Software Requirements Specification

CS684 Projects

Dancing Hexapod

Group G13

Kedar Tatwawadi(09D07022)
Manas Chaudhari (09D26003)
Siddharth Sarangdhar (09026007)
Ayesha Mudassir(09007014)

Table of Contents

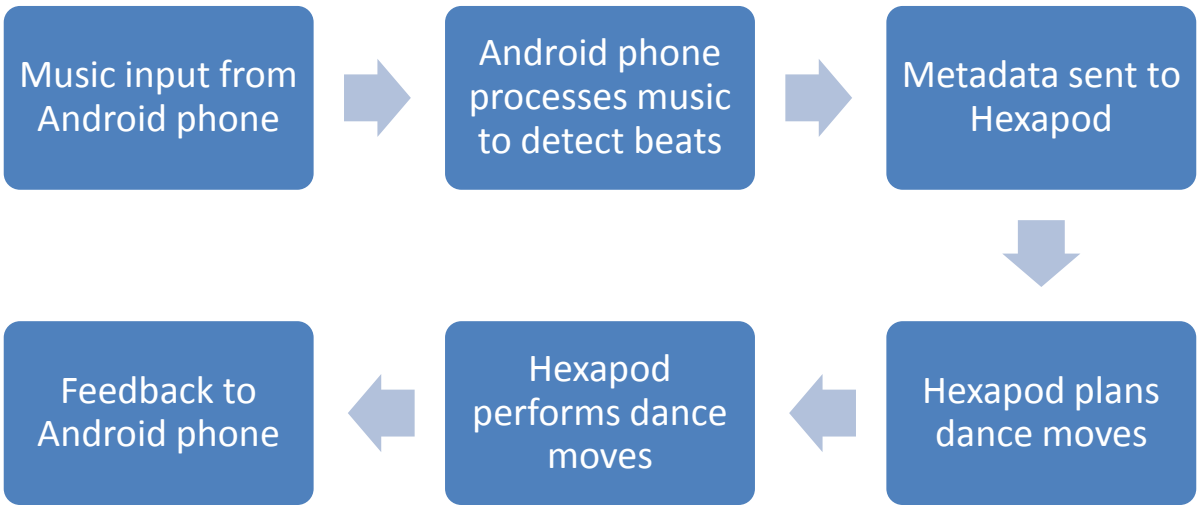
Introduction

We intend to design an Autonomous dancing hexapod, a Hexapod which will dance to any song played.

The user has to play any song on his Android Based Phone, and the hexapod will dance according to the song played. The Android phone will communicate wirelessly with the hexapod.

Our project has 4 important parts:

1. Sound processing- Designing the beat detection and beat identification algorithm.
2. Designing and Implementing different gaits(dance moves) for the hexapod.
3. Developing an android app for implementation of the sound processing algorithms developed.
4. Setting up wireless communication between the android phone and the hexapod.



Definitions, Acronyms and Abbreviations

1. **Gait (dance moves):** Wikipedia gives definition of gaits as:

… the pattern of movement of the limbs of animals, including humans, during locomotion over a solid substrate.

Hexapod robots are biologically inspired by Hexapod locomotion .In the project by "gait" we mean a dance move of the hexapod [defined by a set of sequence of motor actions]

2. **Phase and frequency of beats**: An important part of our project is identifying the beats in any given song. Typically in an upbeat song (dance songs) a wide variety of percussion instruments are used and they constitute the rhythmic element of the song.

These beats can typically be modeled as a superposition of periodic functions with some period and phase (time signature).
[Using sound processing our software will communicate the frequency of these beats and their time signature (phase) so that the bot can match its movements to the beat.]

3. **Metadata:** This is the data sent by the android phone to the hexapod wirelessly. It would consist of all the information necessary for the hexapod to plan and execute the dance moves. This information would be structured in such a way that the hexapod will have to perform minimal processing in order to dance.

4. **Dance Step:** Typically, songs have a beats frequency of 1 – 5 Hz. There is a constraint on speed of performing an action for the hexapod. Hence, any gait (dance move) is broken down into a sequence of basic dance steps. Each dance step will be executed at a beat.

5. **Bot:** Hexapod

References

- http://archive.gamedev.net/archive/reference/articles/article1952.html
- http://www.clear.rice.edu/elec301/Projects01/beat_sync/beatalgo.html
- Hexapod Dance video: http://www.youtube.com/watch?v=pXMnbNoccgA

Overall Description

Product Perspective:

Although dancing hexapods is not a new concept, most dancing hexapods in the market have hardcoded moves. Thus, they can perform only a fixed sequence of moves. We have proposed to make dancing autonomous in the sense that the bot will choose dance moves based on the tempo of the songs and beats detected.

Product Functions: It analyses song sound files and identifies the beats of the song. The information of these beats is then sent to the bot. The bot has some predefined dance moves and executes the appropriate dance step.

User Characteristics:

The user will use our Android phone app "Just Dance" as a controller to the bot. The UI for the app will be like that of any music player app .The user simply has to select a song and click on play.

Constraints (Challenges):

1. Sound processing: There are no standard techniques to detect beats in a song. A large number of percussion beats can be used to produce beats in a song. It will be challenging to come up with a good algorithm which can identify all such beats.
2. Gaits: We will try to implement as many different dance steps as possible. However programming different dance steps is tedious and time consuming.
3. Sync: Our objective will be to achieve complete synchronization. Essentially, we would ensure that the bot makes a best effort to a execute dance step at the planned time (time of beat). The speed and efficiency of sound processing and wireless communication are the main constraints.

Assumptions and Dependencies:

- Dependent on previous project based on hexapod
- Low latency in wireless communication.
- Horizontal smooth floor without any obstacles.

Alternate designs:

The system could have been implemented by using a DSP or any other processor instead of the Android phone. But we chose the Android phone because of the following advantages:

- Providing a user interface for the system

- Built in wireless communication API
- We can deploy multiple apps using different algorithms. Thus, it would be easy to switch between algorithms.
- Also, presence of UI makes it very convenient for debugging and development.
- Cost of DSP is very high compared to an Android phone.

Details

Functional requirement:

1. The hexapod should perform dance -moves after reading the metadata sent to it. (Liveness)
2. It should have at least 5 aesthetic moves to choose from.
3. Delay between the dance step and the corresponding beat must be low so that it is not apparent visually (Timeliness).
4. A UI through which the user will interact with the hexapod. The UI will allow all types of functions such as play/pause/change song.
5. The hexapod should not lose balance or cause damage to itself due to excess stress/strain (Safety).

Non-functional requirements:

1. The android phone must communicate wirelessly with the hexapod.
2. Hexapod movement should be fast and smooth. (This depends on the provided hardware)
3. Hexapod must not leave the arena.
4. The user must be alerted in case of failure.

Supportability

1. Use of standard libraries as far as possible for maintainability/sharing/reuse of code.
2. We will structure the entire code into many different functions.
3. We will use a 'top down' approach for programming i.e. we will first create the skeleton of the code and then create functions for each task.
4. We will maintain code documentation using doxygen.

Design Constraints

1. The Android device performing the sound processing should have a fast enough processor and adequate memory.
2. Floor is smooth and there are no obstacles in the arena.

On-line User Documentation and Help System Requirements

Use of doxygen for code documentation.

We will create a manual for the user which will explain basic functionality of the bot, how to set up the bot etc.

Interfaces

User Interfaces:

The input to our system is an audio file chosen by the user. The user will feed the input through an Android app.

Hardware Interfaces:

- Hexapod: The hardware interface would consist of the connection between the Hexapod's microcontroller and its various motor actuators (i.e. an interface to control each servo motors)
- Charging port for the hexapod.

Software Interfaces:

- Use of Android API
  - Reading audio file from memory card
- Standard sound processing library
- Reuse software developed in past projects related to hexapod

Communications Interfaces:

A protocol for communication between Android and hexapod

Quality Control

Test Data (in increasing order of complexity)

- Audio with only single periodic beat
- Audio having single beat but with variable periods
- Audio with multiple beats with different periods with same intensities
- Audio with multiple beats with different periods with different intensities

- Audio with lyrics or other instruments having rhythmic elements which are not beats.

Evaluating Project:

Our project has three parts which need to be evaluated. They can be evaluated independently.

1. *Detection of beats in music:*

   The different music types specified would be organized in increasing order of difficulty as explained above. One of the parameters for evaluating the progress would be the difficulty level of music reached for the beat detection algorithm.

2. *Performing hexapod moves:*
   - Number of dance moves designed for the bot.
   - Quality and design of the dance moves.

3. *Synchronization and planning of moves:*

   We would consider our project successful if the system is able to plan and execute dance steps in a fashion in which there is no visible delay. This is the most important part of the project as it also involves the application of concepts covered in the course.

   Delay between performing the move and beat can be another evaluating criteria.

Stages of Development:

Sound Processing algorithm development:

- Testing beat detection algorithm on a computer
- Finding periodic patterns in the beats.
- Test on various test inputs in increasing order of complexity

Android app development:

- Reading audio input
- Implementing beat detection on android

Hexapod:

- getting basic hexapod gaits working

- Defining various dance moves and breaking down dance moves into dance steps.
- Scheduling algorithm for choosing dance moves depending on sample beat sequence inputs
- Testing of zigbee module for wireless communication

Integrating the parts:

- Designing a data structure for metadata and protocol for communication
- Get the hexapod to accept inputs from the Android application.
- Implementing a feedback loop for the android phone and hexapod system.

Risk Management:

1. Achieving Real time sound analysis
    - *Reason:* Android device is not able to handle the Sound processing load.
    - *Workaround:* Instead of a full real-time system, we can do some pre-processing of the input.

2. Beats not getting detected in music
    - Try to design a technique to modify the input audio to make it suitable for beat detection. Example: eliminating non-percussion audio

3. Forcing hexapod to do what it cant
    - Initially, we can simulate the moves using flashing leds or by displaying messages before making the bot perform them

4. Music is faster than the bot could dance to
    - Moves would be divided into different steps. For example, moving a leg up could be a step. Performing one move per beat is very fast. So we would allocate the move across several beats so that the steps get aligned to the beats.
    - If the music is faster, we can increase the number of beats allotted for the move.