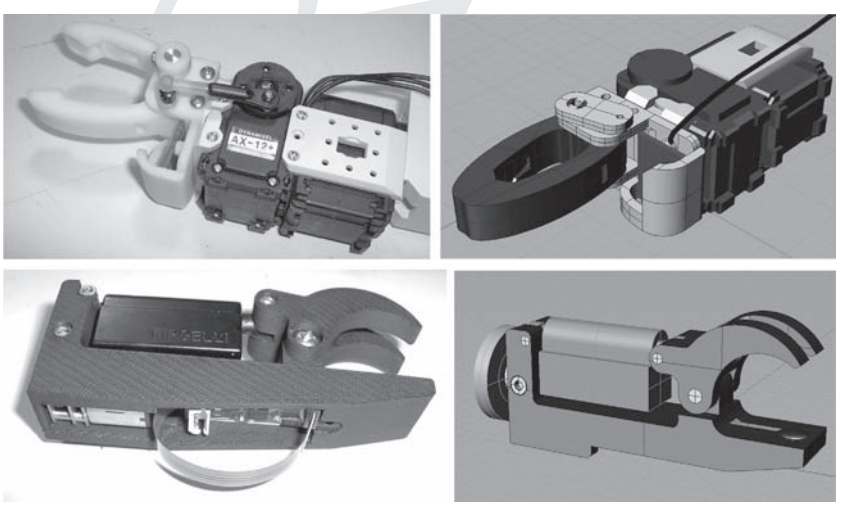

Figure 4.19 Various configurations of virtual and real Brainbot.

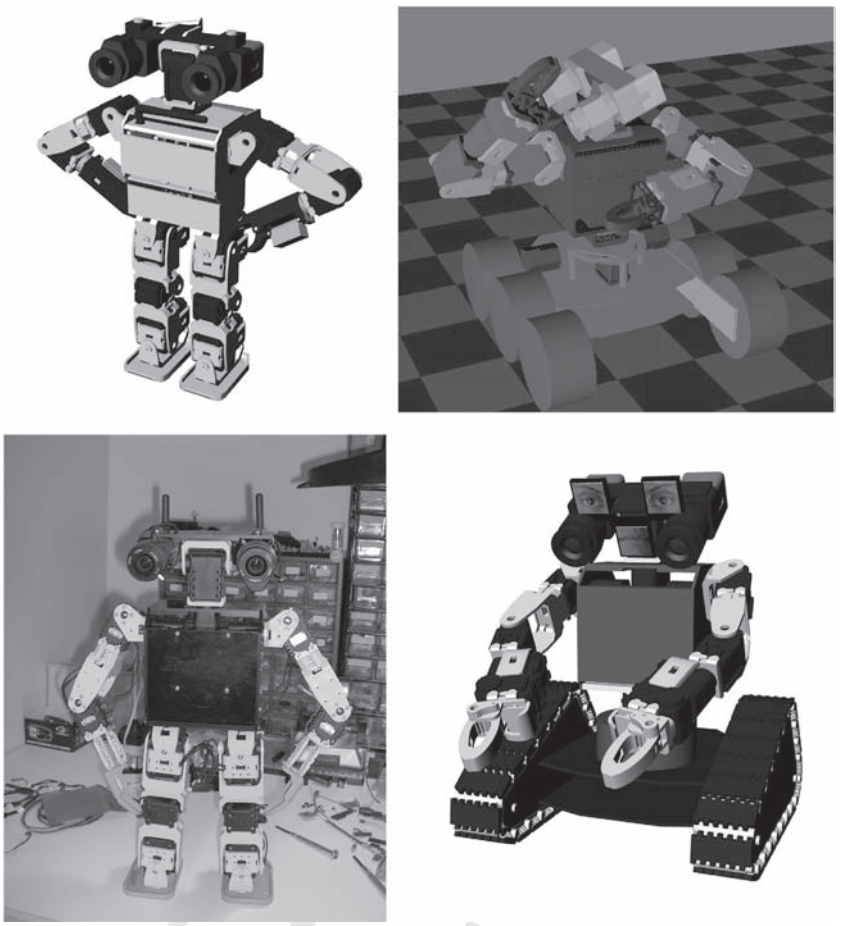

Figure 4.20 Virtual and real gripper designs.

The low cost of Brainbot has allowed research laboratories to purchase teams for studies of social behavior. Furthermore, the open source virtual implementation of multiple Brainbots in a single virtual environment allows the simulation of social cognition experiments for the cost of the simulation program (e.g. Webots). Subsequent transition to the real world is facilitated because each Brainbot is controlled through a separate BrainTalk TCP/IP interface, and these programming interfaces are identical to the real-world Brainbot. The transition both to and from the real world may be further facilitated by using the Brainbot onboard computer as the simulation platform, since the onboard 2.83 GHz Quad-Core Nehalem processor is fully capable of running the Webots simulator while also providing three cores for execution of driving algorithms. In this way, a brain algorithm can be transitioned between the virtual and real worlds by simply changing the designated BrainTalk server IP address. Discrepancies between virtual and real worlds will of course have interesting effects on robot behavior and algorithm performance.

4.5 Discussion

The real world is the ultimate arbiter of embodied perception and action; even carefully devised simulated experiments typically fail to transfer to real-world settings. Yet simulation is often preferable, due to the engineering requirements, programming minutiae, sheer cost, and lack of standardization of robot platforms. Current robot systems exhibit a wide range of dexterity, motor function, appearance, sensors, and computer power, but our interests were overwhelmingly aimed at the study of learning over time, accreting perceptual and motor abilities, in real-world settings. On many advanced computing tasks, including perceptual recognition, motor performance, and processing time-varying data, humans still substantially outperform even the best engineering systems. Living organisms acquire knowledge of the world, organize that knowledge in memories, and use those memories to perform in their learned environments, and transfer the knowledge to novel environments. This suggests that the underlying mechanisms of learning and knowledge organization still are key to identifying the biological mechanisms that so impressively achieve these real-world perceptual and motor tasks. Robot platforms for this research are thus most useful to the extent that they target these two overarching desiderata: extensive learning, occurring in real-world environments. This entails a sturdy (albeit simple) chassis, extensive sensors, and the most powerful possible onboard processors, while dispensing with most other features, especially any that add weight and/or increase power consumption or cost. This overall design decision is embodied in the Brainbot platform: a sensor-rich, lightweight robot with high onboard processing power, using open-source hardware designs and driving software. This powerful toolkit provides most low-level programming necessities, enabling testing of advanced algorithms ranging from learning and perception to reasoning and language, in real-world environments. Current testing includes mechanisms for visual and auditory recognition, real-time processing of time-varying input, performance in the presence of extensive noise, perceptual-motor learning in complex environments, construction of long-term hierarchical memories, exploratory learning in simulated and real settings, and accrual of knowledge over extended time periods.

4.5.1 Future work

The BrainTalk server is implemented in Squeak Smalltalk, a virtual environment that has been ported to many operating systems, such as ARM Linux. The availability of small, low-power systems that support Squeak allows the BrainTalk server to be installed on even very small systems. BrainTalk is currently being adapted to run on an AX-12 based robot with a bipedal dinosaur-like morphology as well as other designs, depicted in Figure 4.21. A miniature 6-inch submarine is also being examined for potential BrainTalk compatibility.

The BrainTalk server itself was designed for the purpose of providing a cross-platform interface for ease of initial programming; other robot operating systems, such

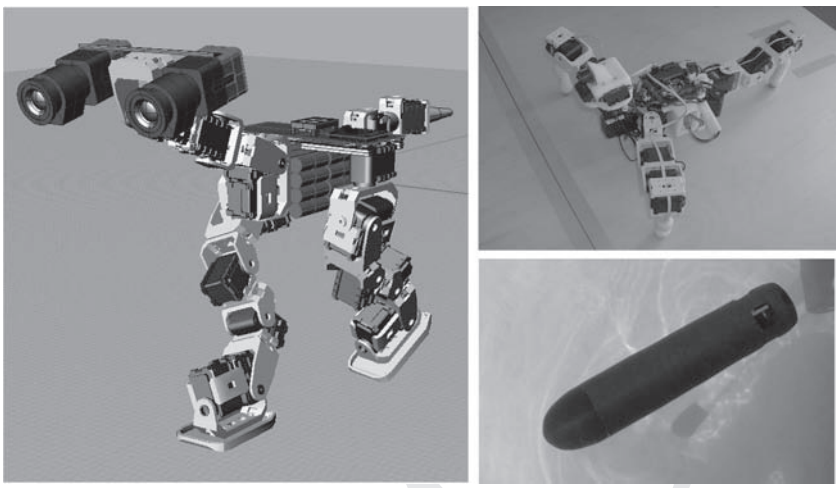

Figure 4.21 Brainbot platforms in development.

as ROS being developed by Willow Garage, or the URBI robot interface, include much more functionality. It is anticipated that the open Brainbot system can be readily ported to either of these (or other) platforms.

References

Asanovic, K., Bodik, R., Catanzaro, B., *et al*. (2006). *The Landscape of Parallel Computing Research*. Technical Report UCB/EECS-2006–183. Berkeley, CA: EECS Department, University of California.

Carmichael, O. (2003). *Discriminative Techniques for the Recognition of Complex-Shaped Objects*. Technical Report CMU-RI-TR-03–34. PhD thesis, The Robotics Institute, Carnegie Mellon University.

DOD (US Department of Defense) (2005). *Unmanned Aircraft System Roadmap 2005–2030*. Washington, DC: US Department of Defense.

Felch, A. and Granger, R. (2008). The hypergeometric connectivity hypothesis: divergent performance of brain circuits with different synaptic connectivity distributions. *Brain Research*, **1202**, 3–13.

Felch, A. and Granger, R. (2009). *Power-Efficient Computation of High-Level Computer Vision in Real Time*. U.S. Patent application.

Felch, A., Chandrashekar, A., Moorkanikara, J., *et al*. (2007). Accelerating brain circuit simulations of object recognition with a Sony Playstation 3. In *International Workshop on Innovative Architectures (IWIA 2007)*, pp. 33–42.

Furlong, J., Felch, A., Nageswaran, J., *et al*. (2007). Novel brain-derived algorithms scale linearly with number of processing elements. In *Proceedings of the International Conference on Parallel Computing (parco.org) 2007*, pp. 767–776.

Granger, R. (2006). Engines of the brain: the computational instruction set of human cognition. *AI Magazine*, **27**, 15–32.

HR (House of Representatives) (2000). H.R. 4205/Public Law no. 106–398 of October 30, 2000.

Rodriguez, A., Whitson, J., and Granger, R. (2004) Derivation and analysis of basic computational operations of thalamocortical circuits. *Journal of Cognitive Neuroscience*, **16**, 856–877.

Thrun, S., Montemerlo, M., Dahlkamp, H., *et al*. (2006). Stanley: the robot that won the DARPA grand challenge: research articles. *Journal of Robotic Systems*, **23**(9), 661–692.

USN (U.S. Navy Unmanned Underwater Vehicle) (2004). *UUV Master Plan*, November 2004. Washington DC: US Department of the Navy.