User's Guide for the Pointer / Power Pulley

Please read this guide carefully before using the Pointer / Power Pulley. Pay particular attention to the safety guidelines on page 2.

The CBL, CBL2 and LabPro interfaces are a tremendous resource for math and science classes, but the task of programming them can be overwhelming. Most people settle for using a few pre-packaged programs as their basis for collecting data through the CBL. Data collection activities certainly ARE a great way to get students actively involved, but they fail to realize the full potential of the equipment. Exclusive emphasis on data collection can also cause students to miss out on seeing how mathematics and science are applied in engineering-style environments. The programming methods described here let students and teachers take control of real equipment that functions much like industrial robots.

The method can help tremendously in transforming math and science from something that students *must* learn into something they *want* to learn. Not very many students come into our classes anxious to learn math and science. With a calculator-controlled robot, pointer or other action-oriented equipment, math and science become more fun, more practical and more effective.

To let students get a quick start on programming, we put the technical details into pre-packaged programming modules. All of our modules are in TI-Basic, so students and teachers are welcome to dig in for themselves to look at the details. Most teachers prefer, however, to have their students focus on the math and science instead of the technical details of programming for the CBL2 or LabPro. With the modules described here, complex actions can be achieved using programs that are simple enough to be entered directly from the calculator keyboard.

```
POISHINE

ú25üA
prgmPOIPIV
5üT
prgmPOISWF
25üA
prgmPOIPIV
```

The 6-line program at right, for example, makes the POINTER pivot clockwise 25°, turn on its 12-V output for 5 seconds, and pivot counterclockwise 25°. POIPIV and the other program modules require a value (A for angle, T for time, etc.) and then operate on that value.

Purists insist that a true robot must not only move but that it must also sense its environment and be able to do something. The Pointer gains the capability to sense its environment by attaching probes and it gains the ability to "do something" through its switch or its "servo motor." Servo motors are low-cost hobby motors that can be set to any desired position between about -75° and +75°. Raising a flag or tipping out a ping pong ball, for example is as simple as setting the angle A and calling the module POIPUT. Students don't need to worry about the details of how the signal makes the servo motor work, unless they or their teacher wants to make that part of the learning experience. The system also provides many different opportunities to challenge students to use functions, for example by asking them to raise a flag in proportion to the light intensity that a sensor finds in different directions.

While we strive to make programming simple enough that students can do it directly from the calculator keypad, it is important to note that more complex programming is far more conveniently done using a computer. The computer also makes it far more convenient for teachers and for students to exchange programs—something we strive to encourage via our web site. Software and connector cables to connect your computer to your TI calculators are available from Texas Instruments.

Important Cautions:

- **Do not use the Pointer / Power Pulley for any purpose except instructional activities in which it controls the rotational motion of light-weight rods, pulleys or similar objects.**
- **Use only the DC wall adapter provided with the Pointer / Power Pulley.**
- **Always protect the Pointer / Power Pulley from water and other liquids and from extreme temperatures. Do not use the Pointer / Power Pulley in the presence of flammable vapors.**
- **Never block or interfere with the flow of air through the cooling vents on the side and bottom of the Pointer / Power Pulley.**
- **Always supervise students using the Pointer / Power Pulley. In any activities that might pose a danger to eyes, always ensure that all students wear proper eye protection.**
- **Never allow anything to be attached to the 12-V DC output that might draw more than 200 mA of current.**
- **Never operate the Pointer / Power Pulley with devices attached simultaneously to the Servo Motor port and the 12-V Output. These are alternative connection points and any signal that activates one also activates the other. A servo motor in particular can be damaged by signals that are normal for the 12-V output.**
- **Be familiar with and adhere to professional safety guidelines for inquiry activities in education. For more information, see *Investigating Safely: A Guide for High School Teachers* published by the National Science Teachers Association. (Texley, Kwan &Summers, 2004, NSTA Press, Arlington, VA).**

This material is based in part upon work supported by the National Science Foundation's Advanced Technological Education program under Grant No. DUE-0202202. The material is now being expanded and disseminated by

Math Machines, Ltd.
1014 Merrywood Drive
Englewood, Ohio 45322
USA

Specifications for the Pointer Program Modules

Basic Pointer Program Modules:

POIPIV	***Pointer Pivot Angle, A***
Required input	A (Pointer's angular displacement in degrees) Positive values give counterclockwise (CCW) rotation; negative values of A give clockwise (CW) rotation.
Optional input	U (Pointer's angular speed in deg/s) Only the absolute value of U is used in this program module.
	W If W = 1, the 12-V output is on during the motion. If W = 4, no changes are made to the calculator screen. If W = 5, both options above are implemented.
Physical Outcome	Pointer pivots CW or CCW by angle, A.
POIPIV2 functions in the same way for a pointer connected to the LabPro’s DIG/SONIC port 2	
POIPIVF	***Pointer Pivot For Time, T***
Required input	T (time in seconds)
Optional input	U (Pointer's angular velocity in deg/s) Positive values of U give CCW rotation; negative values of U give CW rotation.
	W (options for 12-V output &screen display--see above)
Physical Outcome	Pointer pivots for time, T
POIPIVF2 functions in the same way for a pointer connected to the LabPro’s DIG/SONIC port 2	
POIPIVST	***Pointer Pivot For Steps, S***
Required input	S (number of steps)
Optional input	U (Pointer's angular velocity in deg/s) Positive values of U give CCW rotation; negative values of U give CW rotation.
	W (options for 12-V output &screen display--see above)
Physical Outcome	Pointer pivots for S steps
POIPIVW	***Pointer Pivot While* Y_7 is not Zero**
Required input	Function in Y_7
Optional inputs	U (pointer's angular velocity in deg/s)
	W (options for 12-V output &screen display--see above)
Output variables	E (final sensor value from analog channel 1)
	H (angular displacement in degrees)
Physical Outcome	Pointer pivots CCW (or CW if U<0) while measuring the input on analog channel 1until Y_7 becomes zero or the user presses any key
POIPIVW2 functions in the same way for a pointer connected to the LabPro’s DIG/SONIC port 2	
POIPUT	***Pointer Put Servo at Position, A***
Required input	A (servo's angular position in degrees)
Optional input	
Physical Outcome	Servo moves to angular position, A (The Switch is also activated in a pulsed mode.)
POIPUT2 functions in the same way for a pointer connected to the LabPro’s DIG/SONIC port 2	

POISWF	***Pointer Activate Switch for Time, T***
Required input	T (time in seconds)
Optional input	
Physical Outcome	Terminals deliver 12 V (200 mA max) for T seconds **Do not use this program with a servo motor attached, since it also delivers a sustained signal to the servo port which might damage the servo motor. If you don't plan to use the switch, delete the program.**
POISWF2 functions in the same way for a pointer connected to the LabPro's DIG/SONIC port 2	
POISETUP	***Pointer Set Up***
Required input	None except through inputs while the program is running
Optional input	None
Physical Outcome	The Pointer turns under direct user control
Software Outcome	The list $_L$POI is created (or modified if already present). The program can then be deleted and the list itself copied from one calculator to another.

Other Related Program Modules:

DCUINIT (from Vernier Software)	***Verify and Initialize the Connection between the Calculator and the CBL, CBL2, or LabPro***
Required input	None
Optional input	None
Physical outcome	Interface is tested and set for sustained output.
Software outcome	A warning appears on the calculator screen if the calculator is not correctly connected, and the program waits for the user to correct the problem.
READ1	***Read the Value from the Sensor in Analog CH 1***
Required input	None
Optional input	None
Physical outcome	Channel 1 is activated to autoidentify the sensor and take a series of 5 readings at 0.05 s intervals.
Software outcome	Variable E is set to the average value returned from the sensor
READ2	***Read the Value from the Sensor in Analog CH 2***
Required input	None
Optional input	None
Physical outcome	See Read1
Software outcome	Variable F is set to the average sensor value.
READ3	***Read the Value from the Sensor in Analog CH 3***
Required input	None
Optional input	None
Physical outcome	See Read 1
Software outcome	Variable G is set to the average sensor value.
READ4 (LabPro Only)	***Read the Value from the Sensor in Analog CH 4***
Required input	None
Optional input	None
Physical outcome	See Read 1
Software outcome	Variable H is set to the average sensor value.

Variables Used or Designated:

Variables	Use	Changed or deleted in Pointer programs?
A	Angle (degrees)	No
B	Short-term internal use	Yes
C	Short-term internal use	Yes
D	Distance (centimeters)	No
E	Ch1 Sensor	Yes
F	Ch2 Sensor	Yes
G	Ch3 Sensor	Yes
H	Ch4 Sensor (LabPro only) or Displacement	Yes
I	Counter	Yes
J	Counter	Yes
K	Key	Yes
L	*Reserved for higher level programs*	*No*
M	*Reserved for higher level programs*	*No*
N	*Reserved for higher level programs*	*No*
O	*Reserved for higher level programs*	*No*
P	*Reserved for higher level programs*	*No*
Q	Hardware specific data • EDE1200: auto speed settings • Pointer: position within the 4-step cycle for precise positioning	Yes
R	Pulse width	Yes
S	Step Counter	Yes
T	Time (s)	No
U	Angular Velocity (degrees/second) (POI's pivot speed in POI programs, robot pivot speed in SAM programs)	Yes, if outside range
V	Velocity (cm/s)	Yes, if outside range
W	Flag for switches on during movement	No
X	*Reserved for higher level programs*	*No*
Y	*Reserved for higher level programs*	*No*
Z	*Reserved for higher level programs*	*No*
θ	*Reserved for higher level programs*	*No*

Lists Used or Designated:

List	Use	Changed or deleted in Pointer programs?
${}_{L}$A	Angles	No
${}_{L}$B	Short-term use	No
${}_{L}$C	Short-term use	No
${}_{L}$D	Distances	No
${}_{L}$E	Ch1 Sensor	Yes
${}_{L}$F	Ch2 Sensor	Yes
${}_{L}$G	Ch3 Sensor	Yes
${}_{L}$H	Ch4 Sensor (LabPro only) and temporary use	Yes
${}_{L}$I	List of numbers	No
${}_{L}$R	Pulse Widths for Servo	Yes
${}_{L}$S	List of signals	No
${}_{L}$T	Times	Yes
${}_{L}$POI	Interface and Pointer characteristics (see below)	No, except POISET
${}_{L}$SSLB	Setup list for primary board	Yes
${}_{L}$SSOFF	Zero all outputs	Yes

${}_{L}$POI(1)	interface Code (1 = CBL, 2 = CBL2/LabPro, 3=LabPro)
${}_{L}$POI(2)	PCB Code (1 = SVP A 1)
${}_{L}$POI(3)	Step size (degrees)
${}_{L}$POI(4)	Default motor speed for pivots (degrees/second)
${}_{L}$POI(5)	Maximum motor speed (degrees/second)

The sample program below was developed for an activity on the “law of reflection.” Students were given the task of orienting a mirror so it would reflect a light beam 60° left of its original orientation and then sweep the beam in exactly 40 seconds to a point 60° right of its original orientation.

REFLECT

```
prgmDCUINIT
Output(1,1,"MATH MACHINES")
Output(2,2,"TI-83 SOFTWARE")
Output(3,3,"** REFLECT **")
Output(5,1,"CONTROL A MIRROR")
Output(8,1,"<ENTER> TO START")
Pause
ClrHome

Input "ANGLE A (DEG)?  ",X
Input "ANGLE B (DEG)?  ",Y
Input "SPEED (REV/MIN?)",Z
ClrHome
40üU
XüA
prgmPOIPIV
Z*360/60üU
YüA
prgmPOIPIV
Output(8,1,"<ENTER> TO ZERO")
Pause
ú(X+Y)üA
40üU
prgmPOIPIV
ClrHome
```