КОМПЬЮТЕРНЫЙ АУДИОСИНТЕЗ ШТАТНЫМИ СРЕДСТВАМИ AUDACITY® С ВОЗМОЖНОСТЬЮ ИМИТАЦИОННОГО ДИЗАЙН-МОДЕЛИРОВАНИЯ НА ЯЗЫКЕ NYQUIST

COMPUTER AUDIO SYNTHESIS USING STANDARD TOOLS AUDACITY® WITH THE OPPORTUNITY OF SIMULATION DESIGN-MODELLING IN NYQUIST LANGUAGE

T. Taran

Summary. The article discusses general applied issues of computer audio synthesis based on Audacity® audio processing software. The suitability of application of major standard modules aimed at playback and generation of synthesized audio fragments in order to improve quality of the applied procedures when reproducing new audio forms obtained as a result of additive and subtractive types of audio synthesis is emphasized. Step-by-step actions of the application of audio synthesis modules corresponding to specific problem situations are developed. The circuits of regular modules of audio synthesis reproduction used in interface-oriented type of sound processing are schematically demonstrated. The issues of program — oriented approach to sound synthesis by using built-in Nyquist program language are mainstreaming. Some Nyquist scenarios touching LISP syntax are interpreted. The areas of Nyquist language application both at table-wave synthesis of sound and spectrum, frequency and amplitude modulations of audio signal are formulated. Program map of scenario actions emulation required when procedural editing of audio signal characterized as synthesized audio form is unveiled. General suggestions for improvement the processes of applied software based audio synthesis and software usage of macros, functions and some managing logical operators used in conditions of simulation design-modeling of sound as well are presented. Some listing and scenario expressions in Nyquist language used in designated specific individual cases of sound processing and helping to overcome many problem situations are thematically investigated.

Keywords: applied computer audio synthesis, Audacity®, standard modules of audio synthesis, simulation design — modeling, XLISP/ Nyquist, SAL/ Nyquist.

Таран Василий Васильевич
К.культурологии, АНОВО «Московский международный университет»; ФГБУН «Всероссийский институт научной и технической информации РАН»
allscience@lenta.ru

Аннотация. В статье рассматривается общая прикладная проблематика компьютерного аудиосинтеза на базе программного средства обработки звука Audacity®. Акцентируется пригодность применения основных штатных модулей, направленных на воспроизведение и генерирование синтезированных аудиофрагментов с целью повышения качества проведения прикладных процедур при воспроизводстве новых аудиоформ, получаемых в результате аддитивного и субтрактивного видов аудиосинтеза. Разработаны пошаговые действия применения модулей аудиосинтеза, соответствующие конкретным проблемным ситуациям. Схематически продемонстрированы цепи штатных модулей воспроизводства аудиосинтеза, использующиеся при интерфейсно-ориентированном типе обработки звука. Актуализируются проблемы программно-ориентированного подхода к синтезу звука путем встроенного языка программирования Nyquist. Интерпретированы некоторые Nyquist-сценарии, затрагивающие синтаксис XLISP. Сформулированы позиции применения языка Nyquist при таблично-волновом синтезе звука, а также при спектральной, частотной и амплитудной модуляциях аудиосигнала. Представлена программная карта эмуляции сценарных действий необходимых при процедурном редактировании аудио, характеризующегося как синтезированная аудиоформа. Представлены общие предложения по совершенствованию процессов прикладного аудиосинтеза на программной основе, а также по программному использованию макрокоманд, функций и некоторых управляющих логических операторов, применяемых в условиях имитационного дизайн-моделирования звука. Тематически проанализированы некоторые листинги и выражения в сценариях на языке Nyquist, применяемые в обозначенных конкретных частных случаях обработки звука и помогающие преодолеть многие проблемные ситуации.

Ключевые слова: прикладной компьютерный аудиосинтез, Audacity®, штатные модули аудиосинтеза, имитационное дизайн моделирование, XLISP/ Nyquist, SAL/ Nyquist.

В компьютерную эпоху обработки звука, когда главным преимуществом аудиоинженерии с точки зрения генерации акустических форм становится аудиосинтез, актуальной видится проблема его разработки и применения для различных нужд. На современном рынке программного обеспечения не так уж и много решений, способных обеспечить ультратонкую настройку всех необходимых параметров, соответствующих чёткой структуре аудиосинтеза, тем более на некоммерческой основе. Поэтому лучшим выходом являлось бы такое программное обеспечение, которое в равной мере удовлетворяло бы профессионалов в области аудиоинженерии и соответствовало бы концепции некоммерческого использования программных продуктов. И в роли такого программного обеспечения может выступать программный комплекс

Audacity®[1]. Возможности этой программы в области обработки аудиоматериала впечатляют. Программа и её аудиоинтерфейс четко продуманы и могут быть легко трансформированы в соответствии с текущим уровнем пользователя: начинающий специалист или профессионал. К тому же для специалистов в области аудиоинженерии, включающей в себя различные эксперименты по обработке аудиоматериала именно программными средствами, предусмотрены ряд специальных возможностей.

Одной из таких важных, по мнению автора, возможностей является поддержка специализированного языка программирования (применяемого в аудиопроизводстве и аудиоинженерии) Nyquist. Несмотря на то, что Nyquist узкоспециализированный язык программирования[2] его возможности представляют большой интерес с точки зрения искусственного аудиосинтеза. Вообще искусственный аудиосинтез[3] сегодня нашел свое применение в различных сферах. Данная технология активно используется при производстве различных музыкальных аудиопроизведений, прежде всего для определения жанровой направленности и идентификации посредством выделения определенных специализированных инструментов, акустических спецэффектов и смешивания аудиоформ. В цифровом аудиодизайне при проектировании различной аудиопродукции (акдиорекламы, автоинформаторов, отбивок и заставок в радиовещании, а также в цифровом кинематографе)[4]. В сфере машинно-лингвистического перевода аудиосинтез нашел свое применение в компьютерных словарях, переводчиках и справочниках. На его основе активно развивается технология произношения слов, словосочетаний и даже целых фрагментов текста. В образовании (дистанционное образование) аудиосинтез широко применяется для обучения, те же справочные системы на WEB-основе, лингафонные кабинеты с автоматическим аудиосопровождением. Также с внедрением спутниковых технологий навигации аудиосинтез не оставил без внимания и эту область. Здесь он используется для произношения топонимов и различной справочной историко-географической информации.

Итак, для анализа функциональных возможностей в области компьютерного аудиосинтеза Audacity® и её встроенного языка Nyquist нам необходимо разобраться в инсталляционных особенностях программы. Audacity® является кроссплатформенным программным продуктом, который может быть установлен под все современные операционные системы: MS Windows, GNU/Linux AppleMacOS. Штатные средства, имеющие графический интерфейс в Audacity®, располагают широкими возможностями по аудиосинтезу в области низкочастотных и высокочастотных сигналов. В первую очередь, речь идет о встроенных модулях: Chirp, DTMF Tones, Noise, Silence, Tone. Также наряду со штатными средствами аудиосинтеза предусмотрены несколько локальных плагинов для моделирования некоторых звуков на табличноволновой основе среди них Pluck, Rhythm Track, Risset Drum, Sample Data Import.

Для более глубокого понимания технических характеристик, перечисленных выше модулей, опишем их функциональные возможности. Модуль Chirp[5] — является генератором синтезированной звуковой формы, выражающейся заданными спектро-частотными характеристиками 3-х уровней. Частота — задаёт математические значения в Герцах. Амплитуда — устанавливает максимальное значение отклонения заданной величины. Интерполяция — обеспечивает поиск промежуточ-

[1] Audacity® — прикладное программное обеспечение, разработанное в целях осуществления качественной обработки звука на компьютерах различной архитектуры и поддерживающее различные операционные системы.

[2] Nyquist — специализированный язык программирования, использующийся в программной среде Audacity® в целях повышения качества обрабатываемого аудиосигнала, улучшения процессов, включающих в себя обеспечение контроля над обрабатываемыми аудиоформами и возможности тонкой настройки технических характеристик звука, в тех случаях, когда технический арсенал штатных интерфейсно-ориентированных модуей становятся недостаточным. Язык Nyquist основывается на интерпретаторе Lisp и во многом опирается на синтаксис Lisp, данное обстоятельство даёт возможность программировать нестандартные микропрограммы на Lisp/ Nyquist. В целом Nyquist поддерживает два родственных ему языка программирования XLISP и SAL. Язык SAL и XLISP также родственны друг другу и имеют схожую семантику, основное отличие заключается в символьной разметке синтаксиса. Язык Nyquist это удачное решение для синтезирования нелинейных аудиопотоков и создания креативных аудиокомпозиций. **Прим. автора.** Изложенные в статье программные концепции были апробированы автором в среде программирования Nyquist — версии 3.15.

[3] Искусственный аудиосинтез (цифровой аудиосинтез) — синтез различных звуков, получаемый в результате вычислительных действий компьютера под управлением языка программирования, объясняющего машине алгоритм действий. Принципиальное отличие от аналогового аудиосинтеза заключается в способе обработки аудиофрагментов. Аудиофрагменты аналогового синтеза сочетаются посредством соединения воспроизводящих через аудиодинамики различных частот, в то время как цифровой (искусственный) аудиосинтез выполняет последовательные действия с уже перезаписанной цифровой таблицей частот, каждый символ в такой таблице соответствует заданному диапазону спектро-акустических значений.

[4] **Прим. автора.** Более подробно ознакомиться с функционалом языка Nyquist и его применением при проектировании дизайна аудиопродукции (а также возможностями MIDI-синтеза — см. стр. 165–166) можно в следующей статье: Таран В. В. Проектирование дизайна аудиопродукции в программной среде Audacity® с применением языка Nyquist/ В. В. Таран// Современная наука актуальные проблемы теории и практики: Серия естественные и технические науки /// Информатика, вычислительная техника и управление.— 2019.— № 10.— С. 159–171. [ISSN2223–2966].

[5] Модуль имеет довольно странное название Chirp — в переводе с английского обозначает Щебет. Модуль получил такое название благодаря возможности моделировать короткие тона, которые местами бывают похожи на щебет птиц. Диапазон определяемых модулем частот может браться из любой спектро-частотной области аудиоформы между 1–1,5 Гц и половиной текущего уровня аудиопроекта.

ных значений величины по набору заданных значений. Поддерживает два типа представления интерполяции сигнала: линейный и логарифмический. Вид аудиоформы может быть представлен следующими характеристиками:

- Синусоидная
- Квадратичная — Зубцевидная
- Квадратичная — без сглаживания.

DTMF Tones[1] — встроенный штатный модуль программы, призванный обеспечить генерацию мультичастотного аудиосигнала, базирующегося на двух тонах. Изначально данный принцип синтезирования аудиосигнала использовался в автоматических устройствах набора определённой последовательности цифр, знаков и букв, которые кодировались в промежуточный сигнал разной длины и разного спектра с целью обеспечить вызовы абонентов в тональном режиме. Широкое распространение получил в двадцатом столетии с введением аналоговой коммутации радиорелейных линий связи. Цифры от нуля до девяти могут быть представлены промежуточными тонами и расшифруются впоследствии автоматическим коммутатором на узле связи. Сгенерированный сигнал может использоваться и в других направлениях телекоммуникации, он нашёл свое применение в радиовещании и на телевидении при проведении профилактических работ, связанных с ретрансляцией аудио сигнала с целью его декодирования на выходе, чтобы определить насколько точно проходит доставка сигнала. Минимальное отклонение от синусоидальной развертки говорит о неточности воспризведения спектро-акустических характеристик передаваемого в эфир аудиосигнала. В этой же области он зарекомендовал себя вроде метронома для синхронизации видиоконтента, передающего телесигнал в разных часовых поясах и с учётом региональных особенностей технической инфраструктуры.

Noise — модуль генерирует различные виды искусственных шумов: белый шум, розовый шум, броуновский шум. Белый шум используется в компьютерной аудиоинженерии с целью маскировки других звуков. При искажении частоты дискретизации может искусственно подмешиваться в аудиоформу, скрывая некоторые недочеты при повторном изменении частоты дискретизации (передискретизации) обрабатываемого аудиоматериала. Применяется на всех уровнях частот. Розовый шум может использоваться как тестовый материал, эмулирующий работу аналоговых радиоприборов. Также он может быть использован для повышения качества частоты дискретизации (в сторону её увеличения, к примеру, из исходной частоты дискретизации с 22050 Гц до 48 000 Гц, пропуская классическую отметку в 44100 Гц) только при ошибках, возникающих на низкочастотном уровне. То есть низкочастотная обработка сигнала, для его более точного воспроизведения на специализированной аппаратуре, подчеркивающей низкие частоты. Броуновский шум во многом схож с классическим синтезированным Розовым шумом. Единственное отличие (в прикладном понимании процесса обработки звука) проявляется при взаимодействии его спектра (выражающегося в математических значениях) со спектром обрабатываемого аудиосигнала. Сигнал, взаимодействующий с уже сгенерированным Броуновским шумом, может маскировать некоторые ошибки просчёта (рендеринга) проекта, связанные с внезапными или повсеместными провалами средних частот[2].

Silence[3] — используется для генерации псевдоамплитуды равной нулевому значению. Очень полезен при обработке проблемных мест аудиоформы особенно в промежутках между высокой и низкой амплитудами звукового потока. Tone — специфический модуль, позволяющий производить генерацию различных тональных конструкций при моделировании новых частот. Как и с модулем Chirp, модуль Tone поддерживает синусоидную[4], квадратичную[5], зубцевидную[6] и квадратичную без сглаживания[7] формы волны. Рассмотрим несколько ключевых цепей штатных объектно-ориентирванных модулей программы, целью которых будет являться дизайн-моделирование аудиоформы объектно-ориентированного типа.

Вариант № 1. Допустим нам необходимо сгенерировать полезный аудиосигнал, в котором будет содержать-

[1] DTMF (**D**ual-**T**one **M**ulti-**F**requency) двухтоновый мультичастотный сигнал, используемый для коммутации аналоговых линий связи. Кодирование осуществляется аддитивным способом при условии, что сигнал имеет синусоподобное представление.

[2] Примечание автора. Обычно подобный сгенерированный шум в инженерной практике применяется к уже завершенным (выделенным из программы обработки звука) проектам, объединенным в обычный звуковой файл формата RAW (Audio) либо WAV (PCM), имеющим отклонения фона по средним частотам в пределах чувствительности спектрограммы (на визуальном уровне) и на уровне слуха звукоинженера.

[3] Модуль Silence будет очень полезен при замене нежелательного спектра пустыми значениями. Например, при обработке вокала в процессе удаления одышки и прочих артефактов, мешающих адекватному восприятию голоса. Применяется в основном на этапе сведения дорожек, иногда при финализации композиции с целью увеличения пустого интервала между аудиопроизведениями.

[4] Синусоидная форма — представляет собой математическую кривую, описывающую гладкие повторные колебания.

[5] Квадратичная форма — фиксирует минимальные и максимальные спектро-акустические значения, имеющие идентичную продолжительность и устанавливает вертикальный ряд между ними. Такую аудиоформу отличает наличие идентичных пиков и уровней RMS.

[6] Зубцевидная форма — отличается постепенным прогрессивным увеличением амплитуды волны и далее сопровождается регрессивным спадом.

[7] Квадратичная форма (без сглаживания) внешне похожа на прямоугольное представление волны за исключением некоторых искажений на пиках. Увеличивает масштаб времени, чтобы сгенерировать тон.

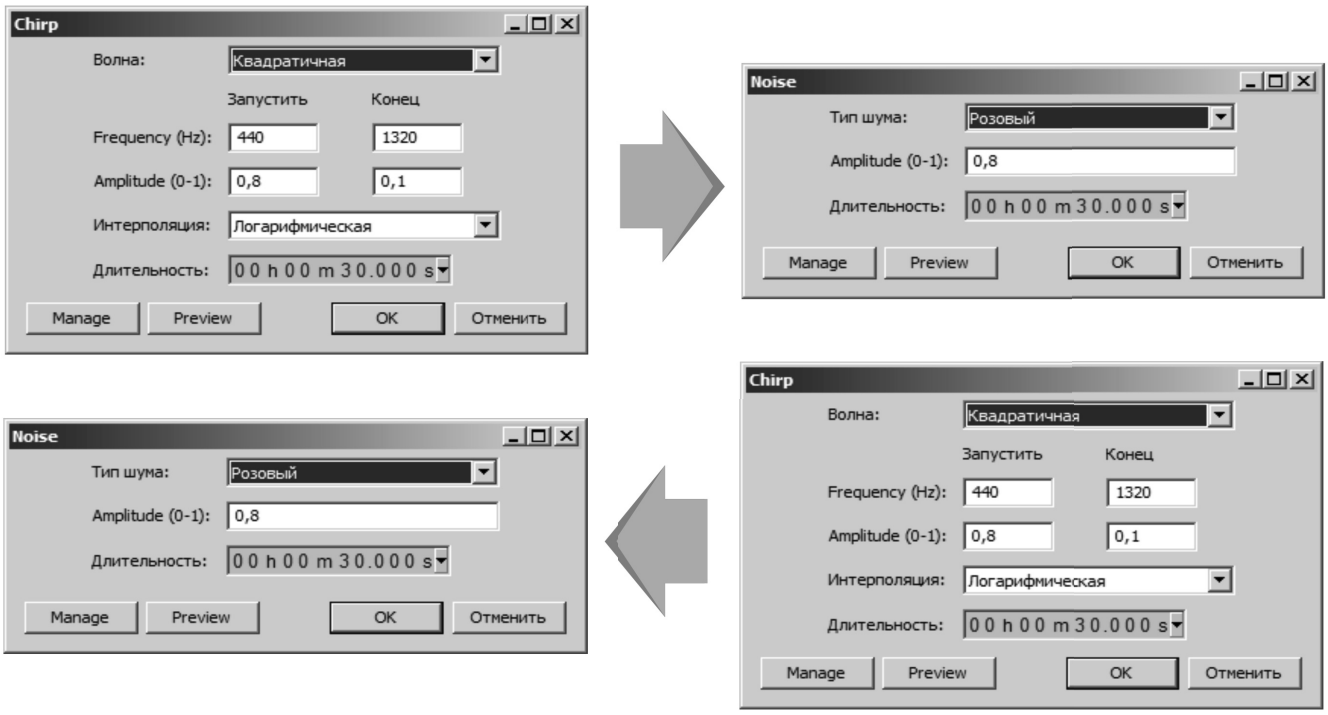

Рис. 1. Штатные модули обработки аудиоматериала для цепочки (Chirp ⟶ Noise, Noise ⟶ Chirp).

ся шум, плавно переходящий в тон. Для этого можно использовать следующую пару модулей:

Chirp ⟶ Noise, Noise ⟶ Chirp (1)

Chirp сгенерирует подходящую аудиоформу, а Noise в соответствии со своими возможностями позволит выбрать тип шума. Данная процедура соединения искусственного шума и специального тона может быть полезна при настройке различных радиопередающих устройств, в том числе и FM-передатчиков, для отладки и тестирования полного спектра воспроизводимых передающим устройством частот.

Как правило, различные тоновые характеристики позволяют получить довольно четкий результат как на цифровых, так и на аналоговых устройствах. Генерация тона и шума разных типов служит своего рода настроечной таблицей для передачи аудиоматериала.

Вариант № 2. В инженерной практике бывают случаи, когда при адаптации технологии VST[1] при подключении различных дополнительных модулей (плагинов), сосредоточенных на одном хосте, возникает необходимость проверки виртуальной маршрутизации аудиосигнала между программой и внешним модулем обработки звука. Чтобы проверить качество распределения аудиопотока, при межпрограммной маршрутизации нужно иметь под рукой обширные средства для дизайн-моделирования аудиоформы, и здесь очень кстати будет цепочка следующих модулей:

Chirp ⟶ DTMF Tones ⟶ Noise ⟶ Silence ⟶ Tone (2)

При необходимости элементы цепи могут выглядеть так:

DTMF Tones ⟶ Tone ⟶ Silence ⟶
Chirp ⟶ Noise (3)

В том случае, если аудиоматериал распределяется через подхост технологии VST, в зависимости от количества подключенных модулей актуальной становится цепочка (3).

Вариант№ 3. VST-технология позволяет подключать не только внешние модули обработки звука, но и виртуальные музыкальные инструменты: струнные, клавишные, ударные, духовые и прочие. Часть из этих технологий внутри себя использует алгоритмы табличноволнового синтеза. Для точной настройки таких ин-

[1] VST (**V**irtual **S**tudio **T**echnology) — программная микросреда, позволяющая межпрограммно взаимодействовать различным модулям, подключаемым к аудиоплагинам, для более эффективной обработки входного аудиосигнала. Основным достоинством данной технологии, по авторскому мнению, можно считать возможность обработки аудиоданных в режиме реального времени. В настоящее время поддержку и развитие этой технологии осуществляет немецкая компания Steinberg.

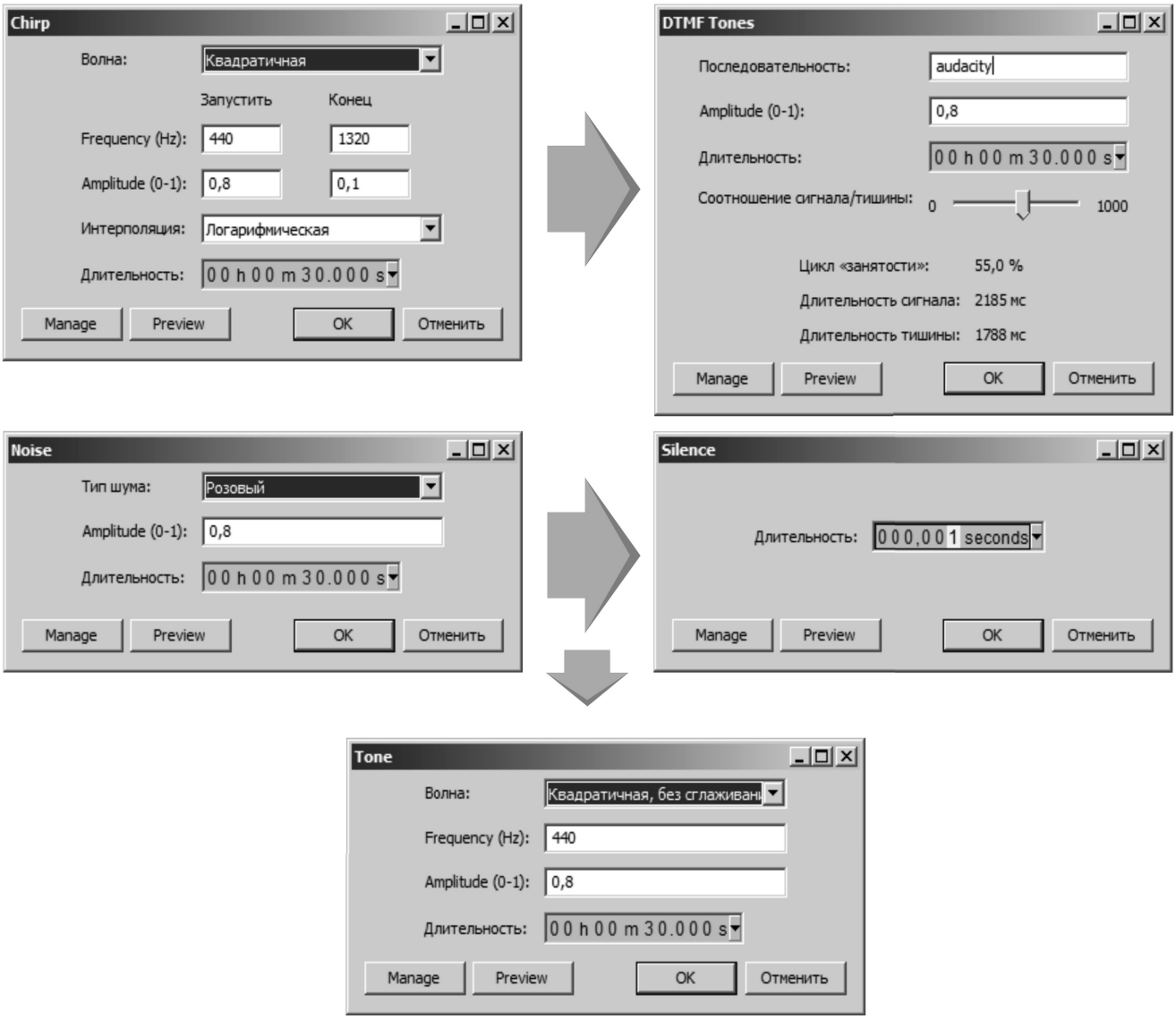

Рис. 2. Штатные модули обработки аудиоматериала для цепочки (Chirp → DTMF Tones → Noise → Silence → Tone).

струментов существует возможность синтезировать тона разных частот для создания эталона настройки для каждого типа компьютерных музыкальных инструментов, в этом случае возможны варианты построения модулей как показано (4), (5).

Tone → Noise → DTMF Tones → Silence → Chirp (4)

Silence → Chirp → Noise → Tone → DTMF Tones (5)

Четвёртый и пятый подварианты распределения модулей обработки звука при поочередности наложения синтезированных семплов также дают возможность создавать новые дизайн-эффекты для различных аудиокомпозиций.

Как было уже сказано выше, наряду со штатными модулями обработки аудиосигнала, существуют локальные модули, базирующиеся на технологии таблично-волнового синтеза: Pluck, Rhythm Track, Risset Drum, Sample Data Import. Pluck — данный модуль предназначен для генерирования тонального сигнала с быстрым или замедленным его исчезновением и регулируется в соответствии с предписаниями MIDI. MIDI-предписания могут использовать следующие параметры:

- C (примечание) 24, 36, 48, 60 (середина C, параметр по умолчанию), 72, 84, 96, 108
- C# (C-sharp) выше середины — C61.

Для более гибкой настройки доступны привилегированные режимы:

Рис. 3. Штатные модули обработки аудиоматериала для цепочки (DTMF Tones → Tone → Silence → Chirp → Noise).

- Режим резкого спада
- Режим плавного спада

Режимы применяются, если необходимо создать искусственные пороги спада между синтезируемыми фрагментами аудиоматериала. Rhythm Track — специальный модуль, призванный генерировать набор последовательностей регулярных импульсных колебаний в соответствии с заданным темпом. Также имеет возможность установки метронома, необходимого для точной звукозаписи.

- Action choice (Выбор действия)
  Отображает возможность типа генерирования дорожки
- Tempo (beats per minute) (Темп и удары в минуту)
  Устанавливает общее количество ударов в минуту (от 30 — до 300 b/m)
- Beats per measure (bar) (мера ударов)
  Регулирует удары по приоритету громкости первый удар обычно громче последующего от 1 до 20 b/m
- Swing amount (количество Swing)
  При нейтральном значении (0) поддерживаемые интервалы (–1,000 до +1,000) каждому удару предопределяет темп (удары в минуту). При -/+ значениях устанавливается задержка ударов для прогнозирования частоты колебаний.
- Number of measures (bars) (число мер)
  Опция служит для повторения ударов при установленных значениях мер. Значение по умолчанию — 16 мер. Комбинация темпа, ударов на меру, а также число мер определяет длину сгенерированной дорожки.

Рис. 4. Штатные модули обработки аудиоматериала для цепочки (Tone ⟶ Noise ⟶ DTMF Tones ⟶ Silence ⟶ Chirp).

- Optional rhythm track duration (minutes, seconds milliseconds) (Дополнительная продолжительность дорожки ритма (мили секунды)
  Поле ввода значений [мили секунд] разделенное пространством, или математическими единицами [секунды] присгенерированная дорожка ритма будет немного дольше, чем оригинальная продолжительность. Конец дорожки расширяется в целую меру, если вводимая продолжительность не производит полную заключительную меру. Используйте только целые числа.
- Start time offset (seconds) (запуск смещения времени по секундам)
  Принуждает дорожку ритма стартовать позже на Временной шкале, чем само начало (нулевые секунды). Максимум 30 секунд и значения по умолчанию — нуль.
- Beat sound (звук удара)
  Выбор звука используемого удара по умолчанию «метроном»
- MIDI pitch of strong click (Подача MIDI сильного щелчка)
  C-отметка 24, 36, 48, 60 (середина C), 72, 84, 96, 108
- C# (C sharp) выше середины C61
  Значение по умолчанию 92 (G#) — [G sharp].
- MIDI pitch of weak click (Подача MIDI слабого щелчка)
  Подача остающихся ударов в каждой мере. Значение по умолчанию 80 (G# октава ниже сильного щелчка).

Рис. 5. Штатные модули обработки аудиоматериала для цепочки (Silence ⟶ Chirp ⟶ Noise ⟶ Tone ⟶ DTMF Tones).

- Risset Drum — специальный модуль, имитирующий звук барабана, основанный на музыкальных наработках композитора Джина Риссета (Jean Claude Risset), базируется на технологии полосно-пропускающего фильтрованного шума, тона гармоник, а также высокой степени волны синуса.
- Frequency (Hz) Частота (Гц)
  Выполняет функции настройки «фундаментальной частоты» и главной ноты звука.
- Decay (seconds) Распад (секунд)
  Определяет длину звука барабана. Более длинные звуки могут казаться подобными гонгу.
- Center frequency of noise (Hz) Частота центра шума (Гц)
  Регулирует частоту центра полосового фильтра, который применен к шуму. Более высокие частоты могут звучать, как будто маленькие тарелки вибрируют с ударом барабана.
- Width of noise band (Hz) Ширина шумовой группы (Гц)
  Минимальные значения ширины полосового фильтра, которые могут быть применены к шуму. Более высокие частоты могут создать впечатление звука гонга.
- Amount of noise in mix (percent) Количество шума в смешивании (процент)

Количество шума в барабане представляет процент смешивания. Частоты близкие к максимуму могут быть похожи на ружейный выстрел, если другие средства управления оставляют в значениях по умолчанию.

- Amplitude (0–1) Амплитуда (0–1)
  Пиковая амплитуда сгенерированного звука
- Sample Data Import — модуль Демонстрационного импорта данных читает математические значения, соответствующие частотной амплитуде нижних, средних и высоких частот из обычного текстового ASCII-файла, и создает PCM-выборку для каждого чтения числового значения. Текстовой документ должен соответствовать формату ASCII, а математические значения должны форматироваться пробелами, разрывами строки либо вкладками.
- File name имя файла (Задаёт имя файла)
- File location (path to file) (Расположение файла) (указывает путь к файлу)
- Invalid data handling (Недопустимая обработка данных)
- Data format (Формат данных)
- Obtaining suitable data (Получение подходящих данных)

Наряду с довольно богатым спектром объектно-ориентированных средств по моделированию и компьютерному синтезу аудиоформ, предусмотренных штатными возможностями программы Audacity®, существуют также более тонкие средства, позволяющие синтезировать различные спектро-частотные характеристики с особыми техническими параметрами.

К таким средствам относится реализация терминальной основы (NyquistIDE Program) программы, взаимодействующей со специальным техническим языком Nyquist, призванный обеспечить полный контроль над выполнением процессов обработки звука, а также расширить те возможности программы, которые отсутствуют в объектно-ориентированных модулях. Открывает большие возможности по имитационному моделированию посредством аудиосинтеза, задействованного в обработке материала.

Стоит отметить, что инфраструктурная среда аппликационной основы NyquistIDE поддерживает связку Lisp[1]/Nyquist — SAL[2]/ Nyquist, данное обстоятельство позволяет найти новую грань в программировании аудиоформ. Связка Lisp/Nyquist — SAL/Nyquist позволяет эффективнее управлять синтаксисом и производить отладку микросценариев внутри среды NyquistIDE, фактически синтаксис SAL/Lisp инкорпорирован в программную среду Audacity® для отладки приложений, исполняемых в среде NyquistIDE. Данные языки похожи друг на друга, и их можно назвать родственными[3]. В обзорно-аналитической статье американского исследователя в области компьютерной аудиоинженерии Генриха Таубе (Heinrich Taube) подробно описываются прикладные процедуры в работе с языком SAL[4]. По мнению автора, язык Nyquist может применяться при следующих видах синтеза:

Спектральная модуляция

- Для оптимизации отображения визуальных характеристик обрабатываемого аудиосигнала
- Для межпрограммной (локальной) многоканальной технической маршрутизации синтезируемого аудиосигнала
- Для более тонкого контроля над процессами аудиосинтеза при выполнении задач, связанных с генерацией нового аудиоматериала.

Частотная модуляция

- При оптимизации объектно-ориентированных модулей для взаимодействия с другими модулями обработки звука, создание карты модулей для обработки звука
- При маршрутизации аудиосигнала, а также при воспроизведении трека в обход библиотек кодирования для воспроизведения напрямую
- При многодорожечном слиянии синтезированного аудиоматериала, имеющего различные спектральные характеристики для ультратонкой настройки

Амплитудная модуляция

- При моделировании изменений математических параметров несущего сигнала амплитудой
- При разработке утилитарных микропрограмм и сценариев для повышения эффективности вывода синтезированного аудиоматериала
- При моделировании спецэффектов различной сложности не предусмотренных штатными модулями обработки звука

Таблично-волновой синтез

- Для создания и коррекции последовательного воспроизведения фиксированных по хрономе-

[1] Lisp (**LIS**t **P**rocessing language) — универсальный алгоритмический язык программирования, способный решать задачи начального, среднего и высокого уровней в разных областях интеллектуальной деятельности.

[2] SAL (**S**imple **A**lgorithmic **L**anguage) — алгоритмический язык программирования, имеющий различные спецификации. По своей структуре очень похож на Алгол (ALGOL-Algorithmic Language). Хорошо зарекомендовал себя в области аудиосинтеза и общей прикладной обработки звука.

[3] McCarthy J. «SuperCollider: A New Real Time Synthesis Language», Proceedings of the International Computer Music Conference, Hong Kong, 1996., Shalit, A. The Dylan Reference Manual. Addison-Wesley, Reading, 1996.

[4] Taube H. SAL: a simple algorithmic language in common music / University of Illinois // School of Music [электронный ресурс] pp.121–124.

тражу аудиоформ, представленных в ячейке памяти в табличном виде
- Для изменения высоты тона сгенерированного аудиоматериала при фрагментарном распределении аудиопотока.
- Для коррекции смещения фаз при линейном стерео потоке[1].

Рассмотрим некоторые примеры использования языка Nyquist для синтеза и моделирования различных спектро-акустических аудиофрагментов и инструментальных партий. Пример располагаемый ниже[2] предусматривает создание аудиофрагмента FM-синтеза посредством спектро-волновой таблицы, где за основу взят фрагмент вокальной партии.

```
(defun vocrap (&optional (pch 16) (dur 1))
(if (not (boundp '*voc-table1*)) (mk-voc1-table))
(fmosc pch (stretch dur (pwl 0 3 0.1–20 0.2 20 0.3 30
0.4–10 0.5 15 0.6 0
0.8–30 1 60 1)) *voc-table1*))
```

Функция позволяет генерировать специальный тест в целях точности обращения к *voc-table1* для проверки инициализации функции. В случае возникновения ошибки по инициализации и обращению к функции запускается проверка введенной переменной для обнаружения взаимосвязи.

В случае с глобальными переменными определение связи обуславливается при присвоении им значений. При введении boundp функция берет атом[3]. Функция предиката boundp при составлении программного кода может интерпретироваться в разных технико-синтаксических конструкциях SAL/XLISP [1]. Приведем пример определения mk-voc1-table, где нужно произвести замену имени файла в зависимости от конфигурации системы:

```
(defun mk-voc1-table ()
(if (not (boundp 'voc-snd1))
(setf voc-snd1 (s-read "./test/voc1.snd")))
(setf *voc-table1* (list voc-snd1 16 T)))
```

Чтобы обеспечить вызов vocrap нужно применить следующий код:

```
(play (seqrep (i 4) (vocrap)))
```

Для синтезирования сложных аудиоформ типа звучания аналогового синтезатора (фрагмент тона) пример фрагмента на языке Nyquist может быть изложен так:

```
(defun warble (&optional (dur 1) (pch 60))
(stretch dur
(sum (mult
(env 0.017 0.1 0.004 1 0.7 0.8 1)
(amosc pch (fmosc (hz-to-step 8)
(pwl 0 4 0.2–4 0.56 9 0.7 0 1–8 1))))
(mult (stretch 0.96 (env 0.2 0.09 0.07 0.92 0.8 0.6 1))
(amosc pch (fmosc (* pch 1.414)
(pwl 0.2 80 0.5 4 0.9 1120 1 200 1)))))))
```

Ключевыми звеньями программной цепочки, указанной на листинге выше, здесь являются объекты pwl и env. Чтобы эффективнее исполнять программный код и для увеличения скорости обработки данных, может потребоваться межпрограммная маршрутизация, обеспечить которую может специальный протокол OSC[4]. Тем самым открывается возможность по оптимизации загрузки процессора в высокоразрядной среде операционной системы MS Windows обрабатываемыми данными [2, стр.141–142].

Объект pwl (кусочно-линейная функция) определяет набор параметров в соответствии с заданными техническими характеристиками, в данном случае используется привязка ко времени (см. рисунок).

Объект env (конвертация) задаёт обычные звуки, имеющие низкочастотный уровень. Функция основывается на принципе умножения конверта, содержаще-

[1] **Прим.автора.** Не путать с процессами преобразования, применяемыми при фазовой модуляции. Речь идёт исключительно о прикладных аспектах связанных со сдвигом фазы на виртуальном канале в целях восстановления общей картины звучания. При этом фаза несущих колебаний не изменяется пропорционально информационному сигналу.

[2] Отдельные фрагменты, схематически иллюстрирующие базовые (шаблонные действия) на языке Nyquist были взяты за основу из нескольких источников: Dannenberg R. B. Nyquist Reference Manual Version 3.15 (2013, 2014, 2015, 2016, 2017, 2018)/Carnegie Mellon University — School of Computer Science, Pittsburgh, PA 15213, U.S.A.-September 11, 2018 p.276; Dannenberg R. B. Nyquist Reference Manual Version 2.36 (2007)/Carnegie Mellon University — School of Computer Science, Pittsburgh, PA 15213, U.S.A.— 5 March 2007 p.205; URL: https://www.audacity-forum.de/download/edgar/nyquist/nyquist-doc/examples/rbd/10-sequence-example.htm. **Прим. автора.** Некоторые программные формулировки, изложенные на языке Nyquist и некоторых диалектах семейства LISP/XLISP, а также языке SAL — доработаны и объяснены автором, с учётом имитации конкретных ситуаций.

[3] Для того чтобы установить отметку обозначения атома (символьного выражения) в LISP (s-expression), необходимо задействовать одинарную кавычку либо имя и значение вводимой переменной true — обозначает, что переменные связаны.

[4] OSC (**O**pen **S**ound **C**ontrol) — Открытый контроль звука, прикладной протокол, обеспечивающий межпрограммную коммуникацию и маршрутизацию данных, прежде всего определяемых по математическим параметрам, идентифицирующим контрольные (промежуточные) значения звука (громкость, амплитудные характеристики и т.д.). Для правильной его эксплуатации в среде MS Windows необходимо ознакомиться с инсталляционными особенностями данного протокола, в частности проверить корневую директорию SystemRoot. Подробнее о пересмотре некоторых взглядов на протокол открытого звукового контроля, и о некоторых его новых возможностях можно узнать из оригинальной статьи: Dannenberg R., Zhang C. «O2: Rethinking Open Sound Control» in Proceedings of the 42nd International Computer Music Conference, Utrecht: HKU University of the Arts Utrecht, 2016, pp. 493–496.

```
-------------------------------------------------
        test acoustic signal input/output — Nyquist
-------------------------------------------------
(pwl 0 4 0.2 -4 0.56 9 0.7 0 1 -8 1))))
Фрагмент длительности воспроизведения(с повторами)№1
0 4 0.2     |(root*1*)|           |*|
0 5 0.2     |(root*2*)|           |*|
0 6 0.2     |(root*3*)|           |*|
0 7 0.2     |(root*4*)|           |*|
-------------------------------------------------
(pwl 0.2 80 0.5 4 0.9 1120 1 200 1)))))))
Фрагмент длительности воспроизведения(с повторами)№2
0.2 80 0.5 (root*1*)      (точка фрагмента времени)
0.3 80 0.5 (root*1*)      (точка фрагмента времени)
0.4 80 0.5 (root*1*)      (точка фрагмента времени)
0.5 80 0.5 (root*1*)      (точка фрагмента времени)
-------------------------------------------------
```

го спектро-акустический фрагмент на другой соседний фрагмент, тем самым обеспечивая прямой последовательный синтез, учитывающий межфрагментарные характеристики. Данный метод и исследуемый программный объект может быть полезен и при моделировании звуковых форм, а также создании взаимосвязи между видеофрагментами и звуком[1], где видео моделирует синтезированный аудиосигнал [3, стр. 3–4].

(env 0.2 0.09 0.07 0.92 0.8 0.6 1))

Для того чтобы осуществить стыковку фрагментов и обеспечить их последовательное умножение вводится функция mult[2]. Фрагменты могут иметь разное количество входных фаз. Количество входных фаз определяется типом аудиофрагмента (моно, стерео, квадро).

Фрагмент, имеющий четыре и более фаз, может быть сгенерирован огибающей функцией. Пример, описанный в специальном учебнике по Nyquist (Nyquist Reference Manual), хорошо иллюстрирует применение функции env.

play seq(seq(env-note(c4), env-note(d4)) ~ 0.25,
seq(env-note(f4), env-note(g4)) ~ 0.5,
env-note(c4))

Пример приводится для более наглядного представления, как осуществляет свою работу функция env в программной среде. Пример показывает конвертацию, которая умножается на аудиофрагмент, в профессиональных кругах данная модель аудиосинтеза называется кольцевой модуляцией. Nyquist использует декларативно-функциональный стиль написания кода, поэтому каждое действие обрабатывается довольно быстро [4, стр. 2–3]

Оператор фрагмента (~) используется для изменения продолжительности. Для того чтобы узнать больше о преобразованиях и абстрактных поведениях, поддерживаемых языком Nyquist, необходимо обратиться к третьей главе (Nyquist Reference Manual/ Behavioral Abstraction — Chapter 3). Чтобы оптимизировать время исполнения некоторых действий и более подробно представить вычислительные процессы, происходящие в реальном времени, аккумулируемые Nyquist, важно представлять структуру временных данных языка Nyquist [5, стр.3–4]. Для комбинации звуков и различия между начальными, промежуточными и конечными выборками, а также в целях уравнивания значений динамического диапазона аудиофрагмента допустимо использовать следующие координаты:

Для расположения каждого звука в соответствии со временем:

[1] **Прим. автора.** Подробнее об описании технологии манипулирования изменяющимся во времени спектром звука, использующим синхронное потоковое видео, можно узнать из статьи Dannenberg R., Neuendorffe T. «Sound Synthesis from Real-Time Video Images» Proceedings of the 2003 International Computer Music Conference. San Francisco: International Computer Music Association, pp. 385–388.

[2] Mult — функция, которая задает параметры простого (последовательного) умножения, представленные параметры также могут быть выражены в числах. При смешивании заданных параметров данная функция используется для масштабирования звука заданными параметрами. В процессе перекрестного умножения аудиофрагментов устанавливается максимальный демонстрационный уровень.

Таблица. Константы, используемые при имитационном дизайн-моделировании аудиоформы и для задач, связанных с широкополосной аудиокомпрессией сигнала

№	Константа	Значения, подверженные динамике			
	lppp	-12.0 (dB)	- 8	-5	-3
	lpp	-9.0	- 6.0	-3.0	-1.5
	lp	-6.0	- 3.0	-1.5	-1.0
	lmp	-3.0	- 0.0	-0.0	-0.0
	lmf	3.0	- 0.0	-0.0	-0.0
	lf	6.0	- 3.0	-1.5	-1.0
	lff	9.0	- 6.0	-3.0	-1.5
	lfff	12.0	- 8	-5	-3
	dB0	1.00	dB		
	dB1	1.122			
	dB10	3.1623			
Продолжительность					
	s(Sixteenth)	0.25			
	i(elghth)	0.5			
	q(Quarter)	1.0			
	h(Half)	2.0			
	w (Whole)	4.0			
sd, id, qd, hd, wd = точки продолжительности st, it, qt, ht, wt = продолжительность возрастающая в трое.					
Язык	Арифметические функции				
LISP	Random (random n)	Вычисляет случайное число между 0 n –1 n верхняя граница (целое число) returns — случайное число			
LISP	Rrandom (rrandom)	Вычисляют случайное реальное число между 0 и 1 включительно Returns — случайное число (с плавающей запятой)			

(sim s1 s2 s3 …)

Для расположения каждого звука последовательно в соответствии со временем:

(seq s1 s2 s3 …)

Для определения сдвигов фрагментов соответствующих секундам:

(at t s)

Подробные характеристики очень хорошо изложены автором языка Nyquist (Roger B. Dannenberg) в обзорной статье: «Nyquist: Язык для композиции и синтеза звука»[1].

Для того чтобы упростить работу с синтезированием сложных форм при имитационном дизайн-моделировании, можно использовать предопределенные константы, предусмотренные Nyquist. Список констант представлен в таблице.

Для создания разряженных и уплотненных аудиотекстур используется частотная модуляция (FM), примером здесь может послужить генерация ключевого аудиофрагмента для инструментальной партии:

```
(defun mod (dur)
(stretch dur
(mult (pwl 0 1000.2 200.5 8000 1 100 1)
(fmosc c4 (pwl 0 1.5 3.25 1.74 1)))))
(defun blurp (dur)
(fmosc c3 (mult (osc 07 dur) (mod dur))))
```

Уплотненные текстуры могут пригодиться в визуальных и текстовых представлениях при интерактивной обработке аудиосигнала [6].

Здесь dur (Durations) задает продолжительность (константы) (см. раздел продолжительность таблицы), stretch расширяет продолжительность, fmosc определяет поведение частоты модуляции для генерации широкого спектра аудиофрагментов.

[1] Dannenberg Roger B. Nyquist: A Language for Composition and Sound Synthesis/ Machine Tongues XIX: Computer Music Journal, 21(3):50–60 (Дополнительно: Roger B. Dannenberg, «The Implementation of Nyquist, a Sound Synthesis Language», Computer Music Journal, 21(3) (Fall 1997), pp. 71–82.)

Так же как было упомянуто выше, Nyquist обладает довольно серьезными возможностями по спектральной модуляции сигнала. Особенности прикладной спектральной модуляции сигнала заключаются в возможности моделирования установленных спектро-акустических значений путём совмещения одного спектра с другим. В среде NyquistIDE хорошо реализованы и интерфейсно-ориентированные функции, помогающие при составлении программного кода, так для синтеза определённых композиций предусмотрено соответствующее расширение pmorales с микропрограммой (pjmg.lsp)[1]:

Простой Синтез (Simple Synthesis)

A4.lsp — Форма колебания + Огибающая кривая, Модуляция огибающей кривой с шумом

A5.lsp — Форма колебания + Огибающая кривая, Модуляция частоты

A6.lsp — Форма колебания + Огибающая кривая, Модуляция частоты, 2

Аддитивный (дополнительный) синтез (Additive Synthesis)

b1.lsp — Звуки похожие на гонг

b2.lsp — Спектральный анализ гармонического сочетания (аккорда) по Риссе[2]

b3.lsp — Колокол Риссе

b4.lsp — Непрерывное управление по тангажу (под углом тангенса) с помощью LFO (генератора низкой частоты)

b7.lsp — Риссе, тибетский

b8.lsp — Риссе, барабан

b9.lsp — Риссе, Бесконечное Глиссандо

c1.lsp — Случайные Сигналы

partial.lsp — Колокол

Субтрактивный Синтез (Subtractive Synthesis)

buzz.lsp — Жужжание с Формантными фильтрами

Синтез Карплуса — Стронга[3] (Karplus Strong Synthesis)

d1.lsp — Простой КАРПЛУС–СТРОНГ(Simple KARPLUS-STRONG)

ks.lsp — Алгоритм Карплус-Стронг (Karplus-Strong Algorithm)

FM-синтез (FM Synthesis)

e2.lsp — Динамическая Спектральная Эволюция Чаунинга

Физическое Моделирование (Physical Modeling)

phm.lsp — Физическое Моделирование Флейты

При моделировании спектро-акустических значений путём совмещения одного спектра с другим необходимо задать слой (L).

```
                                    L=fm-tone/hz
(defun fm-tone (step mi1 mi2 mi3)
```

$$L = \frac{\texttt{(defun fm-tone (step mi1)}}{\texttt{setf}};$$

$$L = \frac{\texttt{(defun fm-tone (step mi2)}}{\texttt{setf}};$$

$$L = \frac{\texttt{(defun fm-tone (step mi3)}}{\texttt{setf}}$$

$$\texttt{partial step} = \frac{\texttt{control-srate-abs *sound-srate*}}{\texttt{pwl 0 mi1 0.5 mi2 1 mi3 1}}.$$

Заданный слой умножает спектро-акустические характеристики и повышает амплитуду сигнала.

```
(defun fm-tone (step mi1 mi2 mi3)
(let ((hz (step-to-hz step)))
(setf mi1 (* mi1 hz))
(setf mi2 (* mi2 hz))
(setf mi3 (* mi3 hz))
(fmosc c4 (partial step
(control-srate-abs *sound-srate*
(pwl 0 mi1 0.5 mi2 1 mi3 1))))))
```

Модулируемый аудиосигнал обусловлен ячейкой из трёх тонов FM. При заданных параметрах было возможно использовать только один тон. Для того чтобы удостовериться, что у сигнала есть амплитуды во многих частотах, можно воспользоваться следующим кодом:

```
(defun mod-snd ()
(sum
(fm-tone c3 15 20 15);; подстройка FM-параметров[4]
(fm-tone d3 15 20 15);; подстройка FM-параметров
(fm-tone e3 15 20 15)));; подстройка FM-параметров
```

Для того чтобы умножить коэффициенты оригинального аудиофрагмента, необходимо вычислить будущую амплитуду другого аудифрагмента. Для подобных действий необходимо задействовать классы, чтобы произвести модуляцию. Объекты приведенного класса отвечают на запросы: next. В такой ситуации: next отправляется обоим источникам (см. рисунок).

В противном случае без процедуры *тестирования* спектральная модуляция не сгенерирует новый аудиосигнал.

[1] Поддерживает вспомогательные функции по синтезу: randi1, randi2, randh1, rndh2.

[2] Жан-Клод Риссе (Jean-Claude Risset) французский композитор, внёсший большой вклад в развитие компьютерной музыки. Один из первых композиторов, занимавшийся экспериментами в области синтеза звука, на основе его экспериментов впоследствии были смоделированы различные музыкальные инструменты, в том числе духовые.

[3] Александр Стронг (Alexander Strong) изобрёл алгоритм, а Кевин Карплус (Kevin Karplus) первым произвёл анализ его работы.

[4] Точная подстройка параметров частотной модуляции выделенного субканала. Определяет степень отклонения частотных характеристик в диапазонах c3 = 15, 20, 15.

```
(setf fft-modulator-class (send class :new '(src1 src2)))

   (send fft-modulator-class :answer :isnew '(s1 s2) '(
       (setf src1 s1)
       (setf src2 s2)))

   (send fft-modulator-class :answer :next '() '(
     (let ((frame1 (send src1 :next))
           (frame2 (send src2 :next))
           n half_n)
       (cond ((and frame1 frame2)
              ; multiply frame2 by the amplitude coefficients of frame1
              (setf (aref frame2 0) (* (aref frame2 0) (aref frame1 0))) ;; DC
              (setf n (- (length frame1) 1))
              ; Subtracted 1 because we already took care of DC component
              (setf half_n (/ n 2)) ; integer divide
              (dotimes (i half_n)
                (let* ((i2 (+ i i 2))
                       (i2m1 (- i2 1))
                       (amp (sqrt (+ (* (aref frame1 i2m1) (aref frame1 i2m1))
                                     (* (aref frame1 i2)   (aref frame1 i2))))))
                  (setf (aref frame2 i2m1) (* (aref frame2 i2m1) amp))
                  (setf (aref frame2 i2) (* (aref frame2 i2) amp))))
              (cond ((= n (+ half_n half_n 2)) ;; n is even -> nyquist component
                     (setf (aref frame2 n) (* (aref frame2 n) (aref frame1 n)))))
              frame2)
             (t nil)))))

   (defun make-fft-modulator (src1 src2)
     (send fft-modulator-class :new src1 src2))
   (defun mod-test ()
     (let ((fs 512)) ;; frame size
       (play-fft1 (make-fft-modulator
                    (file-fft1 sfn fs fs)
                    (make-fft1-iterator (mod-snd) fs fs))
                  fs)))
```

Структура кода: next представляется громоздкой из-за векторного умножения. К сожалению, прародитель Nyquist язык Lisp не идеален при работе с числовыми массивами и алгоритмами [7,8]. Представленный ниже код позволит протестировать новый класс:

```
(defun mod-test ()

(let ((fs 512));; frame size

(play-fft1 (make-fft-modulator

(file-fft1 sfn fs fs)

(make-fft1-iterator (mod-snd) fs fs))

fs)))
```

В данном случае мы осуществляем передачу в iterators для звука в файле и моделируемого аудиосигнала. Пример приведенный выше лучше всего применим при работе с моносодержимым, например вокальная партия или речь диктора.

Можно попытаться имитировать расширение моносодержимого, уменьшая параметр шага до файла (file-fft), таким образом модулируемый сигнал увеличивается во времени. Расширение моносодержимого может быть полезным при стыковке низкочастотных и высокочастотных аудиофрагментов, повышая качество аудиосинтеза [9].

Так же можно поэкспериментирвать с размерами FFTs и задействовать тип телосложения при использовании параметра (Fs). Если вы убедитесь в слишком большой объемности телосложения, то можно разрешить отдельные гармоники аудиоматериала, и напротив если он будет иметь малый объем, то можно не разрешать форманты. Оптимальными числами являются 512 и 256 при работе с частотой дискретизации 44 100 Гц.

Проведённый в статье анализ показывает возможность использования программного обеспечения Audacity® распространяемого на открытой основе в целях проведения экспериментов по синтезированию речи и смешанного содержимого. Возможности встроенной среды исполнения команд NyquistIDE, открывают широкие возможности по интеллектуальному программированию и разработке миниприложений на основе составленных автором алгоритмов предпринимаемых действий, служат безупречной основой для имитационного дизайн моделирования в области прикладной аудиоинженерии. Аудиосинтез, получаемый в результате обработки звука на платформе Audacity®, является очень качественным и пригоден для решения задач, связанных с инженерными разработками в сфере аудиотехнологий, для разработки стилевых конструкций и новых жанров в области электронной музыки, в области коммуникации (на уровне аудиоинтерфейса) при человеко-машинном взаимодействии. Достоинство языка Nyquist заключается ещё и в том, что предусмотренные программой алгоритмы DSP[1] разработаны таким образом, чтобы внутренние циклы вычислений имели хорошо организованную структуру и были оптимизированы под машинный код, но не под относительно медленный транслируемый Lisp-код. Данная рукопись и её отдельные положения служат хорошим подспорьем для инженеров-программистов в области аудиоинформатики и компьютерной инженерии в целом с точки зрения применения показанных в статье технологий и для усовершенствования механизмов обработки звука.

С точки зрения теоретических основ компьютерной аудиоинженерии статья служит необходимым мостиком, соединяющим теоретические наработки в данной области исследований (торию языков программирования, теорию звука) с практико-ориентированным подходом (новые методы в аудионженерии и вопросы разработки компьютерных приложений для аудио постпроизводства), позволяющим более эффективно (качественно) приращать научное знание с помощью теоретической и экспериментальной методологии, открывая новые грани научной дискуссии в данной проблемной области.

ЛИТЕРАТУРА

1. Dannenberg R. B. Nyquist Reference Manual Version 3.15 // Carnegie Mellon University — School of Computer Science/ Pittsburgh, PA 15213, U.S.A. 11.08. 2018, p.276.
2. Dannenberg R. B. Nyquist Reference Manual Version 2.36 // Carnegie Mellon University — School of Computer Science/ Pittsburgh, PA 15213, U.S.A. 05.03. 2007, p.205.
3. Dannenberg R., Bernstein B., Zeglin G., Neuendorffer T., «Sound Synthesis from Video, Wearable Lights, and 'The Watercourse Way'» In Proceedings: The Ninth Biennieal Symposium on Arts and Technology. NewLondon, CT: Ammerman Center for Arts and Technology, (2003), pp. 38–44.
4. Roger B. Dannenberg, «The Implementation of Nyquist, a Sound Synthesis Language», Computer Music Journal, 21(3) (Fall 1997), pp. 71–82.
5. Dannenberg R. B. Mercer C. W. «Real-Time Software Synthesis on Superscalar Architectures» in Proceedings of the 1992 International Computer Music Conference, International Computer Music Association, (October 1992), pp. 174–177.
6. Dannenberg, R. B. «Combining Visual and Textual Representations for Flexible Interactive Audio Signal Processing», Proceedings of the 2004 International Computer Music Conference. San Francisco: International Computer Music Association, pp. 240–247.
7. Betz D.M. XLISP: An Object-Oriented Lisp — Version 3.0 (28.01.1997), 18 Garrison Drive Bedford, NH 03110, p. 41.
8. History, management and technical instructions of a programming language XLISP (The XLISP family) [электронный ресурс] — URL: http://www.edm2.com/index.php/The_XLISP_family (дата обращения к источнику: 15.11.2019).
9. Dannenberg R.B. & Derenyi, I. (1998). Combining instrument and performance models for high-quality music synthesis. Journal of New Music Research, 27(3), pp. 211–238.

Журнал «Современная наука: актуальные проблемы теории и практики»

[1] DSP (**D**igital **S**ignal **P**rocessor) в значении машинной обработки программными языками (**D**igital **S**ignals **P**rocessing) — процессор цифровой обработки сигналов и процессор цифровой обработки сигнала. В нашем случае допустимо употреблять эквивалентный термин ЦОС (Цифровая обработка сигнала).