Theremin Music with Evo

By Richard Born
Associate Professor Emeritus
Northern Illinois University
rborn2@niu.edu

Introduction

A ***theremin***, named after its Russian inventor in the early 1900's, is an electronic musical instrument that is controlled without any contact by the musician. Volume is controlled by moving one hand near one antenna, while pitch is controlled by moving the other hand near a second antenna. The sound is generated by a pair of high-frequency oscillators.

In this lesson, using OzoBlockly, you will turn Evo into a very simple theremin whose pitch is controlled by moving your hand and fingers near Evo's front proximity sensors. In agreement with a traditional theremin, the closer the hand to Evo, the higher the pitch of the sound that is produced. See Figure 1 as well as a short action video at https://youtu.be/fR3WlUblGL4.

Figure 1

The OzoBlockly Program

This OzoBlockly program will sharpen your students' programming techniques with a program maintenance exercise. It will also give them a chance to apply their math slope/intercept skills.

The basic idea of our OzoBlockly program is to have Evo play MIDI notes of varying frequencies that depend upon how far the hand is from Evo's front proximity sensors—the closer to the sensor, the higher the pitch of the note. MIDI notes have integer numbers with a range from 0

to 127. MIDI 0 corresponds to C1 on the musical scale, with a frequency of about 8.176 Hz (cycles per second). MIDI 127 corresponds to G11, with a frequency of 12544 Hz. Notes with very low frequencies have very low volumes and can barely be heard on Evo's speakers. For this reason, our Evo theremin has been designed to produce MIDI 64 (B6) through MIDI 127 (G11). For details on converting MIDI note numbers to frequencies, check out the link at the Web address http://subsynth.sourceforge.net/midinote2freq.html.

Figure 2 shows the blocks making up the OzoBlockly program that turns Ozobot Evo into a theremin. The discussion that follows details the purpose of the blocks, some being straight forward and others subtler in what they accomplish.

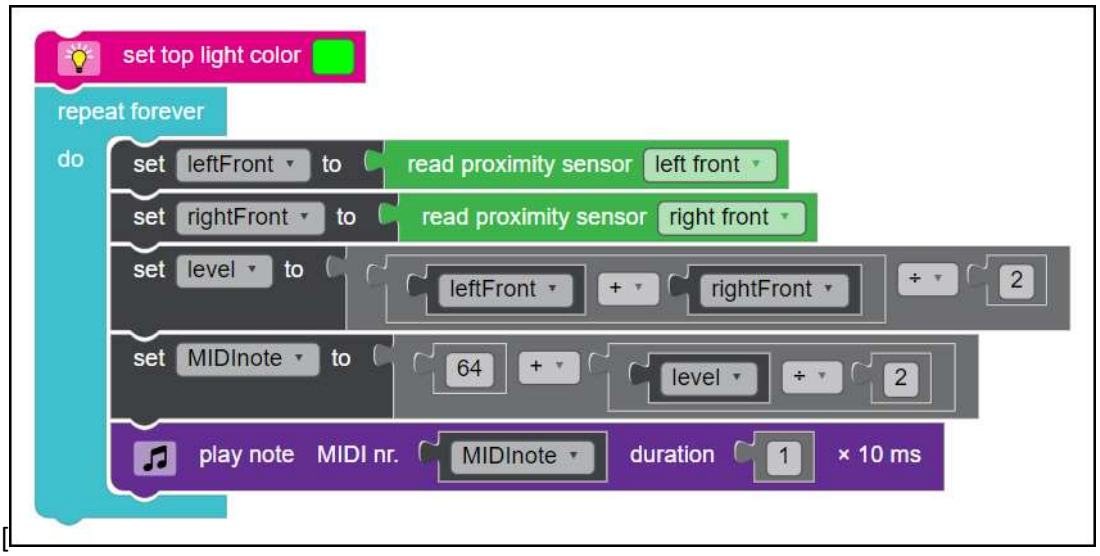

Figure 2

The top light is set to the color green to indicate that the theremin is ready to be played. A *repeat forever* loop then begins by reading both the left and right proximity sensor values. The level is then taken as the average of these two values. According to the online OzoBlockly Reference, *level* will turn out to be a number between 0 and 127, with 0 indicating no obstacle is detected, and increasing values indicating that the hand is getting closer and closer to Evo's sensors.

Figure 3 shows a graph that helps to explain how to convert the average sensor level to MIDI notes between 64 and 127. When the average sensor reading is 0 (hand far from Evos's sensors), we want the MIDI note 64 (lowest frequency in the desired range) played. When the average sensor reading is 127 (hand close to Evo's sensors), we want the MIDI note 127 (highest frequency in the desired range) played. Assuming a linear relationship between level and MIDI note, the red line on the graph expresses this. The slope of the line is 0.5 and the y-intercept is 64. Using the slope-intercept form ($y = mx + b$) of a straight line, we obtain the equation *MIDI note = 0.5*level + 64*. Since Evo deals with integer arithmetic, we re-write the

equation as *MIDI note = level/2 + 64*. The repeat forever loop concludes with the MIDI note being played for 10 milliseconds.

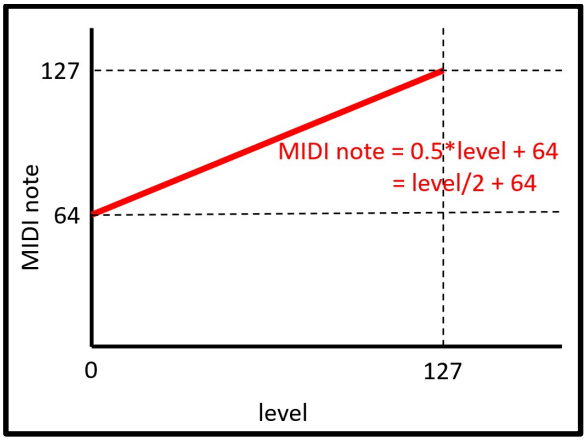

Figure 3

Program Maintenance Exercise

Modify the OzoBlockly program so that Evo's top light will be RED for MIDI note numbers greater than 90, YELLOW for MIDI note numbers between 85 and 90 (inclusive), and green otherwise.