

Robot Music Camp 2013: An Experiment to Promote STEM and Computer Science

CJ ChanJin Chung, Christopher Cartwright, and Chanmee Chung
cchung@LTU.edu, ccartwig@LTU.edu, chanmeec@umich.edu

Abstract – In order to get students interested in STEM (Science, Technology, Engineering, and Math) areas and to increase preparedness in STEM subjects for successful college education, we developed a STEM curriculum connecting music based on Lego® NXT robots and Java MIDI programming. Through this, we are able to teach the STEM subjects in-depth and create interactive musical robots by emphasizing the computer science behind it. According to surveys from the first summer camp in summer 2013 with eleven high school students, the camp achieved the aforementioned goals and we believe it is a STEAM learning environment that could be effective in student recruitment and retention in STEM.

Index Terms – STEAM, robotics education, educational robotics, computer science education

INTRODUCTION

Robots first appeared in U.S. classrooms for educational purposes over 25 years ago [1, 3, 4, 5, 7, 9, 14]. Beginning in 2000 and continuing over the past 14 years through annual Robofest programs [19, 20, 21, 22, 23], we have found “autonomous” robotics to be a highly successful medium for teaching STEM and computer programming skills. We believe that programming a robot that interacts with the physical environment can provide opportunities to learn math, physics, and engineering concepts since computer-programmed robotics is all about STEM [3, 8, 15, 17]. Here we would like to add music components to robotics programs [16, 17, 18] for the following two ultimate goals:

- Get more students interested in STEM (Science, Technology, Engineering, and Math) areas
- Increase preparedness in STEM subjects for successful college education

Why add music to autonomous robotics? Music is a universal language, and students generally enjoy music. We are particularly interested in the integration because computer programming which is essential to robotics and music share deep structural similarities. Both music and computer program codes use formal languages with the same control structures - sequence, selection, and repetition - that are sufficient to express any computable function [2]. The three structures are fundamental concepts for computational and algorithmic thinking to solve problems. For example, see the simple music in Figure 1. “D.S. al

fine” instructs the musician to repeat back to the sign and end the piece at the measure marked *fine*. The procedures are specified with pseudo code in Figure 2, showing that the music can be represented by using sequence, selection, and repetition. Figure 3 is a Visual Programming Language (VPL) program, good for beginners, equivalent to Figure 2. The outer-most rectangular box in the center represents the “Repeat” for “Dal Segno al fine” and the second box represents the two cases of playing the “B” note.

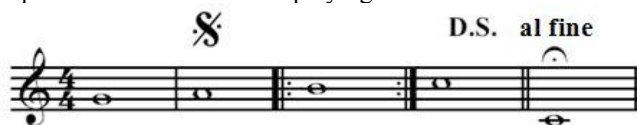


FIGURE 1

MUSIC WITH REPEAT SIGNS AND D.S. AL FINE; THE NOTES ARE TO BE PLAYED IN THIS ORDER: G A B B C A B C LOW-C; WIKIPEDIA [28]

```
Variable firsttime = true;
Play G for 4 time units;
Repeat the following twice { // Dal Segno
  Play A for 4 time units;
  If firsttime is true { // Selection
    Repeat 2 times {
      Play B for 4 time units;
    }
    firsttime = false;
  } else {
    Play B for 4 time units;
  }
  Play high C for 4 time units;
}
Play C for 4+ time units;
```

FIGURE 2

C LIKE STRUCTURED PSEUDO CODE

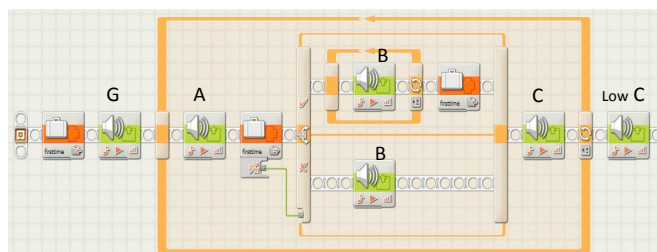


FIGURE 3

CORRESPONDING LEGO NXT-G PROGRAM

In addition to learning opportunities of technology and engineering components through robotics, students can learn more mathematical concepts since music has strong ties with math [11, 18]. As an example, during the camp we have taught the concept of geometric progress by making a robot arm play a castanet, as shown in Figure 4. The n -th term of a geometric sequence with initial value a and common ratio r is given by $a_n = a \cdot r^{n-1}$. We use a as the initial

pause time after hitting (moving the arm down and up) the castanet. Ratio r is used to control the pause. If a is 8 seconds and r is $\frac{1}{2}$, the first pause is 8, the second pause is 4, and the third pause will be 2, and so on. Figure 5 is a NXT-G program to demonstrate the geometric progress by castanet sounds using $9/10$ as the ratio to reduce arm rest time. As workshop exercises, we ask students what they can hear when r is equal to 1, or r is greater than 1.

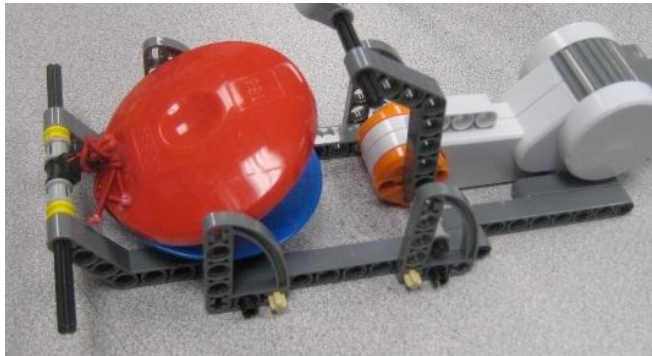


FIGURE 4
A LEGO ROBOT ARM THAT PLAYS A CASTANET

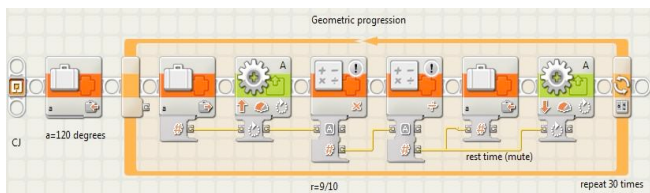


FIGURE 5
NXT-G HITTING A CASTANET WITH GEOMETRIC PAUSE INTERVALS

Robot Music Camp is based on the above principles, experiences, and strategies to provide strong opportunities for effective STEM learning [6, 8, 10, 13, 14, 15, 16, 17, 18]. The first one week-long Robot Music Camp was held July 15-19, 2013 at Lawrence Technological University. A total of 11 high school students participated in the camp. Among them, three were female students, and four were Afro-American students. Four students were in 10th grade and the remaining seven students were 11th grade students. Table 1 shows the agenda for a total of 30 hours, and the following sections describe in detail topics covered.

TABLE 1
ROBOT MUSIC CAMP AGENDA

	Topic	Hours
Day 1	Introduction to Robot Music Camp and music elements	2
	Introduction to Lego NXT robotics and NXT-G	3
	Project (RoboParade with percussion music)	1
Day 2	Advanced NXT-G (Bluetooth and My Block)	4
	RobotC programming and RobotC projects	2
Day 3	RobotC	2
	Introduction to Java and Java projects	4
Day 4	LeJos, USB communication, & Java MIDI program	5
	Group Project	1
Day 5	Bluetooth communication with Java	1
	Survey & Assessment	1
	Group Project	2
	Rehearsal, Public Concert & Wrap-up	2

INTRO TO ROBOT MUSIC CAMP AND MUSIC ELEMENTS

The camp began with a survey of the students to help us understand where they stood in terms of their experience with music, computer science/robotics, mathematics, and their future career plans. It is important to see how their previous experiences may have affected their performance throughout the week, as well as how the camp may affect their career paths as a result of the hands-on experience they received.

The results of the pre-camp survey showed that most of the students had extensive musical backgrounds yet minimal to little computer science and robotics experience. Although many expressed interest in STEM-related fields, many did not have the actual hands-on or academic experience. For two students who did not have enough music background, and to review music theories, we introduced basic music elements as well as reviewed basic music theory for an hour.

INTRO TO NXT AND NXT-G WITH MUSIC AND MATH

On the first day, we introduced LEGO NXT robots with light, touch, sonar, and sound sensors and two motors with rotation sensors together with the programming software Lego NXT-G. Since all the hardware details are encapsulated in Lego boxes and blocks, we found Lego robotics is an effective tool to introduce the concept of programming logic and the integration of music and robotics. One of the challenges we presented to the students was to have them program a simple rhythm integrating one of the robot's motors and a tambourine as shown in Figure 6.



FIGURE 6
TAMBOURINE PLAYING NXT ROBOT

We gave them a specific rhythm to play called the Clave Son Rhythm, as illustrated in Figure 7, with a total of 16 beats in the rhythm and drum beats to be played on beats 1, 4, 7, 11, and 13.

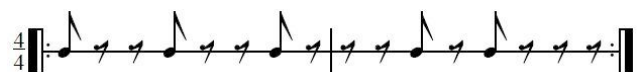


FIGURE 7
THE CLAVE SON RHYTHM WITH MUSIC NOTATION USING THE SMALLEST CONVENIENT DURATIONS OF NOTES AND RESTS [12]

This rhythm can be represented mathematically as a binary string, **1001001000101000** or using interval length

representation, **33424**, where the numbers denote the lengths of the intervals between consecutive beats. Geometric representation of rhythms can also be introduced as shown in Figure 8 for the clave Son. Figure 9 shows a NXT-G program playing the rhythm by changing motor angles and direction.

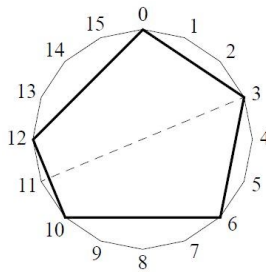


FIGURE 8

THE CLAVE SON REPRESENTED AS A CONVEX POLYGON [12]

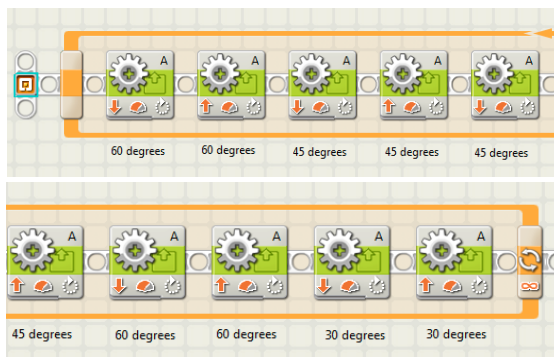


FIGURE 9

NXT-G PROGRAM PLAYING THE CLAVE SON

The students were also introduced to the concept of setting up their own Bluetooth network between the robots in order to program the robots to function with a single trigger, such as a light sensor set to send a message to the paired robot to begin the program. Bluetooth was introduced to the students in order to further their understanding of Bluetooth-capable electronics in their daily lives as well as to include it in their final project to create a robotics music band that requires synchronization between robots.

INTRODUCTION TO A TEXT-BASED LANGUAGE: ROBOTC

Although Lego NXT-G proved to be effective in learning simple programming skills, it is not good at introducing variables. It does not provide arrays either. Therefore we introduced RobotC programming language, which is based on the popular professional language “C”. One of the topics covered in the camp using RobotC was to play broken chords (arpeggios) based on frequencies.

This program was based on a mathematical mapping from a number to a frequency in Hz (shifted from the MIDI, Musical Instrument Digital Interface, mapping so that A440 Hz mapped to the number 0). The frequencies for each note of the chord were computed using the formula below.

$$f(n) = 440(2^{n/12}), -48 \leq n \leq 39$$

The duration in time of each note was computed based on the number of beats and the tempo (in beats per minute). By changing only two arrays in the program, students could program their NXT robots to play any chord as a broken chord with any type of rhythm they desired. Variables were used to set the key of the chord and the tempo. Each note of the chord was specified by how many semitones it differed from the previous note of the chord. The number of beats for each note of the chord was stored in another array. See Figure 10. Figure 11 shows a main program playing the “C” chord when key is equal to 3. Calculated frequencies will be displayed on the LCD screen as 523 (C), 659 (E), 783 (G), 1046 (C), etc.

```
int offsets[] = {4, 3, 5}; // major chord
int key=3; //transpose the key by this number of semitones
// If 0, the key is A
const int num_notes = 7;
int note=0;
float Freqs[num_notes];
float Rhythms[num_notes];
int beats[num_notes] = {10,10,10,1,1,1,1};
int tempo=120; // in bpm
```

FIGURE 10

DATA STRUCTURES OF THE ROBOTC PROGRAM TO PLAY C CHORD

```
task main()
{
    for (int i=0; i<num_notes; i++)
    {
        if (i == 0)
            note = 0;
        else
            note=note+offsets[(i-1)%3];
        Freqs[i] = pow(2.0, (float)(key+note)/12.0)*440.0;
        nxtDisplayCenteredBigTextLine(3, "%d", Freqs[i]);
        //nxtDisplayTextLine(3, "%f", Freqs[i]);
        Rhythms[i] = (float)beats[i]*6000.0/tempo;
        // (1 min = 60 sec) x 100 10Msec
        PlayTone(Freqs[i], Rhythms[i]);
        // duration in units of 10 millisecc, 100 means 1 second
        while(bSoundActive){} // while a sound is actively playing,
        // do not continue until finished playing sound
    }
}
```

FIGURE 11

MAIN TASK OF THE ROBOTC PROGRAM PLAYING “C” CHORD

LEARNING BASIC JAVA WITH PC MIDI PROGRAMMING

Although RobotC is an effective programming language for NXT robots, we chose Java as our main programming language for the camp because it is purely Object-Oriented, standardized, free, as well as its usage in the AP Computer Science course. In addition, most importantly for the camp, Java provides rich library functions for MIDI programming. These characteristics together with the well-known Eclipse IDE prove to be useful to the student that is interested in pursuing computer science.

Students were introduced to MIDI to be the main tool to play musical notes they desired to play. MIDI is a standardized interface that carries messages with specific notes, velocity, pitch, and different instruments. Because the MIDI standard is incorporated into the Java programming language, the MIDI library is needed to be opened via the code in order to play any note. Each note corresponds to a

specific number that is related to a standard 88 key piano [25]. The students were also informed that the library has several instruments to choose from, and that every instrument is listed on a Wikipedia page [26].

One of the first projects the students needed to complete while learning basic Java was to program their PCs to play the note C4 (“Middle C”) from the MIDI database. Students were able to play the note as well as modify the instrument and velocity values, and as shown in Figure 12, changing these values requires knowledge of notation and syntax.

```
import javax.sound.midi.*;
public class HelloMidi0 {
    public static void main(String[] args) throws InterruptedException {
        Synthesizer syn = null;
        try {
            syn = MidiSystem.getSynthesizer();
            syn.open();
        } catch (MidiUnavailableException e1) {
            e1.printStackTrace();
        }
        MidiChannel[] mc = syn.getChannels();
        mc[0].programChange(0, 42); // Set Cello (42) from Bank 0
        mc[0].noteOn(60, 127); // 60: middle C, 127: full velocity
        Thread.sleep(3000); // play 3 seconds
        mc[0].noteOff(64);
        syn.close();
    }
}
```

FIGURE 12

MIDI PROGRAM TO PLAY A MIDDLE C KEY FOR 3 SECONDS WITH CELLO

Furthermore, with the PC MIDI capabilities, the students applied their knowledge of arrays in programming that they learned with RobotC into Java by playing Happy Birthday using two different arrays. These two different arrays are specific to the notes that are playing as well as the duration that each note is held for, as shown in Figure 13.

```
import javax.sound.midi.*;
public class HelloMidiSong {
    public static void main(String[] args) throws InterruptedException {
        Synthesizer syn = null;
        try {
            syn = MidiSystem.getSynthesizer();
            syn.open();
        } catch (MidiUnavailableException e1) { e1.printStackTrace(); }
        final MidiChannel[] mc = syn.getChannels();
        mc[0].programChange(0, 2); // Set Electric Grand Piano from Bank 0
        int[] notes = new int[] { 62, 62, 64, 62, 67, 66 };
        double[] dur = new double[] { 0.5, 0.5, 1, 1, 1, 2 }; //1: a quarter
        int bpm = 120; // beats per minute
        for (int i = 0; i < notes.length; i++) {
            mc[0].allNotesOff();
            mc[0].noteOn(notes[i], 127); // Transpose can be done here.
            Thread.sleep((int)(dur[i]*60/bpm*1000));
        }
        Thread.sleep(1000);
        syn.close();
    }
}
```

FIGURE 13

2 ARRAYS THAT INDICATE WHAT NOTES ARE BEING PLAYED AND DURATION

We also taught them the concept of transposing. Students were amazed to see how simple mathematics can be applied to transpose “Happy Birthday” in C to “Happy Birthday” in C# simply by adding the number 1.

```
mc[0].noteOn(notes[i] + 1, 127);
```

Furthermore, students expanded their understanding of Java syntax by being able to play various types of full chords (majors, minors, major7, minor7, etc.) using a Java method that accepts array reference as noted in Figure 14. The idea is to add the contents of an array to the base note. For

example, to play C chord on 4 channels, you need to play midi note number 60, 60+4, 60+7, and 60+12 at the same time. The number 4, 7, 12 are stored in an array called “major” in the program. With this knowledge, students were also able to incorporate different variations to these chords, including arpeggios. Later, the students were able to modify and combine them suit to their needs for their final musical projects.

```
public class Hello4Chords {
    final static int[] major = new int[] { 4, 7, 12 }; // 4, (4+3), (4+3+5)
    final static int[] minor = new int[] { 3, 7, 12 }; // 3, (3+4), (3+4+5)
    final static int[] seventh = new int[] { 4, 7, 10 }; //4, (4+3), (4+3+5)
    public static void main(String[] args) {
        Synthesizer syn = null;
        try {
            syn = MidiSystem.getSynthesizer();
            syn.open();
        } catch (MidiUnavailableException e1) { e1.printStackTrace(); }
        final MidiChannel[] mc = syn.getChannels();
        mc[0].programChange(0, 2); // Set E Grand Piano from Bank 0
        mc[1].programChange(0, 2);
        mc[2].programChange(0, 2);
        mc[3].programChange(0, 2);
        Chord(60, major, mc, 1000); // C
        Chord(60, seventh, mc, 1000); // C7
        Chord(65, major, mc, 1000); // F
        Chord(57, minor, mc, 1000); // Am
        Chord(62, minor, mc, 1000); // Dm
        Chord(67, seventh, mc, 1000); // G7
        Chord(60, major, mc, 2000); // C
        syn.close();
    }
    public static void Chord(int baseNote, int[] type, MidiChannel[] mc, int d) {
        mc[0].noteOn(baseNote, 127);
        for (int i = 0; i < 3; i++)
            mc[0].noteOn(baseNote + type[i], 127);
        try {
            Thread.sleep(d); // sleep duration
        } catch (InterruptedException e) {}
        mc[0].noteOff(baseNote);
        for (int i = 0; i < 3; i++)
            mc[0].noteOff(baseNote + type[i]);
    }
}
```

FIGURE 14

C – C7 – F – AM – DM – G7 – C CHORDS PLAYED BY MIDI ELECTRIC GRAND PIANO SOUND ON 4 DIFFERENT CHANNELS

NXT CONNECTED TO A PC WITH MIDI THROUGH USB

Lego NXTs can play tones in frequencies or MIDI notes. However, that neither provide various instrument sounds nor ways to amplify the sound. So the idea for the camp was to use Lego sensors attached to NXT to create interactive musical instruments and the NXT sent musical data to a Java program on PC side to play MIDI music. We chose USB communication rather than Bluetooth considering the speed of communication between NXT and PC. For this task, we used leJos NXJ that is available for free from [27]. leJOS NXJ is a Java programming environment for the LEGO NXT. It provides libraries and tools for LEGO robots to be programmed in Java.

I. Touch sensor-based musical instrument

Lego touch sensors can have two states: pressed or unpressed. The following picture in Figure 15 shows an interactive LEGO musical instrument resembling a trumpet made of four touch sensors. The NXT is connected to a laptop using a USB cable. Every laptop has the capability of MIDI sound. We connected external speakers to the laptop to amplify the sound. Human players can select a note out of 15 (16 minus 1) possible combinations. During the camp, students learned about binary number representation.

Figure 16 shows the NXT side program to send sensor data to PC through USB connection.

Figure 17 shows the PC side program which receives signals from NXT to play a MIDI trumpet note. Note that a section of code corresponds to the binary system, shown in table 2, which shows the binary numbers that indicate all the different combinations that can be possible with four touch sensors, together with corresponding musical notes. The notes can be easily changed by modifying the PC side Java.

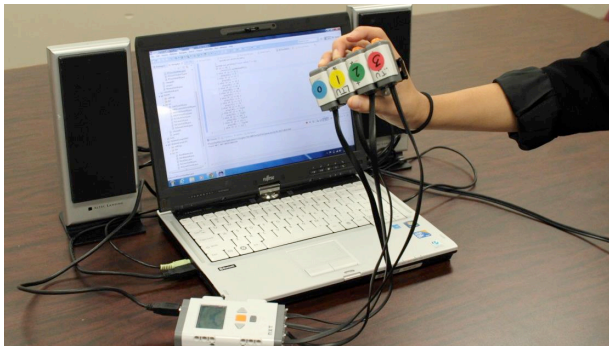


FIGURE 15

TOUCH SENSOR 0 AND 1 ARE PRESSED. TRUMPET SOUND IN MIDDLE C IS BEING PLAYED.

```
import lejos.nxt.*;
import java.io.*;
import lejos.nxt.comm.*;
public class SendTouch2PC {
    public static void main(String[] args) throws Exception {
        LCD.clear();
        USBConnection conn = USB.waitForConnection();
        DataOutputStream dout = conn.openDataOutputStream();
        TouchSensor touch1 = new TouchSensor(SensorPort.S1); // Most Sig. Bit
        TouchSensor touch2 = new TouchSensor(SensorPort.S2);
        TouchSensor touch3 = new TouchSensor(SensorPort.S3);
        TouchSensor touch4 = new TouchSensor(SensorPort.S4); // LSB
        int sent = 99;
        while (Button.ESCAPE.isDown() != true) {
            int x = 0;
            boolean t1, t2, t3, t4 = false;
            t1 = touch1.isPressed();
            t2 = touch2.isPressed();
            t3 = touch3.isPressed();
            t4 = touch4.isPressed();
            if (t1 == false && t2 == false && t3 == false && t4 == false)
                x = 0;
            // code omitted. See table 2
            else if (t1 == true && t2 == true && t3 == true && t4 == true)
                x = 15;
            if (x != sent) { // do not send same code repeatedly
                dout.writeInt(x);
                dout.flush();
                sent = x;
                LCD.drawInt(x, 7, 3); LCD.refresh();
            }
        } // end while
        try {
            dout.close();
            dout = null;
        } catch (IOException ioe) {}
        conn.close();
    }
}
```

FIGURE 16

READS DATA FROM TOUCH SENSORS AND SENDS TO PC VIA USB

TABLE 2

BINARY NUMBERS AND CORRESPONDING NOTES FOR 4 TOUCH SENSORS

TOUCH SENSORS	MIDI NOTE NUMBER	MUSICAL NOTE	TOUCH SENSORS	MIDI NOTE NUMBER	MUSICAL NOTE
0000	N/A	NO SOUND	1000	69	HIGH A
0001	57	A	1001	71	HIGH B
0010	59	B	1010	72	HIGH C
0011	60	C	1011	74	HIGH D
0100	62	D	1100	76	HIGH E
0101	64	E	1101	77	HIGH F
0110	65	F	1110	79	HIGH G
0111	67	G	1111	81	HIGH A

II. Ultrasonic sensor based musical instrument

Lego ultrasonic sensor generates sound waves and reads their echoes to detect and measure distance from objects. Using this sensor, we developed a musical instrument to play chords when an object is detected within a specified range as shown in Figure 18.

```
public class RxTouchViaUSB {
    static MidiChannel[] channel;
    public static void main(String[] args) throws Exception {
        NXTConnector conn = new NXTConnector();
        conn.addLogListener(new NXTCommLogListener() {
            public void logEvent(String message) {
                System.out.println("Log.listener: " + message);
            }
            public void logEvent(Throwable throwable) {
                System.out.println("Log.listener - stack trace: ");
                throwable.printStackTrace();
            }
        });
        if (!conn.connectTo("usb://")) {
            System.err.println("No NXT found using USB");
            System.exit(1);
        }
        InputStream is = conn.getInputStream();
        DataInputStream inDat = new DataInputStream(is);
        int x = 0; openSynthesizer();
        while (Button.ESCAPE.isDown() != true) {
            try {
                x = inDat.readInt();
            } // Blocking; no return until data is read.
            catch (IOException ioe) { System.out.println(ioe); }
            System.out.println("Touch value " + x);
            if (x == 0) channel[3].allNotesOff();
            else if (x == 1) play(57); // A
            // code omitted. See Table 2
            else if (x == 15) play(81); // High A
            else channel[3].allNotesOff();
        } // end while
        channel[3].allNotesOff();
        try {
            inDat.close();
            is.close();
            conn.close();
        } catch (IOException ioe) { System.out.println(ioe); }
    }
    public static void play(int MIDINumber) {
        channel[3].allNotesOff();
        channel[3].noteOn(MIDINumber, 127);
    }
}
```

FIGURE 17

PC PROGRAM TO RECEIVE THE TOUCH SENSOR DATA THE NXT SENDS



FIGURE 18

SINCE THE DISTANCE BETWEEN THE SENSOR AND THE PANEL IS ABOUT 30CM, C 7TH CHORD IS BEING PLAYED

The NXT is connected to a PC through a USB cable. The Java program on PC side receives signals from the NXT robot and plays corresponding chords. Figure 19 shows the NXT side program to send the range data to PC through USB connection.

Figure 20 shows the PC side program which receives signals from NXT to play a MIDI chord sound. Note that the program corresponds to four different ranges of distances from a sonar sensor.

```
public class SendSonar2PC {
    public static void main(String[] args) throws Exception {
        USBConnection conn = USB.waitForConnection();
        DataOutputStream dOut = conn.openDataOutputStream();
        UltrasonicSensor sonar = new UltrasonicSensor(SensorPort.S2);
        int s, sent = 0;
        while (Button.ESCAPE.isDown() != true) {
            s = sonar.getDistance();
            if (s != sent) { // do not send if same as before
                dOut.writeInt(s);
                dOut.flush();
                sent = s;
                LCD.drawInt(s, 8, 0, 1);
            }
        }
        try {
            dOut.close();
        } catch (IOException ioe) {}
        conn.close();
    }
}
```

FIGURE 19

NXT PROGRAM FOR THE SONAR SENSOR TO RECEIVE AND SEND DATA VIA USB CONNECTION

```
int d = 0;
int state = 0;
while (Button.ESCAPE.isDown() != true) {
    try {
        d = inDat.readInt();
    } catch (IOException ioe) {
        System.out.println(ioe);
    }
    System.out.println("Received " + d);
    if (d > 0 && d < 10) {
        if (state != 1) {
            Chord(65, major, mc); // F
            state = 1;
        }
    } else if (d >= 10 && d < 18) {
        if (state != 2) {
            Chord(62, minor7, mc); // Dm7
            state = 2;
        }
    } else if (d >= 18 && d < 26) {
        if (state != 3) {
            Chord(67, minor7, mc); // Gm7
            state = 3;
        }
    } else if (d >= 26 && d < 34) {
        if (state != 4) {
            Chord(60, seventh, mc); // C7
            state = 4;
        }
    } else { // d >= 34
        for(int i = 0; i > 4; i++)
            mc[i].allNotesOff();
    }
} // end of while
```

FIGURE 20

PC PROGRAM THAT RECEIVES THE SONAR SENSOR DATA AND EMITS CHORDS ACCORDINGLY

III. Color sensor based musical instrument

The robots equipped with a color sensor were programmed to identify a specific color and send it to the PC for it to play a specific note or chord for that particular color. See Figure 21. Java programs are similar to Figure 19 and 20. We asked students to create codes.

FINAL PROJECTS

For part of the final project required from the students, the students were broken up into pairs for a total of five pairs and one individual to come up with two different songs to play with at least two different robotic instruments. Each team had the responsibility to select the songs and to

incorporate Java programs with the NXT robots. The students were then required to present their songs. Videos of their performance can be found on a YouTube play list [24].

The second part of the final project required students to set up three NXT robots programmed to play three keyboards using the NXT-G programming language. These three robots were equipped with light sensors to detect a single light trigger that would allow the robots to send a Bluetooth message to three other robots with percussion instruments. These robots would then play “Happy Birthday” as well as “Heart and Soul” simultaneously with four other students who would play along with the songs with the touch sensor trumpet and color sensor chord accompanists. Figure 22 shows the set-up of the final group project.



FIGURE 21

COLOR SENSOR DETECTS BLUE COLOR AND PLAYING MINOR 7TH CHORD

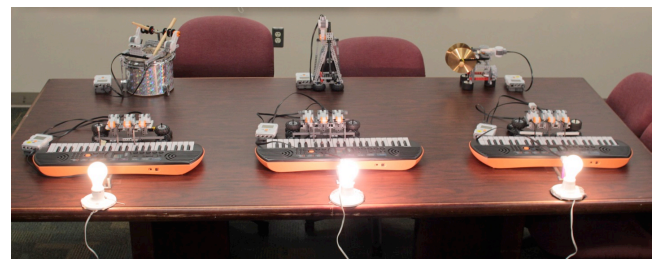


FIGURE 22

SET UP OF THE KEYBOARD PLAYERS, LIGHT BULBS, KEYBOARDS, AND PERCUSSION INSTRUMENTS

SUMMARY AND CONCLUSION

The goals of integrating music with computer science and robotics are successfully demonstrated by the students’ display of their knowledge and application of what they learned throughout the camp week. Music theory, math, Java programming, Lego NXT robotics with NXT-G and RobotC, and LeJos Java were all integrated into this fun camp that both appealed to the music enthusiast and the computer programmer. Furthermore, when comparing the pre-surveys and post-surveys, the results of interest in STEM fields increased overall with the students who were not sure and were not interested at all in STEM changing their minds to being overall “Interested” in the field, as shown in Figures 23 and 24. The Robot Music Camp can be an effective example of STEAM initiative. Rigorous assessment strategies will be introduced for future robot

music camps to determine student's attitude towards STEM and to measure possibly increased preparedness in STEM subjects for successful college education. All the computer program source code files introduced in this paper can be found on the web [29].

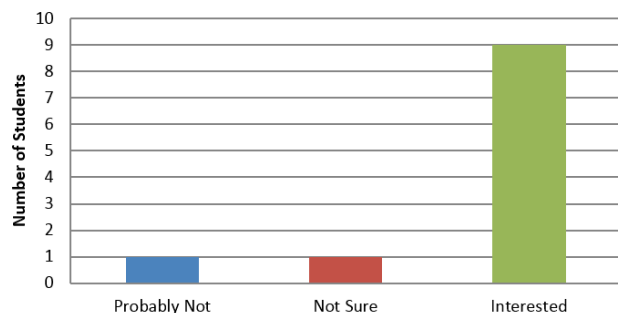


FIGURE 23

PRE-SURVEY RESULTS: ARE YOU INTERESTED IN A STEM RELATED CARRER/JOB?

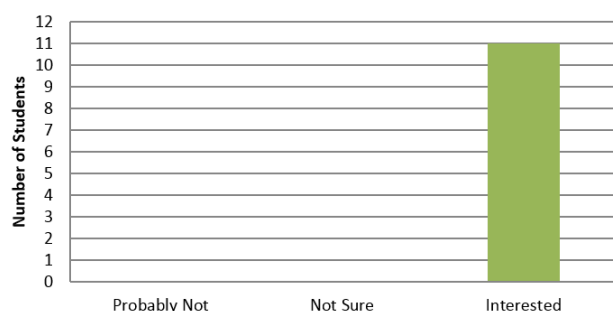


FIGURE 24

POST-SURVEY RESULTS: ARE YOU INTERESTED IN A STEM RELATED CARRER/JOB?

REFERENCES

- [1] Bers, M.U. & Portsmore, M. (2005, March). Teaching Partnerships: Early Childhood and Engineering Students Teaching Math and Science through Robotics. *Journal of Science Education and Technology*, 14 (1), 59-73.
- [2] Böhm, C. and Jacopini, G.: Flow diagrams, Turing machines and languages with only two formation rules, *CACM* 9(5), 1966.
- [3] Cejka, E., Rogers, C. & Portsmore, M. (2006). Kindergarten Robotics: Using Robotics to Motivate Math, Science, and Engineering Literacy in Elementary School. *The International Journal of Engineering Education*, 22 (4), 711-722.
- [4] Chambers, J.M. & Carbonaro, M. (2003). Designing, Developing and Implementing a Course on LEGO Robotics for Technology Teacher Education. *Journal of Technology and Teacher Education*, 11 (2), 209-241.
- [5] Groff, B. H. & Pomalaza-Ráez, C.A. (2001) Robot Omnium Horarum: A Robot for All Seasons. *Proceedings of the American Association for Artificial Intelligence Spring Symposium Series*, March 26 -28, 2001, Stanford University, CA.
- [6] Heines, J.M. & Maloney, J. (2011). Making Music with Scratch, *Proc. of the 40th ACM Tech. Symposium on CS Education*, Dallas, TX: ACM.
- [7] Kolberg, E. & Orlev, N. (2001). Robotics learning as a tool for integrating science technology curriculum in K-12 schools. *Proceedings of the 31st Annual Frontiers in Education Conference*, October 10 -13, 2001, Reno, NV, 1, 12-13.
- [8] Matson, E., DeLoach, S., & Pauly, R. (2004). Building Interest in Math and Science for Rural and Underserved Elementary School Children Using Robots. *Journal of STEM Education*, 5 (3&4), 35-46.
- [9] Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books, Inc.
- [10] Robinson, M. (2005). Robotics-driven activities: Can they improve middle school science learning? *Bulletin of Science, Technology & Society*, 25(1), 73-84.
- [11] Toussaint, G. (2005) The geometry of musical rhythm, *proceedings of the Japan Conference on Discrete and Computational Geometry*, J. Akiyama et al. (Eds.), LNCS 3742, Springer-Verlag, Berlin, Heidelberg, pp. 198-212.
- [12] Toussaint, G. (2010) Computational geometric aspects of rhythm, melody, and voice-leading, *Computational Geometry: Theory and Applications*, Vol. 43, Issue 1, January 2010, pp. 2-22.
- [13] Weiss, S. (2004, December). Teachers' Knowledge and Skills are Key to Improving Student Achievement in Science, Math. *The Progress of Education Reform*, 6 (1), Education Commission of the States.
- [14] Whitman, L.E. & Witherspoon, T.L. (2003). Using LEGOs to interest high school students and improve K-12 stem education. *Proceedings of the 33rd Annual Frontiers in Education*, November 5-8, 2003, 2, F3A 6-10.
- [15] Gil Weinberg, Scott Driscoll and Mitchell Parry, *Musical Interactions with a Perceptual Robotic Percussionist*, 2005 IEEE International Workshop on Robots and Human Interactive Communication, Pages 456-461.
- [16] Nielsen, Jacob, N. K. Brendsen, and Carsten Jessen. "RoboMusicKids-music education with robotic building blocks." In *Digital Games and Intelligent Toys Based Education*, 2008 Second IEEE International Conference on, pp. 149-156. IEEE, 2008.
- [17] Benitti, Fabiane Barreto Vavassori. "Exploring the educational potential of robotics in schools: A systematic review." *Computers & Education* 58, no. 3 (2012): 978-988.
- [18] Majgaard, Gunver, Jacob Nielsen, and Morten Misfeldt. "Robot technology and numbers in the classroom." *Proceedings in Cognition and Exploratory Learning CELDA* (2010): 231-234.
- [19] <http://www.robofest.net>
- [20] Chung, C. (2013). Unveiling 2014 Robofest Challenges, *Robot Magazine*, January/February 2014, pp. 48-51.
- [21] Chung, C. (2011). ROBOFEST 2010 - Little robots perform big missions for STEM, *Robot Magazine*, January/February, pp. 60-65.
- [22] Chung, C. (2010). RoboParade - An opportunity to develop your imagination while having fun, *Robot Magazine*, November/December, pp.78-79.
- [23] CJ Chung and Christopher Cartwright (January 2014), RoboParade: a Fun and Effective Way to Promote STEM Education, *Proceedings of the 12th Hawaii International Conference on Education*, Honolulu, Hawaii
- [24] <http://youtube.com/playlist?list=PL0aa287K8J2ynogx2VJHFL5srw14rd8oF>
- [25] <http://www.phys.unsw.edu.au/jw/notes.html>
- [26] http://en.wikipedia.org/wiki/General_MIDI
- [27] <http://lejos.sourceforge.net>
- [28] http://en.wikipedia.org/wiki/Dal_Segno
- [29] <http://www.robofest.net/2013/roboMusicCamp13src.zip>

ACKNOWLEDGMENT

This camp was sponsored by Lawrence Technological University (LTU) Provost Office and Robofest 2013 major sponsors (ROBOTIS, Nation Defense Industry Association MI Chapter, DENSO, Lego Education, MI Council of Women in Technology Foundation, IEEE Region 4, RobotC, RIIS, TOYOTA, and Realtime Technologies, mindsensors.com)

AUTHOR INFORMATION

CJ ChanJin Chung, Professor, Department of Math and Computer Science, Lawrence Technological University.
Christopher Cartwright, Associate Professor, Department of Math and Computer Science, Lawrence Technological University.
Chanmee Chung, College of Engineering, University of Michigan