Sun SPOT Orchestra:

Algorithmic Composition

for

Wireless Sensor

Networks

Final Project Report

Chet Henry
Kurt Larson
Kyle Murphy

CONTENTS

ABSTRACT

A system for algorithmically composing music for wireless sensor networks (WSN) is proposed. The hardware and software requirements are outlined for the individual WSN motes and the network as a whole. Justification for the project proposal, background information on WSN and algorithmic composition, and overall project vision are presented. Finally, we conclude with the results of project development, challenges faced, design limitations, and future work.

INTRODUCTION

The primary fields of research, study, and implementation for this project were wireless sensor networks (WSN) and synthetic music composition—the latter more commonly known as "algorithmic composition". The team was interested in a musical application of WSN. In addition, the team carried previous experience designing systems for WSN and therefore understood the unique requirements and capabilities of developing applications for the platform. Bringing these very different fields together set the stage for a fun, collaborative, and challenging project for the team.

The overarching concept of dynamically generating music based on environmental conditions and then coordinating the overall composition wirelessly was interesting in the team's opinion. The team hoped to introduce a new application of WSN technology, create a fun system for electronic/computer music enthusiasts, and impact the state-of-the-art of computer generated musical compositions by intelligently mapping environment to emotion and emotion to musical parameters.

Stated simply, the overall vision of the project was to create "appealing" music using only mote hardware (including some primitive circuitry for audio output) distributed across a local environment without the assistance of any sort of base station or master control software/hardware.

Within the field of algorithmic composition there are strikingly beautiful and "appealing" works of music. One example is a piece generated by Karlheinz Essl's computer program, *Lexikon-Sonate* [5]. It's worth a few moments to visit the site to hear what generative algorithms are musically capable of. For this project, the necessity of MIDI hardware was discussed amongst the team and with advisors. MIDI hardware can produce audio that sounds like an actual instrument. The team would like to point out that even *Lexikon-Sonate* is a single instrument—the grand piano—and that it is quite possible to produce interesting and engaging compositions with primitive hardware even if it does not have the dynamic range of a symphonic orchestra. Throughout the project, the team referred to its own audio primitives as "bleeps and bloops".

At first glance, the simple audio primitives appeared to be a significant hindrance to the overall quality of the system. Interestingly, the team soon learned this was a "blessing in disguise". By removing robust and "high-quality" output from the scope of the project, listeners and users of the system are forced to consider the broader scope of the composition. The team considered "the composition" (multiple motes working in tandem and in harmony) greater than the sum of its constituent, rather limited parts. The true quality of this particular musical application of WSN is derived both from the interaction between the user and the system and the system and its environment. The team intended for the system to successfully engage and captivate the user despite its limited audio output capabilities.

Given that the team attempted to generate engaging music with simple tones, researching an effective algorithm for rhythm, melody, and harmony generation was paramount for the music to be perceived as appealing. After several failed attempts to meet with UNL music professors, the team instead decided to conduct its own research online. While it certainly would have been optimal to validate the "musical" aspects of the project with a qualified/trained expert in the field of music (specifically algorithmic composition), the team felt confidant with its ideas for creating simple musical patterns based on the environment. The primary challenge was intelligently determining how to map physical phenomenon to musical properties in software.

Thankfully, the team came across a piece of research [19] which effectively mapped human emotional response to musical parameters. The paper introduced two axes of emotion, valence and arousal, which the team leveraged to intelligently turn environmental readings into musical structure. Valence is a measure of positivity of emotion. High valence corresponds to a positive feeling or emotion. Low valence corresponds to the opposite—a negative feeling or emotion. Arousal is a measure of intensity of emotion. High arousal is energetic while low arousal is lethargic. We can map common motions such as joy, anger, sadness, and gladness on these axes. Joy is considered a high arousal, high valence emotion. Anger is also high arousal but because it is perceived as a negative emotion, it has low valence. Sadness is low valence and low arousal. Gladness on the other hand is high valence but low arousal—when you're glad you're not necessarily bouncing off the walls.

Of course, these axes do not have simply discrete values (e.g. high or low). They are a continuum. The paper cited eight musical parameters or "features": tempo, rhythmic roughness, harmonic mode, upper extensions, loudness, articulation, and pitch register. Each feature is also a continuum that can be mapped to valence and arousal thanks to research in the field of musical psychology [23, 24, 25].

In addition to the emotional/environmental algorithmic challenges for this project, the team researched sources covering the field of algorithmic composition in general [1, 14, 18]. It was clear from early research that a stochastic model would be the most straightforward method of algorithmic composition. A stochastic model of algorithmic composition relies on a random process and probability to produce non-deterministic sounds [8].

Modeling a composition and all its parameters using Markov chains and models [14] seemed an appropriate method for this project considering the somewhat limited power of the motes and the real-time delays that occurred between motes during synchronization.

In the first prototype the team focused efforts around synchronizing mote audio output without any base-station assistance, creating a simple but robust play queue to encapsulate compositional information, and developing its stochastic composition algorithm. For the final prototype the team expanded the stochastic composition algorithm to create compositions in real time that reflect the environment.

DESIGN

The project design centered on building a network of sensor motes equipped with audio output capabilities and enabling them (through software) to create a coherent musical composition based on their environment—each mote contributing a portion of the overall composition. Essentially, each mote acted a basic electronic instrument that determined how and when to play its notes based on its local environmental and those inputs directed to it from a "composer" or "leader" mote. The team's software algorithms converted real-time environmental conditions into coherent music and coordinated a composition across multiple motes.

Because the team's algorithms did not rely on statically defined notes, chords, tempos, or other musical properties and attributes, and because the environments that host the WSN were never exactly the same, no two compositions were identical.

As users interact with the hosting environment they would inevitably impact the overall composition. For instance, if a user shook a mote as it played, future notes were impacted but still fit within the overall composition's style (e.g. tempo and key).

The primary components for the project were four Sun SPOT wireless sensor motes. The motes were equipped with powerful 180MHz ARM9 processors and adequate RAM and Flash storage for audio processing and mathematical calculations. Each Sun SPOT contained a sensor board that included light, temperature, and motion (accelerometer) sensors. These sensors served as the environmental inputs to the algorithmic composition.

The Sun SPOTs were retrofitted with basic audio output circuitry which included a single piezoelectric speaker. Analog filters, amplifiers, and additional could have easily been added but their presence would detract from the portability and power consumption of the network and were scoped out of the project. Hardware for sensing other environmental inputs (such as sound, motion, infrared, or magnetic) could also have been added to increase the number of mappings between the environment and musical properties.

A base station application, while not central to this project or part of its grand vision, was worthy of consideration during the design process. A base station application would provide much greater audio fidelity in the composition. In fact, it would have likely handled the entirety of the composition's output. A base station application would also have been ideal for saving/logging the composition created by the motes so that it could be analyzed or played back at a later date. Despite these obvious benefits, the team decided that the base station application will impact the scope of the project too greatly to be designed or implemented fully in the span of one semester.

Given these design considerations, it is now appropriate to discuss specific hardware and software choices over the course of the project

HARDWARE

A network consisted of one or more Sun SPOT wireless sensor motes that can sense features of their environment, wirelessly communicate with each other, and play music. The core hardware for one of these devices was a mote with an attached speaker and built-in light, acceleration, and temperature sensors, as well as a wireless antenna. The motes were equipped general purpose input and output pins for adding more sensors or other hardware.

Figure 1, on the following page, shows an example wireless sensor network. Each mote sensed different environmental changes when interacted with; a mote reacts by adapting the music it plays and communicating this information over the wireless network, leading the rest of the network to respond during the next composition cycle.

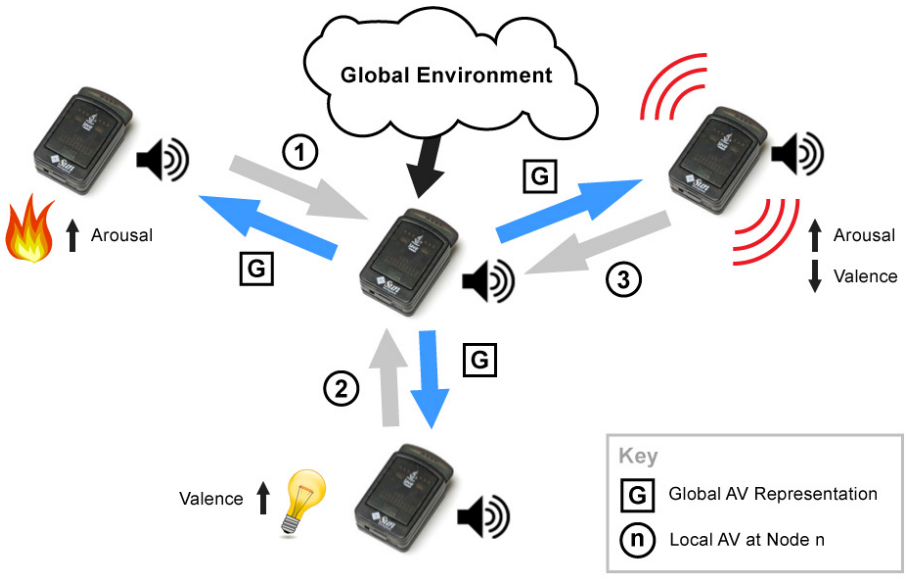

FIGURE 1 - SAMPLE NETWORK TOPOLOGY

SOFTWARE

The bulk of the project's development was in software. The software development was segmented into broad modules, consistent across all motes, including the leader, based on the functional areas described previously.

Because individual motes drive the composition in the network, the team grouped mote behavior into the following areas of related operation:

1. Sensing the environment
2. Planning music
3. Playing music
4. Sending wireless communication
5. Receiving wireless communication
6. Synchronizing time
7. Selecting musical role

Figure 2, below, shows several these behaviors within the so-called "Core" module and how they interact with others (e.g. Audio corresponds to "Playing music" and Wireless responds to "Sending/receiving wireless communication").

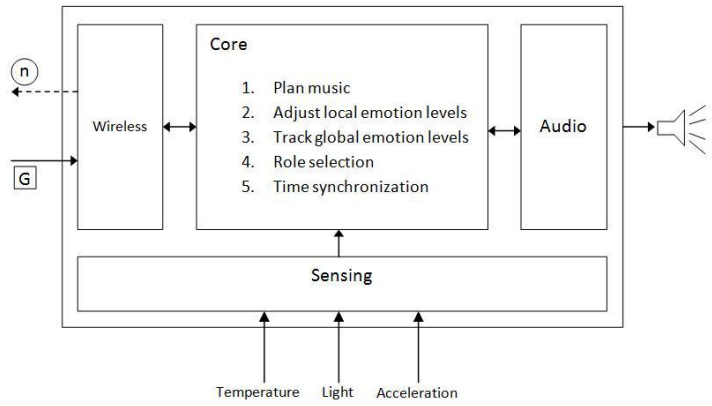

FIGURE 2 - SOFTWARE MODULES

The music composition module was central to the vision of this project – by reacting to environmental changes and wireless communication, this module generates the music played by the mote. While the field of computer music has many complex algorithms based in genetics or artificial intelligence [22], the team elected to implement a simple algorithm due to the motes' limited resources.

For this reason, the music composition module used simple stochastic methods of composition. The module chose notes randomly according to weighted probabilities stored in decision matrices [1, 14].

Composition involved planning the music a little before it's actually played in order to compensate for communication delays between and environment-driven modifications. Strong environmental stimuli caused immediate reactions on the affected mote.

To differentiate the music played by different motes, each mote chose one of the following roles, with each role strongly weighting the types of notes the mote might choose during composition:

1. Melody – a role focused on choosing many different notes and rhythms.
2. Harmony – a more complementary and less flamboyant role than the melody.
3. Bass line – this role focused on a steady pace of notes that delineate a song's chord progression.
4. Rhythm – many notes but few pitches help define a song's beat.

One more role that a single mote could have had in addition to those above is the Leader role—also known as the "composer". The Leader mote in a network performed administrative tasks related to tracking and determining global emotion values and communicating this information to the other motes. The Leader role had no direct effect on compositional decision matrices, however.

A mote's stochastic composition was affected by personal and global emotion levels (In this case, "global" is sort of a mathematical average across the network, as determined by the Leader mote). In the team's model, any emotion can be described by the dimensions of arousal and valence, where arousal describes the amount of energy (through rhythmic roughness, tempo, articulation, and loudness) and valence

describes emotional positivity or negativity (through harmonic mode, upper extensions, and pitch register) [19].

Figure 3, below, shows a graph with valence and arousal as the axes.

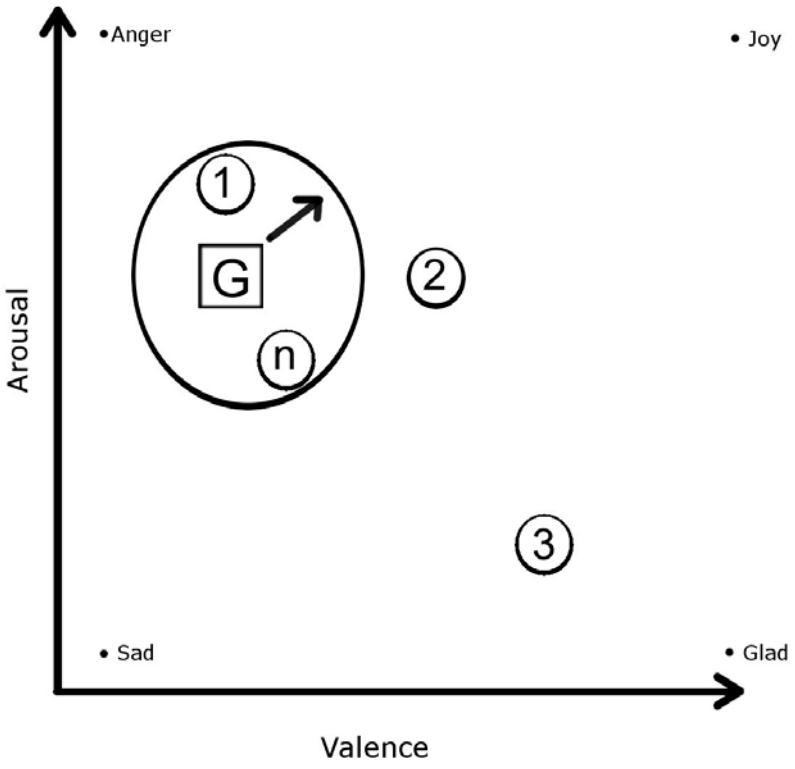

FIGURE 3 - EMOTION LEVELS

Anger, Joy, Gladness, and Sadness, the four emotions mentioned earlier, exist in the extreme corners of this graph. Furthermore, the graph shows an example of personal emotion levels for several motes and the global emotion level determined by the Leader mote; notice that individual motes are allowed some leeway in how closely they follow the global emotion values. Now consider if the motes each sense different environmental stimuli, as in Figure 1. The team is mapping different sensors to different emotion levels, with a focus on a Sun SPOT's built-in sensors that are most easily interacted with:

1. Light – direct mapping to valence
2. Acceleration – direct mapping to arousal

3. Temperature – direct mapping to arousal

The stimuli in Figure 1 affect each mote's personal emotion levels and caused an immediate response in the music that mote plays. The motes communicated this information to the Leader mote which recalculated the global emotion levels that every mote would in turn use in to weight its own calculations; at this point you might see new emotion levels as shown in Figure 4, below.

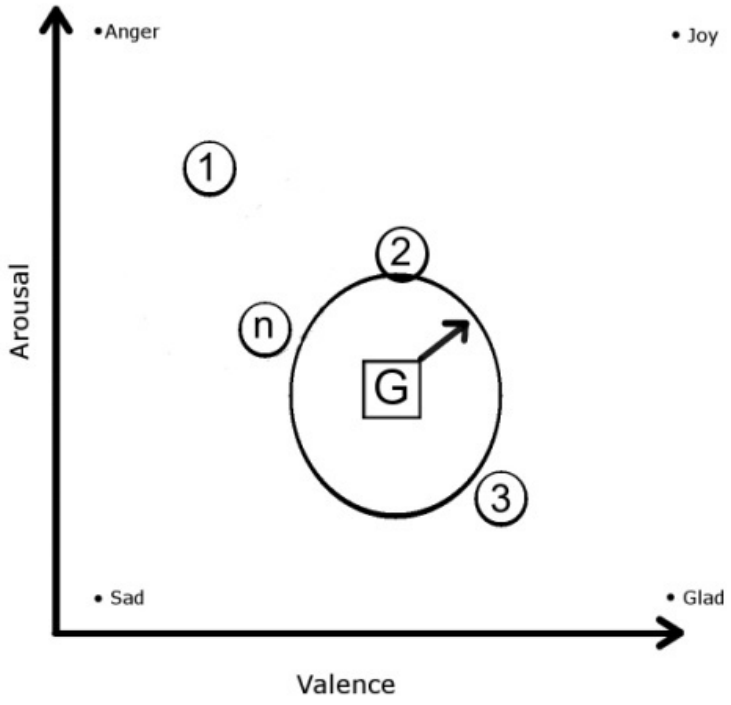

FIGURE 4 - NEW EMOTION LEVELS

Now, with a greater understanding of the primary software and hardware choices, it is appropriate to discuss specific implementation details of the project.

IMPLEMENTATION

This section describes the details of how the high level design has been implemented. The team's implementation procedure involved refactoring the mini-project code and expanding it with new

functional threads while temporarily disabling unfinished modules. Project hardware needed no further implementation work after the completion of the mini-project.

The completed scope includes time synchronization, a queue of musical notes, basic music composition using stochastic processes driven by environmental stimuli, and inter-mote collaboration. Specific implementations of these scope items are discussed in detail below.

Thread Architecture

The project functionality is divided between multiple simultaneous threads as described in the following list:

1. **SunSpotAudio** – This is the main thread, schedules the *SensingThread* and either the *LeaderThread* or *AudioThread*. It also handles the wireless message reception functionality.
2. **LeaderThread** – This thread executes if the mote has been selected as the leader. It coordinates role selection and the exchange of emotion values over wireless.
3. **AudioThread** – This thread interacts directly with the output pins by removing notes from the musical queue and converting the information into audio signals. It also schedules and wakes the *PlanningThread* as necessary.
4. **PlanningThread** – This thread performs all musical composition by using decision matrices. When awoken by *AudioThread*, it plans one measure of notes then returns to sleep.
5. **SensingThread** – This thread periodically reads from environmental sensors to update local emotion values.

Supporting Classes

The project also includes several other classes to provide supporting functionality to the primary threads of execution.

1. **DecisionMatrix** – This class and its sub-classes contain the probabilities, weights, and matrix computations used in the stochastic composition model.
2. **AV** – This simply encapsulates arousal and valence, the abstract emotion values.
3. **AVManager** – This static class stores copies of *AV* objects gathered from the network and provides an easy way to calculate a global average of emotion.
4. **Note** – This class encapsulates musical data needed for playing a single note or rest.

MOTE STARTUP

Every mote has a fixed startup period in which they prepare to play music. The first mote on becomes the leader and advertises this fact and then responds to other motes by assigning musical roles and sending synchronization information.

Sun SPOTs keep fairly accurate internal clocks, but the time differences between SPOTs are still noticeable, so they need some synchronization to play music at the same times. The team's first synchronization tests revealed that wireless communication did not take a noticeable amount of time to propagate across the medium, and the processing time for a mote to read its clock was just barely noticeable. For these reasons the team wrote a very simple algorithm where the leader mote just broadcasts its current time. Other motes compare their current time to the broadcasted time and keep track of the offset, plus an additional small offset (about 50ms) to cover the processing time. After this the motes add their offset to the broadcasted start time to determine when to start playing music.

MUSICAL QUEUE

Music needs to be simultaneously played and planned about a measure ahead. This is accomplished with a queue of musical notes so one thread can play notes while another thread is planning other notes. Musical information is encapsulated in the *Note* class containing pitch, length, and other data to represent musical notes or rests.

The queue of notes is shared between threads, so the *PlanningThread* can add future notes to the queue at the same time the *AudioThread* is popping off the next note it intends to play. A special marker note representing a separation between planned segments of music signals the composition algorithm to run again. In this way, the queue never becomes empty.

STOCHASTIC COMPOSITION

The team created a decision matrix class containing square 2D arrays of floating point numbers. An array with initial weights based on musical role is combined with temporary weights based on emotion levels; these temporary weights are determined by interpolating between values for the highest and lowest possible valence and arousal. In the end, each row represents the current note, each column represents the next note, and each cell of the matrix stores probabilities of picking the next note given the current note. Every row should sum up to 100% and the matrix guarantees this by scaling the cells down by the sum of the entire row any time a matrix computation is performed. The matrix class also encapsulates the random cell selection functionality. Thankfully the Sun SPOTs had plenty of horsepower to handle several iterations of these sorts of computations every time a measure was planned (once every few seconds).

There are several sub-classes of *DecisionMatrix* representing specific musical features that can be manipulated by the environment. *PlanningThread* currently has one of each of the following matrices:

1. **PitchMatrix** – This matrix picks from 12 possible pitches within an octave. It includes special weighting support for chords.
2. **RhythmMatrix** – This matrix picks between note lengths ranging from sixteenth notes to whole notes.
3. **RestMatrix** – This small matrix only determines whether or not the note is a rest.
4. **ArticulationMatrix** – This matrix chooses between three possible articulation values: staccato (short), medium, and legato (long).

Environment Sensing

SensingThread runs every 250ms and updates a mote's local valence and arousal. The Sun SPOT's light sensor maps linearly to valence. For instance, a light value exactly in the center of the sensor's range corresponds to 0.5 (out of 1.0). Readings from accelerometer and temperature sensors contribute to a delta value added to the current arousal.

Emotional Mapping

Local valence and arousal values affect probabilities in each decision matrix while being manipulated by the *PlanningThread*. Local values are also broadcast every measure and collected by Leader mote. Once aggregated into through use of the *AVManager* class, this new global emotion is sent back to each mote. Global valence and arousal are then taken into account by the decision matrices.

TESTING

Early development largely revolved around prototyping single features at a time, so testing each feature involved a lot of module stubbing. Later testing was all done at the integration level.

The team could not find a better way of testing the composition algorithm than just listening to the motes playing aloud. This method allowed for testing of musical quality, musical changes caused by environmental interaction, and musical changes caused by mote collaboration. Unfortunately, the best way of debugging during these testing procedures involves only the use of the Sun SPOTs' LEDs.

To better test flow of program execution, the team used the Solarium emulation program included in to the Sun SPOT SDK. As shown on the next page, Solarium allows the simulation of environment changes on any number of virtual motes. It also displays system output messages, which were especially helpful in finding and resolving exceptions and thread timing issues.

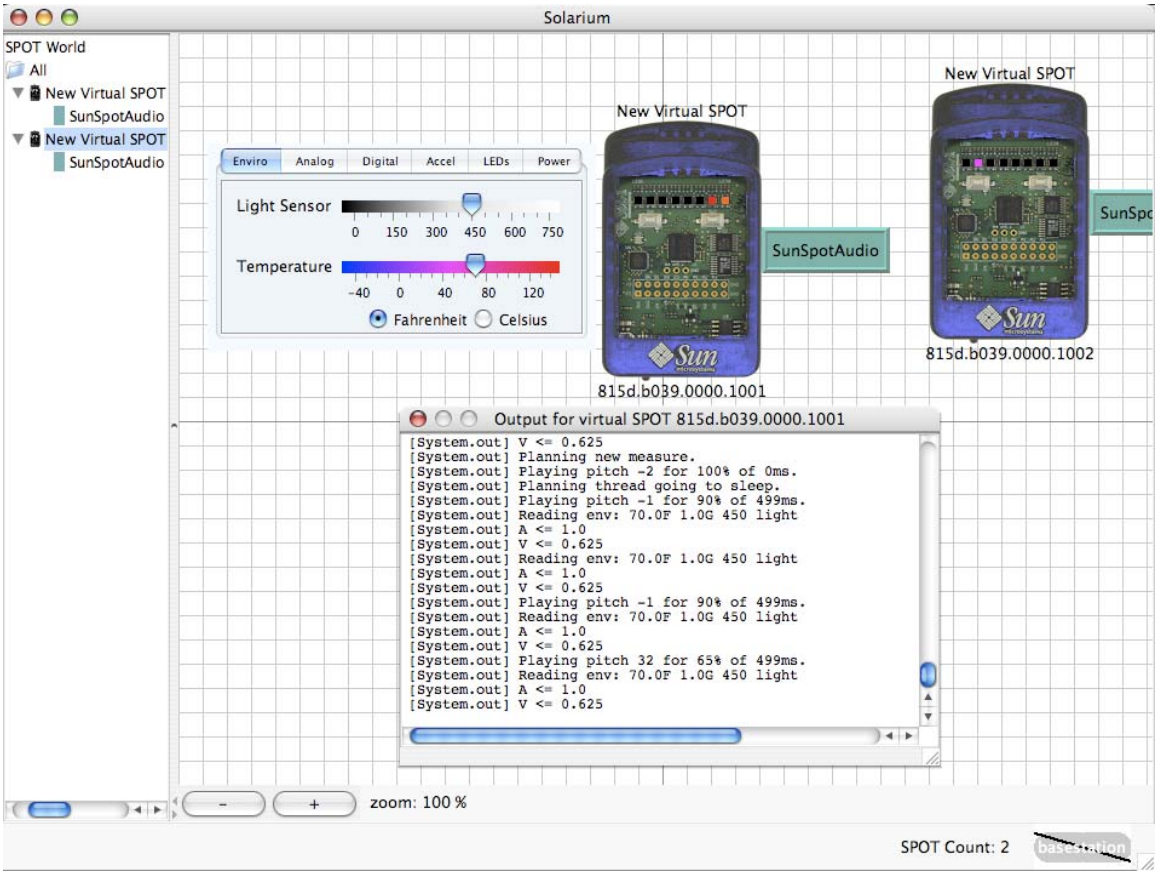

FIGURE 5 - SOLARIUM SCREENSHOT

CHALLENGES

The first challenge the team dealt with was designing the thread level architecture described above. This was a result of both design decisions and trial and error. Once the architecture was implemented, resource contention among the threads was a major hurdle. The team addressed the issue by appropriately managing thread access with lock objects to protect critical sections and limit race conditions and deadlocks.

The second challenge of synchronization was addressed by reducing the time window in which each Sun SPOT must be powered on in order to be part of the current composition world. This shortcoming was accepted by the team because of extra communications required to update the composition world dynamically and does not add anything of interest to the musical composition.

Another challenge relating to synchronization and thread competition was that the playing thread and sensing tread would compete for recourses. This competition would delay the playing thread and the motes would gradually become out of sync over time. The team made many design attempts at re-syncing the motes periodically but was not successful in addressing this problem.

The team's major challenge was designing the composition architecture. This was difficult because of the many design decisions involved. An example of this was the decision to use stochastic algorithm instead genetic algorithm or an expert system based algorithm. Then the implementation of the algorithm proved to be difficult and required many helper methods and classes.

Once these matrices were in place the team was again challenged with how to seed each matrix. Given that in the final version of this project the matrices will be seeded with the environment the team chose a seeding in which the highest and lowest valence and arousal matrices were defined. Then the algorithm would simply interpolate values in-between. This design decision limited the team on the musical transitions that could be achieved. Also it presented the challenge of how to set these extreme matrices to produce the expected musical characteristics.

Minor hardware defects in the piezoelectric buzzers pushed each Sun SPOT slightly out of tune and forced the team to use powered desktop speakers to demonstrate the system.

LIMITATIONS AND FUTURE WORK

The project has shortcomings that could be solved with additional development time. Some shortcomings include areas where originally planned scope could not be met while others are additional features that would be necessary to commercially distribute the system.

Despite early successes with time synchronization, in the end the motes would not stay synchronized. The team abandoned attempts at implementing asynchronous startup routines when it became apparent they were too difficult. Furthermore, though the motes start playing music together, their music quickly becomes unsynchronized due to non-deterministic thread use or some other subtle problem.

The musical quality is somewhat limited by the simplicity of the composition algorithm. More complex musical structure such as phrasing, key changes, tempo changes, and more deeply involved register shifts would be a benefit. Matrix transition logic also suffers from simplicity; a focus on the extreme emotional values led to weak, muddled middle ranges.

Some sort of administrative tool would aid consumers that purchase this system. A mass deploy option would make software setup easier. They might also like to be able to modify the system's various weights and thresholds to tweak system performance for their specific environments. Finally, the tool could let users save particularly good compositions to listen to later.

Finally, the team could expand the project with more advanced hardware. In particular, the Sun SPOTs can't vary their volume as-is. More interactive sensors would also improve the project's intended experience.

COSTS

Although this project had minimal hardware development a full cost analysis is always required. The SunSPOT's cost was easily the greatest at $150.00 per unit. All other small parts were purchased from http://www.sparkfun.com. The prices were as follows:

Item	Cost
4x Sun SPOT	$600.00
4x Piezoelectric speaker	$8.00
4x Audio jack breakout board	$4.80
4x 3.5mm audio jack	$6.00
4x Female headers	$1.50
4x Straight headers	$2.50
Male jumper wires	$3.95
Female jumper wires	$3.95
Development to date	$4,800.00
Learning materials	$30.00
Total	**$5,460.70**

For the development cost the team determined the total number of hours worked by each member for the project (80 hours) then assigned a standard cost of $20.00 per hour, per person. In addition, team purchased one book on algorithmic composition to supplement internet-based research.

CONCLUSION

The team learned a great deal throughout the course of the project and succeeded in its own goal of working on a project that would be enjoyable and provide exposure to new and interesting technologies and algorithms.

Ultimately, in the team's opinion, the project advances the state-of-the-art of both wireless sensor networks (as an application of the field of research) and algorithmic music composition (through a distributed algorithm that composes based on the environment).

REFERENCES

The team researched following sources of information:

1. "Algorithmic Composition", Website. Retrieved November, 2008. http://en.wikipedia.org/wiki/Algorithmic_composition

2. Biles, Al. *Evolutionary Music Tutorial.* GECCO 2005.

3. Collins, Nick. *INFNO: Generating Synth Pop and Electronic Dance Music on Demand.* http://www.informatics.sussex.ac.uk/users/nc81/research.php

4. *cgMusic*. Computer Software, Website. Retrieved November, 2008 http://codeminion.com/blogs/maciek/2008/05/cgmusic-computers-create-music/

5. Essl, Karlheinz. *Lexikon-Sonate – Algorithmic Music Generator*. January, 2007. Computer Software, Website. http://www.essl.at/works/Lexikon-Sonate.html#abstract

6. Gabrielsson and E. Lindstrom. *Music and Emotion: Theory and Research, chapter The Influence of Musical Structure on Emotional Expression*, pp. 224-248, P.N. Juslin and J.A. Sloboda Eds. Oxford UP, 2001.

7. Gallager, R. G., Humblet, P.A., Spira, P.M. *A Distributed Algorithm for Minimum-Weight Spanning Trees.* ACM Transactions on Programming Languages and Systems, Vol. 5, No. 1, January 1983.

8. "javapd". *Sun SPOT GPIO Tone Generator Sample Code*. June, 2007. Video. http://www.youtube.com/watch?v=2dnpLzh9lDg

9. Koelle, David. *The Complete Guide to JFugue – Programming Music in Java*. First Edition. April, 2008. PDF. http://www.jfugue.org/book.html

10. Koelle, David. "Sun SPOT + JFugue". Email Correspondence with Kyle Murphy, Chet Henry, and Kurt Larson.

11. "machizou7777". *Sun SPOT Demo – Playing a melody with a speaker*. May, 2008. Video. http://www.youtube.com/watch?v=ok9xoyaunq4

12. Maurer, John, A, IV. *The History of Algorithmic Composition.* March, 1999. Website/Paper. http://ccrma-www.stanford.edu/~blackrse/algorithm.html

13. Miranda, Eduardo. *Composing with Computers – Algorithmic Composition, Iterative Systems.* Lecture. Retrieved November 2008. http://www.inventionen.de/Studio/Miranda/Composition/Lecture1/CompLecture1.pdf

14. Nierhaus, Gerhard. *Algorithmic Composition: Paradigms of Automated Music Generation*. 2009. Springer-Verlag/Wein. Germany.

15. *Project Sun SPOT – Sun Small Programmable Object Technology.* Website. http://www.Sun SPOTworld.com/

16. Reiners, Paul. *Cellular automata and music*. May, 2004. Website. http://www.ibm.com/developerworks/java/library/j-camusic/

17. Portilla, Jorge, et. all. *Using Wireless Sensor Networks for an Interactive Musical Application*. Research Paper. http://www.cecs.uci.edu/~papers/date07_universitybooth/Sessions/Session7/S77.pdf

18. Taube, Heinrich, K. *Notes from the Metalevel: Introduction to Algorithmic Music Composition*. 2004. Taylor & Francis Group. London, UK.

19. Wallis, Isaac., Ingalls, Todd., Ellen Campana. *Computer-generating Emotional Music: The Design of an Affective Music Algorithm*. Proc. of the 11th Int. Conference on Digital Audio Effects (DAFx-08), Espoo, Finland, September, 2008

20. "Wireless Sensor Networks", Website. Retrieved November, 2008. http://en.wikipedia.org/wiki/Wireless_Sensor_Networks

21. Wolfram, Stephen. *WolframTones*. Website. Retrieved November, 2008. http://tones.wolfram.com/

22. "algorithmic.net: algorithmic composition resources", Website. Retrieved February, 2009. http://www.flexatone.net/algoNet/

23. Hevner, K. *Experimental Studies of the Elements of Expression in Music*. American Journal of Psychology, vol. 48, pp. 246-268, 1936.

24. Gundlach, R.H. *Factors Determining the Characterization of Musical Phrases*. American Journal of Psychology, vol. 47, No. 4, pp. 624-643, 1935.

25. Russell, J.A. *A Circumplex Model of Affect*. J. Pers. Soc. Psychol., vol. 39, no. 6, pp. 1161-1178, Dec. 1980.