

ZOOM ZOOM

By: Bronwyn Ryan
and Chanel Tillman

Introduction

Many mathematics teachers strive to make connections between the topics covered in their classes. The relationship between distance, speed, and time ($d = r \cdot t$) is covered at various depths in both mathematics classes and sciences classes at a variety of grade levels. Additionally, it is an excellent real world example of direct variation. This project employed mechatronics to re-program toy slot cars so that the vehicles moved at a constant speed. Additionally, infrared sensors were used to monitor the time it takes for the vehicles to move down the tracks. The resulting apparatus will be used by students to explore these relationships.

Background

The equation distance equals speed multiplied by time ($d = r \cdot t$) is a real world example of a direct variation. When two variable quantities have a constant (unchanged) ratio, their relationship is called a direct variation. It is said that one variable "varies directly" as the other. The constant ratio is called the constant of variation. The formula for direct variation is $y = kx$, where k is the constant of variation. So, when solving for k : $k = \frac{y}{x}$.

The slope of the line graphed in the lab is not only the constant of variation, but also represents the speed of the car. Thus, the two concepts are interrelated. Furthermore, the lab encourages students to use tables and graphs to investigate. The tables and graphs organize the data so that the slope/constant of variation can be determined and further interpolations can be made.

Rationale

This project lends itself to a "hands-on" lab that provides students the opportunity to explore the relationship between distance, speed, and time and relate it to direct variation. Students will document their data through tables and graphs, and can then make interpolations. Moreover, with the cars set to constant speeds, the tracks can be used to model motion problems. Providing a mobile visual for students, captures interest and may

help assuage frustration in recognizing the six different types of motion problems.

Equipment List

- ❖ 1/32 Carrera Slot Car Race Track Set "Checkered Flag Run"
- ❖ 6 Parallax Infrared Sensors (transmitters and receivers)
- ❖ 2 Servo Motors
- ❖ Several Jumper Wires
- ❖ 2 20 - 25 Ohms Resistors
- ❖ 6 220 Ohms Resistors
- ❖ 6 1000 Ohms Resistors
- ❖ 4 15 Ohms Resistors
- ❖ Piping
- ❖ BS2
- ❖ Breadboard

Experimental Design

The slot cars we purchased are controlled by a hand-held remote. The remote has a large push-button. The further down the button is held, the faster the car goes. Upon taking the hand-held controller apart, we

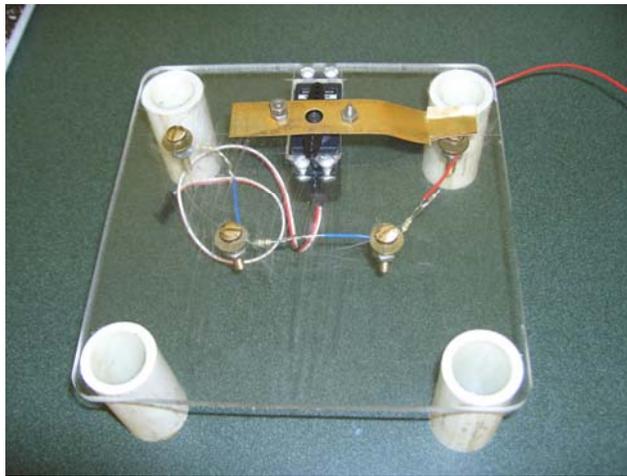


discovered that the inside consisted of two approximately two inch long resistors. The resistor on the left was connected to the live wires. The resistor on the right was connected to ground. The push-button has a shaft with a small metal piece attached. As the push button is held down, the metal piece slides down the resistors, making a connection between the two

sides. When the metal piece connects the two resistors, a circuit is completed, thus the slot car moves down the track.

However, in order to relate $d = r \cdot t$ with direct variation, this lab requires that the cars move at a constant speed. Thus, it is unreliable to simply hold the push-button. We simulated the wiring of the hand-held remote, but we also enlarged it and made a pseudo-potentiometer. Instead of employing a push-button, we used a servo-motor to make a connection between the ground and live wires.

The servomotor will have a metal handle, which is grounded, attached on its propellers. A computer program will cause the servomotor to rotate and the metal handle will make a connection at one of four critical points, corresponding to one of four possible speeds. Between these points (i.e. screws), are resistors. The resistors in series account for an increase in resistance, thus a decrease in speed. We narrowed the range of speeds from a continuum to one of four options. The first option is the fastest and does not use any resistance. The second speed used 15-25 ohms of resistance, the third speed used an additional 15 ohms of resistance, and the fourth, the slowest speed, used an additional 30 ohms of resistance.



As the cars zoom down their straight-away track, infrared sensors were placed approximately every 57 centimeters, are told by the computer program to emit a frequency. The infrared detector senses the frequency unless an object, such as our car, blocks the frequency. These sensors detect when the car zooms by each interval, therefore we can use programming to calculate the amount of time that it took the car to travel to each infrared transmitter/detector pair.

Results

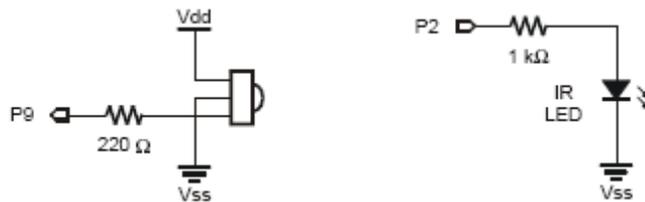
After much trouble shooting we were able to successfully have the two separate components (servo motors and infrared sensors) of this project work together. Thus, students will be able to use this apparatus to help them develop a deeper understanding of the relationship between distance, time, and rate, as well as direct variation.

The first challenge in the servo-motor component involved discovering a design in which the servo-motor would be responsible for various speeds by making the connection from live wires to ground. The next challenge was to

program the servo motors to respond consistently to the input commands on the computer that would determine a constant speed. (The different speeds were represented by 1 - 4: super-fast, fast, medium speed, and slow respectively). Once this program was successfully written and executed we then had to work with our sensors.

Initially, we had planned to use the PING))) Ultra Sonic sensor, which is fantastic for detecting distance. However, as with many tools, it was a failure. Since the PING))) emits frequency in a cone shape, it was going crazy, detecting all sorts of objects around the room. So we switched to infrared sensors.

The concept of wiring the infrared sensors is fairly simple. The transmitter must be positioned towards the receiver so communication is possible. The receiver needs to be connected to a pin (which enables computer programming to manipulate it), Vdd (voltage), and Vss (ground). The transmitter needs to be connected with a pin and ground (see circuit schematic below). Since the straight-away track is over three meters long, a vast amount of jumper wires had to be soldered to the IR sensors and the wiring had to be braided, labeled, and organized. We were able to write a



program that involved six different sensors which would give us the total time the car had been traveling up to each particular sensor.

One unforeseen obstacle was synchronizing the program used to maneuver the servo-motor to establish a speed with the program to track the cars time using the infrared sensors. The harmonization is still not perfect. Sometimes, as the program runs one of two possible errors may occur. The first gaffe maybe that the car is so fast the program has not had an opportunity to begin pulsing out to the first infrared transmitter. The second mishap could be that the car passes through the first infrared sensor while it is pulsing out to other infrared sensors, thus rendering the entire counter within the programming useless. However, providing that the car is detected by the first infrared sensor, the system does work!

Computer Program

```
' {$STAMP BS2}
```

```
' {$PBASIC 2.5}
```

We declared the variables below. Each infrared sensor required a counter and the detector displays either a 0 or 1 depending on whether or not it the car is blocking the frequency. StopA was a variable made to tell the program to exit when the last infrared sensor detected the car. The last counter variable was used to keep track of the time.

```
timeCounter VAR Word
```

```
counterA VAR Word
```

```
counterB VAR Word
```

```
counterC VAR Word
```

```
counterD VAR Word
```

```
counterE VAR Word
```

```
counterF VAR Word
```

```
irDetectA VAR Bit
```

```
irDetectB VAR Bit
```

```
irDetectC VAR Bit
```

```
irDetectD VAR Bit
```

```
irDetectE VAR Bit
```

```
irDetectF VAR Bit
```

```
value VAR Nib
```

```
stopA VAR Bit
```

```
counterA=0
```

```
counter VAR Word
```

The debug commands tell the participant how to interact with the program. Debugin tells the computer program to wait for a response from the participant.

```
DEBUG CLS, "Choose a speed for your race car!", CR
```

```
DEBUG "Type in a number 1 - 4 for the RIGHT track.", CR
```

```
DEBUG "1 is SUPER FAST! 4 is the slooooooowest speed.", CR
```

```
DEBUG "Type in a number 5 - 8 for the LEFT track.", CR
```

```
DEBUG "5 is SUPER FAST! 8 is the slooooooowest speed.", CR
```

```
DEBUGIN DEC value
```

The values, 1 - 8, correspond to the four different speed settings on the left and right tracks and to the respective servo-motors.

```
IF value = 1 THEN GOTO case1
```

```
IF value = 2 THEN GOTO case2
```

```
IF value = 3 THEN GOTO case3
```

```
IF value = 4 THEN GOTO case4
```

```
IF value = 5 THEN GOTO case5
IF value = 6 THEN GOTO case6
IF value = 7 THEN GOTO case7
IF value = 8 THEN GOTO case8
```

STOP

The PULSOUT command sends current to the specified pin. Pin 14 controls one servo motor, which is wired to the left track. Pin 15 controls the other servo motor, which is wired to the right track. The GOTO commands send the computer to the part of the program that controls the infrared sensors.

case8:

```
DEBUG "BLAST OFF! WE LOVE TOUCHING PIN 4", CR, CR
FOR counter = 1 TO 25
PULSOUT 14, 260
PAUSE 20
NEXT
GOTO Detect
```

case7:

```
DEBUG "Come on baby light my fire!!!! on PIN 3", CR, CR
FOR counter = 1 TO 25
PULSOUT 14, 545
PAUSE 20
NEXT
GOTO Detect
```

case6:

```
DEBUG "I am soooo excited! we are all over PIN 2", CR, CR
FOR counter = 1 TO 25
PULSOUT 14, 870
PAUSE 20
NEXT
GOTO Detect
```

case5:

```
DEBUG "RAISE YOUR HANDS UP AND SHOUT! we love touching pin 1"
FOR counter = 1 TO 25
PULSOUT 14, 1190
PAUSE 20
```

NEXT
GOTO Detect

case4:
DEBUG "BLAST OFF! WE LOVE TOUCHING PIN 4", CR, CR
FOR counter = 1 TO 25
PULSOUT 15, 240
PAUSE 20
NEXT
GOTO Detect

case3:
DEBUG "come on baby light my fire!!!! on PIN 3", CR, CR
FOR counter = 1 TO 25
PULSOUT 15, 545
PAUSE 20
NEXT
GOTO Detect

case2:
DEBUG "i am soooo excited! we are all over PIN 2", CR, CR
FOR counter = 1 TO 25
PULSOUT 15, 870
PAUSE 20
NEXT
GOTO Detect

case1:
DEBUG "RAISE YOUR HANDS UP AND SHOUT! we love touching pin 1"
FOR counter = 1 TO 25
PULSOUT 15, 1170
PAUSE 20
NEXT
GOTO Detect

The detect component of the program commands the infrared LEDs to emit a frequency of 38,500 hertz. Each infrared is connected to its own pin. The infrared LEDs communicate with a receiver. Each receiver is assigned its own variable, for example, irDetectC is located at pin four.

Detect:

DO

FREQOUT 1, 1, 38500

irDetectA = IN0

IF (irDetectA =1) THEN

DO

PAUSE 100

counterA = counterA+1

The pause allows the computer program to add one to the counter every tenth of a second.

FREQOUT 3, 1, 38500

irDetectB = IN2

FREQOUT 5, 1, 38500

irDetectC = IN4

FREQOUT 7, 1, 38500

irDetectD = IN6

FREQOUT 10, 1, 38500

irDetectE = IN9

FREQOUT 12, 1, 38500

irDetectF = IN11

The series of IF... THEN... ELSE IF statements tells the computer how to monitor the cars progress as it passes through the sensors.

IF (irDetectB=1) THEN

counterB=counterA

ELSEIF (irDetectC=1) THEN

counterC=counterA

ELSEIF (irDetectD=1) THEN

counterD=counterA

ELSEIF (irDetectE=1) THEN

counterE=counterA

ELSEIF (irDetectF=1) THEN

counterF=counterA

stopA=irDetectF

```
EXIT
ENDIF
LOOP
ENDIF
```

These debug commands display the time it took for the car to pass through each infrared sensor in tenths of a second.

display:

```
DEBUG CLS,HOME, "constant timer: ", DEC5 counterA,CR,CR
DEBUG "Infrared B ", DEC5 counterB, " tenths of a second", CR, CR
DEBUG "Infrared C ", DEC5 counterC, " tenths of a second", CR, CR
DEBUG "Infrared D ", DEC5 counterD, " tenths of a second", CR, CR
DEBUG "Infrared E ", DEC5 counterE, " tenths of a second", CR, CR
DEBUG "Infrared F ", DEC5 counterF, " tenths of a second", CR, CR
IF (stopA=1)THEN EXIT
LOOP
```

Future Works

The basic design of our system can be improved upon. One suggestion for future research would be to ensure that the first infrared sensor always detects the car. Moreover, it may be interesting to devise a system that detects the distance, rather than time. Distance would be more desirable since intuitively; one would expect time to be the independent variable and distance to be the dependent variable.

Reference

BASIC Stamp Syntax and Reference Manual Version 2.2

Lindsay, Andrew. [What's a Microcontroller?](#)

www.parallax.com

www.societyofrobots.com

www.regentsprep.org