УДК 791.4

Особливості аналізу звукових ефектів в обчислювальній платформі iOS

Борисов Г. О., ORCID 0000-0003-2780-2700
e-mail borusov5364@gmail.com
Факультет електроніки, каф. ЗТ та РІ
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Київ, Україна

***Анотація*—Побудовано алгоритм дослідження найбільш розповсюджених звукових ефектів на основі використання можливостей інформаційної платформи iOS. Описано архітектуру "Модель-вид-контролер" у формі компонентів додатку, що є базовим підгрунтям до проведення експерименту з аналізу особливостей мультимедійного контенту на основі використання мобільних гаджетів та спеціальної мови програмування. Побудовано простий алгоритм використання спеціальних фреймворків в середовищі XCode, який дозволив провести тестування на обраному семплі ряд звукових ефектів. Знайдено, що побудований алгоритм аналізу семплів на основі ресурсів iOS дозволяє значно спростити сам процес дослідження звукових сигналів різної природи, через те, що інструменти симулятора описуються через код мови програмування, і отже їх налаштування є зрозумілим підходом до проведеного аналізу. Крім цього, описані елементи в статті можуть бути використані при створенні власних звукових ефектів, та на різних етапах конструювання віртуальних генераторів звуку.**

Бібл. 7, рис. 6, табл. 1.

Ключові слова — клас; ефект; додаток; мультимедійний контент; платформа; симулятор; доріжка; мікшер; моделювання.

I. Вступ

Система iOS — обчислювальна платформа і разом перспективне сучасне середовище для розробки різних мультимедійних додатків. Використовуючи ресурси цієї платформи у користувача мобільного гаджету з'являються по суті необмежені можливості зі створення різних звукових сигналів, які, наприклад, можуть повністю замінити звуки природи, звуки тварин а їх обробка може стати основою до створення різних спеціальних звукових ефектів, звукових фільтрів та відповідних інструментів. Перевагою такого підходу є прозорість усіх дій з звукової обробки, адже код, спеціальна мова програмування, спеціальні аудіо модулі є відкритими в середовищі мережі Інтернет і розробники власноруч на основі засобів тестування можуть провести цілий ряд експериментів з обраним звуковим семплом. Наприклад, мобільна платформа в середовищі iOS дозволяє не тільки прослуховувати звукові файлі, але й завдяки відкритому коду, обробляти звуковий ряд, накладаючи на нього різні звукові ефекти, наприклад, цифрову затримку, притлумлення, і взагалі, у розробника з'являється можливість провести дослідження з імітації звукової картини віртуального приміщення, де були б враховані переважно усі головні акустичні характеристики. При цьому, зрозуміло, що усі налаштування можна провести лише в коді, враховуючи відомі особливості розповсюдження звукових сигналів, зокрема час загасання сигналів, тобто час реверберації, особливості відбиття сигналів їх поглинання на певній віртуальній поверхні. Фактично, на основі інформаційної платформи iOS та засобів запису, прослуховування та відповідних бібліотек можливо забезпечити корегування параметрів наявних звукових сигналів, не лише через стандартні звукові ефекти, але й шляхом внесення змін у записані сигналограми. Так, традиційно використовуючи спеціальні програми, наприклад Sound Forge, Adobe Audition звуковий файл можна обробляти лише як єдиний цільовий об'єкт, тоді як на основі засобів підходу XCode можна провести корекцію окремих звукових семплів за окремими складовими на основі відповідних відкритих бібліотек, та написаної програми на основі коду.

Програмування для системи iOS являє собою абсолютно новий досвід, що відрізняється від будь-якої іншої платформи. Поширення мобільних пристроїв означає, що люди можуть використовувати програмне забезпечення всюди, де вони захочуть, без безпосереднього зв'язку з персональним комп'ютером, без врахування ліцензійних обмежень програм і все це за допомогою як телефонів, так і інших мобільних аксесуарів, наприклад Apple Watch. З появою системи iOS 10, середовища Xcode 8, мови Swift 3 у користувача з'являється можливість створити програмну основу для аналізу звукових сигналів. Через це, відкрита платформа є базисом до проведення різних досліджень з наявним мультимедійним контентом, причому в даному випадку можна використовувати не лише стрімовий потік записаної телевізійної

програми, але й сигнал, який записано через використання мікрофону мобільного гаджету. В останньому випадку, через спеціальний програмний код в середовищі iOS можна взагалі змінити параметри запису аудіовізуальної інформації в режимі реального часу.

Метою роботи є розроблення алгоритму, який дозволив би провести тестування різних звукових семплів через розроблену програмну основу на основі коду в мові програмування платформи iOS. При цьому, в рамках дослідження необхідно врахувати той факт, що усі складові з дослідження звукових сигналів є відкритими та прозорими і визначаються виключно особливостями xCode.

II. Особливості програмного налаштування

По суті, всі додатки чинної версії iOS використовують архітектуру «Модель — вид — контролер» (Model-View-Controller, MVC). Модель, вид і контролер — це три основних компоненти додатку (з точки зору його архітектури) в операційній системі iOS (рис.1) [1].

Модель (Model) містить основні дані, наприклад, змінні, підключення до зовнішніх RSS-каналів або зображень, функції і числову інформацію. Цей шар повністю відділяється від візуального оформлення, і через це можна легко змінити вигляд дисплея, і при цьому це не вплине на самі дані [2].

Вид (View) відповідає за стиль відображення інформації на дисплеї. В якості уявлення виступає екран з графічними елементами, на якому, наприклад, можна змінити стиль або видаляти різні елементи [2].

Контролер (Controller) управляє запитами користувача і використовує модель для реалізації необхідної реакції [2].

XCode надає широкі можливості для роботи з мультимедійним контентом [3-5]. У плані роботи зі звуком, компанія Apple знаходиться попереду своїх конкурентів. Так, компанія пропонує удосконалений інструментарій для програвання, запису та обробки треків [6-7]. Завдяки цьому на AppStore можна побачити величезну кількість додатків, які так чи інакше працюють з аудіоконтентом. В їх число входять програвачі з добрим звуком (Vox), аудіоредактор з інструментами для редагування і накладення ефектів (Sound Editor), додатки, які змінюють голос (Voicy Helium), різні симулятори музичних інструментів, які дають досить точну імітацію відповідного звучання (Virtuoso Piano), і навіть симулятори DJ-установок (X Djing).

Для роботи з аудіо сигналами розробники Apple визначили фреймворк AVFoundation, який об'єднує в собі кілька інструментів. Так, інструмент AVAudioPlayer використовується для програвання одиночного треку. Інструмент AVAudioRecorder є основним для запису звуку з мікрофону. Натомість, для накладення будь-якого ефекту, програвання кілька доріжок одночасно, мікшування їх, обробка чи редагування аудіо або запис звуку з виходу певного аудіовузла, використовується інструмент AVAudioEngine. Найбільше даний клас приваблює можливістю накладення ефектів на трек. Якраз на основі цих ефектів побудовано сьогодні безліч додатків з еквалайзерами з можливістю змінювати голос. Крім того, Apple дозволяє розробникам створювати свої ефекти, генератори звуку і інструменти.

AVAudioEngine є групою сполук аудіовузлів, які генерують і обробляють аудіосигнали і група за формою має аудіовходи і виходи. Цю групу можна описати програмно як схему аудіовузлів, які розташовані певним чином для досягнення необхідного результату при використанні різного звукового контенту.

Наведемо послідовність роботи зі звуком в платформі iOS:

1) створення AVAudioEngine;
2) створення вузла / вузлів AVAudioNode;
3) прикріплення вузлів до AVAudioEngine (attach);
4) з'єднання вузлів між собою (connect);
5) запуск AVAudioEngine.

Програмний елемент AVAudioEngine працює з різними “дочірніми” класами відповідно класу AVAudioNode, які до нього прикріплені. Як і основний програмний елемент, AVAudioNode має шини входу і виходу, які можна розглядати як точки з'єднань.

Працюючи з цим класом, необхідно в певному порядку виконати процедури ініціалізації і методи, аби всі операції здійснювалися коректно. В таблиці 1 наведено обрані для дослідження “дочірні класи” AVAudioNode

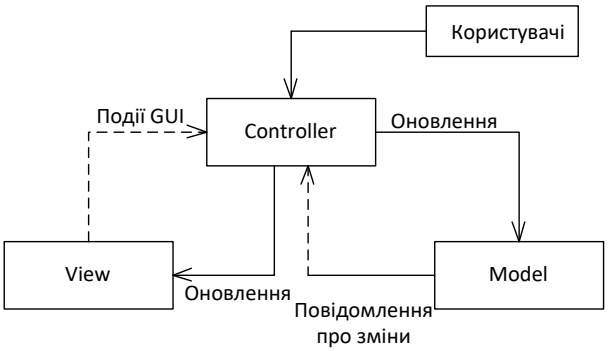

Рис. 1 Архітектура MVC

Таблиця 1 Дочірні класи AVAudioNode

Найменування класу (вузла)	Опис
AVAudioPlayerNode	Використовується для відтворення аудіо
AVAudioUnitEQ	Еквалайзер
AVAudioUnitDelay	Затримка
AVAudioUnitReverb	Ефект відлуння
AVAudioUnitTimePitch	Ефект прискорення або уповільнення, і зміни тону
AVAudioUnitDistortion	Реалізує багатоступінчастий ефект спотворення
AVAudioMixerNode	Мікшер

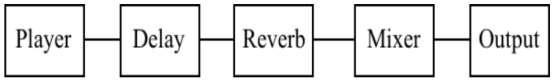

Рис. 2 Схема підключення вузлів

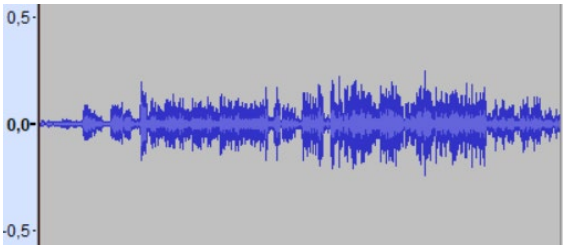

Рис. 3 Сигналограма аудіо файлу до накладання ефектів

Зазначимо, що обрані класи дозволяють на обраному звуковому треку провести тестування звукових ефектів не вносячи при цьому зміни у вихідний матеріал мультимедійного контенту.

Елемент AVAudioPlayerNode програє звук або з заданого буфера уявлення AVAudioBuffer, або з сегментів аудіофайлу, відкритого за допомогою AVAudioFile. Вузли AVAudioUnitEQ, AVAudioUnitDelay, AVAudioUnitReverb, AVAudioUnitTimePitch, AVAudioUnitDistortion — це вузли, які дозволяють накласти певні ефекти для набору файлів аудіо контенту. Вони мають широкий функціонал і різні налаштування. Наприклад, елемент AVAudioUnitDistortion дозволяє накласти 22 різних ефектів на одну аудіо доріжку, тоді як клас AVAudioUnitReverb має 12 моделей, які описують ефект луни в різних приміщеннях.

AVAudioMixerNode на відміну від інших вузлів, у яких суворо по одному вхідному вузлу, може мати кілька входів. Тобто, можна об'єднати вузли-програвачі з вузлами ефектів і еквалайзером, і після чого вихід вузла підключається вже до AVAudioOutputNode для виведення кінцевого звуку.

На рисунку 2 наведена схема експерименту з обробки заданого аудіо сигналу через використані вузли-класи обчислювальної платформи iOS.

Визначено, що модуль Player є програмною реалізацією програвача на основі команд та процедур xCode. Так, окрім традиційних елементів програвання звукових сигналів (Play, Stop, Pause, Rev) в платформі iOS у розробника є можливість заблокувати управління підключеного файлу, змінити порядок підключення і програвання, а також через код можна адаптувати програвач не лише під звукові сигнали, але й під будь-яку візуальну інформацію. Наприклад, після корегування програмного коду, можна завантажувати цифрові зображення з різною роздільною здатністю, і застосувати до них різні цифрові фільтри зображення. Іншою функцією програмного програвача є можливість після доопрацювання здійснювати стиснення мультимедійного контенту на основі вбудованих алгоритмів стиснення.

Наведена схема підключення вузлів-класів для обробки звукових файлів на рисунку 2 має характерні ознаки послідовного виконання операцій обробки. Такий підхід обумовлений тим, що у розробника на основі класів та коду є можливість змінювати параметри обробки в режимі реального часу, тобто безпосередньо при програванні файлу. Це є перевагою, адже можна провести аналіз записаної інформації і внести необхідні корективи ще до того моменту, коли буде виконано етап накладання звукового треку на динамічне зображення. Іншими словами, можливо провести попередню обробку, ще до того, як параметри цифрового звукового файлу будуть змінені відповідно до вимог спеціалізованої програми для роботи зі звуком на персональному комп'ютері.

На рисунку 3 наведена сигналограма файлу до накладання обраних звукових ефектів, відповідно до даних таблиці 1 та схеми експерименту з рисунку 3.

На рисунку 4 зображені інструменти які необхідні для зчитування, обробки та програвання семплу. Звуковий двигун "engine" за структурою містить в собі як вузли для програвання семплу та накладання на нього ефектів, так і вузли, які відповідають за програвання audioPlayer та накладання ефектів reverb, delay. Крім цього скрипт 1 представляє елемент file за допомогою якого можна відкрити для зчитування семпл.

```
let engine = AVAudioEngine()
var file: AVAudioFile = AVAudioFile()
let audioPlayer: AVAudioPlayerNode = AVAudioPlayerNode()
let reverb: AVAudioUnitReverb = AVAudioUnitReverb()
let delay: AVAudioUnitDelay = AVAudioUnitDelay()
```

Скрипт 1 Створення аудіо двигуна та вузлів для диформації треку

Скрипт 2 відображає прикріплення до звукового двигуна вузлів з заданими параметрами, і після прикріплення, всі вузли були з'єднані між собою згідно з схемою (рис. 2).

```
engine.attach(audioPlayer)

delay.delayTime = TimeInterval(pitchSlider.value)
engine.attach(delay)

reverb.loadFactoryPreset(AVAudioUnitReverbPreset.cathedral)
reverb.wetDryMix = reverbSlider.value
engine.attach(reverb)

engine.connect(audioPlayer, to: delay, format: nil)
engine.connect(delay, to: reverb, format: nil)
engine.connect(reverb, to: engine.mainMixerNode, format: nil)
engine.connect(engine.mainMixerNode, to: engine.outputNode, format: nil)
```

Скрипт 2 Реалізація накладення ефектів на аудіо трек

На семпл, сигналограма якого показана на рисунку 4, в рамках дослідження було накладено два ефекти: ефект відлуння та ефект затримки. Для ефекту відлуння (рис.5) була обрана модель собору. На виході було отримано семпл, який накладався сам на себе через одну секунду та при цьому імітувалось враження у слухача композиції, що цей звук відтворюється в соборі (рис.6).

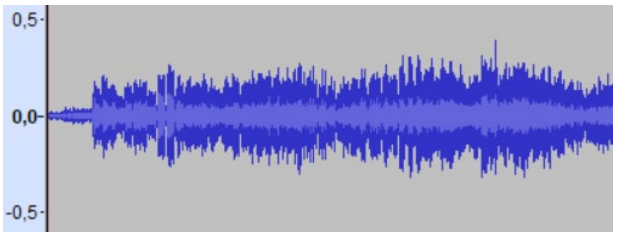

Рис. 4 Сигналограма семплу після накладання ефекту затримки

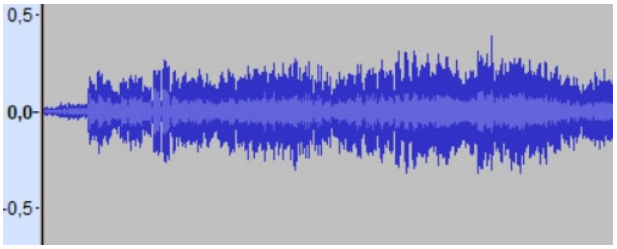

Рис. 5 Сигналограма семплу після накладання ефекту відлуння

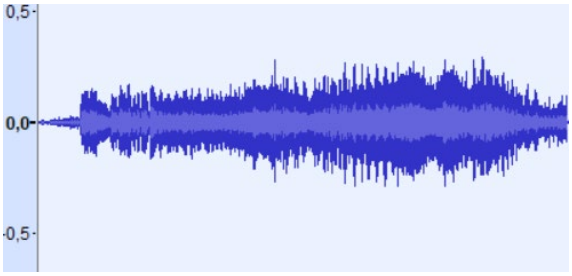

Рис. 6 Сигналограма семплу після обох ефектів

На основі проведеного дослідження можна підкреслити, що звуковий сигнал, до якого було застосовано послідовно 2 ефекти визначається тим, що енергетична складова сигналу значно збільшилась, що обумовлено акустичними особливостями віртуального приміщення-собору. При цьому, тональні складові сигналу для обраного семплу відповідають при роді у випадку обробки сигналів, яку можна провести в спеціалізованих програмних комплексах. Крім цього, з аналізу сигналограм на рисунках 4-6 можна підкреслити той факт, що аналіз звукових сигналів засобами обчислювальної платформи iOS вирізняється своєю простотою, але і режим програвання і режим обробки сигналів можна визначити через скрипт, який написано в середовищі xCode.

Висновки

Побудовано алгоритм дослідження найбільш розповсюджених звукових ефектів на основі використання можливостей інформаційної платформи iOS. Описано архітектуру “Модель-вид-контролер”
у формі компонентів додатку, що є базовим підгрунтям до проведення експерименту з аналізу особливостей мультимедійного контенту на основі використання мобільних гаджетів та спеціальної мови програмування. Побудовано простий алгоритм використання спеціальних фреймворків в середовищі XCode, який дозволив провести тестування на обраному семплі ряд звукових ефектів. Знайдено, що побудований алгоритм аналізу семплів на основі ресурсів iOS дозволяє значно спростити сам процес дослідження звукових сигналів різної природи, через те, що інструменти симулятора описуються через код мови програмування, і отже їх налаштування є зрозумілим підходом до проведеного аналізу. Крім цього, описані елементи в статті, зокрема схема підключення вузлів в середовищі Xcode обчислювальної платформи iOS може бути використана для додавання власних звукових ефектів між елементами програвача та виходу, та на різних етапах конструювання віртуальних генераторів звуку.

Перелік посилань

[1] L. O. Velikanova, « PRINCIPLES OF DEVELOPMENT OF APPLICATIONS FOR IOS OPERATING SYSTEM », Krasnodar, 2017.

[2] V. Nahavandipur, «iOS. Priemyi programmirovaniya [iOS. Programming techniques] », Piter, 2014.

[3] Vasiliy Usov, «Swift. Osnovyi razrabotki prilozheniy pod iOS i macOS [Swift. Basics of developing applications for iOS and macOS]», 3-e izd., dop. i pererab. — SPb.: Piter, 2017 — 368p.

[4] Noyburg M., «Programmirovanie dlya iOS 7. Osnovyi Objective-C, Xcode i Cocoa [Programming for iOS 7. Basics of Objective-C, Xcode and Cocoa]», Vilyams, 2014, 384 p.

[5] Dalrimpl M., Knaster S., «Objective-C 2.0 i programmirovanie dlya Mac [Objective-C 2.0 and Mac programming]», Vilyams, 2009, 320 p.

[6] Velikanova L.O., Gayvuk A.R., «Primenenie mobilnyih tehnologiy dlya avtomatizatsii biznes-protsessov na torgovom predpriyatii. Ekonomicheskoe prognozirovanie: modeli i metodyi [The use of mobile technologies to automate business processes in a commercial enterprise. Economic Forecasting: Models and Methods]», Voronezh, 2016, 314 p.

[7] V. Nahavandipur «iOS. Razrabotka prilozheniy dlya iPhone, iPad i iPod [iOS. Application development for iPhone, iPad and iPod]», Piter, 2013. — 864 p

Надійшла до редакції 10 квітня 2019 р.

УДК 791.4

Особенности анализа звуковых эффектов в вычислительной платформе iOS

Борисов Г. А., ORCID 0000-0003-2780-2700
e-mail borusov5364@gmail.com
Факультет електроніки, каф. ЗТ та РІ
Национальный технический университет Украины
"Киевский политехнический институт имени Игоря Сикорского"
Киев, Украина.

***Аннотация*—Построен алгоритм исследования наиболее распространенных звуковых эффектов на основе использования возможностей информационной платформы iOS. Описаны архитектуру "Модель-вид-контроллер" в форме компонентов приложения, является базовым основанием к проведению эксперимента по анализу особенностей мультимедийного контента на основе использования мобильных гаджетов и специального языка программирования. Построен простой алгоритм использования специальных фреймворков в среде XCode, который позволил провести тестирование на выбранном семпле ряд звуковых эффектов. Найдено, построенный алгоритм анализа сэмплов на основе ресурсов iOS позволяет значительно упростить сам процесс исследования звуковых сигналов различной природы, потому что инструменты симулятора описываются через код языка программирования, и, следовательно, их настройки понятно подходом к проведенного анализа. Кроме этого, описаны элементы в статье могут быть использованы при создании собственных звуковых эффектов, и на разных этапах конструирования виртуальных генераторов звука.**

Библ. 7, рис. 6, табл. 1.

Ключевые слова — класс, эффект, приложение мультимедийным контентом, платформа, симулятор, дорожка, микшер, моделирование.

UDC 791.4

Features of Analysis of Sound Effects in the Computing Platform iOS

H. O. Borysov, ORCID 0000-0003-2780-2700
e-mail borusov5364@gmail.com
Faculty of Electronics.
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute",
Kyiv, Ukraine

***Abstract*—The iOS system is a computing platform and together is a promising modern environment for the development of various multimedia applications. Using the resources of this platform, the user of the mobile gadget appear essentially unlimited opportunities to create different audio signals. In terms of working with sound, Apple is ahead of its competitors, and this is no accident. The company offers good tools for playing, recording and processing tracks. Thanks to this, on the AppStore you can see a huge number of applications that somehow work with audio content. These include players with good sound (Vox), audio editors with tools for editing and applying effects (Sound Editor), voice changing applications (Voicy Helium), and various musical instrument simulators that give a fairly accurate simulation of the corresponding sound (Virtuoso Piano) , and even simulators DJ-installations (X Djing). To work with audio, Apple provides the AVFoundation framework, which combines several tools. If you need to impose any effect, play several tracks at the same time, mix them, do processing or editing audio, or record sound from the output of a certain audio node, AVAudioEngine will help you with this. Most of all this class attracts the possibility of imposing effects on the track. Just on the basis of these effects is built a lot of applications with equalizers and the ability to change the voice. In addition, Apple allows developers to create their own effects, sound generators and instruments. The algorithm for the study of the most common sound effects based on the use of the capabilities of the iOS information platform has been built. The Model-View-Controller Architecture is described in the form of application components, which is the basis for an experiment on analyzing the features of multimedia content based on the use of mobile gadgets and a special programming language. A simple algorithm for using special frames in an XCode environment was constructed, which allowed testing on a selected sample a series of sound effects. It has been found that the built-in algorithm for analysis of samples based on iOS resources can greatly simplify the process of studying sound signals of various nature, because the simulator tools are described through the code of the programming language, and therefore their configuration is a clear approach to the analysis. In addition, the described elements in the article can be used to create their own sound effects, and at various stages of the design of virtual sound generators.**
Ref. 7, fig. 6, tabl. 1.

Keywords — class, effect, application, multimedia content, platform, simulator, track, mixer, simulation.