ActivMedia Robotics
MobileROBOTS.com

Pioneer 3™ Operations Manual

with

NEW Renesas SH2-based Controller

&

*Activ*Media's Robot Control & Operations Software

Pioneer 3-SH Operations Manual, version 1, September 2004

Important Safety Instructions

- ✓ Read the installation and operations instructions before using the equipment.
- ✓ Avoid using power extension cords.
- ✓ To prevent fire or shock hazard, do not expose the equipment to rain or moisture.
- ✓ Refrain from opening the unit or any of its accessories.
- ✓ Keep wheels away from long hair or fur.
- ✓ Never access the interior of the robot with charger attached or batteries inserted.

Inappropriate Operation

Inappropriate operation voids your warranty! Inappropriate operation includes, but is not limited to:

- ✓ Dropping the robot, running it off a ledge, or otherwise operating it in an irresponsible manner
- ✓ Overloading the robot above its payload capacity
- ✓ Getting the robot wet
- ✓ Continuing to run the robot after hair, yarn, string, or any other items have become wound around the robot's axles or wheels
- ✓ Opening the robot with charger attached and/or batteries inserted
- ✓ All other forms of inappropriate operation or care

Use *Activ*Media Robotics' authorized parts *ONLY*;
warranty void otherwise.

Table of Contents

Chapter 1 Introduction

Figure 1. Pioneer Mobile Robots first appeared commercially in 1995.

Congratulations on your purchase and welcome to the rapidly growing community of developers and enthusiasts of *Activ*Media Robotics' intelligent mobile robots.

This *Pioneer 3-SH Operations Manual* provides both the general and technical details you need to operate your new Pioneer 3-DX or –AT robot and to begin developing your own robotics hardware and software.

For operation of previous versions of Pioneer 2 and 3 which use the Siemens C166- or Hitachi H8S-based robot controllers, original motor-power boards, and support systems, please contact sales@activmedia.com or access our support website: http://robots. activmedia.com for their related documentation.

ROBOT PACKAGES

Our experienced manufacturing staff put your mobile robot and accessories through a "burn in" period and carefully tested them before shipping the products to you. In addition to the companion resources listed above, we warrant your *Activ*Media robot and our manufactured accessories against mechanical, electronic, and labor defects for one year. Third-party accessories are warranted by their manufacturers, typically for 90 days.

Even though we've made every effort to make your *Activ*Media Robotics package complete, please check the components carefully after you unpack them from the shipping crate.

Basic Components (all shipments)

- ✓ One fully assembled mobile robot with battery
- ✓ CD-ROM containing licensed copies of *Activ*Media software and documentation
- ✓ Hex wrenches and assorted replacement screws
- ✓ Replacement fuse(s)
- ✓ Set of manuals
- ✓ Registration and Account Sheet

Optional Components and Attachments (partial list)

- ✓ Battery charger (some contain power receptacle and 220VAC adapters)
- ✓ Automated dock and recharge station
- ✓ Onboard PC computer and accessories
- ✓ Radio Ethernet
- ✓ Supplementary and replacement batteries
- ✓ 3-Battery Charge Station (110/220 VAC)
- ✓ Added sonar arrays
- ✓ 2-DOF Gripper
- ✓ 5-DOF P2 Arm with gripper
- ✓ *Activ*Media Color Tracking System (ACTS)
- ✓ Stereo Vision Systems
- ✓ Pan-Tilt-Zoom Surveillance Cameras
- ✓ Custom Vision System
- ✓ Range-finding laser
- ✓ Global Positioning System

- ✓ Heading-correction gyro
- ✓ Compass
- ✓ Bumper rings
- ✓ Serial cables for external connections
- ✓ Many more…

User-Supplied Components / System Requirements

- ✓ Client PC: 586-class or later PC with Microsoft Windows© or Linux OS
- ✓ One RS-232 compatible serial port or Ethernet
- ✓ Four megabytes of available hard-disk storage

ADDITIONAL RESOURCES

New *Activ*Media Robotics customers get three additional and valuable resources:

- ✓ A private account on our support Internet website for downloading software, updates, and manuals
- ✓ Access to private newsgroups
- ✓ Direct access to the *Activ*Media Robotics technical support team

Support Website

We maintain a 24-hour, seven-day per week World Wide Web server where customers may obtain software and support materials:

http://robots.activmedia.com

Some areas of the website are restricted to licensed customers. To gain access, enter the username and password written on the *Registration & Account Sheet* that accompanied your robot.

Newsgroups

We maintain several email-based newsgroups through which *Activ*Media robot owners share ideas, software, and questions about the robot. Visit the support `http://robots.activmedia.com` website for more details. To sign up for pioneer-users, for example, send an e-mail message to the `-requests` automated newsgroup server:

To: **pioneer-users-requests@activmedia.com**
From: <*your return e-mail address goes here*>
Subject: <*choose one command:*>
help (returns instructions)
lists (returns list of newsgroups)
subscribe
unsubscribe

Our SmartList-based listserver will respond automatically. After you subscribe, send your email comments, suggestions, and questions intended for the worldwide community of Pioneer users:[1]

To: **pioneer-users@activmedia.com**
From: <*your return e-mail address goes here*>
Subject: **<something of interest to pioneer users>**

Access to the `pioneer-users` newslist is limited to subscribers, so your address is safe from spam. However, the list currently is unmoderated, so please confine your

[1] Note: Leave out the –requests part of the email address when sending messages to the newsgroup.

comments and inquiries to issues concerning the operation and programming of Pioneer or PeopleBot robots.

Support

Have a problem? Can't find the answer in this or any of the accompanying manuals? Or do you know a way that we might improve our robots? Share your thoughts and questions with us from the online form at the support website:

`http://robots.activmedia.com/techsupport`

or by email:

`support@activmedia.com`

Please include your robot's ***serial number*** (look for it beside the `Main Power` switch)—we often need to understand your robot's configuration to best answer your question.

Tell us your robot's SERIAL NUMBER.

Your message goes directly to the *Activ*Media Robotics technical support team. There a staff member will help you or point you to a place where you can find help.

Because this is a support option, not a general-interest newsgroup like `pioneer-users`, we reserve the option to reply only to questions about problems with your robot or software.

See Chapter 8*, Maintenance & Repair,* for more details.

Use *Activ*Media Robotics' authorized parts *ONLY*;
warranty void otherwise.

Chapter 2 What Is Pioneer?

Pioneer is a family of mobile robots, both two-wheel and four-wheel drive, including the Pioneer 1 and Pioneer AT, Pioneer 2™ -DX, -DXe, -DXf, -CE, -AT, the Pioneer 2™-DX8/Dx8 *Plus* and -AT8/AT8 *Plus*, and the newest Pioneer 3-DX and -AT mobile robots. These small, research and development platforms share a common architecture and foundation software with all other *Activ*Media robots including AmigoBot™, PeopleBot™ V1, Performance PeopleBot™, and PowerBot™ mobile robots.

Figure 2. Some ActivMedia robots

PIONEER REFERENCE PLATFORM

*Activ*Media robots set the standards for intelligent mobile platforms by containing all of the basic components for sensing and navigation in a real-world environment. They have become reference platforms in a wide variety of research projects, including several US Defense Advanced Research Projects Agency (DARPA) funded studies.

Every *Activ*Media robot comes complete with a sturdy aluminum body, balanced drive system (two-wheel differential with caster or four-wheel skid-steer), reversible DC motors, motor-control and drive electronics, high-resolution motion encoders, and long-life, hot-swappable battery power, all managed by an onboard controller and mobile-robot server software.

Besides the open-systems robot-control server software onboard the robot controller, every *Activ*Media robot also comes with a host of advanced robot-control client software applications and applications-development environments. Software development includes our own foundation *Activ*Media Robotics Interface for Applications (ARIA), released under the GNU Public License, and complete with fully documented C++, Java, and Python libraries and source code. Several third-party robotics applications development environments also have emerged from the research community for *Activ*Media robots, including Saphira from SRI International, Ayllu from Brandeis University, Pyro from Bryn Mawr and Swarthmore Colleges, Player/Stage from the University of Southern California, and Carmen from Carnegie-Mellon University.

Every ActivMedia robot comes with a plethora of expansion options, including built-in hardware support for sonar and bump sensors and lift/gripper effectors, as well as serial port and server software support for a number of sensors, effectors, and control accessories, like an onboard PC system, automated docking/recharging system, laser range-finder, 5-DOF arm, robotic pan-tilt cameras, and much, much more.

PIONEER FAMILY OF ROBOT CONTROLLERS AND OPERATION SYSTEM SOFTWARE

The original Pioneer 1 mobile robot contained a controller based on the Motorola 68HC11 microprocessor and powered by Pioneer Server Operating System (PSOS) software. The first generation of Pioneer 2 and PeopleBot robots used a Siemens C166-based controller with Pioneer 2 Operating System (P2OS) software. Until now, the Pioneer 3, Performance PeopleBot, and PowerBot robots had an Hitachi H8S-based controller with *Activ*Media Robotics Operating System (AROS) software.[2]

[2] AmigoBot has an H8S-based controller, too, but uses the AmigoBot Operating System tailored for its electronics.

All *Activ*Media robots now use a revolutionary new, high-performance controller with *Activ*Media's Robot Control and Operations Software (ARCOS) based on the new-generation 32-bit Renesas SH2 microprocessor. This new P3-SH controller with completely renovated and improved code provides unprecedented performance and flexibility in robotics control. But you might not even notice. Because we have taken great care to ensure backward compatibility across *Activ*Media's entire history of robots, client software written to operate an ancient P2OS-based Pioneer AT will work with a brand new P3-AT with little or no modification. Client-server communication over a serial communication link remain identical as do support for all robotics commands. See *Chapter 6, ActivMedia's Robot Control Operations Software,* for details.

PORTS AND POWER

Your *Activ*Media robot has a variety of expansion power and I/O ports for attachment and close integration of a client PC, sensors, and a variety of accessories—all accessible through a common application interface to the robot's server software, ARCOS. Features include:

- ✓ 44.2368 MHz Renesas SH2 32-bit RISC microprocessor with 32K RAM and 128K FLASH
- ✓ 4 RS-232 serial ports (5 connectors) configurable from 9.6 to 115.2 kilobaud
- ✓ 4 Sonar arrays of up to 8 sonar each
- ✓ 2 8-bit bumpers/digital input connectors
- ✓ Gripper/User I/O port with 8-bits digital I/O, analog input, and 5/12 VDC power
- ✓ Heading correction gyro port
- ✓ Tilt/roll sensor port
- ✓ 2-axis, 2-button joystick port
- ✓ User Control Panel
- ✓ Controller HOST serial connector
- ✓ Main power and bi-color LED battery level indicators
- ✓ 2 AUX power switches (5 and 12 VDC) with related LED indicators
- ✓ RESET and MOTORS pushbutton controls
- ✓ Piezo buzzer
- ✓ Motor/Power Board (drive system) interface with PWM and motor-direction control lines and 8-bits of digital input
- ✓ I^2C interface with 4-line X 20-character LCD support

With the onboard PC option, your *Activ*Media robot becomes an autonomous agent. With Ethernet-ready onboard autonomy, your robot even becomes an agent for multi-intelligence work.

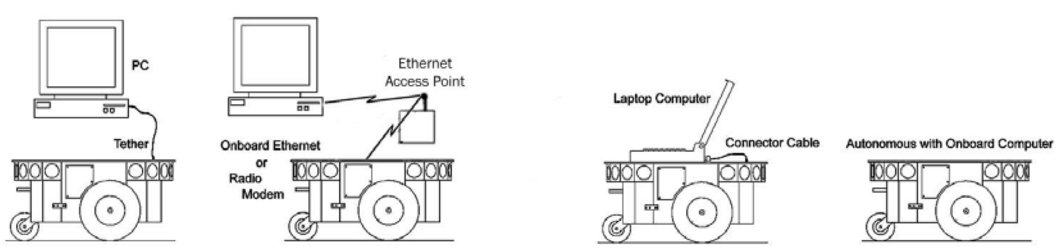

Figure 3. ActivMedia's robot servers require a PC to run client software for intelligent robotics command and control operations.

CLIENT SOFTWARE

All *Activ*Media robots operate as the server in a client-server environment: Their controllers handle the low-level details of mobile robotics, including maintaining the platform's drive speed and heading over uneven terrain, acquiring sensor readings, such as from the sonar, and managing attached accessories like the Gripper. To complete the client-server architecture, *Activ*Media robots require a PC connection: software running on a computer connected with the robot's controller via the HOST serial link and which provides the high-level, intelligent robot controls, including obstacle avoidance, path planning, features recognition, localization, navigation, and so on.

An important benefit of *Activ*Media Robotics' client-server architecture is that different robot servers can be run using the same high-level client. Several clients also may share responsibility for controlling a single mobile server, which permits experimentation in distributed communication, planning, and control.

ARIA

The *Activ*Media Robotics Interface for Applications (ARIA) software comes with every *Activ*Media robot. It is a C++-based open-source development environment that provides a robust client-side interface to a variety of intelligent robotics systems, including your *Activ*Media robot's controller and accessory systems.

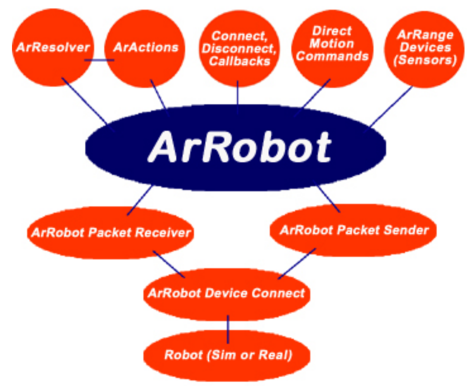

Figure 4. ARIA's architecture

ARIA is the ideal platform for integration of your own robot-control software, since it neatly handles the lowest-level details of client-server interactions, including serial communications, command and server-information packet processing, cycle timing, and multithreading, as well as a variety of accessory controls, such as for a scanning laser-range finder, motion gyros, among many others.

What's more, ARIA comes complete with source code so that you may examine the software and modify it for your own sensors and applications.

Mapping, Navigation, and Localization

*Activ*Media Robotics' also has a comprehensive suite of client tools and applications by which, with a laser range-finder enabled *Activ*Media robot, you automatically create, edit, and use maps and floorplans for advanced robotics applications, including localization and navigation. For much more information about ARNL, MobileEyes™, and our many commercial ventures, visit http://www.mobilerobots.com and contact *sales@mobilerobots.com*.

MODES OF OPERATION

You may operate your Pioneer 3 robot in one of four modes:

- ✓ Server
- ✓ Joydrive
- ✓ Maintenance
- ✓ Standalone

Server Mode

The new Renesas SH2 microprocessor-based controller comes with 128K of reprogrammable FLASH and 32K dynamic RAM memory. We don't recommend that you start learning SH2 programming. Rather, the robot comes to you installed with the latest ARCOS servers.

In conjunction with client software like ARIA running on an onboard or other user-supplied computer, ARCOS lets you take advantage of modern client-server and robot-control technologies to perform advanced robot tasks. Most users run their *Activ*Media robot in server mode, because it gives them quick, easy access to its robotics functionality while working with high-level software on a familiar host computer.

Maintenance and Standalone Modes

For experiments in controller-level operation of your robot's functions, you may reprogram the onboard FLASH for direct and standalone operation of your *Activ*Media robot. We supply the means to download and debug (ARSHstub embedded GDB interface), but not the controller's programming software, for you to work in standalone mode.

The utilities we provide for you to reprogram the SH2-based controller's FLASH also may be used to update and upgrade your robot's ARCOS. In a special Maintenance Mode, you also adjust your robot's operating parameters that ARCOS uses as default values on startup or reset. See Chapter 7, *Updating & Reconfiguring ARCOS,* for much more detail.

We typically provide the maintenance utilities and ARCOS upgrades free for download from our website, so be sure to sign up for the `pioneer-users` email newslist**.** That's where we notify our customers of the upgrades, as well as where we provide access to *Activ*Media robot users worldwide.

Joydrive Mode

Finally, we provide onboard software and controller hardware that let you drive the robot from a tethered joystick when not otherwise connected with a controlling client. See Chapter 4 for more details.

THE PIONEER LEGACY

Figure 5. The original Pioneer 1 and Pioneer AT

Commercially introduced in the summer of 1995, Pioneer 1 was the original platform. Intended mostly for indoor use on hard, flat surfaces, the robot had solid rubber tires and a two-wheel differential, reversible drive system with a rear caster for balance. It came with a single-board 68HC11-based robot controller and the Pioneer Server Operating System (PSOS) software. The Pioneer 1 also came standard with seven sonar range finders (two side-facing and five forward-facing) and integrated wheel encoders. Its low-cost and high-performance caused an explosion in the number of researchers and developers who now have access to a real, intelligent mobile robotic platform.

Software-wise, the Pioneer 1 initially served as a platform for SRI International's AI/fuzzy logic-based Saphira robotics applications development. But it wasn't long before Pioneer's open architecture became the popular platform for the development of a variety of alternative robotics software environments.

Pioneer AT

Functionally and programmatically identical to the Pioneer 1, the four-wheel drive, skid-steer Pioneer AT was introduced in the summer of 1997 for operation in uneven indoor and outdoor environments, including loose, rough terrain.

Except for the drive system, there were no operational differences between the Pioneer AT and the Pioneer 1: The integrated sonar arrays and controllers were the same; they shared accessories; and applications developed for the Pioneer 1 worked with little or no porting on the AT.

Pioneer 2™ and PeopleBot™

The next generation of Pioneer, including the Pioneer 2-DX, -CE, and -AT, introduced in fall of 1998 through summer of 1999, improved upon the Pioneer 1 legacy while retaining its many important advantages.[3] Indeed, in most respects, particularly with applications software, Pioneer 2 worked identically to Pioneer 1 models, but offered many more expansion options, including a client PC onboard the robot.

The *Activ*Media Robotics Pioneer 2 models -DX, -DE, -DXe, -DXf, and -AT, and the V1 and Performance PeopleBot robots used a 20 MHz Siemens 88C166-based controller, with independent motor-power and sonar controller boards for a versatile operating environment. Sporting a more holonomic body, larger wheels and stronger motors for better indoor performance, Pioneer 2-DX, -DXe, -DXf, and -CE models, like Pioneer 1, were two-wheel, differential-drive mobile robots.

Figure 6. The Performance PeopleBot sports an attractive body design and bundled systems, including voice synthesis and recognition for human-interaction research and applications.

The four-wheel drive Pioneer 2-AT had four independent motors and drivers. Unlike its Pioneer AT predecessor, the Pioneer 2-AT came with a stall-detection system and inflatable pneumatic tires with metal wheels for much more robust operation in rough terrain, as well as the ability to carry nearly 30 kilograms (66 lbs) of payload and climb a 60-percent grade.

Other Pioneer 2-like robots include the Performance PeopleBot robots, which were introduced in 2000. They are architecturally Pioneer 2 robots, but with stronger motors and integrated human-interaction features, including a pedestal extension, integrated voice and sound synthesis and recognition—ideal for human-interaction studies as well as for commercial and consumer mobile-robotics applications.

Pioneer 3™ and Recent Pioneer 2-DX8™, -AT8™, and *Plus*™ Mobile Robots

Two new models of Pioneer 2 appeared in the summer of 2002, two more at the beginning of 2003, and the Pioneer 3 debuted in the summer of 2003. All used a controller based on the Hitachi H8S microprocessor, with new control systems software

[3] Price/performance ratio included! The much more capable and expandable Pioneer 2 was introduced four years later for just a few hundred dollars (US) more than the original Pioneer 1.

(AROS) and I/O expansion capabilities. The Pioneer 3 and 2-*Plus* robots also had new, more powerful motor/power systems for better navigational control and payload.[4]

Pioneer 3™ SH Robots

Hardware-wise, the latest Pioneer, PeopleBot™, and PowerBot™ robots—all introduced in summer of 2004—are identical to their predecessors except for their revolutionary new Renesas SH2-based controller. Software-wise, these new robots are fully compatible with all other *Activ*Media robots, including Pioneer 1. The new *Activ*Media Robot Control & Operation Software (ARCOS) provides unprecedented performance and expansion, yet can interface and run client programs originally developed for Pioneer 1, 2, as well as 3 platforms. Of course, you will have to extend your old client software, as we have done with ARIA, in order to take full advantage of ARCOS.

To the relief of those who have invested years in developing software for Pioneer 1 and 2, Pioneer 3™ truly does combine the best of the new mobile robot technologies with *Activ*Media's tried-and-true robot architecture.

Figure 7. PowerBot™ carries over 100 kg of payload.

[4] The interim Pioneer 2-DXf had the same, more-powerful motors as the DX8s and AT8 *Plus*.

Chapter 3 Specifications & Controls

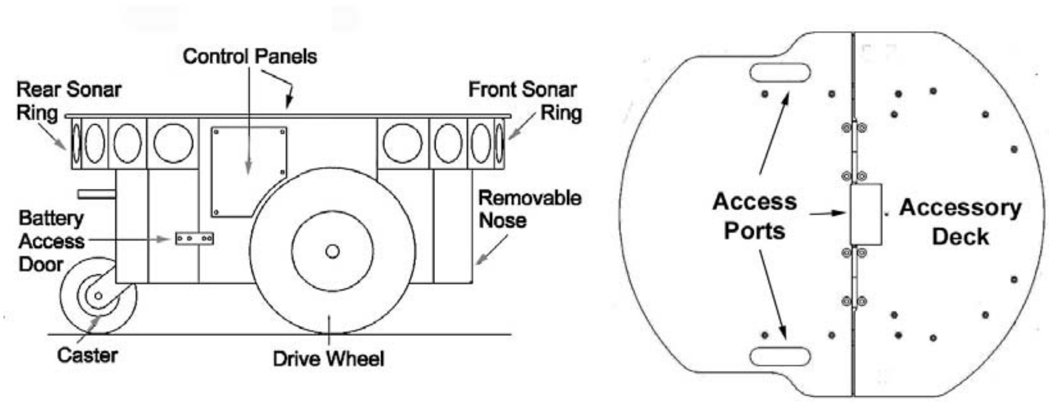

Figure 8. Pioneer 3-DX features

*Activ*Media's Pioneer robots may be smaller than most, but they pack an impressive array of intelligent mobile robot capabilities that rival bigger and much more expensive machines. The Pioneer 3-DX with onboard PC is a fully autonomous intelligent mobile robot. Unlike other commercially available robots, Pioneer's modest size lends itself very well to navigation in tight quarters and cluttered spaces, such as classrooms, laboratories, and small offices. With its powerful ARCOS server and *Activ*Media Robotics client software, the Pioneer 3 is fully capable of mapping its environment, finding its way home, and performing other sophisticated path-planning tasks.[5]

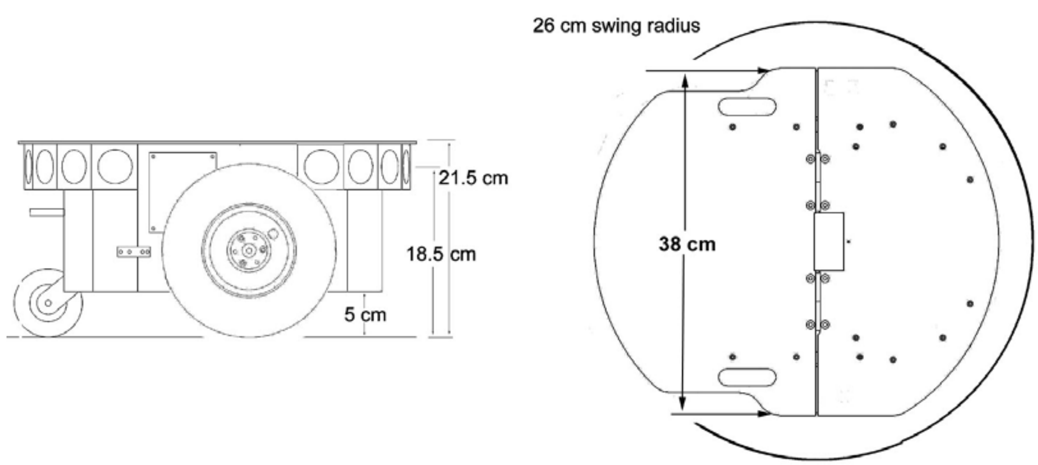

Figure 9. Pioneer 3-DX's physical dimensions and swing radius.

Physical Characteristics and Components

Weighing only 9 kg (20 pounds with one battery), the basic Pioneer 3-DX mobile robots are lightweight, but their strong aluminum body and solid construction make them virtually indestructible.

[5] Requires a laser range-finder accessory and special Navigation and Localization software, too.

These characteristics also permit them to carry extraordinary payloads: The Pioneer 3-DX can carry up to 23 Kg (50 lbs.) additional weight; the 3-AT can carry over 35 Kg (70 lbs.) more! Yet, Pioneer 3s are lightweight enough that it is also as easy to transport as a suitcase—a task made even easier by the DX's built-in handle.

*Activ*Media robots are composed of several main parts:

- ✓ Deck
- ✓ Motor Stop Button
- ✓ User Control Panel
- ✓ Body, Nose, and Accessory Panels
- ✓ Sonar Array(s)
- ✓ Motors, Wheels, and Encoders
- ✓ Batteries and Power

DECK

All Pioneer 3 models have hinged top-plates which give you much easier access to the internal components of the robot. See Chapter 8, *Calibration & Maintenance*, for access details.

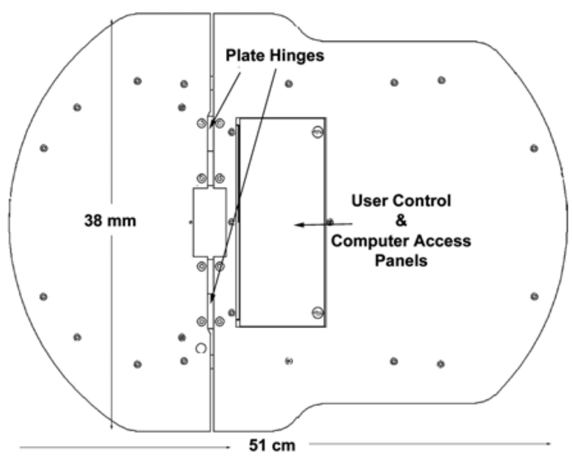

Figure 10. Pioneer 3-AT's console and hinged deck

The robot's deck is simply the flat surface for mounting projects and accessories, such as the PTZ Robotic Camera and a laser range finder. Feed-through slots on each side of the DX deck let you conveniently route cables to the accessory connectors on the side panels of the robot. A removable plug in the middle of the deck on all models gives you convenient access to the interior of the robot.

When mounting accessories, you should try to center the robot's payload over the drive wheels. If you must add a heavy accessory to the edge of the deck, counterbalance the weight with a heavy object on the opposite end. A full complement of batteries helps balance the robot, too.

MOTOR STOP BUTTON

All Pioneer 3-AT and, upon request, some Pioneer 3-DX robots have a `STOP` button at the rear of their deck. Press and release it to immediately disengage the robot's motor power. It will also cause a stall and can result in incessant beeping from the onboard piezo speaker (see *User Controls* below).

Press the `STOP` button in to re-engage motor power and stop that incessant beeping noise. Note that you may also have to re-enable the motors when connected with client software, either by manually pressing the `MOTORS` button on the User Control Panel, or through a special client command #4.

USER CONTROL PANEL

The User Control Panel is where you have access to the ARCOS-based onboard controller. Found inside the AT's hinged access panel on the deck or on the left sidepanel of the DX, it consists of control buttons and indicators and an RS-232 compatible serial port (9-pin DSUB connector).

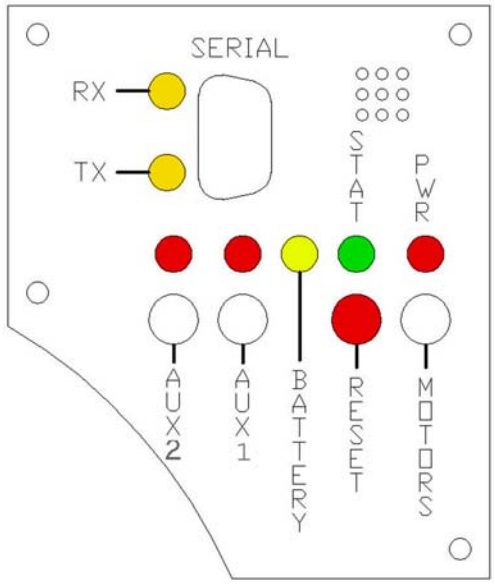

Figure 11. P3-DX User Control Panel

Power and Status Indicators

The red `PWR` LED is lit whenever main power is applied to the robot. The green `STAT` LED state depends on the operating mode and other conditions. It flashes slowly when the controller is awaiting a connection with a client and flashes quickly when in joydrive mode or when connected with a client and the motors are engaged. It also flashes moderately fast when the controller is in maintenance mode.

The `BATTERY` LED's apparent color depends on your robot's battery voltage: green when fully charged (>12.5 volts) through orange, and finally red when the voltage drops below 11.5. When in maintenance mode, the BATTERY LED glows bright red only, regardless of battery charge.

Buzzer

A built-in piezo buzzer (audible through the holes just above the `STAT` and `PWR` LEDs) provides audible clues to the robot's state, such as upon successful startup of the controller and a client connection. An ARCOS client command lets you program the buzzer, too, to play your own MIDI sounds.

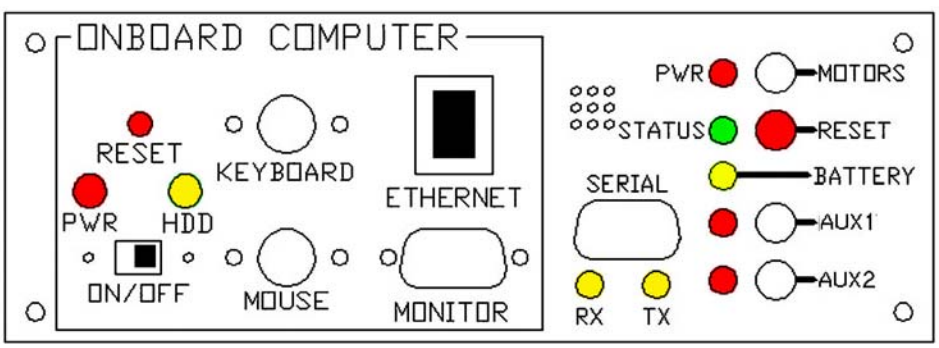

Figure 12. P3-AT computer and user controls

Serial Port

The `SERIAL` connector, with incoming and outgoing data indicator LEDs (`RX` and `TX`, respectively), is through where you may interact with the ARCOS controller from an offboard computer for tethered client-server control and for controller software maintenance. The port is shared internally by the `HOST` serial port, to which we connect the onboard computer or an Ethernet-to-serial device. Either the `SERIAL` or `HOST` connector may be used for client-server and maintenance mode communication with the controller.

To avoid communication conflicts, digital switching circuitry disables the internal `HOST` serial port if the attached serial device hasn't opened the port. However, serial port interference will be a problem if the `HOST` and User Control `SERIAL` ports are both occupied and engaged. Accordingly, remove the cable from the User Control `SERIAL` port if you plan to connect with the controller through the HOST port.

In particular, if you have a serial cable connected to the User Control Panel SERIAL port, with the attached PC has that serial port opened for communications, and you then reset or power up the robot and controller, ARCOS automatically goes into maintenance mode.

Power Switches

`The AUX1 and AUX2` switches on the User Control Panel are pushbuttons which engage or disengage power to 5 and 12 VDC connectors on the Motor-Power board to which we or you attach power for various accessories. For example, 12 VDC power for the PTZ camera typically gets switched via the AUX1 pushbutton. See *Appendix B* for power connections. Respective red LEDs indicate when power is ON.

Reset and Motors

The red `RESET` pushbutton acts to unconditionally reset the controller, disabling any active connections or attached devices, including the motors.

The white `MOTORS` pushbutton's actions depend on the state of the controller. When connected with a client, push it to enable and disable the motors manually, as its label implies.[6]

To manually engage ARCOS maintenance mode, press and hold the white `MOTORS` button, press and release the red `RESET` button, then release MOTORS. Note that while this manual operation was required to engage maintenance mode with previous robot controllers, it is no longer necessary with ARCOS.

BODY, NOSE, AND ACCESSORY PANELS

Your *Activ*Media robot's sturdy, but lightweight aluminum body houses the batteries, drive motors, electronics, and other common components, including the front and rear sonar arrays. The body also has sufficient room, with power and signal connectors, to support a variety of robotics accessories inside, including an A/V wireless surveillance system, radio Ethernet, onboard computer, laser range finder, and more.

On all models except those outfitted with the docking-charging system, a hinged rear door gives you easy access to the batteries, which you may quickly hot-swap to refresh any of up to three batteries.

Nose

The nose is where we put the onboard PC. The nose is readily removable for access: Simply remove two screws from underneath the front sonar array. A third screw holds the nose to the bottom of the AT's body. The DX nose is hinged at the bottom.

[6] A client command lets you engage/disengage the motors programmatically. See chapter 6.

Once the mounting screws are removed, simply pull the nose away from the body.[7] This provides a quick and easy way to get to the accessory boards and disk drive of the onboard PC, as well as to the sonar gain adjustment for the front sonar array. The nose also is an ideal place for you to attach your own custom accessories and sensors.

Access Panels

All DX's come with a removable right-side panel through which you may install accessory connectors and controls. A special sidepanel comes with the onboard PC option, for example, which provides connectors for a monitor, keyboard, mouse, and 10Base-T Ethernet, as well as the means to reset and switch power for the onboard computer.

AT's come with a single access panel in the deck. Fastened down with finger-tight screws, the User Control Panel and onboard computer controls are accessible beneath the hinged door.

All models come with an access port near the center of the deck through which to run cables to the internal components.

SONAR

Natively, ARCOS-based *Acti*Media robots support up to four sonar arrays, each with up to eight transducers that provide object detection and range information for collision avoidance, features recognition, localization, and navigation. The sonar positions in all Pioneer 3 sonar arrays are fixed: one on each side, and six facing outward at 20-degree intervals. Together, fore and aft sonar arrays provide 360 degrees of nearly seamless sensing for the platform.

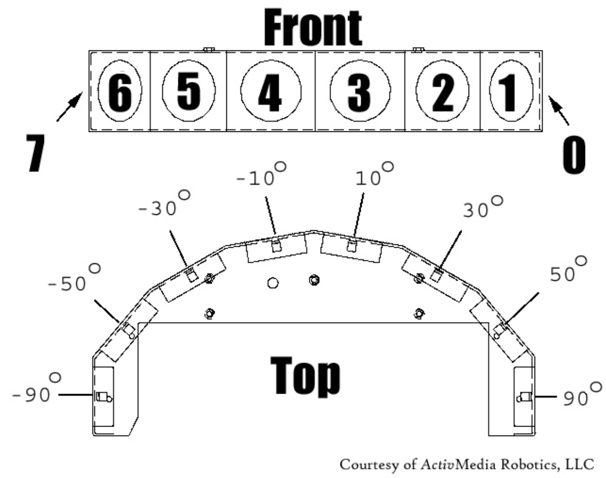

Figure 13. Pioneer 3 sonar array

Multiplexed Operation

Each sonar array's transducers are multiplexed: Only one disc per array is active at a time, but all four arrays fire one transducer simultaneously. The sonar ranging acquisition rate is adjustable, normally set to 25 Hz (40 milliseconds per transducer per array). Sensitivity ranges from 10 centimeters (six inches) to over four meters, depending on the ranging rate. You may control the sonar's firing pattern through software, too; the default is left-to-right in sequence 0 to 7 for each array. See the ARCOS Chapters 6 and 7 for details.

Sensitivity Adjustment

The driver electronics for each array is calibrated at the factory. However, you may adjust the array's sensitivity and range to accommodate differing operating environments. The sonar gain control is on the underside of the sonar driver board, which is attached to the floor of each sonar module.

[7] With older Pioneer 2 models, you also needed to remove the Gripper before removing the nose. With P3 models, the robot's nose and Gripper come off together, so you only need to remove the nose mounting screws.

Sonar sensitivity adjustment controls are accessible directly, although you may need to remove the Gripper to access the front sonar, if you have that accessory attached. For the front sonar, for instance, locate a hole near the front underside of the array through which you can see the cap of the sonar-gain adjustment potentiometer. Using a small flat-blade screwdriver, turn the gain control counterclockwise to make the sonar less sensitive to external noise and false echoes.

Low sonar-gain settings reduce the robot's ability to see small objects. Under some circumstances, that is desirable. For instance, attenuate the sonar if you are operating in a noisy environment or on uneven or highly reflective floor—a heavy shag carpet, for example. If the sonar are too sensitive, they will "see" the carpet immediately ahead of the robot as an obstacle.

Increase the sensitivity of the sonar by turning the gain-adjustment screw clockwise, making them more likely to see small objects or objects at a greater distance. For instance, increase the gain if you are operating in a relatively quiet and open environment with a smooth floor surface.

MOTORS, WHEELS, AND POSITION ENCODERS

Pioneer 3 drive systems use high-speed, high-torque, reversible-DC motors, each equipped with a high-resolution optical quadrature shaft encoder for precise position and speed sensing and advanced dead-reckoning. Motor gearhead ratios, encoder ticks-per-revolution, and tire sizes can vary by robot model. However, ARCOS corrects for tire mismatches and converts most client commands and reported server information from platform-independent distance and heading units into platform-dependent encoder ticks, as expressed in the `driftFactor`, `ticksMM`, and `revCount` FLASH parameters. Please read Chapter 6 for more details.

All Pioneer 3-DX robots come with foam-filled solid tires with knobby treads.[8] Pioneer 3-AT tires are pneumatic so that you may configure your robot for differing terrains. In any configuration, be careful to inflate the 3-AT tires evenly and adjust the respective `driftFactor`, `ticksMM`, and `revCount` FLASH parameters for proper operation. We ship 3-AT's with the tires inflated to 23 psi each.

BATTERIES AND POWER

Except when the DX is outfitted with the docking-charging system (see below), Pioneer 3 robots contain up to three, hot-swappable, seven ampere-hour, 12 volts direct-current (VDC) sealed lead/acid batteries (total of 252 watt-hours), accessible through a hinged and latched rear door. We provide a suction cup tool to help grab and slide each battery out of its bay. Spring contacts on the robot's battery power board alleviate the need for manually attaching and detaching power cables or connectors.

> Balance the batteries in your robot.

Battery life, of course, depends on the configuration of accessories and motor activity. AT charge life typically ranges from two to three hours. The DX runs continuously for six hours or more; up to four hours with onboard computer. If you don't use the motors, your robot's controller will run for several days on a single battery charge.

IMPORTANT: Batteries have a significant impact on the balance and operation of your robot. Under most conditions, we recommend operating with three batteries. Otherwise, a single battery should be mounted in the center, or two batteries inserted on each side of the battery container.

[8] A ribbed-tread tire is available optionally. Contact *Activ*Media sales for details.

Battery Indicators and Low Voltage Conditions

The User Control Panel has a bi-color LED labeled `BATTERY` that visually indicates current battery voltage. From approximately 12.5 volts and above, the LED glows bright green. The LED turns progressively orange and then red as the voltage drops to approximately 11.5 volts.

Aurally, the User Control Panel's buzzer, if active (see the ARCOS `SoundTog` client command and FLASH parameter), will sound a repetitive alarm if the battery voltage drops consistently below the FLASH `LowBattery` (11.5 VDC, by default) level. If the battery voltage drops below the FLASH `ShutdownVolts` (11 VDC, by default) the controller automatically shuts down a client connection and notifies the computer, via the `HOST RI` (ring indicator) pin, to shut down and thereby prevent data loss or systems corruption due to low batteries.

Recharging

Typical battery recharge time using the recommended accessory (800 mA) charger varies according to the discharge state; it is roughly equal to three hours per volt per battery. The Power Cube accessory allows simultaneous recharge of three swappable batteries outside the robot.

With the high-speed (4A maximum current) charger, recharge time is greatly reduced. It also supplies sufficient current to continuously operate the robot and onboard accessories, such as the onboard PC and radios. But with the higher-current charger, care must be taken to charge at least two batteries at once. A single battery may overcharge and thereby damage both itself and the robot.

The new automated docking/recharging system is the best option. Because its integrated charge-management system has sufficient power and actively adjusts to system loads, it can run your DX's onboard systems while properly and optimally recharging its batteries. And because the charging mechanism may be operated independently of your robot's systems power, you may start up and shut down your robot and its onboard systems without disturbing the battery charging cycle.

All our recommended chargers are specifically designed for safe lead-acid battery recharging. Indicators on the module's face show fast-charge mode (typically an orange LED) in which the discharged batteries are given the maximal current, and trickle mode (green LED indicator), which the batteries are given only enough current to remain at full charge.

SAFETY ARCOS WATCHDOGS

ARCOS contains a communications `WatchDog` that will halt the robot's motion if communications between a PC client and the robot server are disrupted for a set time interval. The robot will automatically resume activity, including motion, as soon as communications are restored.

ARCOS also contains a stall monitor. If the drive exerts a PWM pulse that equals or exceeds a configurable level (`StallVal`) and the wheels fail to turn, motor power is cut off for a configurable amount of time (`StallWait`). The server software also notifies the client which motor is stalled. When the `StallWait` time elapses, motor power automatically switches back on and motion continues under client control.

You may reconfigure the various FLASH-based parameter values to suit your application. See Chapter 7, *Updating & Reconfiguring ARCOS*, for details.

Chapter 4 Accessories

Pioneer 3 robots have many accessory options. For convenience, we include a description of the more common integrated accessories in this document. Please also refer to the detailed documents that come with the accessory.

JOYSTICK AND JOYDRIVE MODE

Although not all models come standard with an exposed joystick connector, your Pioneer 3 robot's controller has a joystick port and ARCOS contains a `joydrive` server for manual operation.[9]

Start driving your robot with a joystick any time it is not connected with a client software program. Simply plug it into the joystick port and press the "fire" button to engage the motors.

To drive your robot with a joystick while it is connected with an ARIA client (overrides client-based drive commands for manual operation while recording a map, for instance), you must have the client software send the ARCOS command #47 with an integer argument 1 to enable the ARCOS `joydrive` servers. Have your client send the ARCOS `joydrive` command #47 with an integer argument of 0 to disable the joystick drive override.

The joystick's fire button acts as the joydrive "deadman"—press it to start driving; release it to stop the robot's motors. The robot should drive forward and reverse, and turn left or right in response and at speeds relative to the joystick's position.

While driving forward, pull back on the joystick into full-reverse to decelerate faster than normal.

When not connected with a client control program, releasing the joystick fire button stops the robot. However when connected with a client, the client program resumes automatic operation of your robot's drive system. So, for example, your robot may speed up or slow down and turn, depending on the actions of your client program.

You may adjust the maximum translation and rotation speeds and even disable joydrive mode, through special ARCOS FLASH configuration parameters. See Chapter 7, *Updating & Reconfiguring ARCOS*, for details.

BUMPERS

Bump rings fore and aft provide contact sensing for when other sensing has failed to detect an obstacle. The accessory rings also are segmented for contact positioning.

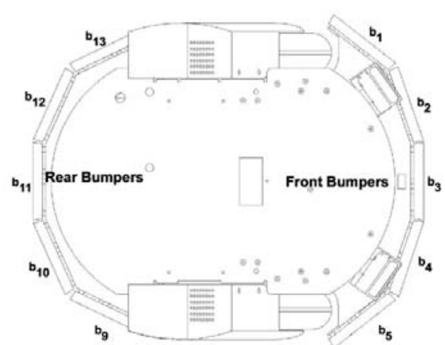

Figure 14. Pioneer 3 bumpers and associated STALL bits

Electronically and programmatically, the bumpers trigger digital events which are reflected in the `STALL` values of the standard server-information packet that ARCOS automatically sends to a connected client. Your client also may request a special `IOpac`

[9] A joystick adaptor kit for the DX is available for nominal fee through sales@activmedia.com. Also note that this port is different than the USB-based joystick port found on the back of the Laser bracket for the optional equipment and used to manually drive from ARIA-based clients.

server information packet that contains additional, more-detailed bumper, stall, IR, and other I/O related information.

Your robot won't move if you unplug one or both bumpers.

ARCOS itself also monitors and responds to protection triggers. For example, ARCOS' `bump_stall` server triggers a stall in the robot whenever one or more bumper segments get triggered while the robot is moving in the same direction (front forward or rear reverse). Please consult the *Appendix A* for interface details and the chapters on ARCOS, particularly the section which describes the contents of the `IOpac` server information packet, later in this manual for configuration and programming details.

Docking-Charging Accessory

The Pioneer 3-DX docking-charging accessory[10] is both a manual and an automated mechanism. Onboard controls, triggered either by the `DEPLOY CHARGER` button near the manual `CHARGE` port, or by ARCOS controller-mediated client commands, deploy actuated contacts on the bottom of the robot, which in turn seat onto the charging platform. Then, when activated by an IR-based, unique frequency-modulated signal from the robot, the charger platform delivers up to 17 VDC @ 11.5 A to its plates.

While connected, onboard circuitry conditions the power to optimally charge the three 7-Ahr, 12 VDC lead-acid batteries (6A charging current max) and provides sufficient power (up to 5.5 A) for operation of all onboard systems.

Manual Operation (Robot Power OFF)

With `MAIN POWER` off, place the robot over the charge platform so that its charging contacts are perpendicular to and, when deployed, contact the charger plates. Note that no charging power is applied to the plates on the platform; only low signal (5VDC @ <300mA) power for the IR detectors.

Press and hold the `DEPLOY CHARGER` button to manually deploy the charge mechanism on the bottom of the robot. Hold for a few seconds, but not more than 10 seconds. Charging is activated by positive contact with the charging platform. In that case, the charge lamp on the charger unit will light and the robot's contacts will remain deployed when you release the `DEPLOY CHARGER` button. Otherwise, the mechanism will retract. In that case, re-position the robot and try again.

The robot's charging mechanism automatically retracts if you press the `DEPLOY CHARGER` button while charging, if you move the robot on the docking platform and lose positive charging contact, or if you remove power from the charger unit. In all cases, charging power is removed immediately from the docking platform when not actively engaged by the robot.

Manual Operation (Robot Power and Systems ON)

Because the automated docking/charging system's charger and integrated circuitry actively adjusts to system loads, it can run your robot's onboard systems while properly and optimally recharging its batteries. And because the charging mechanism may be operated independently of your robot's systems power, you may start up and shut down

[10] The charging mechanism and onboard power conditioning circuitry can be retrofitted to all Pioneer 3 and some Pioneer 2 and Performance PeopleBot robots. All require return to the factory.

your robot and its onboard systems without disturbing the battery charging cycle, if engaged.

For example, with MAIN POWER on, use joystick mode to position the robot onto the charging platform. Then manually deploy the charging mechanism as described in the section above. Thereafter, switch MAIN POWER off, or conversely, start up and shut down other onboard systems, including the PC, camera, laser, and other accessories, to proceed with development work without disturbing battery recharging.

The same conditions apply to remove charging power and retract the robot's charging mechanism with the robot's MAIN POWER on as well as off. Since the ARCOS controller always is active while the robot's power is on, you also may connect and disconnect a client program, run in maintenance mode, or engage Joydrive mode. However, engaging the motors, such as when you press the "fire" button on the joystick, immediately and automatically disengages the charger and retracts the charging mechanism. And the charging mechanism will not activate manually via the DEPLOY CHARGER button until you disengage the motors.

RADIO CONTROLS AND ACCESSORIES

All *Activ*Media robots are servers in a client-server architecture. You supply the client computer to run your intelligent mobile-robot applications. The client can be either an onboard piggy-back laptop or embedded PC, or an off-board PC connected through radio modems or wireless serial Ethernet. In all cases, that client PC must connect to the internal HOST or User Control Panel SERIAL port in order for the robot and your software to work.

For the piggyback laptop or embedded PC, that serial connection is via a common "pass-through" serial cable. Radio modems may replace that serial cable with a wireless tether. Accordingly, if you have radio modems, one is inside your robot and connected to the controller's HOST serial port, and the other modem plugs into a serial port on some offboard computer where you run your client software.[11] Hence, in these configurations, there is one dedicated client computer.

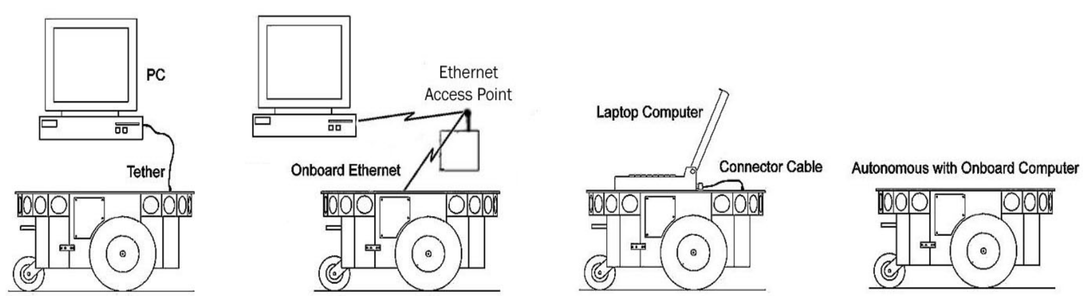

Figure 15. Client-server connection options.

Radio Ethernet is a little more complicated, but this preferred method because it lets you use many different computers on the network to become the robot's client. If you have a PC onboard (either integrated or piggyback), it can supply the radio Ethernet connection through a PCMCIA-based wireless Ethernet card.

We also sell a special wireless Ethernet-to-serial accessory which connects directly to your robot's controller. It works by automatically translating network-based Ethernet packet communications into streaming serial for the robot controller and back again.

[11] We no longer offer a radio modem accessory.

Running your robot through a wireless Ethernet to an onboard computer is different than with the Ethernet-to-serial device. In the first case, you run your robot client software on the onboard PC and use wireless Ethernet to monitor and control that PC's operation. In the latter case, you run the client software on a remote LAN-based PC.

Accordingly, a major disadvantage of the wireless Ethernet-to-serial device is that it requires a consistent wireless connection with the robot. Disruption of the radio signal—a common occurrence in even the most modern installations—leads to poor robot performance and very short ranges of operation.

This is why we recommend onboard client PCs for wider, much more robust areas of autonomous operation, particularly when equipped with their own wireless Ethernet. In this configuration, you run the client software and its interactions with the robot controller locally and simply rely on the wireless connection to export and operate the client controls. Moreover, the onboard PC is often needed for local processing, such as to support a laser range finder or to capture and process live video for vision work.

INTEGRATED PC

Mounted just behind the nose of the robot, the Pioneer 3 integrated PC is a common EBX form-factor board that comes with up to four serial ports, 10/100Base-T Ethernet, monitor, keyboard, and mouse ports, two USB ports, and support for floppy, as well as IDE hard-disk drives. For additional functionality, such as for sound, video framegrabbing, firewire or PCMCIA bus, and wireless Ethernet, the onboard PC accepts PC104 and PC104-plus (PCI bus-enabled) interface cards that stack on the motherboard.

Necessary 5 VDC power comes from a dedicated DC:DC converter, mounted nearby. A hard-disk drive is specially shock-mounted to the robot's nose, in between a cooling fan and computer speaker.

The onboard PC communicates with the robot's controller through its HOST serial port and the dedicated serial port `COM1` under Windows® or `/dev/ttyS0` on Linux systems. The controller automatically switches in that HOST-to-PC connection when PC-based client software opens the serial port. Otherwise, the PC doesn't interfere with externally connected clients through the shared `SERIAL` port on the User Control Panel.

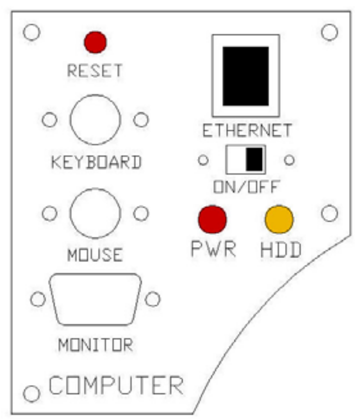

Figure 16. DX computer control side panel

Note also that some signals on the controller's HOST serial port as connected with the onboard PC or other accessory can be used for automated PC shutdown or other utilities: Pin 4 (DSR) is RS-232 high when the controller operates normally; otherwise it is low when reset or in maintenance mode. Similarly, pin 9 (RI) normally is low and goes RS-232 high when the robot's batteries drop below a set (FLASH `ShutdownVolts` parameter) voltage level.

Computer Control Panel

User-accessible communication and control port connectors, switches, and indicators for the onboard PC are on the Computer Control Panel, found on the right side panel of the DX or in the hinged control well next to the user controls of the AT.

The controls and ports use common connectors: standard monitor DSUB and PS/2 connectors on the mouse and keyboard. The Ethernet is a 10/100Base-T standard RJ-45 socket.

The ON/OFF slide switch directly controls power to the onboard PC—through Main Power, unlike some earlier versions of the onboard system which included a delayed power shutdown. The PWR LED lights when the computer has power.

The HDD LED lights when the onboard hard-disk drive is active. The RESET button restarts the PC.

Operating the Onboard PC

This is a brief overview of operating the onboard PC. Please consult the *Computer Systems Documentation* and the OS manufacturer's documentation for more detail. *Activ*Media Robotics' software runs on either Microsoft Windows® (currently Windows 2000®) or RedHat® or Debian Linux. Accordingly, we prefer (the latter, in particular) and support those OSes on the onboard PC.

When we perform the installation and configuration, we install our robotics and accessory software typically in `/usr/local` on Linux systems, or in `C:\Program Files\ActivMedia Robotics` under Windows®. Of course, we install the appropriate drivers for the various accessory expansion cards, such as for a framegrabber or sound card. Please consult the respective *Activ*Media Robotics application software manuals, such as the *Activ*Media Color Tracking System (ACTS) for the video framegrabber or Festival for the sound card.

The first time you access the onboard PC, we recommend that you put the robot up on blocks so that it cannot inadvertently move and wreak havoc with external connections. Then attach a keyboard, monitor, and mouse to their respective sockets on the Computer Control Panel. Switch Main Power and then the computer power switch on.

After boot up, log in to the system. We've already created two users: one with common systems and file read/write permissions ('guest') and one with full-access to the PC software and OS—root (Linux) or 'administrator' (Windows®). If there is a password (usually not), it's 'activmedia'. When connected directly, we recommend you log in with full-access capabilities so that you can do systems set up and maintenance, such as change passwords, add users, and set up the network. Do note that with Linux systems, you cannot log in remotely over the network as root; you must log in as a common user and use the 'su -' command thereafter to attain superuser ('root' login) status.

Once logged into a Windows® system, it's simply a matter of clicking the mouse to select programs and applications. With Linux, use the 'startx' command to enable the X-Windows desktop and GUI environment. You might perform some of the *QuickStart* activities this way, although motion is impractical because of the monitor, mouse, and keyboard tethers. You may remove these while the system is active at your own risk.

Rather, we suggest that you run the *QuickStart* activities from an offboard computer first (onboard PC off), and then tackle the networking issues to establish a remote, preferably wireless connection with your robot.

PC Networking

The RJ-45 connector on the Computer Control Panel provides wired 10/100Base-T Ethernet networking directly with the onboard PC. With the purchased option, we also install a PCMCIA adaptor card on the PC's accessory stack and insert a wireless Ethernet card in one of its slots. The wireless Ethernet antenna sits atop the robot's deck.

To complete the wireless installation, you will need to provide an Access Point module (comes as an accessory with most units). Attach the Access Point to one of your LAN hubs or switches. No special configuration is required. We use the default operating mode: `'managed'` client-server.

We ship installed PC systems' preset and tested at a fixed IP address with Class-C network configuration. We allocate the same IP to both the wired and wireless Ethernet ports, typically `192.168.1.32`. Although you need not fuss with drivers or low-level device settings, before you may establish a network connection with the onboard PC (*not* the robot's controller!), even if just through a "cross-over" Ethernet cable to another PC, you'll need to reconfigure the robot's PC network settings. Please consult with your network systems administrator for networking details.

Briefly, with Windows®, go to the Control Panel's `Network and Dial-up Connections` wizard and choose the networking device's `Properties` to change the IP address and other details. Under Linux, there are similar, GUI-based tools under X-Windows to help you set up the network, such as `netcfg`, but we prefer to edit (*emacs* or *vi*) the salient network settings in `/etc/sysconfig/network` and in the specific device configuration files found in `/etc/sysconfig/network-scripts/`, such as `ifcfg-eth0` (wired Ethernet) and `ifcfg-eth1` or `ifcfg-wvlan0` (wireless).

From Windows®, use the Control Panel `Network and Dial-up Connections` tool to enable or disable a particular device. From Linux, use `ifup` and `ifdown` to enable or disable an Ethernet device. For example, as superuser, type `'ifdown eth0; ifup eth1'` to switch from a tethered to a wireless Ethernet connection.

For remote connections over Ethernet to your onboard PC, simply use *telnet* or the more secure *ssh* to log in to your Linux system. Allow X-windows server connections at your remote PC (`xhost`) if you plan to export the X-Windows display from the robot PC for remote GUI-based controls (`export DISPLAY=remote's hostname or IP:0`, for example).

With Windows®, you will need a special remote-control application to establish a GUI-based connection from a remote computer to the onboard PC over the network; *VNCserver*, for example, or *XWin32*.

Please note that, with the onboard PC and wireless Ethernet, as opposed to the wireless Ethernet-to-serial device, you may ***not*** connect with the robot's controller directly over the network: That is, you cannot run a client application like the ARIA `demo` on a remote PC and choose to directly connect with the robot server by selecting the robot PC's IP address. Rather, either run the client application on the onboard PC and export the display and controls over the network to the remote PC (preferred), or use the ARIA-based `IPTHRU` programs (see program sources in `Aria/examples`) to negotiate the IP-to-serial conversions needed by the client-server connection.

UPS and Genpowerd

To protect your robot's onboard PC data, we've enabled a detection scheme in ARCOS and UPS-like software on the PC that invoke shutdown of the operating system in the event of a persistent low-battery condition.[12]

ARCOS raises the HOST serial port's DSR pin 6 to RS-232 high and puts the RI pin 9 to low when the controller is operating normally and your robot's battery power is above the FLASH-parameter set `ShutdownVolts` value, which default is 11 VDC.[13] The RI pin goes high when power drops consistently below `ShutdownVolts`. *Genpowerd* running on the onboard Linux system or *ups.exe* running under Windows®, detects the change of state

---

[12] The original Pioneer 2 Motor-Power boards implemented a similar strategy in hardware.
[13] RI and DSR on the HOST serial port are RS-232 low during reset or when the controller is in Maintenance Mode.

and initiates OS shutdown after a short wait, during which the shutdown may be canceled by raising the battery voltage, such as by attaching a charger.

Genpowerd monitors the HOST serial RI port on `/dev/ttyS0`. Windows®' *ups.exe* requires a dedicated serial port—`COM2` on current systems, and prefers to monitor the CTS line. Consequently, we wire the onboard PC serial connector differently for Linux versus Windows® PCs. Please consult the ARCOS Chapters 6 and 7 for more details.

Chapter 5 Quick Start

This chapter describes how to quickly set up and operate your new *Activ*Media robot with the ARIA demonstration software.

PREPARATIVE ASSEMBLY

Your *Activ*Media robot comes fully assembled and ready for out-of-the-box operation. However, you may need to attach some accessories that were shipped separately for safety. The procedures we describe herein are for control of the basic robot.

If you have the onboard PC option, we recommend that you leave it off and perform the following tests first with a laptop or desktop computer tethered to the `SERIAL` port on the User Control Panel, then attack the many networking issues before you establish a remote-control connection with the onboard PC.

Install ARIA

The ARIA client software-development environment, including the ARIA demonstration program, come on CD-ROM with your new robot. They also come installed in your robot's onboard PC, if you purchased this option.

*Activ*Media Robotics customers also may obtain ARIA and related software and updates from our support website:

`http://robots.activmedia.com`

When installed, ARIA typically requires 10 or more megabytes of hard-disk space.

The Windows® version of ARIA is a self-extracting archive. Simply double-click its `.exe` icon and follow the extraction program's instructions. Normally, ARIA is put into a directory named `C:\Program Files\ActivMedia Robotics\ARIA`. The demonstration program gets put into the `bin\` subdirectory. For convenience, you also may access it from the `Start Menu's Programs` option. The demonstration program's source code and .NET® project and workspace files are in the `examples\` subdirectory.

Linux users must have `superuser` (`root`) permissions in order to install ARIA. It comes as an RPM installation archive:

```
rpm -ihv aria...
```

and gets installed in `/usr/local/Aria`. The ARIA demonstration program gets put into the `bin/` subdirectory. The demonstration sources and Makefile are in the `examples/` subdirectory.

Linux users should also be sure they have permission to read/write through their PC's serial port that connects with the robot. The default is `/dev/ttyS0`. ARIA is a terminal application that does not include a GUI, so its programs do not require X-Windows.

CAREFUL
Slide the batteries into the robot TERMINALS LAST.
Otherwise, you will damage the robot.

Install Batteries

Out of the box, your *Activ*Media robot comes with its batteries fully charged, although shipped separately, unless you have the special docking/charging system. Slide at least one and up to three batteries into the robot's battery box through the back door. Balance them: one in the center; if two, then one on each side.

Client-Server Communications

Your robot requires a serial communication link with a client PC for operation. The serial link may be:

- ✓ A tether cable from the robot's 9-pin serial connector on the User Control Panel to a computer
- ✓ A piggyback laptop cabled to the User Control Panel serial port
- ✓ Serial Ethernet
- ✓ Radio Modem
- ✓ An integrated onboard PC wired internally for direct onboard control

STARTING UP CLIENT AND SERVER

ARIA's examples are text-based "terminal" applications that do not include a GUI, so its programs do not require X-Windows over Linux or special software on a remote PC client—a simple `telnet` session will do the trick.

First, please note well that you cannot connect with and control your *Activ*Media robot through its controller directly from a remote client over the network without special hardware (wireless Ethernet-to-serial device) or, alternatively, special software that runs on the onboard computer and converts IP packets into serial data.[14] Otherwise, you must run the client software on the robot's PC or on a PC that is connected to the robot's controller `HOST` or User Control Panel `SERIAL` port. You may, of course, export the controls and display of your onboard PC over the network from X-windows or with special Windows software, such as VNCserver.

If you are using a wireless Ethernet-to-serial device to communicate with the robot's controller from a desktop PC, now is a good time to power up the unit. The `AUX1` power switch for the integrated radio is on the User Control Panel. You might test your connection, too—either peer-to-peer or through an access point on your LAN—from your PC to the radio Ethernet installed in your robot with the common `ping` program.

Windows® users may select the ARIA `demo` from the `Start` menu, in the `ActivMedia Robotics` program group. Otherwise, start if from the ARIA `bin\` directory.

Linux users will find the compiled `demo` in `/usr/local/Aria/bin/` or in `examples/`. Start it:

```
% ./demo
```

Demo Startup Options

By default, the ARIA `demo` program connects with the robot through the serial port `COM1` under Windows® or `/dev/ttyS0` under Linux. And, by default, `demo` connects with an attached laser rangefinder accessory through serial port `COM3` or `/dev/ttyS2`. To change those connection options, either modify the ARIA source code (`examples/demo.cpp` and related files in `src/`) and recompile the application, or use a startup argument on the command line (Table 1).

---

[14] Look in the ARIA/examples directory for a program called `ipthru`. It converts IP to serial and back again for remote-control clients connected through the onboard PC.

For example, from the Windows Start:Run dialog, choose Browse… and select the ARIA demo program: C:\Program Files\ActivMedia Robotics\ARIA\bin\demo.exe. Then, type a command line argument at the end of the text in the Run dialog. To connect through the Ethernet-to-serial radio device over the wireless network, for example, try the command:

C:\Program Files\ActivMedia Robotics\ARIA\bin\demo.exe --remoteHost 192.168.1.32

Table 1. ARIA demo command line arguments

--remoteHost <Host Name or IP>	Connect with robot through a remote host over the network instead of a serial port; requires special serial Ethernet hardware or IPTHRU software mediation.
--robotPort <Serial Port>	Connect with robot through specified serial port name; COM3, for example.
--remoteRobotTcpPort <Number>	Remote TCP host-to-robot connection port number; default is 8080.
--laserPort <Serial Port>	Connect with laser rangefinder through the specified serial port name; /dev/ttyS3, for example.
--remoteLaserTcpPort <Number>	Remote TCP host-to-laser connection port number; default is 8081.

A Successful Connection

ARIA prints out lots of diagnostic text as it negotiates a connection with the robot. If successful, the client requests various ARCOS servers to start their activities, including sonar polling, position integration, and so on. The controller sounds an audible connection cue, and you should hear the robot's sonar ping with a distinctive and repetitive clicking. In addition, the motors-associated STATUS LED on the User Control Panel should flash very fast (was flashing slowly while awaiting connection). Note that the ARIA demo automatically engages your robot's motors though a special client command. Normally, the motors are disengaged when first connecting.

Table 2. Keyboard teleoperation

KEY	ACTION
↑	forward
↓	reverse
←	turn left
→	turn right
space	all stop

The amber SERIAL port indicator LEDs on the robot's User Control Panel should blink to indicate ARIA-client to ARCOS-server communications, too.

OPERATING THE ARIA DEMONSTRATION CLIENT

When connected with the ARIA demo client, your robot becomes responsive and intelligent. For example, it moves cautiously. Although it may drive toward an obstacle, your *Activ*Media robot will not crash because the ARIA demo includes obstacle-avoidance behaviors which enable the robot to detect and actively avoid collisions.

The ARIA demo displays a menu of robot operation options. The default mode of operation is teleop. In teleop mode, you drive the robot manually, using the arrow keys on your keyboard or a joystick connected to the client PC's joystick port (as opposed to a joystick port on the robot).

While driving from the keyboard, hold down the respective keys to simultaneously drive the robot forward or backward and turn right or left. For instance, hold down the up-arrow key to have the robot accelerate forward to its cruising speed of around 400 millimeters per second (defined in the source code). Release the arrow key to have the robot slow down and stop. Press and hold the right- or left-arrow key to have the robot rotate or turn in an arc if you also hold down the up- or down-arrow key.

The other modes of ARIA demo operation give you access to your robot's various sensors and accessories, including encoders, sonar, laser, Gripper, a pan-tilt-zoom robotic camera, I/O port states, bumpers, and more. Accordingly, use the ARIA demo not only as a demonstration tool, but as a diagnostic one, as well, if you suspect a sensor or effector has failed or is working poorly. The demo also is useful for calibrating your robot's drive system.

Table 2. ARIA demo operation modes

MODE	HOT KEY	DESCRIPTION
`laser`	l	Displays the closest and furthest readings from the robot's laser range finder
`io`	i	Displays the state of the robot's digital and analog-to-digital I/O ports
`position`	p	Displays the coordinates of the robot's position relative to its starting location
`bumps`	b	Displays the status of the robot's bumpers
`sonar`	s	Displays the robot's sonar readings
`camera`	c	Controls and exercises the robot's pan-tilt-zoom robotic camera
`gripper`	g	Controls, exercises, and displays status of the robot's Gripper
`wander`	w	Sends the robot to move around at its own whim, while avoiding obstacles
`teleop`	t	Drive and steer the robot via the keyboard or a joystick; avoids collisions
`unguarded`	u	Same as teleop, except no collision avoidance
`direct`	d	ARCOS-direct command mode

Access each ARIA demo mode by pressing its related hot-key: '`t`', for instance, to select `teleoperation`. Each mode includes onscreen instructions and may have sub-menus for operating of the respective device.

DISCONNECTING

When you finish, press the `Esc` key to disconnect the ARIA client from your robot server and exit the ARIA demonstration program. Your ActivMedia robot should disengage its drive motors and stop moving, and its sonar should stop firing. You may now slide the robot's `Main Power` switch to `OFF`.

QUICKSTART TROUBLESHOOTING

Most problems occur when attempting to connect the ARIA client with a robot for the first time. The process can be daunting if you don't make the right connections and installations.

Proper Connections

Make sure you have ARIA properly installed and that your robot and connections are correct. A common mistake with Linux is not having the proper permissions on the connecting serial port.

Make sure your robot's batteries are fully charged (battery LED green). The robot servers shut down and won't allow a connection at under `ShutdownVolts`.

ATTENTION!

The ARIA-to-robot connection is ***SERIAL*** *only*. Accordingly, run the ARIA demo client with the onboard or piggyback computer, over radio modems, or over the network with the radio Ethernet-to-serial device.

If you are using the onboard PC or radios, the serial connection is internal and established at the factory; you should not have problems with those cables. Simply make sure the `AUX1` switch on the User Control Panel is engaged (associated LED lit), for example. And remove any serial cable that is plugged into the User Control Panel as it may interfere with internal serial communication.

With other serial connections, make sure to use the proper cable: a "pass-through" one, minimally connecting pins 2, 3, and 5 of your PC's serial port to their respective contacts of the robot's serial port on the User Control Panel.

If you access the wrong serial port, the ARIA demonstration program will display an error message. If the robot server isn't listening or if the serial link is severed somewhere between the client and server (cable loose or the radio is off, for instance), the client will attempt "Syncing 0" several times and fail. In that case, `RESET` the robot and check your serial connections.

If for some reason communications get severed between the ARIA client and ARCOS server, but both the client and server remain active, you may revive the connection with little effort: If you are using wireless communications, first check and see if the robot is out of range.

Communications also will fail if the client and/or server is somehow disabled during a session. For instance, if you inadvertently switch off the robot's `Main Power` or press the `RESET` button, you must restart the connection. Turning the `Main Power` switch `OFF` and then back `ON`, or pressing the `RESET` button puts the robot servers back to their wait state, ready to accept client connections again. If the ARIA demo or other client application is still active, simply press `esc` and restart.

Chapter 6 ARCOS

All *Activ*Media robots use a client-server mobile robot-control architecture. In the model, the robot's controller servers work to manage all the low-level details of the mobile robot's systems. These include operating the motors, firing the sonar, collecting and reporting sonar and wheel encoder data, and so on—all on command from and reporting to a separate client application, such as the ARIA demo.

With this client/server architecture, robotics applications developers do not need to know many details about a particular robot server, because the client insulates them from this lowest level of control. Some of you, however, may want to write your own robotics control and reactive planning programs, or just would like to have a closer programming relationship with your robot. This chapter explains how to communicate with and control your *Activ*Media robot via the *Activ*Media Robot Control and Operations Software (ARCOS) client-server interface. The same ARCOS functions and commands are supported in the various client-programming environments that accompany your robot or are available for separate license.

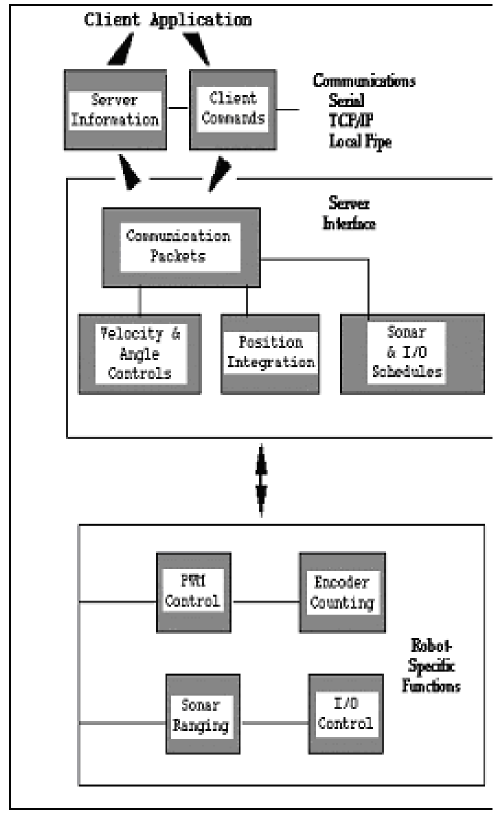

Figure 17. ActivMedia Robotics' client-server control architecture

Experienced *Activ*Media robot users can be assured that ARCOS is upwardly compatible with all *Activ*Media robots, implementing the same commands and information packets that first appeared in the Pioneer 1-based PSOS, in the original Pioneer 2-based P2OS, and more recent AROS-based Pioneer 2s and 3s, as well as PeopleBot and PowerBot. ARCOS, of course, extends the servers to add new functionality, improve performance, and provide additional information about the robot's state and sensing.

CLIENT-SERVER COMMUNICATION PACKET PROTOCOLS

*Activ*Media robots communicate with a client using special client-server communication packet protocols, one for command packets from client to server and another for Server Information Packets (SIPs) from the server to client. Both protocols are bit streams consisting of five main elements: a two-byte header, a one-byte count of the number of subsequent packet bytes, the client command or SIP packet type, command data types and argument values or SIP data bytes, and, finally, a two-byte checksum. Packets are limited to a maximum of 207 bytes each.

The two-byte header which signals the start of a packet is the same for both client-command packets and SIPs: 0xFA (250) followed by 0xFB (251). The subsequent count byte is the number of all *subsequent* bytes in the packet including the checksum, but not including the byte count value itself or the header bytes.

Data types are simple and depend on the element (see descriptions below): client commands, SIP types, and so on, are single 8-bit bytes, for example. Command arguments and SIP values may be 2-byte integers, ordered as least-significant byte first. Some data are strings of up to a maximum 200 bytes, prefaced by a length byte. Unlike common data integers, the two-byte checksum appears with its *most-significant* byte first.

Table 3. Client command packet protocol

COMPONENT	BYTES	VALUE	DESCRIPTION
HEADER	2	0xFA, 0xFB	Packet header; same for client and server
BYTE COUNT	1	N	Number of command/argument bytes plus Checksum's two bytes, but not including Byte Count itself or the header bytes. Maximum of 249.
COMMAND NUMBER	1	0 - 255	Client command number; see Table 7.
ARGUMENT TYPE	1	0x3B or 0x1B or 0x2B	Required data type of command argument: positive integer, negative or absolute integer, or string
ARGUMENT	n	data	Command argument; always 2-byte integer or string containing length prefix
CHECKSUM	2	computed	Packet integrity checksum

Packet Checksum

Calculate the client-server packet checksum by successively adding data byte pairs (high byte first) to a running checksum (initially zero), disregarding sign and overflow. If there are an odd number of data bytes, the last byte is XORed to the low-order byte of the checksum.

```
AREXPORT ArTypes::Byte2 ArRobotPacket::calcCheckSum(void)
{
  int i;
  unsigned char n;
  int c = 0;

  i = 3;
  n = myBuf[2] - 2;
  while (n > 1) {
    c += ((unsigned char)myBuf[i]<<8) | (unsigned char)myBuf[i+1];
    c = c & 0xffff;
    n -= 2;
    i += 2;
  }
  if (n > 0)
    c = c ^ (int)((unsigned char) myBuf[i]);
  return c;
}
```
(from *Activ*Media Robotics' ARIA ArRobotPacket.cpp)

NOTE: The checksum integer is placed at the end of the packet, with its bytes in the reverse order of that used for data; that is, b_0 is the high byte and b_1 is the low byte.

Packet Errors

ARCOS ignores a client command packet whose byte count exceeds 204 (total packet size of 207 bytes) or has an erroneous checksum. The client should similarly ignore erroneous SIPs.

Because of the real-time nature of client-server mobile-robotics interactions, we made a conscious decision to provide an unacknowledged communication packet interface. Retransmitting server information or command packets typically serves no useful purpose because old data is useless in maintaining responsive robot behaviors.

Nonetheless, the client-server interface provides a simple means for dealing with ignored command packets: Most of the client commands alter state variables in the server. By examining those values in respective SIPs, client software may detect ignored commands and re-issue them until achieving the correct state.

THE CLIENT-SERVER CONNECTION

Before exerting any control, a client application must first establish a connection with the robot server via a serial link through the robot controller's `HOST` serial port either via the internal HOST or the User Control Panel SERIAL connector. After establishing the communication link, the client then sends commands to and receives operating information from the server.

When first started or reset, ARCOS is in a special wait state, listening for communication packets to establish a client-server connection.[15] To establish a connection, the client application must send a series of three synchronization packets containing the `SYNC0`, `SYNC1`, and `SYNC2` commands in succession, and retrieve the server responses.

Specifically, and as examples of the client command protocol described below, the sequence of synchronization bytes is:

```
SYNC0:  250, 251, 3, 0, 0, 0
SYNC1:  250, 251, 3, 1, 0, 1
SYNC2:  250, 251, 3, 2, 0, 2
```

When in wait mode, ARCOS echoes the packets verbatim back to the client. The client should listen for the returned packets and only issue the next synchronization packet after it has received the appropriate echo.

Autoconfiguration (SYNC2)

ARCOS automatically sends robot identifying information back to the client following the last synchronization packet (`SYNC2`). The configuration values are three NULL-terminated strings that comprise the robot's FLASH-stored `name`, `class`, and `subclass`. You may uniquely `name` your *Activ*Media robot with the FLASH configuration tool we provide. The `class` and `subclass` are constants normally set at the factory and not changed thereafter. (See next chapter for details.)

The `class` string typically is `Pioneer`. The `subclass` depends on your robot model; P3`DX-SH` or P3`AT-SH`, for example. Clients may use these identifying strings to self-configure their own operating parameters. ARIA, for instance, loads and uses the robot's related parameter files found in the special `Aria/params` directory.

Opening the Servers—OPEN

Once you've established a connection with ARCOS, your client should send the `OPEN` command #1 (250, 251, 3, 1, 0, 1) to the server, which causes the robot controller to perform a few housekeeping functions, start its various servers, such as for the sonar and motor controllers, and begin transmitting server information to the client.

Note that when at first connected, your robot's motors are disabled, regardless of their state when last connected. To enable the motors after starting a connection, you must

[15] There also is maintenance mode for ARCOS downloads and parameter updates; see next chapter for details.

either do it manually (press the white MOTORS button on the User Control Panel) or have your client send an `ENABLE` client command #4 with an integer argument of 1. See Client Commands below).

Keeping the Beat—PULSE

An ARCOS safety watchdog expects that, once connected, your robot receives at least one client command packet from the attached PC every `watchdog` seconds, as defined in your robot's FLASH (default is two seconds). Otherwise, it assumes the client-server connection is broken and stops the robot.

Some clients—ARIA-based ones, for instance—use the good practice of sending a `PULSE` client command #0 (250, 251, 3, 0, 0, 0) just after `OPEN`. And if your client application will be otherwise distracted for some time, periodically issue the `PULSE` command to let your robot server know that your client is indeed alive and well. It has no other effect.

If the robot shuts down due to lack of communication with the client, it will revive upon receipt of a client command and automatically accelerate to the last-specified speed and heading motion setpoints.

Closing the Connection—CLOSE

To close the client-server connection, which automatically disables the motors and other server functions like sonar, simply issue the client `CLOSE` command #2 (250, 251, 3, 2, 0, 2).

> Once connected, send the `ENABLE` command
> or press the white `MOTORS` button on the User Control Panel
> to enable your robot's motors.

Most of the controller's operating parameters return to their FLASH-based default values upon disconnection with the client.[16]

SERVER INFORMATION PACKETS

Once `OPEN`ed, ARCOS automatically and repeatedly sends a packet of information over the `HOST` serial port back to the connected client. The standard ARCOS SIP informs the client about a number of operating states and readings, using the order and data types described in the nearby Table.

ARCOS also supports several additional SIP types. See following sections for details.

CLIENT COMMANDS

ARCOS has a structured command format for receiving and responding to directions from a client for control and operation of your *Activ*Media robot. Client commands are comprised of a one-byte command number optionally followed, if required by the command, by a one-byte description of the argument type and then the argument value.

---

[16] With earlier OSes, the changes persisted between sessions and reverted to the FLASH defaults only after the controller was reset.

Table 4. ARCOS standard SIP contents

NAME	VALUE	DESCRIPTION
HEADER	2 bytes	Exactly in order 0xFA (250), 0xFB (251)
BYTE COUNT	byte	Number of data bytes + 2 (checksum), not including header or byte-count bytes
TYPE	0x3s	Motors status; s = 2 when motors stopped or 3 when robot moving.
XPOS	int	Wheel-encoder integrated coordinates in millimeters (*DistConvFactor*‡ = 1.0).
YPOS	int	
THPOS	int	Orientation in degrees (*AngleConvFactor*‡ = 1.0).
L VEL	int	Wheel velocities in millimeters per second (*VelConvFactor*‡ = 1.0)
R VEL	int	
BATTERY	byte	Battery charge in tenths of volts (101 = 10.1 volts, for example)
STALL AND BUMPERS	uintǂ	Motor stall and bumper indicators. Bit 0 is the left wheel stall indicator, set to 1 if stalled. Bits 1-7 correspond to the first bumper I/O digital input states (accessory dependent). Bit 8 is the right wheel stall, and bits 9-15 correspond the second bumper I/O states, also accessory and application dependent.
CONTROL	int	Setpoint of the server's angular position servo in degrees
FLAGS	uint	Bit 0 motors status; bits 1-4 sonar array status; bits 5,6 STOP; bits 7,8 ledge-sense IRs; bit 9 joystick fire button; bit 10 auto—charger power-good.
COMPASS	byte	Electronic compass accessory heading in 2-degree units
SONAR COUNT	byte	Number of new sonar readings included in SIP
NUMBER	byte	If Sonar Count>0, is sonar disc number 0-31; readings follow
RANGE	uint	Corrected sonar range value in millimeters (RangeConvFactor‡ = 1.0)
…REST OF THE SONAR READINGS…		
GRIP_STATE	byte	Gripper state byte.
ANPORT	byte	Selected analog port number 1-5
ANALOG	byte	User Analog input (0-255=0-5 VDC) reading on selected port
DIGIN	byte	Byte-encoded User I/O digital input
DIGOUT	byte	Byte-encoded User I/O digital output
BATTERYX10	int	Actual battery voltage in 0.1 V (especially useful for battery voltages > 25.5)
CHECKSUM	int	Packet-integrity checksum

‡ Client-side data-conversion factor. Consult the ARIA parameter file your robot.
ǂ Explicitly, an unsigned integer; all others sign-extended

The number of client commands you may send per second depends on the `HOST` serial baud rate, average number of data bytes per command, synchronicity of the communication link, and so on. ARCOS' command processor runs on a one millisecond interrupt cycle, but the server response speed depends on the command. Typically, limit client commands to a maximum of one every 3-5 milliseconds or be prepared to recover from lost commands.

MOTION COMMANDS

The ARCOS motor-control servers accept several different motion commands of two types: either independent-wheel (`VEL2`) or translation/rotation movements (`VEL`, `ROT`, etc). Actually, `VEL2` commands are composed at the server into their translation and rotation components and the corresponding limits and (de)accelerations get applied. However, the ARCOS servers automatically abandon any translation or rotation setpoints and switch to independent wheel velocity- or translation/rotation-type controls when your client issues a command of the opposite type.

Table 5. ARCOS client-side command set

COMMAND	#	ARGS	DESCRIPTION	ARCOS VERSION
			Before Client Connection	
SYNC0	0	none	Start connection. Send in sequence. ARCOS echoes synchronization commands back to client, and robot-specific auto-synchronization after SYNC2.	1.0
SYNC1	1	none		
SYNC2	2	none		
			After Established Connection	
PULSE	0	none	Reset server watchdog.	1.0
OPEN	1	none	Start up servers.	1.0
CLOSE	2	none	Close servers and client connection.	1.0
POLLING	3	str	Change sonar polling sequence.	1.0
ENABLE	4	int	1=enable; 0=disable the motors.	1.0
SETA	5	int	Set translation acceleration, if positive, or deceleration, if negative; in mm/sec^2.	1.0
SETV	6	int	Set maximum/move translation velocity; mm/sec.	1.0
SETO	7	none	Reset local position to 0,0,0 origin.	1.0
MOVE	8	int	Translate (+) forward or (-) back mm distance at SETV speed	1.0
ROTATE	9	int	Rotate (+) counter- or (-) clockwise degrees/sec.	1.0
SETRV	10	int	Sets maximum/turn rotation velocity; degrees/sec.	1.0
VEL	11	int	Translate at mm/sec forward (+) or backward (-) (SETV limit).	1.0
HEAD	12	int	Turn at SETRV speed to absolute heading; ±degrees (+ = ccw).	1.0
DHEAD	13	int	Turn at SETRV speed relative to current heading; (+) counter- or (–) clockwise degrees.	1.0
SAY	15	str	Play up to 20 pairs of duration, tone sound pairs through User Control Panel piezo speaker.	1.0
CONFIG	18	none	Request a configuration SIP.	1.0
ENCODER	19	int	Request one (1), a continuous stream (>1), or stop (0) encoder SIPs.	1.0
RVEL	21	int	Rotate robot at (+) counter- or (–) clockwise; degrees/sec (SETRV limit).	1.0
DCHEAD	22	int	Adjust heading relative to last setpoint; ± degrees (+ = ccw)	1.0
SETRA	23	int	Change rotation de(-) or (+)acceleration, in $degrees/sec^2$	1.0
SONAR	28	int	1=enable, 0=disable all the sonar; otherwise, use bit 0 to enable (1) or disable (0) a particular array 1-4, as specified in argument bits 1-4.	1.0
STOP	29	none	Stop the robot; motors remain enabled	1.0
DIGOUT	30	2 byte	Set (1) or reset (0) User Output ports. Bits 8-15 is a byte mask that selects, if set (1), the output port(s) for change; Bits 0-7 set (1) or reset (0) the selected port(s).	1.0
VEL2	32	2 byte	Set independent wheel velocities; bits 0-7 for right wheel, bits 8-15 for left wheel; in 20mm/sec increments.	1.1
GRIPPER	33	int	Gripper server commands. See the Gripper or PeopleBot Manual for details.	1.0
ADSEL	35	int	Selects the A/D port number for reporting `ANPORT` value in standard SIP.	1.0
GRIPPERVAL	36	int	Gripper server values. See Gripper or PeopleBot Manual for details.	1.0
GRIPREQUEST	37	none	Request one (1), a continuous stream (>1), or stop (0) Gripper SIPs.	1.0
IOREQUEST	40	none	Request one (1), a continuous stream (>1), or stop (0) IO SIPs.	1.0
TTY2	42	str	Sends string argument to serial device connected to AUX1 serial port.	1.0

GETAUX	43	uint	Request to retrieve 1-200 bytes from the AUX1 serial port; 0 flushes the buffer.	1.0
BUMP_STALL	44	int	Stall robot if no (0), only front (1) while moving forward, only rear (2) while moving backward, or either (3) bumpers contacted when robot moving in related direction.	1.0
TCM2	45	int	TCM2 module commands; see *TCM2 Manual* for details.	1.0
JOYDRIVE	47	int	1=allow joystick drive from port while connected with a client; 0 (default) disallows.	1.0
SONAR_CYCLE	48	uint	Change the sonar cycle time; in milliseconds.	1.0
HOSTBAUD	50	int	Change the HOST serial port baud rate to 0=9600, 1=19200, 2=38400, 3=57600, or 4=115200.	1.0
AUX1BAUD	51	int	Change the AUX1 serial port baud rate (see `HOSTBAUD`).	1.0
AUX2BAUD	52	int	Change the AUX2 serial port baud rate (see `HOSTBAUD`).	1.0
AUX3BAUD	53	int	Change the AUX3 serial port baud rate (see `HOSTBAUD`).	1.0
E_STOP	55	none	Emergency stop; very abrupt by overriding deceleration.	1.0
TTY4	60	str	Send string argument out to device connected at AUX3 serial port.	1.0
GETAUX3	61	int	Request to retrieve 1-200 bytes from the device connected at the AUX3 serial port; 0 flushes the buffer.	1.0
TTY3	66	str	Send string argument out to device connected at AUX2 serial port.	1.0
GETAUX2	67	int	Request to retrieve 1-200 bytes from the device connected at the AUX2 serial port; 0 flushes the buffer.	1.0
CHARGE	68	int	0=release; 1=deploy autocharging-docking mechanism.	1.0
ARM	70-80	int	Pioneer Arm-related commands. See Arm manual for details.	1.1
ROTKP	82	int	Change working rotation Proportional PID value.	1.0
ROTKV	83	int	Change working rotation Derivative PID value.	1.0
ROTKI	84	int	Change working rotation Integral PID value.	1.0
TRANSKP	85	int	Change working translation Proportional PID value.	1.0
TRANSKV	86	int	Change working translation Derivative PID value.	1.0
TRANSKI	87	int	Change working translation Integral PID value.	1.0
REVCOUNT	88	int	Change working differential encoder count.	1.1
DRIFTFACTOR	89	int	Change working drift factor.	1.0
SOUNDTOG	92	int	0=mute User Control piezo; 1 = enable.	1.0
TICKSMM	93	int	Change working encoder ticks per millimeter tire travel.	1.1
BATTEST	250	int	Artificially set the battery voltage; argument in tens volts (100=10V); 0 to revert to real voltage	1.0
RESET	253	none	Force a power on-like reset of the controller.	1.0
MAINTENANCE	255	none	Engage controller maintenance (ARSHstub) mode.	1.0

Note that once connected, *Activ*Media robots' motors are, regardless of their state when last connected. Accordingly, you must either enable the motors manually (white `MOTORS` button on the User Control Panel) or send the motors `ENABLE` client command #4 with the argument value of one.[17] Monitor the status of the motors with bit 0 of the `Flags` integer in the standard SIP.

[17] Alternatively, disable the motors with the `ENABLE` command argument of zero.

Table 6. ARCOS motion commands

Rotation	
HEAD (#12)	Turn to absolute heading at SETRV max velocity
DHEAD (#13), DCHEAD (#22)	Turn to heading relative to control point at SETRV max velocity
ROTATE (#9)	Rotate at SETRV velocity
Translation	
VEL (#11)	Translate forward/reverse at prescribed velocity (SETV maximum)
MOVE (#8)	Translate distance at SETV max velocity
Independent	**Wheel**
VEL2 (#32)	Set velocity for each side of robot

ActivMedia Robots in Motion

When the robot controller receives a motion command, it accelerates or decelerates the robot at the translation `SETA (command #5)` and rotation `SETRA` (command #23) rates until the platform either achieves its `SETV` (command #6) maximum translation and `SETRV` (command #10) maximum rotation speeds (or VEL2 equivalents), or nears its goal. Accordingly, rotation headings and translation setpoints are achieved by a trapezoidal velocity function, which ARCOS recomputes each time a new motion command is received.[18]

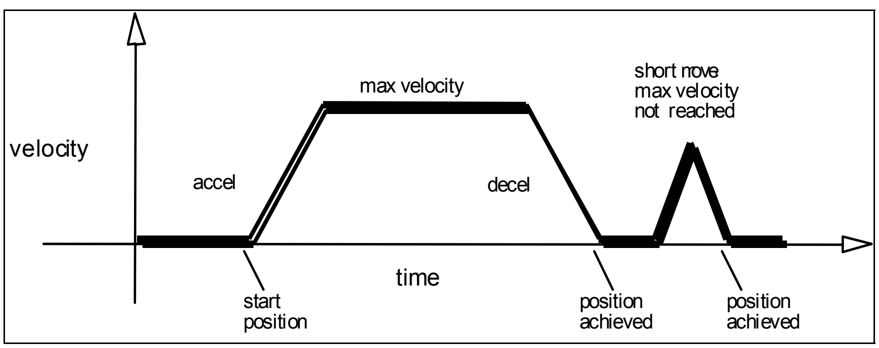

Figure 18. ActivMedia robot's trapezoidal velocity profile

ARCOS automatically limits `VEL2`-, `VEL`-, and `RVEL`-specified velocities to previously imposed, client-modifiable `SETVEL` and `SETRV` maximums, and ultimately by absolute, platform-dependent, FLASH-embedded constants (`TOP` values). Similarly, the distinct acceleration and deceleration parameters for both translation and rotation are limited by FLASH constants. ARCOS initializes these values upon controller startup or reset from related FLASH parameters. The speed limits, either from FLASH or when changed by SETV or SETRV commands, take effect on subsequent commands, not on current translation or rotation. The maximums revert to their FLASH defaults when disconnected from a client.

[18] Note that acceleration and deceleration are distinct values, settable via SETA for translation and SETRA for rotation.

The orientation commands `HEAD` (#12), `DHEAD` (#13), `DCHEAD` (#22) turn the robot with respect to its internal dead-reckoned angle to an absolute heading (0-359 degrees), relative to its immediate heading, or relative to its current heading setpoint (achieved or last commanded heading), respectively. In general, positive relative heading command arguments turn the robot in a counterclockwise direction. However, the robot always turns in the direction that will achieve its heading most efficiently. Accordingly, relative-heading arguments greater than 179 degrees automatically get reduced to 179 or less degrees with a concomitant change in direction of rotation.

The `E_STOP` command #55 or the `STOP` button that is found on some *Activ*Media robots override deceleration and abruptly stop the robot in the shortest distance and time possible. Accordingly, the robot brakes to zero translation and rotation velocities with very high deceleration and remains stopped until it receives a subsequent translation or rotation velocity command from the client or until the STOP button is reset. (See *E_STOP and E_STALL* later in this chapter.)

PID Controls

The ARCOS drive servers use a common Proportional-Integral-Derivative (PID) control system to adjust the PWM pulse width at the motor drivers to smooth the power to the motors. The motor-duty cycle is 50 microseconds (20 KHz); pulse-width is proportional 0-500 for 0-100% of the duty cycle.

The ARCOS drive servers recalculate and adjust your robot's trajectory and speed every five milliseconds based on feedback from the wheel encoders.

The default PID values for translation and rotation and maximum PWM are stored as FLASH parameters in your robot's controller and may be changed. You also may temporarily update the PID values with the ARCOS client commands #84 through #87. On-the-fly changes persist until the client disconnects. Translation PID values apply to independent wheel-velocity mode.

The P-term value Kp increases the overall gain of the system by amplifying the position error. Large gains will have a tendency to overshoot the velocity goal; small gains will limit the overshoot but cause the system to become sluggish. We've found that a fully loaded robot works best with a Kp setting of around 15 to 30, whereas a lightly loaded robot may work best with Kp in the range of 20 to 50.

The D-term Kv provides a PID gain factor that is proportional to the output velocity. It has the greatest effect on system damping and minimizing oscillations within the drive system. The term usually is the first to be adjusted if you encounter unsatisfactory drive response. Typically, we find Kv to work best in the range of 10 to 30 for lightly to heavily loaded robots, respectively. If your robot starts to vibrate or shutter, reduce Kv.

The I-Term Ki moderates any steady state errors thereby limiting velocity fluctuations during the course of a move. At rest, your robot will seek to "zero out" any command position error. Too large of a Ki factor will cause an excessive windup of the motor when the load changes, such as when climbing over a bump or accelerating to a new speed. Consequently, we typically use a minimum value for Ki in the range of 0 to 10 for lightly to heavily loaded robots respectively.

Position Integration

*Activ*Media robots track their position and orientation based on dead-reckoning from wheel motion derived from encoder readings. The ARCOS-based robot maintains its internal coordinate position in platform-dependent units, but reports the values in platform-independent millimeters and degrees in the standard SIP (Xpos, Ypos, and Thpos).

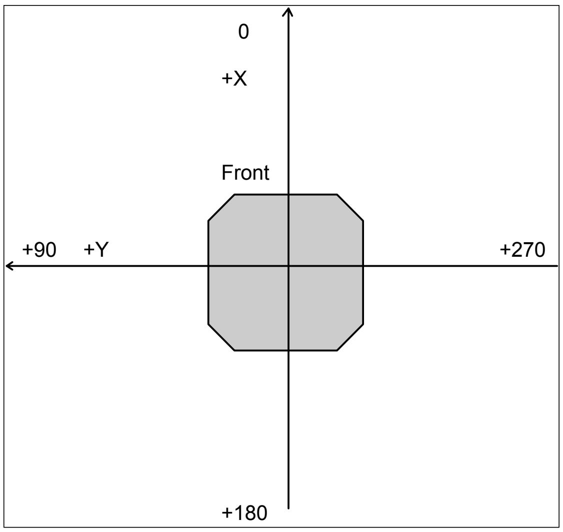

Figure 19. Internal coordinate system

Be aware that registration between external and internal coordinates deteriorates rapidly with movement due to gearbox play, wheel imbalance and slippage, and many other real-world factors. You can rely on the dead-reckoning ability of the robot for just a short range—on the order of a few meters and one or two revolutions, depending on the surface. Carpets tend to be worse than hard floors.

Also, moving either too fast or too slow tends to exacerbate the absolute position errors. Accordingly, consider the robot's dead-reckoning capability as a means of tying together sensor readings taken over a short period of time, not as a method of keeping the robot on course with respect to a global map.

On start-up, the robot is at the origin (0, 0, 0), pointing toward the positive X-axis at 0 degrees. Absolute angles vary between 0 and 359.

You may reset the internal coordinates back to 0,0,0 with the `SETO` command #7.

DriftFactor, RevCount, and TicksMM

Three client commands let you change, albeit momentarily for the current client-server connection, those values that affect translation, rotation, and drift in your robot. The `driftFactor` is a signed value in 1/8192 increments that gets added to or subtracted from the left wheel encoder to correct for tire circumference differences and consequent translation and rotation drift. `DriftFactor` defaults to its FLASH value on start up or reset, and can be changed on-the-fly with the `DRIFTFACTOR` client command #89 with signed integer argument.

The `revCount` parameter is the differential number of encoder ticks for a 180-degree rotation of the robot, and is used to compute and execute headings. Like `driftFactor` and `ticksMM`, `revCount` defaults to its FLASH value on startup or reset, and can be changed on-the-fly with the `REVCOUNT` client command #88 and unsigned integer argument.

`TicksMM` is the number of encoder ticks per millimeter tire rotation for translation speed and distance computations. The default FLASH value can be changed on-the-fly during a client connection session with the `TICKSMM` client command #93 and unsigned integer value.

SONAR

When connected with and opened by the client, ARCOS automatically begins firing your robot's sonar, one disc each simultaneously for each of up to four arrays, as initially sequenced and enabled in your robot's FLASH parameters. The sonar servers also begin sending the sonar-ranging results to the client via the standard SIP.

Enable/Disabling Sonar

Use the `SONAR` client command #28 to enable or disable all or individual sonar arrays. Set ("1") bit zero of the SONAR argument to enable or reset it ("0") to disable the sonar pinging. Set argument bits two through four to an individual array number one through four to enable or disable only that array. Array zero, the form of the original P2OS command, affects all the arrays at once.

For example, an argument value of one enables all the sonar arrays, whereas an argument value of six silences array number three. Monitor the status of the sonar arrays in the `FLAGS` integer bits 1-4 of the standard SIP: the respective bit is set if the array is engaged.

Polling Sequence

Each array's sonar fire at a rate and in the sequence defined in your robot's FLASH parameters. (Consult the next chapter on how to change the FLASH settings.) Use the sonar `POLLING` command #3 to have your client change the firing sequence, and the `SONAR_CYCLE` command #48 to change the rate. The changes persist until you restart the client-server connection.

The `POLLING` command string argument consists of a sequence of sonar numbers one through 32. Sonar numbers one through eight get added to the polling sequence for sonar array number one; numbers nine through 16 get added to the sequence for sonar array number two; 17-24 specify the sequence for array three; and 25-32 are for array four. You may include up to 16 sonar numbers in the sequence for any single array. Only those arrays whose sonar numbers appear in the argument get re-sequenced. You may repeat a sonar number two or more times in a sequence. If a sonar number does not appear in an otherwise altered sequence, the disc will not fire.

Note that for compatibility with earlier *Activ*Media robot operating systems, if the string is empty, all the sonar get disabled, but their polling sequences remain unaltered, just as if you had sent the `SONAR` command with an argument value of zero.

Polling Rate

For earlier Pioneer controller versions, the sonar polling rate was fixed for each array: one sonar disc per array got polled every 40 milliseconds. So, for an eight-disc array with a sequential polling sequence (12345678, or example), any one sonar transducer would be read every 320 milliseconds. That common cycle timing accommodates ranging out to the maximum of the sonar of over six meters for general applications, including features recognition and localization. For other applications, such as close-in obstacle avoidance, a shorter range but faster rate of update is better.

Use the `SONAR_CYCLE` client command #48 to change the cycle timing on the fly to the command integer's argument value in milliseconds. Minimum and maximum values are two and 120 milliseconds, respectively. The default value is set in FLASH, normally 40 milliseconds.

STALLS AND EMERGENCIES

With a robot equipped with forward and/or rear bumpers, ARCOS may immediately stop the robot and notify the client of a stall if any one or more of the contact sensors get triggered and the robot is going in the direction of the bump (forward/front or backward/rear). Send the `BUMPSTALL` command #44 with an integer argument of zero to disable that bump-stall behavior. Give the argument value of one to re-enable

BUMPSTALL only when a forward bump sensor gets triggered; two for rear-only BUMPSTALLs; or three for both rear and forward bump contact-activated stalls.

In an emergency, your client may want the robot to stop quickly, not subject to normal deceleration. In that case, send the E_STOP command (#55).

Like BUMPSTALL, use ARCOS' built-in E_STALL feature to simulate a stall when someone presses the robot's STOP button.[19] An integrated switch in the STOP button toggles a dedicated digital I/O port on the controller thereby notifying ARCOS of the condition. ARCOS stops the robot's motors, puts on the brakes, and throws continuous stalls.

Unlike other stalls, E_STALL also disables the motors. You must either re-enable the motors manually (MOTORS button) or programmatically (ENABLE command #4).

Table 7. The FLAGS bits

BIT	CONDITION IF SET
0	Motors enabled
1	Sonar array #1 enabled
2	Sonar array #2 enabled
3	Sonar array #3 enabled
4	Sonar array #4 enabled
5	STOP button pressed
6	E_stall engaged
7	Far ledge detected (IR)
8	Near ledge detected (IR)
9	Joystick button 1 pressed
10	Recharging "power-good"
11-15	Reserved

The E_STALL server notifies your client software through the stall bytes and in bit 5 of the FLAGS byte in the standard so that your client may respond to a STOP E_STALL differently than a regular stall.

Normally enabled (default was disabled in P2OS), change E_STALL by sending the ARCOS command #56. With argument of zero, E_STALL gets disabled. An argument value of one re-enables E_STALL.

ACCESSORY COMMANDS AND PACKETS

Several types of alternative server information packets (SIPs) come with ARCOS to better support the *Activ*Media Robotics community. On request from the client by a related ARCOS command, the ARCOS server packages and sends one or a continuous stream of information packets to the client over the HOST serial communication line. Extended packets get sent immediately before (such as GYROpac) or after (such as IOpac) the standard SIP that ARCOS sends to your client every SIP milliseconds (typically 100).[20]

The standard SIP takes priority so you may have to adjust the HOST serial baud rate to accommodate all data packets in the allotted cycle time, or some packets may never get sent.

Packet Processing

Identical with the standard SIP, all ARCOS server information packets get encapsulated with the header (0xFA, 0xFB; 250, 251), byte count, packet type byte, and trailing checksum. It is up to the client to parse the packets, sorted by type for content. Please consult the respective client application programming manuals for details.

CONFIGpac and CONFIG Command

Send the CONFIG command #18 without an argument to have ARCOS send back a CONFIGpac SIP packet type 32 (0x20) containing the robot's operational parameters. Use

[19] Available only on some robots.

[20] You may have to adjust the HOST serial baud rate to accommodate the additional communications traffic.

the `CONFIGpac` to examine many of your robot's default FLASH_based settings and their working values, where appropriate, as changed by other client commands, such as `SETV` and `ROTKV`.

Table 8. CONFIGpac contents

LABEL	DATA	DESCRIPTION
HEADER	int	Common packet header = 0xFAFB
TYPE	byte	ENCODERpac = 0x20
BYTE COUNT	byte	Number of following data bytes
ROBOT TYPE	str	Typically "Pioneer"
SUBTYPE	str	Identifies the *Activ*Media robot model; e.g. "dxsh",
SERIALNUM	str	Serial number for the robot.
4MOTS	byte	Antiquated (=1 if AT with P2OS)
ROTVELTOP	int	Maximum rotation velocity; deg/sec
TRANSVELTOP	int	Maximum translation speed; mm/sec
ROTACCTOP	int	Maximum rotation (de)acceleration; deg/sec^2
TRANSACCTOP	int	Maximum translation (de)acceleration; mm/sec^2
PWMMAX	int	Maximum motor PWM (500=fully on; typically limited to 400 for DX or AT).
NAME	str	Unique name given to your robot.
SIP	byte	Server information packet cycle time; ms.
HOSTBAUD	byte	Baud rate for client-server HOST serial: 0=9.6k, 1=19.2k, 2=38.4k, 3=56.8k, 4=115.2k.
AUXBAUD	byte	Baud rate for AUX1 serial port; see HostBaud.
GRIPPER	int	0 if no Gripper; else 1.
FRONT SONAR	int	1 if robot has front sonar array enabled, else 0.
REAR SONAR	byte	1 if robot has rear sonar enabled, else 0.
LOWBATTERY	int	In 1/10 volts; alarm activated when battery charge falls below this value.
REVCOUNT	int	Working number of differential encoder ticks for a 180 degree revolution of the robot.
WATCHDOG	int	Ms time before robot automatically stops if it has not received a command from the client. Restarts on restoration of connection.
P2MPACS	byte	1 means alternative SIP enable; not used by ARCOS.
STALLVAL	int	Maximum PWM before stall. If > PWMMAX, never.
STALLCOUNT	int	Ms time after a stall for recovery. Motors lax during this time.
JOYVEL	int	Joystick translation velocity setting, mm/sec
JOYRVEL	int	Joystick rotation velocity setting in deg/sec
ROTVELMAX	int	Current max rotation speed; deg/sec.
TRANSVELMAX	int	Current max translation speed; mm/sec.
ROTACC	int	Current rotation acceleration; deg/sec^2
ROTDECEL	int	Current rotation deceleration; deg/sec^2
ROTKP	int	Current Proportional PID for rotation
ROTKV	int	Current Derivative PID for rotation
ROTKI	int	Current Integral PID for rotation
TRANSACC	int	Current translation acceleration; mm/sec^2
TRANSDECEL	int	Current translation deceleration; mm/sec^2
TRANSKP	int	Current Proportional PID for translation.
TRANSKV	int	Current Derivative PID for translation.
TRANSKI	int	Current Integral PID for translation.
FRONTBUMPS	byte	Number of front bumper segments.
REARBUMPS	byte	Number of rear bumper segments.
CHARGER	byte	1 if P3/PeopleBot or 2 if PowerBot automated charger mechanism and circuitry installed in robot; otherwise 0.
SONARCYCLE	byte	Sonar duty cycle time in milliseconds.
AUTOBAUD	byte	1 if the client can change baud rates; 2 if auto-baud implemented.
HASGYRO	byte	1 if robot equipped with the gyro heading correction device; otherwise 0.
DRIFTFACTOR	int	Working drift factor value.
AUX2BAUD	byte	Baud rate for AUX2 serial port; see HostBaud.

AUX3BAUD	byte	Baud rate for AUX3 serial port; see HostBaud.
TICKSMM	int	Encoder ticks per millimeter tire motion

SERIAL

The baud rates for the HOST and Aux ports initially are set from their respective FLASH-based defaults and get reset to those values whenever the controller is reset or upon client disconnection. For advanced serial port management from the client side, ARCOS provides four client commands which let your software reset the HOST (`HOSTBAUD` #50), Aux1 (`AUX1BAUD` #51), Aux2 (`AUX2BAUD` #52), and Aux3 (`AUX3BAUD` #53) serial port baud rates, respectively. Use the integer command argument values: 0=9600, 1=19.2K, 2=38.8K, 3=57.6K, or 4=115.2K baud, respectively.

With auto-bauding, the HOST serial port automatically reverts to its FLASH default baud rate if, after being reset by the HOSTBAUD client command, it does not receive a subsequent and valid client-command packet within 500 milliseconds.

HOST-to-AUX Serial Transfers

Use the client-side `TTY2` command #42 with a string argument to have that string sent out the Aux1 port to the attached serial device, such as a robotic camera. Similarly, use the `TTY3` command #66 to send a string argument out the Aux2 port, or `TTY4` command #60 to send a HOST-mediated client string out the Aux3 port.

ARCOS also maintains three circular buffers for incoming serial data from the respective Aux ports. On request, ARCOS sends successive portions of the buffer to your client via the HOST serial in the respective `SERAUXpac` (type = 176; 0xB0), `SERAUX2pac` (type = 184; 0xB8), and SERAUX3 (type = 200; 0xC8) SIPs. Use the `GETAUX` #43 for Aux1, `GETAUX2` command #67 for Aux2, and GETAUX3 command #61 for Aux3.

Use the integer argument value of zero to flush the contents of the respective buffer. Otherwise, use an argument value of up to 253 bytes to have ARCOS wait to collect the requested number of incoming AUX-port serial bytes and them send them in the respective `SERAUXpac`, `SERAUX2pac, or SERAUX3pac` SIP.

ENCODERS

Issue the `ENCODER` command #19 with an argument of one for a single, or with an argument value of two or more for a continuous stream of `ENCODERpac` (type 144; 0x90) SIPs. Discontinue the packets with the `ENCODER` command #19 with an argument of zero.

Table 9. ENCODERpac SIP contents

Header	integer	Exactly 0xFA, 0xFB
Byte Count	byte	Number of data bytes + 2 (checksum)
Left Encoder	integer	Least significant, most significant portion of the
	integer	current accumulated encoder counts from the left wheel
Right Encoder	integer	Least significant, most significant portion of the
	integer	current accumulated encoder counts from the left wheel
Checksum	integer	Checksum for packet integrity

BUZZER SOUNDS

Pioneer 3 robots have a piezo buzzer on the User Control Panel that aurally notifies you of system conditions, such as low batteries or stalls. For stealthy operation, issue the

SOUNTOG command #92 with an argument of zero to mute the controller's buzzer or argument of one to re-enable it. (See also the SOUNDTOG FLASH parameter in the next chapter to set its default state.)

The SAY command #15 lets you play your own sounds through the buzzer. The argument consists of a length-specified string of duration,tone pair bytes. The duration is measured in 20 millisecond increments.

A tone value of zero means silence (musical rest). The next 127 frequencies (1-127) are the corresponding MIDI notes. The remaining tones are frequencies computed as:

Tone – 127 * 32

Meaning frequencies from 1 to 4096, in 32 Hz increments.

Except for the MIDI notes, you'll just have to experiment with tones. Here is the sequence that generates the ARCOS distress wail when the robot stalls or the batteries are low:

```
50,100,20,0,50,60,0
```

TCM2

The TCM2 accessory is an integrated inclinometer, magnetometer, thermometer, and compass that attaches to one of the AUX serial ports of the ARCOS controller. When attached and enabled, special TCM2 compass servers read and report the heading in ±2 degree increments as the compass byte in the standard SIP. Use the TCM2 command 45 to request additional information from the device in the form of the TCM2pac. See the *TCM2 Manual* and supporting software that accompanies the device for details.

ONBOARD PC

Communication between the onboard PC and the robot controller is RS-232 serial through the respective COM1 (Windows) or /dev/ttyS0 (Linux) and internal HOST ports. Set the HostBaud FLASH communication rate to match the PC client-software's serial port rate.

The RI pin 9 on the HOST port initializes to low and goes high when the batteries discharge to below the FLASH-set ShutdownVolts value. We use the *genpowerd* software under Linux to detect that low-power signal and automatically shut down the PC. Windows PCs are a bit more problematic.

The Windows *genpowerd*-like *ups.exe* program requires a dedicated serial port and prefers to use the CTS line to indicate low power. Accordingly, we jumper the RI signal of HOST COM1 to the CTS signal pin of the adjacent COM2 port of the onboard PC for the feature. For convenience, the Versalogic VSBC8 PC found onboard most recent Pioneers shares its 20-pin connector on the PC's motherboard with COM1 and COM2. So, to implement Windows *ups.exe*-enabled low-power shutdown, we jumper pin 8 (COM1 RI) to pin 16 (COM2 CTS) on that VSBC8 serial connector. Use a similar strategy for other implementations; the UPS configuration dialog lets you select COM1-4.

Once the port is wired, start up Windows and, as Administrator, go to the Start:Settings:Control Panel:Power Options dialog and select the UPS tab. Click Select and in the UPS Selection dialog, select COM2 (or other) port, Generic manufacturer, and Custom model. Then click Next.

In the UPS Interface Configuration On: COM2 dialog, check the Power Fail/On Battery and its related Position options. Uncheck to disable the Low Battery and UPS Shutdown options. Then click Finish to save the settings and close the dialog. Click OK or Apply to enable the UPS shutdown programs.

Change a registry value so that the PC shuts down one minute instead of two minutes after low-power notification by the controller: Use *regedit* and navigate to `[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\UPS\Config`. Change the `ShutdownOnBatteryWait` dword value to 1 (from 2).

Use the ARCOS client maintenance command #250 to test your *genpowerd* or `ups.exe` setup. Send a bogus battery voltage as its integer argument below `ShutdownVolts` to simulate the low battery condition. ARCOS should issue warnings first, then disconnects from the client after about a minute and sets the PC-shutdown signal on `RI`. Restore to the real battery voltage by sending the #250 command with 0 as its argument. Resetting the controller cancels shutdown, too, unless battery power really is very low.

Put the controller into maintenance mode and fix your onboard PC settings if the computer falsely engages *genpowerd* or *ups.exe*.

INPUT OUTPUT (I/O)

Your SH2-based controller comes with a number of I/O ports that you may use for sensor and other custom accessories and attachments. See Appendix A for port locations and specifications. Some I/O states and readings appear in the standard SIP and may be manipulated with ARCOS client commands. There also is an `IOpac` SIP for convenient access to all of your robot's I/O.

Gripper

Please consult the respective Gripper manual for details.

ARCOS supports a `GRIPPERpac` (type=224; 0xE0) packet type and related `GRIPREQUEST` P2OS command #37 to retrieve setup and status information from the servers.

Table 10. GRIPPERpac packet contents

HEADER	int	Exactly 0xFA, 0xFB
BYTE COUNT	byte	Number of data bytes + 2 (checksum)
TYPE	byte	Packet type = 0xE0
HASGRIPPER	byte	Gripper type: 0=none; 1=Pioneer; 2=PeopleBot
GRIP_STATE	byte	See *Table 11 below.*
GRASP_TIME	byte	Ms time controls grasping pressure.
CHECKSUM	integer	Computed checksum

Normally disabled, your client program may request one or a continuous stream (command argument > one) of Gripper packets. Send `GRIPREQUEST` with the argument value zero to stop continuous packets.

Table 11. GRIPPERpac GRIP_STATE byte

BIT	FUNCTION	STATE
0	Grip limit	Paddles fully open when 0; otherwise between or closed.
1	Lift limit	Lift fully up or down when 0; otherwise in between.
2	Outer breakbeam	Obstructed when 0; nothing in between when 1.
3	Inner breakbeam	Obstructed when 0; nothing in between when 1.
4	Left paddle	Grasping when 0.
5	Right paddle	Grasping when 0.
6	Lift	Moving when 1.
7	Gripper	Moving when 1.

Note that the Gripper status information bits 0-5 also may be obtained from the respective `DIGIN` and `DIGOUT` values of the standard SIP as related to the User I/O port states. See *Appendix A* for connection details.

Heading Correction Gyro

With the new rate-gyro accessory, your client software may detect and compensate for robot heading changes that aren't detected by the wheel encoders, such as from slipping wheels. The controller supports the gyro via attachment to its AN6 and AN7 analog-to-digital input ports.

ARCOS collects 10-bit (0-1023) gyro rate and 8-bit (0-255) temperature data and will, upon request, send the collected data to a connected client in a new `GYROpac` (type=0x98) server information packet for processing. Analysis of the gyro data and subsequent modifications to the robot's heading are done on the client side, as supported in the latest versions (1.3 and later) of ARIA.

To enable the gyro, you must set the `hasGyro` FLASH parameter to one using the ARCOScf tool (see next chapter). Set it to 0 if the gyro isn't attached. Then to acquire gyro data, send the `GYRO` command #58 with integer argument of one; zero disables the gyro SIP. The gyro SIP is stopped upon client disconnection or controller reset, too.

ARCOS collects the gyro rate and temperature readings at the maximum rate of once every 25 milliseconds and reports each of these values to the client, when enabled, in the `GYROpac` SIP that gets sent just before the standard Server Information Packet every `sInfoCycle`, typically every 100ms. `GYROpac` consists of a count byte of the rate and temperature data pairs accumulated since the last cycle (typically 4 for a 100ms cycle time), followed by that number of rate/temperature integer/byte pairs.

Gyro rates are 10-bit integers of value 0-1023. When not moving, the rate is centered around 512 or so, depending on the gyro's temperature and other calibration factors which drift with use and should be corrected on the fly. Values below that center point indicate counter-clockwise rotation rates; values above the resting center measure clockwise rotation rates.

Table 12. GYROpac SIP contents

LABEL	BYTES	CURRENT VALUE	DESCRIPTION
HEADER	2	0xFA, 0xFB	Common header
BYTE COUNT	1	xx	Varies
TYPE	1	0x98	Packet type
N PAIRS	1	x	Number of gyro data pairs
FOR N PAIRS			
RATE	2	varies 0-1023	Gyro rate
TEMPERATURE	1	varies 0-255	Gyro temperature
CHECKSUM	2	varies	Computed checksum

User I/O

The User I/O connector on the Pioneer 3 controller contains eight digital input and eight digital output ports, as well as an analog-to-digital (A/D) port.[21] The bit-mapped states of the sixteen digital ports and analog port automatically and continuously appear in the standard SIP, in their respective `DIGIN`, `DIGOUT`, and `ANALOG` bytes. When not physically connected, the digital input and A/D port values may vary and change without warning.

Use the ARCOS client command #30 to set one or more of the eight `DIGOUT` ports on the ARCOS controller. Electrically, the ports are digital high (1) at ~5 VDC (V_{cc}) and low (0) at ~0 VDC (`GND`). `DIGOUT` uses a two-byte (unsigned integer) argument. The first byte is a mask whose bit pattern selects (1) or ignores (0) the state of the corresponding bit in the second byte to set (1) or unset (0) the digital output port.

[21] Many of these ports are used by the Gripper and other accessories. Alternative I/O is available.

For example, here's the ARCOS client command to set digital output ports one and three (OD1 and OD3), reset port four (OD4), and leave all the rest alone (hexadecimal notation):

```
250, 251, 6, 30, 27, 25, 9, 55, 36
```

Bumper and IR I/O

Two 10-position microfit connectors on the ARCOS controller provide 16 digital input ports that are normally used for the bumper accessory, but also available for your own attachments. See *Appendix A* for connector details.

Similarly, the Motor-Power connector on the controller contains eight digital inputs that we normally use for IR sensors on the Performance PeopleBot and PowerBot, and whose states are digitally mapped. See *Appendix B* for connector details.

Normally pulled high (5 VDC=digital port bit value 1), all the bumper and IR bit-mapped switches go low (digital 0) when the respective port gets triggered. Bumper inputs also appear with the stall bits in the standard SIP, but unlike in the IOpac, are modified by the InvertBumps mask. All the bumper and IR data bits appear in the IOpac packet.

IO packets

Table 13. IOpac packet contents

LABEL	BYTES	CURRENT VALUE	DESCRIPTION
HEADER	2	0xFA, 0xFB	Common header
BYTE COUNT	1	22	Number of data bytes + 2
TYPE	1	0xF0	Packet type
N DIGIN	1	4	Number of digital input bytes
DIGIN	1	varies 0-255	ID0-8 bits mapped
FRONTBUMP*	1	varies 0-255	Front bumper bits mapped
REARBUMP*	1	varies 0-255	Rear bumper bits mapped
IR	1	varies 0-255	IR inputs
N DIGOUT	1	1	Number of digital output bytes
DIGOUT	1	varies 0-255	Digital output byte(s)
AN	1	9	Number of A/D values
A/D	10	5 integers varying 0-1023	A/D ports 0-7 input values at 10-bit resolution = 0-5 VDC
BATTERY	2	0-1023	Battery A/D input (AN3 Pioneer 3)
CHECKSUM	2	varies	Computed checksum

* Actual, not affected by InvertBumps since bumper bits may be used for other digital input besides bumpers.

Not all analog and digital I/O appears in the standard SIP. Accordingly, your client software may request the IOpac SIP (type = 240; 0xF0), which contains all common I/O associated with the H8S controller and which appear on the various connectors, including User IO, General IO, Bumpers, and IRs.

Use the ARCOS client IOREQUEST command number 40 with an argument value of zero, one, or two. The argument value one requests a single packet to be sent by the next client-server communications cycle. The request argument value of two tells ARCOS to send IOpac packets continuously, at approximately one per cycle depending on serial port speed and other pending SIPs. Use the IOREQUEST argument value zero to stop continuous IOpac packets.

DOCKING/CHARGING SYSTEM

The docking/charging system's mechanism and associated charge-management circuitry on the robot may be controlled from the robot's controller and ARCOS servers.

Digital Port Controls

When set digital high (1), the "inhibit" port OD4 on pin 10 of the User I/O connector (see Appendix A) causes the charging mechanism to disengage and retract from the charging platform and inhibits its future deployment. The "deploy" port OD5 pin 12, when set high with port OD4 low, deploys the charging mechanism with full force to seat it onto the charging platform.[22]

At the fully deployed position, the mechanism is mechanically stabilized and requires much less force to maintain contact. If in positive contact with the charger base, the robot's onboard circuitry activates and thereafter maintains the actuated mechanism at that lower force as long as it receives power. To minimize heat and eventual damage to the actuator, the deploy line should be activated for only short periods; maximally for 10 seconds at a time.

Your client software may run the charging mechanism by individually activating/ deactivating the digital output ports, such as with the ARCOS `DIGOUT` command #30. However, for best results, we recommend using the automated charging control commands and systems we provide with the latest ARCOS.

Docking/Charging Servers

To use ARCOS' docking/charging servers, you must first enable the controller's automated charger servers through your robot's FLASH parameters. Use the ARCOScf configuration tool and set the `Charger` parameter value to one (zero to disable) and save the value.

Thereafter, for autonomous operation of the robot with the charging platform, establish a client-server connection between an ARIA- or similar client-enabled PC and the robot's controller. Use the ARCOS `CHARGE` command #68 with an integer argument of one to automatically halt robot motion and deploy the docking mechanism. The docking mechanism automatically retracts after five seconds if the robot does not engage with the docking platform, during which time the robot's drive system is unresponsive. So your client should wait at least that long before attempting to resume activity.

The CHARGE command automatically disables the robot's motors, so you will have to re-engage them from the client (command #4) or by manually pressing the MOTORS button on the controller. Re-engaging the motors automatically retracts the charging mechanism.

While the motors are engaged, the charging mechanism cannot be deployed, except by the `CHARGE` command. For best control and safety, consider also using the ARCOS `CHARGE` command number 68 with integer argument of zero to gracefully cancel charging and retract the charging mechanism.

In addition to the client-mediated commands, you also may cancel recharging and retract the charging mechanism manually with the `Charge Deploy` button, as described in an earlier chapter. Note that client-mediated docking/charging behaviors may act to reverse your actions.

For example, the client may, upon untimely loss of recharging power resulting from someone pressing the Charge Deploy button, may re-engage the motors and have the robot automatically attempt to re-dock with the charging platform and restart charging.

---

[22] These output ports and the charge-sensing User I/O-based digital input ports (see below) do not interfere with the Pioneer/PeopleBot Gripper.

Your client software may disengage and re-engage the client-server connection without disrupting recharging, as long as the robot remains positively engaged with the charging platform and you don't do anything else to otherwise disrupt recharging. Once disengaged from the client, the rules for engaging and disengaging the recharge mechanism and power manually apply.

Monitoring the Recharge Cycle

Three digital signals indicate battery recharging states with the docking/recharging system. All appear in the standard SIP.

Table 14. Recharging cycle states

Charge State	Overcharge (ID7)	~Volts	Charge current
Bulk	1	discharge-~14V	6A
Overcharge	0	~14-14.7	decreases to ~1A
Float	1	~13.5	< 1A

The "power-good" signal appears as both User I/O `DIGIN` bit 6 and as bit 10 of the `FLAGS` integer in the standard SIP, but their states are inversely related: `DIGIN` bit 6, normally high (1) when not charging or when the charging system is not installed, goes low (0) when the recharge system is engaged on the charge platform. Conversely, the power-good bit 10 in `FLAGS` normally is low and goes high when the robot is docked and charging. For compatibility with future docking systems, we recommend that your client monitor the power-good FLAGS bit and not the DIGIN line to determine if the robot is getting power from the charging platform.

The `DIGIN` and `DIGOUT` bytes of the Standard SIP also reflect the states of the associated charging digital input and output bits. `DIGOUT` bits 4 and 5 are the inhibit and deploy output ports described earlier. `DIGIN` bit 7, corresponding to the User I/O connector digital input port ID7, pin 15, reflects the battery recharge cycle and, with the `Battery Voltage` SIP value, helps the autonomous robot client determine immediate battery life and operation times.

The "overcharge" bit ID7 is set (1) when the batteries are well below full charge and the charger is at full charging current. During this bulk-charging period, the battery voltage rises to around 13.8-14V. The overcharge bit ID7 then drops to low (0) while the batteries charge from approximately 80% to 90% of full charge: from ~13.8 to 14.7V. The charger then reverts to "float mode", maintaining full charge at much lower current and charger voltage (~13.5V).

In float mode, the overcharge bit ID7 is set.

Accordingly, by monitoring the power-good and overcharge bits, as well as the battery voltage, your client may make recharging strategy decisions. The thing to remember is that lead-acid batteries last longest when routinely charged into float mode, typically once per day.

Chapter 7 Updating & Reconfiguring ARCOS

The ARCOS software and a set of operating parameters get stored in your *Activ*Media robot controller's FLASH ROM. With special upload and configuration software tools, you change and update ARCOS, too. No hardware modification is required.

WHERE TO GET ARCOS SOFTWARE

Your *Activ*Media robot comes preinstalled with the latest version of ARCOS. And the various ARCOS configuration and update tools come with the robot on CD-ROM. Thereafter, stay tuned to the `pioneer-users` newsgroup or periodically visit our support website to obtain the latest ARCOS software and related documentation:

`http://robots.activmedia.com`

The main utility, `ARCOScf`, is a multi-functional application for both uploading new ARCOS versions as well as modifying your robot's onboard FLASH-based parameters.

ARCOS MAINTENANCE MODE

To connect with and update your robot's ARCOS servers and its FLASH-based operating parameters, you need to first connect a serial port on the PC from which you will run ARCOScf to the HOST port of your robot's controller:

- ✓ If you are running from an onboard PC, the computer-to-`HOST` connection already is made.
- ✓ If you have an onboard PC, but prefer to use an external computer for maintenance, simply power down the onboard computer.
- ✓ If you use radio or Ethernet wireless, switch its power OFF (typically AUX1).
- ✓ When connecting from an external PC, directly tether (no radios) its serial port to the 9-pin DSUB serial connector on the User Control Panel.

Enabling Maintenance Mode on the Controller

You have three ways in which to put the controller into ARCOS Maintenance Mode:

- ✓ Start Up
- ✓ Manual
- ✓ Automatic

If for any reason your robot's FLASH parameters get erased or your ARCOS software encounters a code fault, your Pioneer 3 controller automatically reverts to ARSHstub-based Maintenance Mode.[23] Or, if you attach a PC to the SERIAL port on the User Control Panel, open the port through software such as by running ARCOScf on that PC, and then reset or otherwise restart the controller, it will automatically revert into Maintenance Mode.

Like with previous Pioneer controllers, you may manually engage Maintenance Mode manually:

1. Press and hold the white `MOTORS` button on the User Control Panel
2. Press and release the adjacent red `RESET` button
3. Release the `MOTORS` button.

[23] ARSHstub is a resident GDB-like interface for FLASH updates and other debugging uses.

Unlike previous controllers, and certainly with much more convenient since you don't need to be right next to the robot, ARCOScf *automatically* engages Maintenance Mode when run through an onboard PC. Just start it up and it forces the controller into maintenance mode. It uses the ARCOS command #255 to do that.

The `STATUS` LED on the User Control Panel should flash twice the rate than when in server ("wait") mode and the `BATTERY` LED should shine bright red.

ARCOScf

The ARCOS update and configuration program, `ARCOScf`, is part of a collection of utilities and files for comprehensive management of your *Activ*Media robot's onboard servers and FLASH-based operating parameters. The distribution archive for the software is simply named `ARCOSV_v` (`V` and `v` are the version major and minor numbers, such as 1_0), with a "`.tgz`" suffix for Linux-based PCs or "`.exe`" for Windows computers.

Install the utilities and files on the PC you plan to use for maintaining your robot's operating system and parameters by double-clicking the distribution software's onscreen icon or otherwise executing the self-extracting, self-installing package. For Linux, `uncompress` and `untar` the files. For example,

```
% tar –zxvf ARCOS1_0.tgz
```

The expanded archive creates an `ARCOS/` directory in the selected Windows or current Linux path and stores the ARCOS software within.

Starting ARCOScf

`ARCOScf` is a text-based console application which like `demo` is built with ARIA. It runs in two stages: Startup followed by interactive mode. When invoked, you may start `ARCOScf` with the various ARIA command-line options. With an X-terminal under Linux, for example, navigate to the ARCOS directory and invoke the program:

```
% cd /usr/local/ARCOS
% ./ARCOScf <options>
```

With Windows® PCs, you may double-click the `ARCOScf` icon to automatically open a console window and start the program without any options. To start up with command-line options, `Run` the program from the `Start` menu, or run `Command` from the `Start` menu, then navigate to the ARCOS directory and start `ARCOScf` with options.

For example (after invoking the MSDOS-like command window):

```
C:\> cd Program Files\ActivMedia Robotics\ARCOS
C:\Program Files\ActivMedia Robotics\ARCOS\> ARCOScf <options>
```

Normally (without any command-line arguments), `ARCOScf` starts up expecting to connect your PC's `COM1` or `/dev/ttyS0` serial port with your robot's controller. If successfully connected, the program automatically retrieves your robot's FLASH-stored operating parameters and enters interactive mode.

You may still operate many of ARCOScf's interactive features without a connection, such as maintain disk-based copies of your robot's operating parameters. And there is an interactive `connect` command that lets you establish a maintenance connection with your robot. See the next section for ARCOScf commands and operating features.

Include each of the selected ARCOScf's startup-mode options as a key letter with a dash ("-") prefix, followed by any required arguments, separated by spaces. For example, to start up ARCOScf and make a connection with a serial port other than the default `COM1` or `ttyS0`:

```
C:\Program Files\ActivMedia Robotics\ARCOS> ARCOScf –p COM3
```

Similarly, this Linux `xterm` command uploads a fresh copy of ARCOS to your robot's controller and then exits, much like the simple `dl_ARCOS1_0` program:

```
% ./ARCOScf -d ARCOS1_0.mot -n -b
```

Table 15. ARCOScf startup options

KEY	ARGUMENT	DESCRIPTION
-b	command arguments	Batch mode executes list of ARCOScf interactive-like mode commands with arguments, then exits automatically.
-d	motfile	Automatically upload ARCOS (.mot) motfile after connecting with the controller.
-l	paramsfile	Load the disk-stored params (.rop) file instead of the robot's copy
-n	none	Don't automatically connect with controller
-rp	serial-device	Uses specified serial port for connection
-rb	baudrate	Specify the port connection baud rate
-s	paramsfile	On exit from ARCOScf, automatically save the current parameter values to the named .rop paramsfile

CONFIGURING ARCOS PARAMETERS

Your *Activ*Media robot has several parameters stored in FLASH that ARCOS uses to configure its servers and auxiliary attachments and to uniquely identify your robot. For instance, the default maximum translation velocity is stored in the `TransVelMax` parameter. Its value takes effect when starting your robot or after resetting the controller, and may be changed temporarily by a client command. Use ARCOScf's batch or interactive modes to modify these operating parameters, and hence your robot's default operating characteristics.

Start up ARCOScf as described in the previous section. As discussed earlier, ARCOScf normally downloads the set of operating parameters from your robot's FLASH for your review and modification. Or you may load a disk-stored version of those parameters.

Interactive Commands

To operate ARCOScf in interactive mode, simply type a keyword at the command line. Some keywords affect the operation of ARCOScf, the status of the parameters file as a whole, or the connection between ARCOScf and your robot's controller. For instance, to review the list of current ARCOS FLASH variables, type `'v'` or '`view`' followed by a return (`Enter`). Each successive return will display additional variable keywords and current values. Similarly, type '`?`' or `'help'` to see a list of ARCOScf interactive commands.

Changing Parameters

Most keywords refer to the operating parameters themselves. Alone, a parameter's keyword simply asks ARCOScf to display the parameter's value. Provide an argument with the parameter keyword separated by a space to change its value. That value may be a string (no quotes or spaces) or a decimal or hexadecimal ("`0xN`") number. For example, to change the `watchdog` timeout to four seconds, type:

```
> watchdog 4000
```

or

```
> watchdog 0xfa0
```

See the respective control command and parameter Tables nearby for a full description of ARCOScf operation.

Table 16. ARCOScf control commands

COMMAND	DESCRIPTION
KEYWORD <value>	Alone, a keyword displays current, edited value. Add argument to change current value.
v or **view**	Display FLASH parameters.
u or **upload <motfile>**	Upload specified ARCOS image (.mot) file to the controller.
r or **restore** <paramsfile>	Restore parameters to values currently stored in FLASH or, if given, from a paramsfile (.rop) on disk
save <paramsfile>	Saves current edited values to FLASH or saves current edited values to paramsfile (.rop) on disk for later reference.
q or **quit**	Exits ARCOScf.
connect <portname**>**	Connects ARCOScf with controller through serial port (COM1 or /dev/ttyS0 default)
disconnect	Disconnects ARCOScf from your robot's controller
? or **help**	Displays these commands and descriptions.

SAVE YOUR WORK

While changing parameter values in ARCOScf interactive mode, you are editing a temporary copy; your changes are not put into effect in your robot's FLASH until you explicitly "`save`" them to the controller.

Also use the ARCOScf `save` command to save a copy of the parameters to a disk file for later upload. We strongly recommend that you save each version of your robot's parameter values to disk for later retrieval should your controller get damaged or its FLASH inadvertently erased. Default parameter files come with each ARCOS distribution, but it is tedious to reconstruct an individual robot's unique configuration.

Table 17. ARCOS FLASH configuration parameters with values for Pioneer 3–DX

KEYWORD	Type	Default	Description
TYPE	str	Pioneer	Identifies the robot type.
SUBTYPE	str	P3DX-SH	Identifies the *Activ*Media robot model.
NAME	str	not_set	Unique name for your robot. Maximum of 20 characters, no spaces.
SERNUM	str	factory	Serial number for the robot.
TICKSMM	int	132	Encoder ticks/mm: (ticks per rev x gear-ratio) / (wheel_diameter x PI)
REVCOUNT	int	18150	The number of differential encoder ticks for a 180 degree revolution of the robot.
DRIFTFACTOR	int	0	Value in 1/8192 increments to be added or subtracted from the left encoder ticks in order to compensate for tire differences.
BATTCONV	byte	0	0 if a 12V system; 1 if 24V
LOWBATTERY	int	115	In 1/10 volts; controller alarm activated when

			battery charge falls below this value.
SHUTDOWNVOLTS	int	110	In 1/10 volts; controller disconnects client and signals onboard PC to shutdown when battery charge falls below this value.
HOSTBAUD	byte	0	Baud rate for client-server HOST serial: 0=9.6k, 1=19.2k, 2=38.4k, 3=56.8k, 4=115.2k.
AUXBAUD1	byte	0	Baud rate for AUX serial port 1; see HostBaud
AUXBAUD2	byte	0	Baud rate for AUX serial port 2; see HostBaud
AUXBAUD3	byte	0	Baud rate for AUX serial port 3; see HostBaud
SIPCYCLE	byte	100	Server information packet cycle time in 1 ms increments. Default is classic 100 ms.
WATCHDOG	int	2000	Ms time before robot automatically stops if it has not received a command from a client. Restarts on restoration of connection.
SOUNDTOG	byte	1	0 disables the buzzer
SONARCYCLE	byte	40	Sonar cycle time in milliseconds
SONAR1	str	12345678	Ping sequence for sonar array #1. Up to 16 number characters 1-8; 0 to disable the array
SONAR2	str	0	Ping sequence for array #2. See sonar1 above
SONAR3	str	0	Ping sequence for array #3. See sonar1 above
SONAR4	str	0	Ping sequence for array #4. See sonar1 above
HASGYRO	byte	0	Set to 1 if you have the gyro accessory
HASBRAKES	byte	0	1 if your robot has brakes; 0 if not.
CHARGER	byte	0	Set to 1 if P3/PeopleBot or 2 if PowerBot autocharger mechanism and circuitry installed; otherwise 0
GRIPPER	byte	0	Set to 1 if DX/AT Gripper; 2 if Gripper on Performance PeopleBot
TCM2	byte	0	TCM2 module, if connected, specify AUX port 1,2, or 3
FRONTBUMPS	byte	0	Number of front bumper segments
REARBUMPS	byte	0	Number of rear bumper segments
INVERTBUMP	byte	0	0=none; 1=front; 2=rear; or 3=invert both; affects STALL bits in std. SIP only.
BUMPSTALL	byte	0	0=disable bump stall; 1=enable rear; 2=enable front; 3=enable both front and rear bump stalls
STALLVAL	int	300	Maximum PWM before stall. If > PwmMax, never.
STALLCOUNT	int	100	Ms time after a stall for recovery. Motors not engaged during this time.
PWMMAX	int	400	Maximum motor PWM (500 maximum).
ROTVELTOP	int	360	Maximum rotation velocity; deg/sec
TRANSVELTOP	int	1400	Maximum translation speed; mm/sec
ROTACCTOP	int	600	Maximum rotation (de)acceleration; deg/sec^2
TRANSACCTOP	int	1000	Maximum translation (de)acceleration; mm/sec^2
ROTVELMAX	int	200	Max rotation speed; deg/sec.
TRANSVELMAX	int	600	Max translation speed; mm/sec.
ROTACC	int	100	Rotation acceleration; deg/sec^2
ROTDECEL	int	100	Rotation deceleration; deg/sec^2
ROTKP	int	40	Proportional PID for rotation
ROTKV	int	20	Differential PID for rotation
ROTKI	int	0	Integral PID for rotation
TRANSACC	int	300	Translation acceleration; mm/sec^2
TRANSDECEL	int	300	Translation deceleration; mm/sec^2
TRANSKP	int	40	Proportional PID for translation
TRANSKV	int	30	Differential PID for translation
TRANSKI	int	0	Integral PID for translation
JOYSTICK	byte	1	Joystick type: 0=analog, 1=inductive
JOYVELMAX	int	1200	Joydrive maximum translation velocity
JOYRVELMAX	int	50	Joydrive maximum rotation velocity

PID PARAMETERS

The ARCOS configuration parameters include settings for the PID motor controls for translation and rotation of the robot. The translation values also are used for independent-wheel mode. The default values are for a lightly loaded robot. Experiment with different values to improve the performance of your robot in its current environment.

The Proportional PID (Kp) values control the responsiveness of your robot. Lower values make for a slower system; higher values make the robot "zippier", but can lead to overshoot and oscillation.

The Derivative PID (Kv) dampens oscillation and overshoot. Increasing values gives better control of oscillation and overshoot, but they also make the robot's movements more sluggish.

The Integral PID (Ki) adjusts residual error in turning and velocity. Higher values make the robot correct increasingly smaller errors between its desired and actual angular position and speed.

DRIFTFACTOR, TICKSMM, AND REVCOUNT

ARCOS uses the `ticksMM` and `revCount` parameters to convert your platform-independent speed and rotation commands—typically expressed in millimeters or degrees, respectively—into platform-dependent units. And it uses the `driftFactor` to compensate for tire difference.

The `ticksMM` value is the number of encoder pulses ("ticks") per millimeter of wheel rotation. The value is, of course, dependent upon the wheel encoder's resolution, the motor-to-wheel gear ratio, and the wheel's diameter.

Table 18. Some platform-dependent parameters

PARAMETER	VALUES		
	P3DX	P3AT	Perf PB
ENCODER TICKS/REV	500	100	500
GEAR RATIO	38.3	67.5	38.3
WHEEL DIAM (MM)	195	220	195
ENCODER TICKS/MM	132	138	132

The `revCount` value is the number of encoder ticks for one-half revolution of the robot. It depends on a number of factors, principally the length of the wheel base, which may change due to payload, tire wear, operating surface, and so on.

The `driftFactor` is a value in 1/8192 units that gets added or subtracted from the left-wheel encoder count at each motor cycle. In doing so, it compensates for tire difference and thereby straightens the robot's translation forward and backward.

`The ticksMM` and `revCount` parameters affect the conversion of your motion command arguments into platform-dependent values used by ARCOS. Unlike previous controller software, ARCOS also uses `ticksMM` and `revCount` to convert its internal measures into platform-independent position, heading, and velocity values reported back from the server, such as `X-Pos` and `Th`. Accordingly, you'll notice that the respective ARIA client parameters have many conversion factors like `DistConvFactor` set to 1.0.

STALLVAL AND STALLCOUNT

An ARCOS stall monitor maintains a running average of PWM values for each wheel over a 500 millisecond integration period. PWM values get added to the sum if the wheel speed is below 100 mm/sec. The average is then compared with the `stallval` FLASH value. If it exceeds that value, in other words the motors are being given lots of power but are barely moving if at all, a stall occurs. Once stalled, power is removed and the motors relax for the `stallwait` period, after which power gets reapplied.

BUMPERS

Use the `BumpStall` FLASH parameter to set the default for the robots behavior when its front and/or rear bumper gets triggered. Normally, `BumpStall` is engaged for both front and rear (default value of 0) bumpers. Reset it to 3 to disengage bump stalls altogether; 1 to trigger stalls only when the rear bumpers engage; or 2 for front bumps only.

You may over-ride the `BumpStall` FLASH default with the `bump_stall` client command number 44, although the command arguments are the reverse: enabling versus disabling the various bumper-stall combinations.

Your robot's `BumpStall` behavior reverts to the FLASH default on reset and up disconnection from the client.

ARCOS implements three FLASH parameters that specify states and numbers of front and rear bumper segments. Set the `FrontBumps` and `RearBumps` parameters to the number of bumper segments for the front and rear bumpers, respectively; or to 0 if you don't have a particular bumper. The number of segments is used to isolate the bumper bits, if any, so that a triggered bumper event is reported correctly in the `STALL` values of the standard SIP. Use the `InvertBump` FLASH parameter's value to invert those bumper-related STALL values, but not the hardware-related states reported in the `IOpac`.

The `FrontBumps` and `RearBumps` byte values also are reported near the end of the `CONFIGpac`.

If for any reason you remove a bumper from your robot, you MUST reset the associated `FrontBumps or RearBumps` FLASH value. Otherwise, the robot will stall incessantly or ARIA won't let you drive.

Chapter 8 Calibration & Maintenance

Your *Activ*Media robot is built to last a lifetime and requires little maintenance.

TIRE INFLATION

Maintain even tire inflation for proper navigation of your Pioneer 3-AT. We ship with each pneumatic tire inflated to 23 psi. If you change the inflation, remember to adjust the `driftFactor`, `ticksMM`, and `revCount` FLASH values.

CALIBRATING YOUR ROBOT

Your robot comes with FLASH parameters adjusted for operation on smooth, flat surfaces with its original payload. If you operate your robot on some different surface and with lighter or heavier loads, you probably will need to recalibrate many of its operating parameters, such as `driftFactor`, `revCount`, `ticksMM`, and the PIDs.

The ARIA `demo` program has two modes to help you do that. In '`p`'osition mode, `demo` displays current heading and position. Press the '`r`' key to reset these to 0 at any time. Press the right or left arrow key to have the robot rotate 90 degrees either clockwise or counterclockwise, respectively. The up and down arrow keys tell the robot to advance forward or backward one meter, respectively.

ARIA `demo`'s '`d`'irect mode lets you change the variety of operating parameters on-the-fly by sending the respective ARCOS client command number and value to be used during the current session. To replace the default values for permanent use by ARCOS, use ARCOScf.

Accordingly, to properly calibrate your robot, first use `ARCOScf` and record the default values for the respective parameters, such as for `driftFactor` and `revCount`. Then connect with the ARIA demo in position mode and move the robot. As accurately as possible, measure its actual motion and position and use direct mode to adjust the reported values for your robot's current configuration and operating environment.

For example, start with `driftFactor` since its value affects both `ticksMM` and `revCount`. Draw a line on the floor parallel to the robot's translation travel and drive the robot forward or back at least five meters. Adjust `driftFactor` (command #89) to minimize the robot's drift off that line.

Then drive the robot forward or back one or more meters and compare its actual translation distance you accurately measure with `demo`'s ARCOS-reported distance (x) in millimeters. Adjust `ticksMM` (command #93) so that the numbers match.

Likewise, rotate your robot and compare your measured rotation angle to the reported heading (`th`). Adjust `revCount` (the measure of differential encoder ticks to achieve 180-degrees rotation) accordingly (command #88).

Finally, drive the robot around and adjust its PID, velocity and acceleration values to achieve the desired performance for the operating configuration.

When you are satisfied that the robot moves and rotates the proper distances and headings, and drives with the proper performance, commit the new values into their related FLASH parameters in your robot with `ARCOScf` and don't forget to save a copy in a `.rop` file for later reference.

DRIVE LUBRICATION

Pioneer 3 drive motors and gearboxes are sealed and self-lubricating, so you need not fuss with grease or oil. An occasional drop or two of oil on the axle bushings between the wheels and the case won't hurt. And keep the axles clear of carpet or other strings that may wrap around and bind up your robot's drive.

BATTERIES

Lead-acid batteries like those in your *Activ*Media robot last longest when kept fully charged. In fact, severe discharge is harmful to the battery, so be careful not to operate the robot if the battery voltage falls below 11 VDC.

Changing Batteries

CAREFUL!
The Batteries slide in
TERMINALS LAST!

Except for those equipped with the automated docking/charging system, your Pioneer robot has a special battery harness and latched doors for easy access to the onboard batteries. Simply unlatch the rear door, swing it open and locate the one to three onboard batteries inside.

To remove a battery, simply grasp it and pull out. We provide a suction-cup tool to help. Spring-loaded contacts eliminate the need to detach any connecting wires.

Similarly, insert batteries by simply sliding each one into a battery box compartment. Load the batteries so that their weight gets distributed evenly across the platform: Center a single battery and place two batteries one on each side.

Hot-Swapping the Batteries

You may change the batteries on some of your *Activ*Media robots without disrupting operation of the onboard systems (except the motors, of course): Either connect the charger, which powers the robot's systems while you change the battery or batteries. Or, if you have two or three batteries, swap each with a freshly charged one individually, so that at least one battery is in place and providing the necessary power.

Charging the Batteries

If you have the standard charger accessory, insert it into a standard 110 or 220 (Europe/South America/Asia) VAC wall power receptacle. (Some users may require a special power adapter.) Locate the round plug at the end of the cable that is attached to the charger and insert it into the charge socket that is just below your robot's Main Power switch. The LEDs on the charger indicate charge status, as marked on its case.

It takes fewer than 12 hours—often just a few hours, depending on the level of discharge—to fully charge a battery using the accompanying charger (roughly, three hours per volt per battery). Although you may operate the robot while recharging, it restricts the robot's mobility.

Automated Docking/Charging System

The automated docking/charging system accessory optimally conditions power to charge the three 21-Ahr, 12 VDC lead-acid batteries (6 A charging current max) and provides sufficient power (up to 5A) for operation of all onboard systems.

The charging mechanism and onboard power conditioning circuitry can be retrofitted to all Pioneer 3 and some Pioneer 2 and PeopleBot robots. All require return to the factory.

Alternative Battery Chargers

The center post of the charger socket is the positive (+) side of the battery; the case is the negative (-) side. A diode protects against the wrong charger polarity. Nonetheless, if you choose to use an alternative battery charger, be sure to connect positive to positive and negative to negative from charger to robot.

An alternative AC to DC converter/battery charger should sustain at least 0.75A at 13.75 to 14 VDC per battery, and not more than 2-2.5 amperes per battery. The High-Speed Charger accessory, for example, is a four ampere charger and should be used with at least two of the standard batteries.

An alternative charger also should be voltage-and current-limited so that it cannot overcharge the batteries.

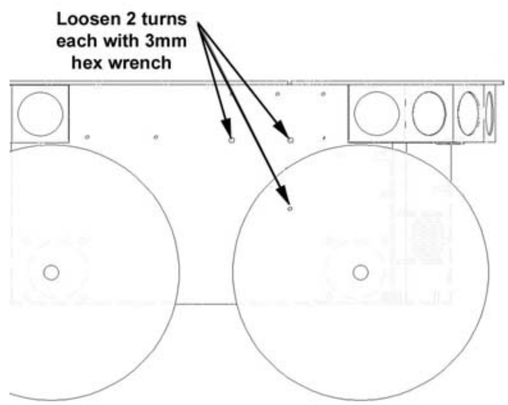

Figure 20. Loosen the AT drive belt retainer screws first.

TIGHTENING THE AT DRIVE BELT

Occasionally, particularly after heavy use, the Pioneer 3- or 2-AT drive belts that mechanically link the front and rear motors on each side will loosen and slip, resulting in a load popping noise. To start, use a 3mm hex key to loosen, but not remove, the three screws on the side of the robot near the front wheel. One screw is partly behind the wheel, so with our parts kit, we included a 3mm hex key with a shortened "L" section to fit behind the wheel.

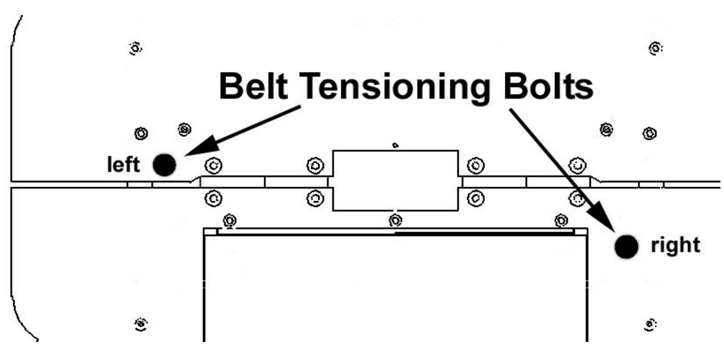

Figure 21. Locations of the P3-AT's belt-tensioning bolts

Remove the small plastic plug which is near the hinge on the top plate and near the edge by the wheel. Under it, you will see the head of a large hex bolt. This bolt tightens (clockwise) or loosens (counter-clockwise) the drive belt for that side of the robot. Turn it using a 5mm hex key probably not more than 1 full rotation. Avoid over tightening.

Test to make sure that it is tight enough by holding the wheel while running the self test. When adjusted satisfactorily, re-tighten the screws on the side and replace the plug.

GETTING INSIDE

We normally discourage you from opening up your robot. However, on occasion, you may need to get inside, for instance to access the user power connections on the Motor-Power board and attach your custom electronics. Or you may need to get to your onboard computer and its accessories.

Open the robot AT YOUR OWN RISK,
unless explicitly authorized by the factory.

REMOVE THE BATTERIES FIRST!

We describe here how to remove your robot's nose to get at the onboard computer. And we describe how to access the contents of the body of your Pioneer 3 robot.

Removing the Nose

The Pioneer 3-DX and –AT onboard computer sits just behind the robot's nose. And you may have to remove the nose to access the front sonar array's gain adjustment pot. Two screws hold the nose to the front sonar (or blank) array. The AT also has a screw at the bottom of the nose that attaches to the body; the DX's nose is hinged at the bottom.

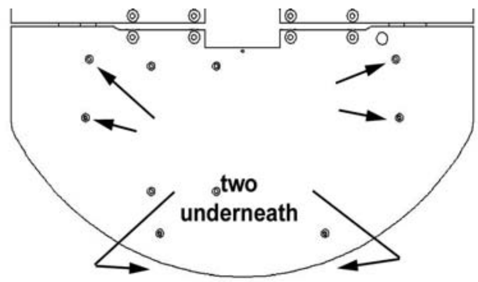

Figure 22. Remove indicated screws to access the front plate of Pioneer 3-DX and -AT robots.

Remove all nose retaining screws with the 3mm hex wrench supplied with your robot. Unlike earlier Pioneer 2 models, you do not have to remove the Gripper or the front bumper accessories.

Once loosened, the DX nose pivots down on a hinge. For the AT model, four pins along the nose's back edges guide it onto the front of the robot. Simply pry the nose out and away from the body.

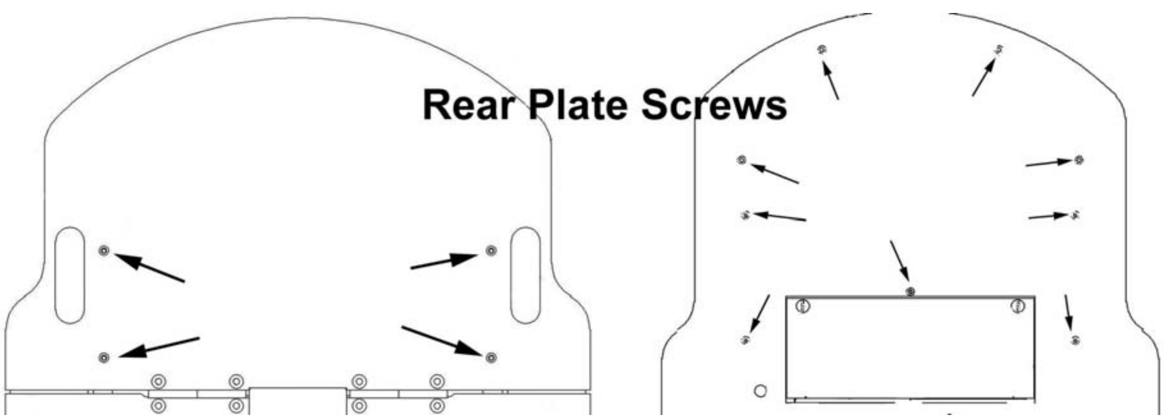

Figure 23. Remove indicated screws from Pioneer 3-DX or -AT rear deck to open plate.

Careful: The computer's hard-drive, fan, and speaker have attached wire harnesses that you need to relieve before completely detaching the nose from the body. We recommend unplugging the speaker wire and simply rotating the nose out of the way to access the onboard computer.

Opening the Deck

All Pioneer 3 robots have a center hinge in the deck which let you easily open and access internal components without completely removing the top plate. Simply remove the indicated 3mm screws shown in the Figures nearby from the section of the deck that you want to access. You may need to first remove any accessories that are bolted to the top plate through the indicated holes.

Remove the batteries BEFORE opening the robot.

FACTORY REPAIRS

If, after reading this manual, you're having *hardware* problems with your *Activ*Media robot and you're satisfied that it needs repair, contact us:

http://robots.activmedia.com/techsupport

Tell us your robot's SERIAL NUMBER

Describe the problem in as much detail as possible. Also include your **robot's serial number** (IMPORTANT!) as well as name, email and mail addresses, along with phone and fax numbers. Tell us when and how we can best contact you (we will assume email is the best manner, unless otherwise notified).

Use *Activ*Media Robotics' authorized parts *ONLY*;
warranty void otherwise.

We will try to resolve the problem through communication. If the robot must be returned to the factory for repair, obtain a shipping and repair authorization code and shipping details from us first.

We are not responsible for shipping damage or loss.

Appendix A

CONTROLLER PORTS & CONNECTIONS

This Appendix contains pinout and electrical specifications for the external and internal ports and connectors on the SH2-based controller for the Pioneer 3, PeopleBot, and PowerBot, including motor-power interface and User Control boards.

Note that layered connectors are numbered differently, depending on the socket type. IDC ones are odd and even layers; microfit connectors use successive-position numbering. See the Figures nearby for examples.

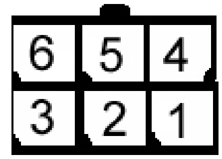

Figure 24. Mini- and micro-fit style connector numbering

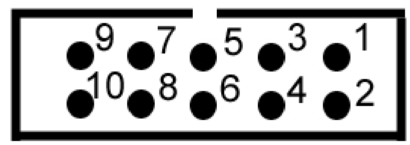

Figure 25. IDC-type connector

SH2 CONTROLLER

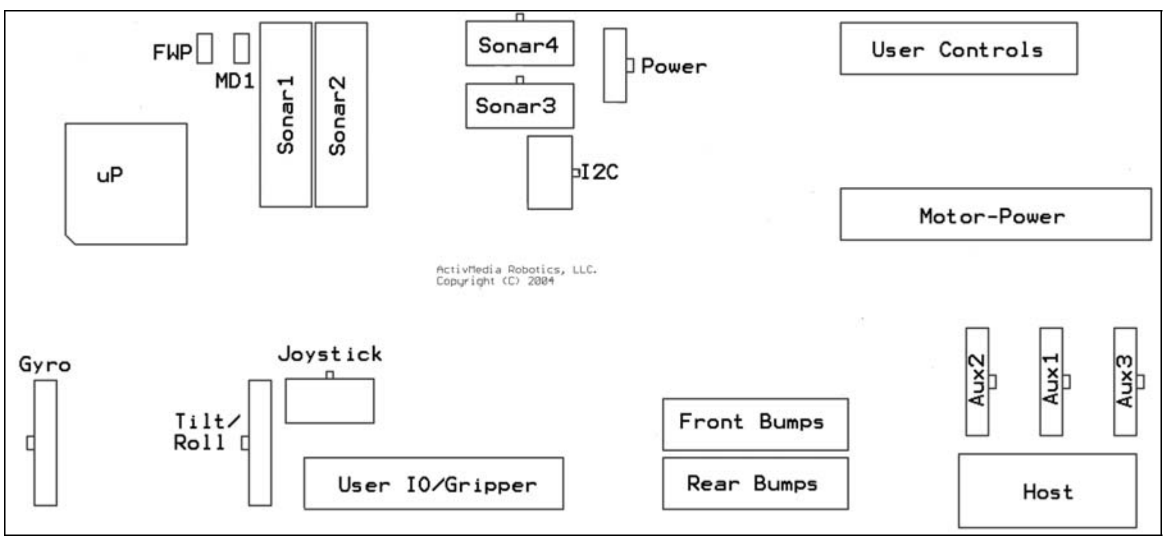

Figure 26. Connectors on ActivMedia's SH2-based controller

Power Connector

The power connector is a 3-pin microfit socket that delivers 12 and 5 VDC to the controller, including power ground.

Table 19. Controller Power Connector

PIN	DESCRIPTION
1	12 VDC (battery)
2	GND
3	5VDC

Serial Ports

One DSUB-9 and three 5-position microfit sockets provide the HOST and Aux1-Aux3 auxiliary serial ports for the SH2 controller. All are RS-232 compatible. The HOST port is shared on the User Control Panel's `SERIAL` port and is for ARCOS client-server and maintenance connections. The HOST serial connector also has signal lines for detecting an attached device (DTR pin 4) and for notifying the attached PC of low-power condition (DSR pin 6 and HRNG pin 9). The HOST serial connectors are wired DCE for direct connection (straight-through cable, not NULL-modem) to a standard PC serial port. See the nearby Tables for details.

The Aux1 through Aux3 serial ports are for RS232-compatible serial device connections, such as for the TCM2 accessory or any of several pan-tilt-zoom robotic systems.[24]

The serial ports operate at any of the common data rates: 9.6, 19.2, 38.4, 57.8, or 115.2 kilobits per second, and with eight data bits, one stop bit, no parity or hardware handshaking.

Table 20. HOST serial ports on controller and on User Control (*) (DSUB-9 socket)

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	nc			2	*TXD	Signal out
3	*RCV	Signal in		4	DTR	Input detects attached device and switches TxD and RxD into the uC
5	*GND	Common		6	*DSR	Output when controller powered
7	nc			8	nc	
9	†HRING	Output lowered to signal PC shutdown				

† Shared on User Control Board interface

Table 21. Aux1, Aux2, and Aux3 serial ports (5-pos microfit sockets)

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	DTR	Aux1 only		2	TXD	Signal out
3	RCV	Signal in		4	DSR	Output active
5	GND	Common				

User I/O, Gripper, Docking/Charging Interface

A 20-pin latching IDC socket on the controller provides the digital, analog, and power ports for user connections and for the Gripper and automated docking/charging accessories, if installed. Indicated ports (*) are shared on other connectors. Digital inputs are buffered and pulled high (digital 1); outputs are buffered and normally low (digital 0).

Table 22. User I/O – Gripper (20-pos latching IDC)

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	OD0	DIGOUT bit 0; Gripper enable		2	ID0	DIGIN bit 0; Paddles open limit
3	OD1	DIGOUT bit 1; Gripper direction		4	ID1	DIGIN bit 1; Lift limit
5	OD2	DIGOUT bit 2; Lift enable		6	ID2	DIGIN bit 2; Outer breakbeam IR
7	OD3	DIGOUT bit 3; Lift direction		8	ID3	DIGIN bit 3; Inner breakbeam IR
9	ID4	DIGIN bit 4; Left paddle contact		10	OD4	DIGOUT bit 4; Automated

[24] Note that on some original boards the Aux ports 1 and 2 were mislabeled. The Figure is correct for all boards.

					docking/charging “inhibit”
11	ID5	DIGIN bit 5; Right paddle contact	**12**	OD5	DIGOUT bit 5; Automated docking/charging “deploy”
13	ID6	DIGIN bit 6; Automated docking/charging ”power good”	**14**	OD6	DIGOUT bit 6; User only
15	ID7	DIGIN bit 7; Automated docking/charging ”overcharge”	**16**	OD7	DIGOUT bit 7; User only
17	*AN0	A/D port 0	**18**	Vcc	5VDC < 1A
19	Vpp	Battery 12VDC < 1A	**20**	Gnd	Signal/power common

Motors, Encoders, and IR Sensors

A 26-position latching IDC connector on the controller provides interface to the Motor-Power Board (Appendix B). Line descriptions also can be found in the following Motor-Power Interface section.

Table 23. Motors, encoders, and IRs interface (26-pos latching IDC)

PIN	SIGNAL	DESCRIPTION	PIN	SIGNAL	DESCRIPTION
1	LPWM	Left motors PWM	**2**	LDIR	Left motors direction
3	RPWM	Right motors PWM	**4**	RDIR	Right motors direction
5	MEN	Motors enable	**6**	LEA	Left encoder channel A
7	E-STOP	E-Stop detect input	**8**	REA	Right encoder channel A
9	RPWR	Aux1 power enable	**10**	REB	Right encoder channel B
11	APWR	Aux2 power enable	**12**	LEB	Left encoder channel B
13	CHRG	Charge port detect	**14**	IR6	IR input bit 6
15	IR7	IR input bit 7	**16**	IR4	IR input bit 4
17	IR5	IR input bit 5	**18**	IR2	IR input bit 2
19	IR3	IR input bit 3	**20**	IR0	IR input bit 0
21	IR1	IR input bit 1	**22**	VBAT	Battery voltage detect
23	Gnd	Signal common	**24**	AN1*	Analog input
25	Gnd	Signal common	**26**	AN2*	Analog input

* Board versions C and earlier pin 24 HOST RI and pin 26 ground.

Joystick Port

An 8-position microfit socket provides signal lines for connection to the inductive joystick accessory. Indicated lines (*) are shared on other connectors.

Table 24. Joystick connector (8-pos microfit)

PIN	SIGNAL	DESCRIPTION	PIN	SIGNAL	DESCRIPTION
1	Vcc	5 VDC	**2**	FB0	Fire button 0
3	*AN4	Turn Y-axis	**4**	AGND	Analog gnd
5	*AN3	Translate X-axis	**6**	FB1	Fire button 1
7	*AN0	Max speed control	**8**		nc

Bumper Ports

Two 10-position latching IDC connectors provide general-purpose digital inputs, typically used for the robot’s bumpers. All inputs are buffered and pulled high (digital 1).

Table 25. Bumper ports (10-pos latching IDC)

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	BP0	Bumper bit 0		2	BP1	Bumper bit 1
3	BP2	Bumper bit 2		4	BP3	Bumper bit 3
5	BP4	Bumper bit 4		6	BP5	Bumper bit 5
7	BP6	Bumper bit 6		8	BP7	Bumper bit 7
9	Gnd	Common		10	Gnd	Common

User Control Board Interface

A 16-position latching IDC connector provides interface with the User Control Panel board and functions. See description in a following section.

Table 26. User Control Panel interface

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	Vcc	5 VDC power		2	Vcc	5 VDC power
3	RST	RESET button		4	MOT	MOTORS button
5	RPWR	Radio power switch		6	APWR	Aux power switch
7				8	BZR	Buzzer PWM
9	PLED	Main power		10	SLED	Status
11	Vpp	Battery 12 VDC		12	Gnd	Signal/power common
13	Gnd	Signal/power common		14	HTXD	HOST serial transmit
15	HDSR	HOST serial enabled		16	HRCV	HOST serial receive

Sonar Connectors

Four connectors—two latching 10-pos IDC and two 10-pos microfits—provide signal and power for the four sonar arrays SONAR1 through SONAR4, respectively.

Table 27. Sonar ports

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	A0	disc address		2	A1	disc address
3	A2	disc address		4	BINH	inhibits return signal
5	INIT	starts sonar ping		6	VCC	5 VDC
7	VCC	5 VDC		8	SGND	Common
9	SGND	Common		10	ECHO	Goes high when echo threshold reached

Heading Correction Gyro

The heading-correction gyro accessory attaches directly with the controller through its respective 6-position microfit connector. Indicated lines (*) are shared on other connectors.

Table 28. Heading correction gyro connector

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	nc			2	VCC	5 VDC power
3	RATE	AN6		4	TEMP	AN7
5	AGND	Analog gnd		6	GND	Power ground

Tilt/Roll Sensor

Another six-position connector provides signal and power for an accelerometer-based tilt/roll accessory. Indicated lines (*) are shared on other connectors.

Table 29. Tilt/roll accessory connector

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	nc			**2**	VCC	5 VDC power
3	XAXIS	Pitch AN4		**4**	YAXIS	Roll AN5
5	AGND	Analog gnd		**6**	GND	Power gnd

I²C Interface

A six-position microfit contains the signals and power to interface with I²C bus-enabled devices, such as the 4-line x 20-character LCD accessory.

Table 30. I²C bus connector

PIN	SIGNAL	DESCRIPTION		PIN	SIGNAL	DESCRIPTION
1	SDA	Data		**2**	nc	
3	VCC	5 VDC power		**4**	SCL	Clock
5	CGND	Shield gnd		**6**	GND	Power/signal gnd

Appendix B

Motor-Power Distribution Board

*Activ*Media Robotics' original Pioneer 2 robots had two separate boards which interface with their respective controller to provide power for the motors as well as conditioned power and signal paths for the standard and accessory onboard electronics. Pioneer 3s have just a single Motor-Power Board. Consult *Appendix A* for controller and User Control Panel interface details.

MOTOR-POWER BOARD

The Motor-Power Board contains all the features of the two-board legacy system and lots more.

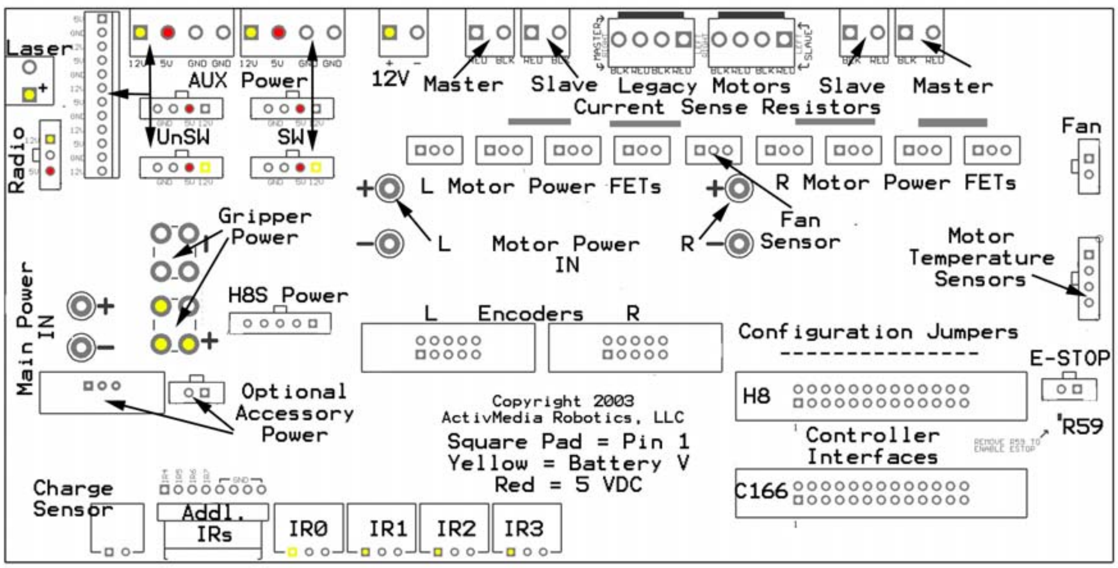

Figure 27. Motor-Power Board

Configuration for Current and Temperature Sensing

The motor drivers are configured to limit 10 amps per motor and to share the drivers with both motors on each side of the AT. Accordingly, there are two additional motor-current sense resistors added to the AT versus DX board: R3 and R26, as well as R1 and R2.

The new Motor-Power board also has a set of 0-ohm resistor pads that may be configured to engage the analog-to-digital input ports AN1 and AN2. By adding jumpers to R60 and R62, for example, the board is configured to sense motor current draw on AN1 and AN2, respectively.

Instead, by jumpering R77 and R78 and by attaching temperature sensors to two motors via the Motor Temperature Sensors connector, the AN1 and AN2 ports respectively may be used to protect against motor overheating. This configuration is currently enabled in the new ATs, but not yet supported in ARCOS.

Table 31. Motor Temperature Sensors Connector (4-pos microfit)

PIN	SIGNAL	DESCRIPTION
1	Vcc	5 VDC
2	T2	To AN2-based temp sensor circuit
3	T1	To AN1-based temp sensor circuit
4	GND	Signal/power common

Otherwise, a jumper across R76 connects the AN1 port to the fan sensor system that is attached to the FET heat sink. Note, too, that with or without attachment of AN1 via R76, but with the heat sensor in place, a fan may be attached and activated whenever the motor-driver FETs get overheated, as implemented in all new AT systems.

Controller Power and Interface

Individual 26-pos IDC connectors and cables provide signal for the H8S- and SH2-based controllers or for the legacy C166-based controllers. A separate cable and connector provides for the SH2 controller and sonar power. Power and signal are shared on the C166 controller connector.

Table 32. Power connector (5-pos microfit)

PIN	SIGNAL	DESCRIPTION
1	Vbat	Battery power
2	Gnd	Power common
3	Vcc	5 VDC for sonar
4	Vcc	5 VDC for sonar
5	nc	No connection

Radio, Auxiliary, and User Power Connectors

Various connectors provide conditioned 5 VDC @ 1.5A total and unconditioned battery power for the variety of accessories and custom user attachments. Some are AUX1 and AUX2 power switched from the User Control Panel. Use the 12-position latchlock connector for legacy installations. Otherwise, screw-down auxiliary user-power connectors make custom attachments easy. Four-position microfit connectors also provide AUX power for standard accessories.

Table 33. User Control Panel-switched AUX1 (formerly RADIO) power connector (3-pos microfit)

PIN	SIGNAL	DESCRIPTION
1	Vpp	AUX1 (formerly radio) switched battery 12 VDC
2	Gnd	Power common
3	Vcc	AUX1 (formerly radio) switched 5 VDC

Table 34. User Control Panel-switched and unswitched Aux power connectors (4-pos microfits and screw-down terminal blocks)

PIN	SIGNAL	DESCRIPTION
1	Vpp	Aux switched battery 12 VDC
2	Vcc	Aux switched 5 VDC
3	Gnd	Power common
4	Gnd	Power common

Table 35. User Power connector (12-pos latchlock; unswitched)

PIN	CONNECTION		PIN	CONNECTION
1	Vcc		**7**	Vcc
2	Gnd		**8**	Gnd
3	Vpp		**9**	Vpp
4	Vcc		**10**	Vcc
5	Gnd		**11**	Gnd
6	Vpp		**12**	Vpp

IR Signal and Power

Four connectors provide power and signal for fixed-range IR sensors. A separate connector provides signal path for an additional four IR sensors.

Table 36. IR power and signal connectors (3-pos microfits)

PIN	SIGNAL	DESCRIPTION
1	Vpp	Battery 12 VDC
2	IRn	Switching signal
3	Gnd	Power/signal ground

Table 37. Additional IR connector (8-pos latchlock 0.1 header)

PIN	SIGNAL	DESCRIPTION
1-4	IR4-7	IR signals
5-8	GND	Signal common

Appendix C

ETHERNET-TO-SERIAL DEVICE SETTINGS

The Ethernet-to-Serial device settings are made at the factory and stored in the device FLASH. Pressing and holding the `test` button for more than five seconds restores those factory settings.

<u>Server name</u>

AMR-EW-1

<u>Wireless</u>

SSID: WaveLAN Network

Mode: Infrastructure

Speed: 1 Mbps

<u>TCP/IP</u>

Address: 192.168.1.11 (.12, .13, … for successive units on a single order)

Gateway: 192.168.1.1

Subnet mask: 255.255.255.0

Boot protocol: static

<u>Serial Port (S1)</u>

Disable console mode

Disable flow control

<u>Serial port service (AMR-EW-1_S1)</u>

Disable queuing

TCP port 8101

Disable NetWare

<u>Misc Network</u>

Disable AppleTalk

Disable POP3

Disable SMTP

LAN IP SETTINGS

You need to modify your Ethernet-to-Serial device network setting in order to use it with your own LAN and Access Point. You have two ways to change those settings: From a serial console or from the device's support webpage.

Console mode:

1. Power off.
2. Attach a cross-over serial cable between your PC and the serial port on the device.
3. Start `minicom` (Linux), `HyperTerminal` (Windows®) or comparable serial console on your PC.
4. Serial settings are 115,200 baud, 8 bits, one stop, no parity, and hardware handshaking.
5. Hold in the test button and power the device.
6. Press the `Return` key to get the `Local>` prompt
7. Type:
    8. set ip address aa.bb.cc.dd (aa.bb.cc.dd is the device IP address)
    9. set ip router aa.bb.cc.dd (aa.bb.cc.dd is the gateway IP address)
    10. set ip subnet aa.bb.cc.dd (aa.bb.cc.dd is the subnet IP mask)
    11. save
    12. init
    13. exit
14. Restart the device

Webpage

1. Start web browser and access `http://192.168.1.xx` (last two digits depend on device's factory setting, typically .11)
2. The default password is `'access'`
3. Select `Configure TCP/IP`
4. Change the fields for the IP address, subnet mask, and gateway
5. Click `Submit`
6. Restart the device

Peer-to-Peer Networking

If you don't have an established LAN or access to the wireless network, you may operate your robot wirelessly directly from a PC that contains wireless Ethernet in what is known as peer-to-peer mode.

1. From console mode (see above), at the `Local>` prompt
2. Type:
    3. `set enet mode adhoc`
    4. `save`
    5. `init`
    6. `exit`
7. Restart the device

1. Or from the webpage (access as above)
2. Select `Configure WiFi`
3. Choose `Ad-hoc` from the `Mode` menu
4. `Submit`
5. Restart the device

Appendix D

SPECIFICATIONS

	Pioneer 3-DX	Pioneer 3-AT	Performance PeopleBot
Physical Characteristics			
Length (cm)	44.5	50.1	47
Width (cm)	39.3	49.3	38
Height (cm)	23.7	27.7	124
Clearance (cm)	6.0	8.4	3.5
Clearance bumpers (cm)	3.5	5.4	3.5
Weight (kg)	9	14	21
Payload (kg)	25	40	11
Power			
Batteries 12VDC lead-acid	3	3	3
Charge (wtt-hrs)	252	252	252
Run time (hrs)	8–10	4-6	8-10
with PC (hrs)	3-4	2-3	3-4
Recharge time hr/battery std charger	6	6	6
High-Speed (3 batteries)	2.4	2.4	2.4
Mobility			
Wheels	2 foam-filled	4 pneumatic	2 foam-filled
tread	knobby	wave	knobby
diam (mm)	195.3	220	195.3
width (mm)	47.4	75	50
Caster (mm)	75	na	75
Steering	Differential	Skid	Differential
Gear ratio	38.3:1	67:1	38.3:1
Swing (cm)	26.7	34	33
Turn (cm)	0	0	0
Translate speed max (mm/sec)	1,400	700	900
Rotate speed max (deg/sec)	300	140	150
Traversable step max (mm)	20	89	15
Traversable gap max (mm)	89	127	50
Traversable slope max (grade)	25%	40%	11%
Traversable terrains	Wheel-chair accessible	Unconsolidated. ***No carpets***!	Wheel-chair accessible

Sensors	Pioneer 3-DX	Pioneer 3-AT	Performance PeopleBot
Sonar Front Array (one each side, six forward @ 20° intervals)	8	8 optional	8
Rear Sonar Array (one each side, six rear @ 20° intervals)	8 optional	8 optional	8
Top Deck Sonar (one each side, six forward @ 20° intervals)	na	na	8
Rear Deck Sonar (one each side, six forward @ 20° intervals)	na	na	8 optional
Encoders (2 ea) counts/rev	76,600	34,000	76,600
counts/mm	128	49	128
counts/rotation	33,500	22,500	33,500
Bumpers	Optional	Optional	Standard

Controls and Ports

Main Power	Standard	Standard	Standard
Charge	Standard	Standard	Standard
Joydrive	Optional	Standard	Standard
Motor Stop	Optional	Standard	Standard
Aux1 Power	5 & 12 VDC sw'd	5 & 12 VDC sw'd	5 & 12 VDC sw'd
Aux2 Power	5 & 12 VDC sw'd	5 & 12 VDC sw'd	5 & 12 VDC sw'd
System Serial	Standard	Standard	Standard
Motors	Standard	Standard	Standard
Controller Reset	Standard	Standard	Standard

Warranty & Liabilities

Your *Activ*Media robot is fully warranted against defective parts or assembly for one year after it is shipped to you from the factory. Accessories are warranted for 90 days. Use only ActivMedia-authorized parts, or warranty void. This warranty also explicitly *does not include* damage from shipping or from abuse or inappropriate operation, such as if the robot is allowed to tumble or fall off a ledge, or if it is overloaded with heavy objects.

The developers, marketers, and manufacturers of *Activ*Media Robotics products shall bear no liabilities for operation and use of the robot or any accompanying software except that covered by the warranty and period. The developers, marketers, or manufacturers shall not be held responsible for any injury to persons or property involving *Activ*Media Robotics products in any way. They shall bear no responsibilities or liabilities for any operation or application of the robot, or for support of any of those activities. And under no circumstances will the developers, marketers, or manufacturers of *Activ*Media Robotics product take responsibility for support of any special or custom modification to *Activ*Media robots or their software.

ActivMedia Robotics
MobileROBOTS.com

19 Columbia Drive
Amherst, NH 03031
(603) 881-7960
(603) 881-3818 fax
http://www.MobileRobots.