

Assessment – Advanced Computer Music

There are two programming projects for this course. Task 1 is to be completed for hand-in on Thursday of Week 6 of the Autumn term. Task 2 is to be handed in on the first Thursday of Week 1 in the Spring term. Each task is worth 50% of the marks. We will use electronic submission via Study Direct. For each task, you will upload a zip file containing:

- 1) Everything required to run the programs
- 2) An accompanying write-up of 1000 words

You must either use Mac OS X, or your code must be compatible in principle with cross-platform compilation. Whilst the exact package you use for each task is not set in stone, the recommended package for almost all work will be SuperCollider. You might also investigate (amongst others) Java, MATLAB, Csound, Processing and C/C++ code solutions, but should always check with the lecturer early on that your work will use suitable formats for submission. This can be discussed in the weekly workshop sessions.

Note that the upload limit on Study Direct is 20MB. If you need large audio files as part of the project, you should place them online within another zip and provide a link to download this in a README document with your submission. The only external data that will be accepted in this fashion is auxiliary audio or video data; no programming code should be placed externally.

As well as the code, each submission should be accompanied by a write-up detailing your solution to the task, with appropriate computer music context (you will need to do some further reading and will refer to different resources you've accessed). You should provide 1000 words (there is however no upper word limit, within reason). You may wish to submit audio examples of your solutions (perhaps placed on SoundCloud with a link provided), though this is optional, and direct assessment will be undertaken of any code whenever possible independent of this. For any complicated system using a non-standard set-up, you may be required to provide a demonstration to the lecturer and second marker to prove that the system is working, prior to or at 4pm on the day of submission itself.

Your solutions will be judged on evidence of learning from the lectures and workshops, coding elegance, creativity of work, and the contextualization in the write-up. Because these are creative tasks, they remain relatively open-ended: use your common sense when assigning proportional time to these based on the marks. However, guideline marking criteria are provided below so you know what to expect.

Hints:

- Code must run out of the box! Build things incrementally: set a modest target, make sure that works, then add extensions.
- Code must be laid out neatly and commented.
- Include all required sound samples and extra data
- Contextualise your work in your report: refer to appropriate research literature and artworks. Neat referencing is expected.

Task 1: Direct Sample Level DSP Routine for sound synthesis/audio effect

Implement a DSP (Digital Signal Processing) routine at the level of individual samples, which synthesizes and/or effects sound.

The implementation could involve SuperCollider language code using Signal and SoundFile, Java or Processing via the starting templates provided, MATLAB code, a SuperCollider plugin (via C code or FAUST), or even an Audio Unit, VST plugin or other demonstrable sample level routine. Well documented code and an explanation of the DSP principles are required for the write-up; relate your work to historical sound synthesis and processing research.

‘At the sample level’ is critical here; the point of this assignment is for you to demonstrate that you can work at the lowest level of sound, and not to take advantage of existing SuperCollider functionality (like a SinOsc UGen!) to do it for you. It does not matter if you choose to re-implement an existing method (like FM) or explore non-standard synthesis; but the creativity and effort in your response is important. Note that FM is quite straight-forward to achieve: building your own vocal tract physical model is more impressive. An advanced solution may incorporate both sound synthesis and audio effects, for example, a band-limited sawtooth wave with swept frequency, going through a reverb unit.

Task 2: Interactive Music System or Interface

EITHER:

Create a new MIDI or audio responsive interactive music system

OR:

Create a novel musical interface for live performance which incorporates a new physical musical controller. You might use one of the lab Arduino/Wiring boards and sensors on negotiation with the lecturer.

Your write-up should explain the principles and contextualise the system with respect to the work of others. You must also show some evidence of considering the user of the system! The most important keyword here is ‘interactive’; the interface or interactive system must be engaging to work with, and you must justify this in your write-up. Try not to build very simple one-dimensional interactions. This is your chance to make exciting new computer music machines and perhaps employ some AI!

Assessment criteria for the tasks:

85-100% Original and exciting creative solutions showing significant effort in their construction, which deeply acknowledge and analyse the network of prior art and technology upon which they rest.

70-85% Individual solutions with some original and well built code are presented which are carefully contextualized with respect to other work, demonstrating competent integration of computer music principles.

60-70% Good ideas are pursued, with a working practical instantiation and broad appreciation of context.

50-60% The solutions show some evidence of creative engagement without a fully convincing realisation, and mediocre understanding of the computer music background.

40-50% Minimal functioning solutions are provided which are poorly contextualized, with at least some redeeming feature of note.

30-40% Basic constructions are provided which show no real attempt to engage with the creative task, little appreciation of context, and may not fully function

15-30% Solutions are presented but are incomplete and do not function.

0-15% Little or nothing has been accomplished.

Task 1 Marking scheme:

Proportion of marks out of 100	Being tested	How to get high marks
10	Coding neat and commented	10 = industry level commenting and formatting
10	Code runs out of the box and is difficult to break, despite a large codebase	10 = no problems, no crashes, easy to get going, but a large codebase. Note that a three line program will not score well here since there is no challenge in getting it to run!
20	Write-up well contextualized, with appropriate references	Give evidence of wide ranging further reading which places this system in the context of computer music research and projects.
20	DSP theory	Demonstrate understanding of the DSP theory underlying your solution, both in the write-up, and in the code itself.
20	Originality of sonic output	High marks here correspond to a creative response which provides novel sonic stimuli, whether synthesizing new sounds, or manipulating soundfile or live input.
20	Architectural challenge	This category reflects that a three line program in SuperCollider to create white noise is far less demanding than a VST

		plug-in which does the same. To get high marks, either show that you can tackle a significantly new API (such as creating a VST plug-in), or demonstrate a large program in SC or Processing which presents a solution to a significant challenge.
--	--	--

Task 2 Marking scheme:

Proportion of marks out of 100	Being tested	How to get high marks
10	Coding neat and commented	10 = industry level commenting and formatting
10	Code runs out of the box and is difficult to break, despite a large codebase	10 = no problems, no crashes, easy to get going, but a large codebase. Note that a three line program will not score well here since there is no challenge in getting it to run!
20	Write-up well contextualized, with appropriate references	Show evidence of wide ranging further reading which places this system in the context of computer music research and projects.
20	Interaction level: is there a sense of mutual influence on musical outcomes, and considerate support of human participation? What does the system feel like to the user?	There may be some variation in interaction aims based on player or instrument paradigms, but justify your thinking in the write-up and try to build some sensitivity into your system to support real music making.
20	Input processing; what mechanisms for analyzing user input are explored?	This category rates the technical competence in using machine listening facilities, and interface building. Show you've picked up some techniques from the course and explored further reading.
20	Output generation: how profound are the sonic outputs?	The marker will rate the quality and range of output options allowed by the system. Aim for an extensive repertoire of appropriate responses, within an effective sound space.