**Introduction**

AudioVisual is an OpenGL/Audio project that has the intention of providing an interface between graphical models and audio streams. The objectives of this implementation are:

* **Audio** *(Meta: These are goals not features)*
  + Smooth, error-free, realtime, threaded access to an audio buffer.
  + Accessible interface providing audio analysis metrics; including pitch tracking, energy analysis..
* **User Interface**
  + Modular implementation of the immediate-mode GUI
* **Visualisation Engine**
  + Simple graphics engine with: *lighting, AA, normal mapping, texture & model loading.*
  + Model-view based rendering
  + Action-queue framework to build behaviour.
  + A High-dimensional model, with a dimension-free polytope winding and interpolation algorithm.
* **Visualisations**
  + A set of visualisations, built within the bounds of the engine, with their own setting systems
* **Render Configuration**
  + The ability to run the application in two modes:
    - Realtime Mode: *View and configure audiovis from realtime buffers.*
    - Scripted Render Mode: *Configure audiovis in advance, and save output to a file*
* **Development Process**
  + Standardised Versioning: unit testing, installers, changelogs.

**Status**

**Requirements**

**Scheduled Features (Next version is V0.3)**

**Candidate Features**

1. **Audio:**
   1. **Buffer:**
      1. Audio Analysis Metrics: Either: Import GIST correctly *or* Implement DFT, energy metrics etc yourself.
      2. Long & Interpolated buffer: *The user should be able to choose not only the sample size of the recorded buffer, but a suitable multiple (1-224) of these samples to be stored in a longer term buffer. (Note: you should look up deques/queues for this)*
   2. **Stream**
      1. Configurable audio settings: *Build an interface to the Recorder that allows you to (a) see all stream properties via a public interface, (b) Modify specific stream properties, re-initialising PortAudio safely with different settings: Channel No, Left/Right, Sample Size, HostAPI, Device*.
      2. Audio files Pt I: Implementation: *Build a*
      3. Audio files Pt I: Abstraction: *The audio file stream should be an implementation of an abstract stream, sharing type information with the PortAudio recorder class.*
2. **User Interface**
   1. Main Window: *A window from which you can show/hide all other windows, itself toggleable with a hotkey.*
   2. **Audio Window**:
3. **Visualisation Engine**
   1. **High-Dimensional Model**:
4. **Render Configuration**
   1. Command-Line interface:
   2. Configuration XML files:
5. **Development Process**
   1. Implement Unit Testing for the main modules
   2. Create installation instructions for github, with a list of dependencies
   3. Make sure that all build configurations (x64 & x86 Debug/Release) work correctly
   4. Create a publishable installer, with a policy to update the installer
   5. Create

**Completed Features**