ЗМІСТ

Перелік умовних позначень, символів, скорочень і термінів

ПЗ – програмне забезпечення;

ГР – графічний рушій;

VR – віртуальна реальність.

GVRf – Gear VR framework

SDK – software development kit

MFCC – мел-частотні кепстральні коефіцієнти

HMM – прихована модель Маркова

DCT – дискретне косинусне перетворення

Рендеринг – в комп'ютерній графіці — це процес отримання зображення за моделлю з допомогою комп’ютерної програми. Тут модель – це опис тривимірних об’єктів (3D) на визначеній мові програмування і у вигляді структури даних. Такий опис може містити геометричні дані, положення точки спостерігача, інформацію про освітлення. А зображення – це цифрове растрове зображення.

Стереоскопія – це техніка для створення ілюзії глибини картинки за допомогою стереоскопічних засобів для бінокулярного зору.

ВСТУП

Майбутнє віртуальної реальності часто представляють у вигляді антиутопії – мільйони самотніх людей, занурившись у фантастичні світи, проводять купу часу з величезними пристроями на головах.

Насправді, мільйони цих людей не будуть такими вже й самотніми. Адже вони проводитимуть час разом з друзями, рідними, колегами і новими знайомими. Вони зможуть спілкуватися і взаємодіяти, незважаючи на фізичну відстань між ними.

Найшвидша і найбільш ефективна взаємодія між людьми відбувається безпосередньо за допомогою усного мовлення. За допомогою мови можуть бути передані різні почуття і емоції, а головне – корисна інформація. Необхідність створення комп'ютерних інтерфейсів звукового вводу-виводу не викликає сумнівів, оскільки їх ефективність заснована на практично необмежених можливостях в будь-яких областях людської діяльності.

Віртуальна реальність дуже швидкими темпами набирає популярність. В даний час розробляється дуже багато ігрових додатків, які націлені продемонструвати колосальні можливості цієї технології. Виходячи з того, якими темпами вона розвивається, можна передбачити, що ця технологія дуже швидко проникне у всі галузі нашого життя. Постає проблема того, що з часом люди захочуть взаємодіяти не тільки із своїм віртуальним світом, а і з іншими людьми, які також знаходяться у віртуальній реальності. На даний момент представлені технології, які примітивно, також у вигляді ігр дозволяють реалізувати це. Отже, з більш серйозним розвитком цього явища, надається більше уваги тому, щоб віртуальний світ мало чим відрізнявся від реального. Виникне, так звана, соціальна віртуальна реальність.

Соціальна віртуальна реальність – це новий рівень розвитку даної технології. Тут люди будуть не просто самотньо спостерігати віртуальні світи або грати в ігри, а зможуть ділитися віртуальним контентом різного роду зі своїми друзями, спілкуватись з ними, грати, вчитись, дискутувати, запрошувати у самостійно створений віртуальний світ. І цим можливості соціальної складової

не обмежуються.

Для спілкування людей дуже важливим є не тільки передача голосу, а і візуальне його відображення. Синхронний природний рух губ та роту під час розмови з анімацією віртуального персонажа у контексті соціальної віртуальної реальності є важливою проблемою, яка досі не є повністю вирішена.

Актуальною проблемою цього підходу є синхронізація тривалості звукового потоку та тривалостей фонем разом із відповідними їм віземами. Даний процес повинен відбуватись потоково в режимі реального часу. Тобто потрібно потрібно з мінімальною затримкою розпізнавати голос, виділяти з нього фонеми та їх довжини та, на основі цих даних, синхронно візуалізувати мовлення на віртуальному персонажі.

Отже, метою даної роботи є дослідження можливих на даний момент принципів роботи систем розпізнавання мовлення та аналіз можливих відкритих із них з подальшим вибором тієї, яка найкраще підходить для розпізнавання в режимі реального часу із можливістю виділення фонем та їх тривалостей. У якості результату, розробити прототип алгоритму синхронізації анімації обличчя віртуального персонажа із голосом, який в подальшому можна застосовувати в концепції віртуальної реальності.

1 ВІРТУАЛЬНА РЕАЛЬНІСТЬ

Віртуальна реальність – це неймовірне середовище для занурення. Воно надає сенсаційні можливості бути повністю перенесеним в віртуальний (або реальний, але відтворений за допомогою цифрових технологій) тривимірний світ. Крім того, воно дає можливість зануритися куди глибше, ніж звичні технології з використанням звичайного екрану.

Розробники несуть відповідальність за те, щоб їх контент відповідав усім стандартам індустрії, слідував всім рекомендаціям з безпеки і комфорту, а також вони зобов'язані ознайомитися зі всією необхідною науковою літературою по цій темі.

Якщо контент для віртуальної реальності ігнорує фундаментальні рекомендації, він може викликати неприємні відчуття у деяких користувачів. Так звана «хвороба симуляції» являє собою комбінацію з симптомів, пов'язаних із зоровим дискомфортом, дезорієнтацією та нудотою. Історично велика частина цих проблем була викликана використанням не оптимізованого належним чином обладнання для віртуальної реальності – наприклад, з наявністю системної затримки. Але навіть з бездоганною апаратною оптимізацією неправильно розроблений контент може привести до неприємного для користувача досвіду.

В даний час існують два основних напрямки VR-розробки – tethered і untethered, тобто на основі комп'ютерів і на основі мобільних пристроїв. Прив'язаний до комп'ютера tethered VR використовує потужні графічні карти настільних комп'ютерів для створення картинки в таких висококласних VR-пристроях, як Oculus Rift і HTC Vive. Для мобільного untethered VR використовують мобільний пристрій користувача, вставлене в VR-пристрій з пластмаси, картону або піноматеріалу; при цьому використовуються обчислювальні ресурси тільки користувацького пристрою, характеристики якого можуть змінюватися в широкому діапазоні.

Для створення якісного контенту віртуальної реальності потрібно звертати увагу на аспекти, які наведені у наступних пунктах[4].

1.1 Обмеження

У мобільного VR є свої обмеження, які необхідно врахувати, перш ніж вибирати підтримувану платформу. По-перше, як зрозуміло з назви, контент мобільного VR надається мобільними пристроями, а не потужними комп'ютерами. Хоча нові пристрої і мають набагато більші обчислювальними ресурсами, проте все одно існують обмеження глибини і складності сцени, яка рендериться в. Звичайно ж, це не означає, що пристрої абсолютно неефективні: щоб згенерована 3D сцена була переконливою для користувачів, від неї не потрібно фотореалістичності. Згенеровані на мобільних пристроях низькополігональні світи досить цікаві і ефективні в VR, При цьому вони забезпечують високу продуктивність.

Ще один аспект полягає в тому, що незважаючи на незалежність від комп'ютера і надання користувачеві свободи пересування без кабелів, поки неможливо відстежувати переміщення очей в просторі: неважливо, як рухається користувач, положення камери оновлюється тільки в трьох напрямках. Це означає, що безліч відчуттів, одержуваних при tethered VR (присідання, наближення до персонажів з будь-якого кута, вільне переміщення в межах кімнати), недоступні для мобільного VR. Однак ведуться розробки обладнання, яке повинно рано чи пізно вирішити цю проблему.

Отже, у мобільного VR менша обчислювальна потужність і неможливо відстежити рух голови. Однак в них відсутні будь-які способи реальної взаємодії, користувач може тільки дивитися на щось, тому деякі не вважають це «правильною» VR. Інші також вважають, що завдяки покращенням мобільного VR і зростанням обчислювальної потужності розробникам слід націлюватися на повну, високоякісну VR, тому що розрив буде тільки звужуватися, і якщо сьогодні вже розробляються програми для мобільного VR, вони скоро застаріють протягом декількох місяців або років у зв'язку з швидким розвитком обладнання[3,4].

1.2 Oculus SDK. Можливості та рекомендації.

1.2.1 Рендеринг

- Використання шейдерів спотворення Oculus VR. Впровадження свого власного рішення для спотворення, навіть коли воно виглядає вірно, часто призводить до дискомфорту користувачів.
- Використання точної проекції матриці і застосування моделі голови Oculus за замовчуванням. Будь-які відхилення від оптичного потоку, які в реальному середовищі викликаються рухами голови, викликають проблеми з рухом очей і дискомфорт в тілі.
- Підтримання занурення у віртуальну реальність від початку і до кінця. Наприклад, не можна закріплювати статичне зображення перед користувачем – на кшталт єдиної заставки на весь екран, яка не відповідає на рухи голови – так як це може викликати дезорієнтацію в просторі.
- Зображення надаються для кожного ока окремо, і тому вони повинні відрізнятися в залежності від кута зору; ефекти післяобробки (наприклад, спотворення кольору, сама передача кольору) повинні йти для двох очей послідовно так само, як і повинна рендеритись правильна глибина по осі Z, щоб створити правильно сформоване зображення.
- Ефект спрощення і/або згладжування для усунення ефекту низької роздільної здатності, яка проявлятиметься найгіршим чином в центрі екрану для кожного ока[4].

1.2.2 Мінімізація затримки

- Код повинен працювати з рівною або більшою частотою кадрів в порівнянні з тим, як оновлюється дисплей, його віртуальна синхронізація і буферна пам'ять. Лаги і завислі кадри стають причиною вібрації, яка викликає дискомфорт у віртуальній реальності.
- Ідеально, якщо буде отримана затримка в 20 мс або менше. Потрібно мінімізувати затримку між зняттям даних із сенсорів і рендерингом.
- Затримка ігрового циклу не є єдиною постійною і змінюється час від часу. SDK використовує кілька трюків (наприклад, прогнозоване відстеження

TimeWarp, щоб захищати користувача від ефектів затримки, але зробити все можливе, щоб звести до мінімуму варіативність затримки протягом усього сеансу.

1.2.3 Оптимізація

- Потрібно зменшити розмір буфера для рендерингу на кожне око для того, щоб зберегти відеопам'ять і поліпшити частоту кадрів.
- Незважаючи на те, що зниження розширення екрану може бути гарним методом для поліпшення продуктивності, отримана в результаті перевага обумовлена в першу чергу його впливом на здатність екрану для рендерингу кожного ока. Зниження розміру буфера для рендерингу кожного ока при одночасному збереженні розширення екрану може поліпшити продуктивність з меншою втратою якості візуалізації, ніж при зниженні їх обох[4].

1.2.4 Відслідковування голови і точки погляду

- Потрібно уникати візуальних ефектів, які порушують почуття стабільності у користувача під час його перебування у віртуальному середовищі. Обертання або рух лінії горизонту або інших великих компонентів середовища користувача в конфлікті з реальними рухами користувача в просторі (або їх відсутністю) може викликати дискомфортні відчуття.
- Дисплей повинен відповідати на рухи користувача протягом усього часу, без винятків. Навіть в меню або під час заставок користувач повинен мати можливість озирнутися і подивитися навколо себе.
- Потрібно застосовувати позиційне відстеження SDK і модель голови для того, щоб віртуальні камери оберталися і рухалися відповідно до рухів голови і тіла; диспропорції викликають почуття дискомфорту.

1.2.5 Камери

- Наближення чи віддалення за допомогою камери може викликати або посилювати «хворобу симуляції», особливо, якщо при цьому рухи голови користувача і камери не відповідають один одному.
- Для контенту від імені третьої особи необхідно мати на увазі, що рекомендації по рухах застосовуються до камери так само, як для контенту

від першої особи, незалежно від того, що робить аватар. Крім того, користувачі повинні мати повну свободу дивитися навколо, що може додати нові вимоги до розробки контенту.

- Потрібно уникати використання кутів Ейлера всюди, де це можливо; краще використовувати кватерніони.
- Потрібно уникати використання ефектів «гойдання голови». Це створює серію невеликих, але при цьому некомфортних прискорень.

1.2.6 Зовнішній вигляд віртуального персонажа

- При створенні контенту можна вибрати: або щоб користувач відчував себе як привид (без фізичної присутності) або знаходився в іншому тілі, яке сильно відрізняється від власного. Наприклад, можна реалізувати перенесення користувача в рамках контенту в історичну постать, вигаданого персонажа, анімацію, дракона, велетня, орка, амебу або будь-який інший варіант зі всієї безлічі можливостей. Будь-які персонажі такого роду не викликатимуть проблеми у користувачів, якщо буде дотримано рекомендацій по оптимальних методах забезпечення комфорту і дано можливість користувачам отримати інтуїтивно зрозуміле управління.
- Коли аватар необхідний для того, щоб представляти самого користувача у віртуальному середовищі, він може відволікати від глибокого занурення, якщо користувач дивиться вниз і бачить тіло або руки, які сильно відрізняються від власних. Наприклад, відчуття глибокого занурення жінки може бути порушено, якщо вона дивиться вниз і бачить руки або тіло чоловіка. Надання можливості користувачам налаштовувати відображення своїх рук і тіл може значно поліпшити відчуття повного занурення. Якщо це вимагає занадто багато витрат або ускладнює проект, все одно можна вжити заходів для того, щоб звести до мінімуму різницю між віртуальною реальністю і реальним світом. Наприклад, можна уникнути надмірно очевидних чоловічих або жіночих особливостей тіла в видимих частинах персонажа[4].

1.3 Oculus Mobile SDK

Даний пакет для розробників віртуальної реальності містить у собі бібліотеки, інструменти та ресурси для розробки для телефонів Samsung з гарнітурою Gear VR.

До складу Oculus Mobile SDK входять натупні компоненти:

● VrApi – дозволяє інтегруватись в більшість популярних графічних движків

● Фреймворк для розробки додатків на мові C++

● Додаткові бібліотеки для забезпечення функцій, які можуть знадобитись при розробці віртуальної реальності.

● Приклади додатоків та їх код для створення власних додатків на основі них.

● Інші інструменти і ресурси для полегшення нативної розробки.

До складу Oculus Native SDK входять наступні компоненти:

● VrApi – надає можливість візуалізації у VR. Обробляє запити додатка для зчитування даних із сенсорів для орієнтації, обробляє текстури для їх викривлення, застосування даних із сенсорів та композицію.

● VrAppFramework – фреймворк для нативних додатків. Представлений у вигляді обгортки для Android activity, тобто за допомогою нього відбувається керування життєвим циклом додатку.

● VrAppSupport – підтримка додаткових фунцій (наприклад звуку);

● LibOVRKernel – бібліотека для контейнерів, математичних операцій і т.д[4].

1.4 Gear VR Framework

Gear VR Framework – це потужний відкритий графічний рушій, з відкритим вихідним кодом, який надає можливість розробки мобільних ігор VR і додатків для Gear VR на мові програмування Java.

GVRf має такі особливості:

- Орієнтований на мобільні платформи. Легка і проста робота рушія на системному рівні Android OS.
- Простота розробки. Мова програмування Java та Android Studio як середовище розробкиі. Не потрібні поглиблені знання про рендеринг Open GL та Oculus.

- Продуктивний. Специфічні для VR засоби рендерингу та звернення до графічного конвеєра.
- Відкритий вихідний код. Немає ліцензійних зборів або ліцензійних платежів.
- Ефективний. Інтерфейс GVRf абстрагується від цільової мобільної платформи VR SDK.

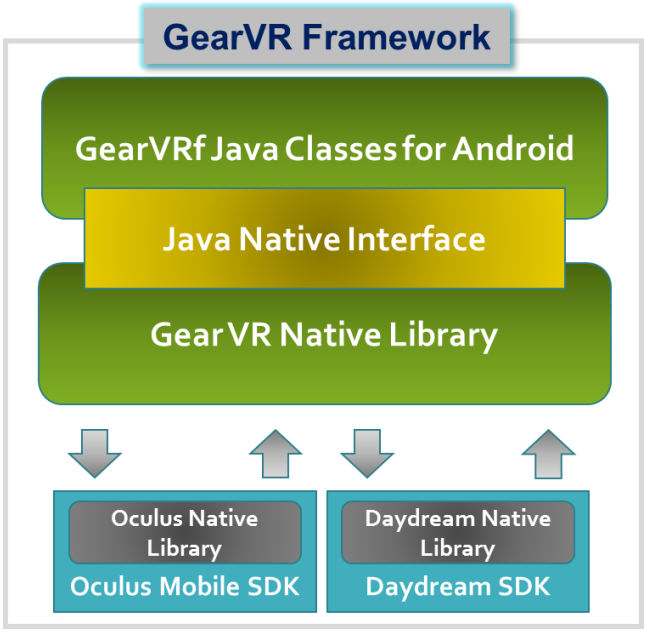

Рисунок 1.1 – Архітектура Gear VR Framework

GearVRf є нативним 3D рушієм для рендерингу для використання на Android. Він надає можливість створення контенту, використовуючи тільки вбудовані об'єкти. Додавання об’єктів в сцену, об’єктів з або без шейдерів – GearVRf виконує це на апаратному рівні. Вся розробка ведеться на мові програмування Java.

Ієрархія об'єктів GVRf формує 3D-сцену. Кожен видимий об'єкт рендериться за допомогою трикутників. При використанні GVRf не відбувається явний виклик OpenGL. Замість цього, фреймворк управляє всім рендерингом, забезпечуючи більш високий рівень абстракції для графіки[3].

При побудові Android програми, треба створити підклас класу Activity. При використанні GVRf аналогічним чином потрібно унаслідуватись від класу GVRActivity, який надає код ініціалізації для створення GVRMain, що створює початкову сцену і обробляє вхідні події.

Під час ініціалізації GVRActivity створює GVRViewManager. Цей клас відповідає за планування задач, 3D-рендеринг, анімації та завантаження ресурсів[3].

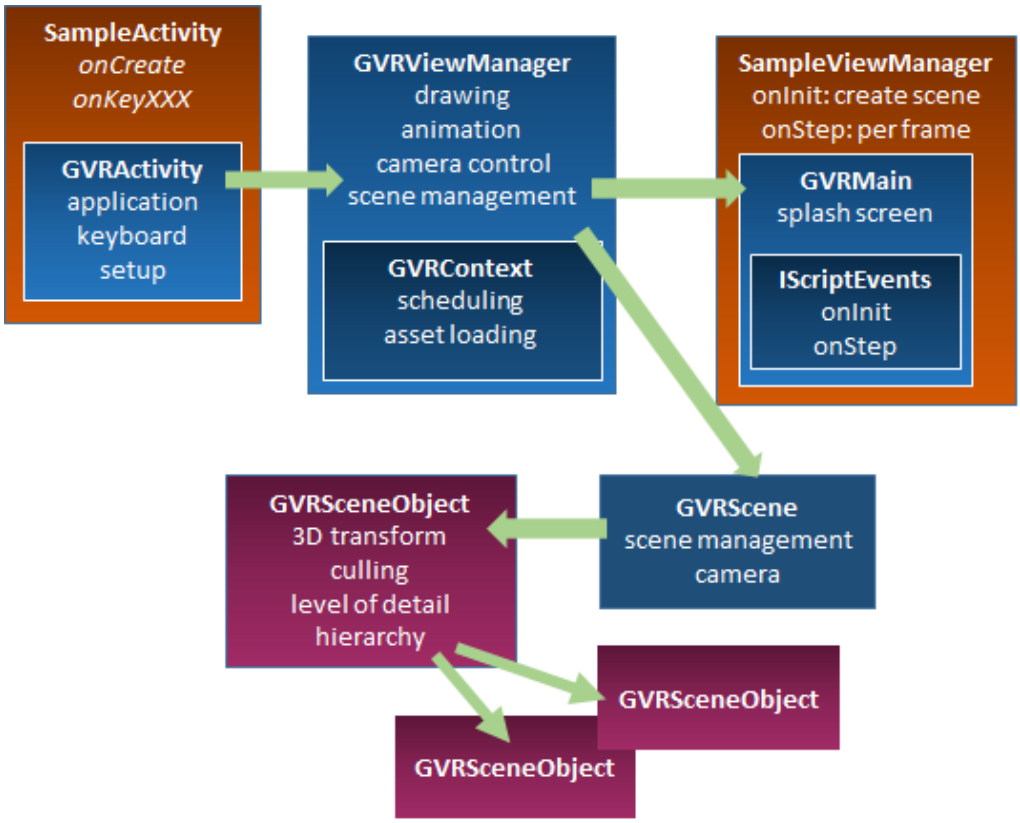

Рисунок 1.2 – Приклад роботи Gear VR Framework

Висновок

В даному розділі було описано основні принципи роботи додатків віртуальної реальності. Також надано рекомендації для правильної побудови таких додатків. Проаналізувавши рішення для відображення контенту у віртуальній реальності, найбільш прийнятним і з потрібним функціоналом виявся графічний рушій GVRf. Також було описаноструктуру та основні компоненти Oculus Mobile SDK.

2 РОЗПІЗНАВАННЯ ГОЛОСУ

Згідно з лінгвістичними особливостями людської мови, додаткові артикуляційні дані дозволяють більш точно виявити мову диктора і автоматично розбити звукову хвилю на окремі фрагменти. Також, при загальному аналізі аудіовізуального голосового сигналу у часовій динаміці є перспектива фіксування відкритих і закритих складів, дзвінких, шиплячих, наголошених, ненаголошених голосних/приголосних та інших мовних одиниць. Саме тому в задачі високоякісного розпізнавання мови вкрай важливим є створення бібліотеки даних, яка повинна містити всі ці критерії. Даний напрямок може бути реалізовано в тому випадку, якщо є відкритий доступ до мовних одиниць. Саме тому для вирішення поставленої задачі (реалізація синхронізації анімації обличчя з голосом для персонажу у віртуальній реальності) вкрай важливо розглянути аудіо-системи розпізнавання мови з відкритим вихідним кодом.

2.1 Загальні принципи розпізнавання голосу

Почнемо з того, що наша мова - це послідовність звуків. Звук в свою чергу – це суперпозиція (накладення) звукових коливань (хвиль) різних частот. Хвилі ж, як нам відомо з фізики, характеризуються двома атрибутами - амплітудою і частотою.

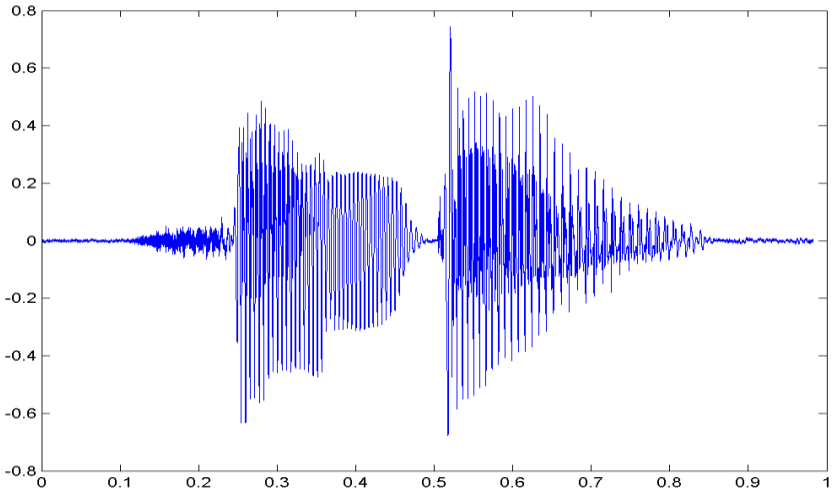

Рисунок 2.1 – Звукова хвиля

Для того, що б зберегти звуковий сигнал на цифровому носії, його необхідно розбити на безліч проміжків і взяти деяке «усереднене» значення на

кожному з них[5].

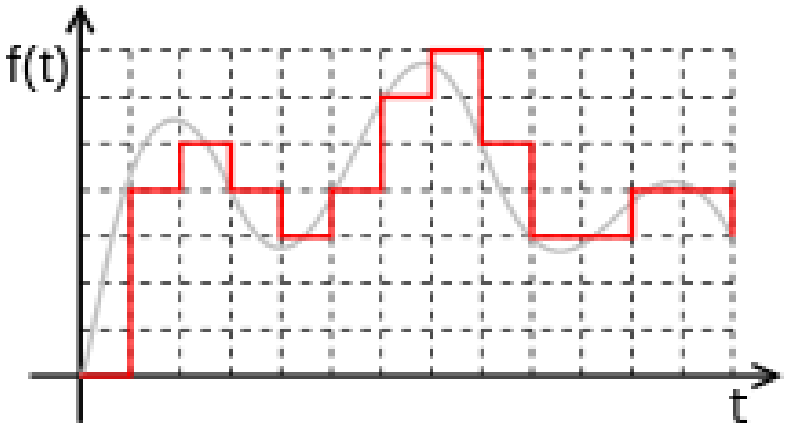

Рисунок 2.2 – Оцифрована звукова хвиля

Таким чином механічні коливання перетворюються в набір чисел, придатний для обробки на сучасних ЕОМ. Звідси випливає, що завдання розпізнавання мови зводиться до «порівняння» безлічі чисельних значень (цифрового сигналу) і слів з деякого словника (англійської мови, наприклад).Розберемося, як, власне, це саме «співставлення» може бути реалізовано.

2.1.1 Вхідні дані

Припустимо у нас є деякий файл / потік з аудіоданими. Перш за все нам потрібно зрозуміти, як він влаштований і як його прочитати. Розглянемо найпростіший варіант – WAV файл.

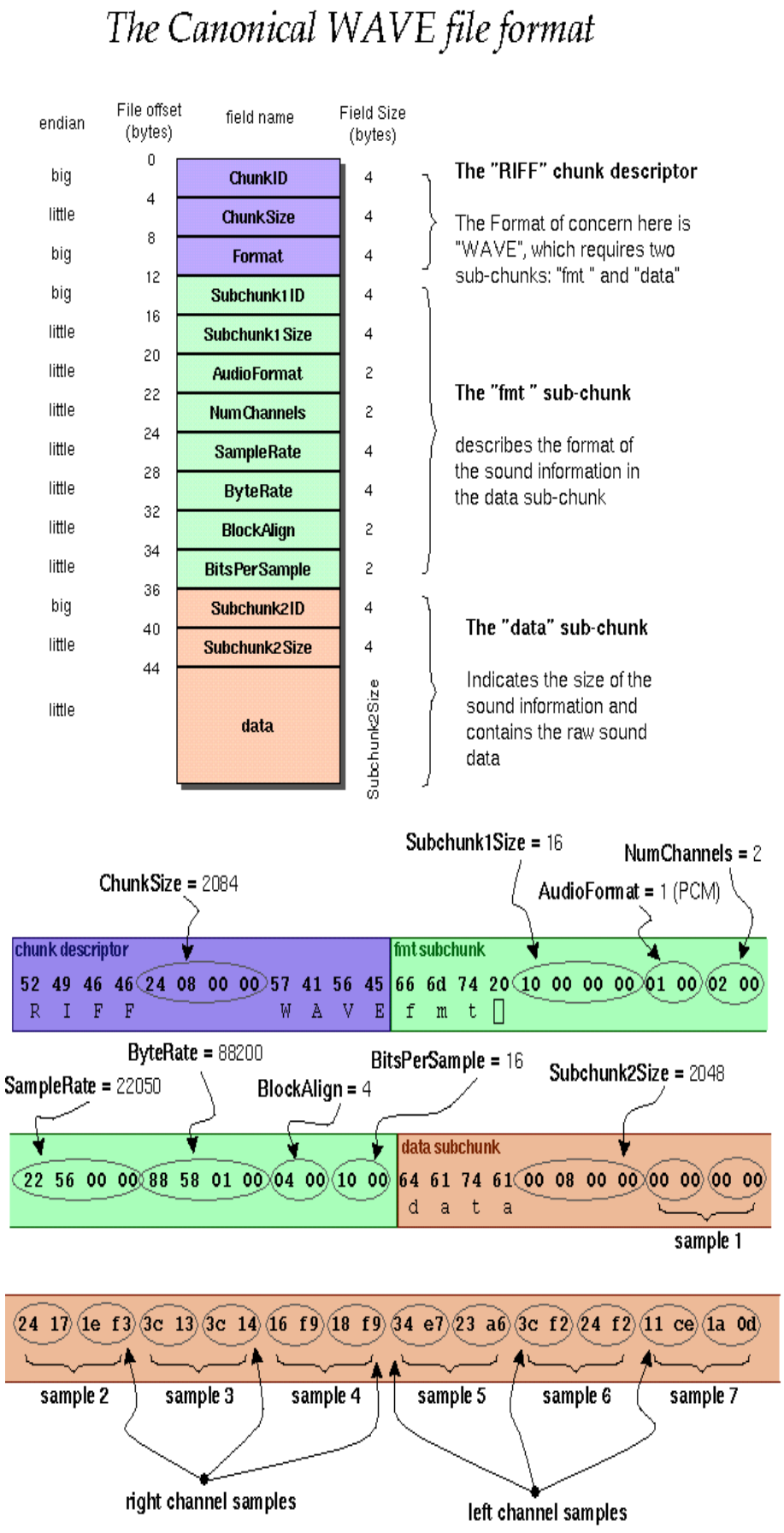

Рисунок 2.3 – Структура звукового файлу

У файлі даного формату наявні два блоки. Перший блок - це заголовок з

інформацією про аудіопотік: бітрейт, частота, кількість каналів, довжина файлу і т.д. Другий блок складається з «сирих» даних - того самого цифрового сигналу, набору значень амплітуд.

Логіка читання даних в цьому випадку досить проста. Зчитуємо заголовок, перевіряємо деякі обмеження (відсутність стиснення, наприклад), зберігаємо дані в спеціально виділений масив.

2.1.2 Розпізнавання

Тепер ми можемо порівняти (поелементно) наявний у нас зразок з яким-небудь іншим, текст якого нам вже відомий. Тобто спробувати «розпізнати» мовлення. Але це є неправильно з точки зору стійкості.

У даному випадку підхід повинен бути стійкий до зміни тембру голосу (людини, яка вимовляє слово), гучності і швидкості вимови. Поелементним порівнянням двох аудіосигналів цього неможливо досягти. Тому розглянемо далі детальніше розглянемо підхід який буде стійким[6].

2.1.3 Фрейми

Насамперед потрібно розбити дані на невеликі часові проміжки - фрейми. Причому фрейми повинні йти не один за одним, а з накладанням. Тобто кінець одного фрейма повинен перетинатися з початком іншого.

Фрейм є більш придатною одиницею аналізу даних, ніж конкретні значення сигналу, так як аналізувати хвилі набагато зручніше на деякому проміжку, ніж в конкретних точках. Розташування ж фреймів з накладанням дозволяє згладити результати аналізу, перетворюючи ідею фреймів у "вікно", яке рухається вздовж вихідної функції (значень сигналу).

Дослідним шляхом встановлено, що оптимальна довжина фрейму повинна відповідати проміжку в 10 мс, накладання – 50%. З урахуванням того, що середня довжина слова становить 500 мс – такий крок дасть приблизно 500 / (10 * 0.5) = 100 фреймів на слово[7,8].

2.1.4 Розбиття слів

Першим завданням, яке доводиться вирішувати при розпізнаванні мовлення, є його розбиття на окремі слова. Для простоти припустимо, що в нашому випадку мова містить в собі деякі паузи (проміжки тиші), які можна

вважати "роздільниками" слів.

У такому випадку потрібно знайти деяке значення, поріг - значення вище якого є словом, нижче - тишею. Варіантів тут може бути декілька:

- задати константою (спрацює, якщо вихідний сигнал завжди генерується при одних і тих же умовах, одним і тим же способом);
- кластеризувати значення сигналу, явно виділивши безліч значень відповідних тиші (спрацює тільки якщо тиша займає значну частину вихідного сигналу);
- проаналізувати ентропію;

Найбільш прийнятним варіантом є аналіз ентропії. Почнемо з того, що ентропія - це міра безладу, "міра невизначеності будь-якого досвіду". У даному випадку ентропія означає те, як сильно "коливається" наш сигнал в рамках заданого фрейму.

Для того, що б підрахувати ентропію конкретного фрейму слід виконати наступні дії:

- припустимо, що сигнал пронормовано і всі його значення лежать в діапазоні [-1; 1];
- побудуємо гістограму (щільність розподілу) значень сигналу фрейму:

розрахуємо ентропію, як

$$E = \sum_{i=0}^{N-1} P[i] * \log_2(P[i])$$

І так, ми отримали значення ентропії. Але це всього лише ще одна характеристика фрейму, і для того, щоб відокремити звук від тиші, потрібно її з чимось порівнювати. Рекомендовано брати поріг ентропії рівним середньому між її максимальним і мінімальним значеннями (серед всіх фреймів).

Ентропія – величина відносно самостійна. Це дозволяє підібрати значення її порогу у вигляді константи (0.1).

Ентропія може просідати на середині слова (на голосних), а може раптово збільшуватись через незначний шум. Для того, щоб боротися з першою

проблемою, доводиться вводити поняття "мінімальна відстань між словами" і "склеювати" набори фреймів, які знаходяться поруч та розділені через просідання. Друга проблема вирішується використанням "мінімальної довжини слова" і відсіканням всіх кандидатів, які не пройшли відбір (і не використаних в першому пункті).

Якщо ж мова в принципі не є "членороздільною", можна спробувати розбити вихідний набір фреймів на підготовлені підпослідовності, кожна з яких буде піддана процедурі розпізнавання[9,10,11].

2.1.5 MFCC

Отже, отримано набір фреймів, які відповідають певному слову. В якості чисельної характеристики фрейму використаємо середній квадрат всіх його значень (Root Mean Square). Однак, така метрика несе в собі вкрай мало придатної для подальшого аналізу інформації.

Тут знадобляться Мел-частотні кепстральні коефіцієнти (Mel-frequency cepstral coefficients). MFCC – це своєрідне представлення енергії спектра сигналу. Плюси його використання полягають у наступному:

- Використовується спектр сигналу (тобто розклад по базису ортогональних синусоїдальних функцій), що дозволяє враховувати хвильову "природу" сигналу при подальшому аналізі;
- Спектр проектується на спеціальну mel-шкалу, дозволяючи виділити найбільш значущі для сприйняття людиною частоти;
- Кількість обчислюваних коефіцієнтів може бути обмежена будь-яким значенням, що дозволяє "стиснути" фрейм і як наслідок, кількість оброблюваної інформації;

Розглянемо процес обчислення MFCC коефіцієнтів для деякого фрейма. Представимо фрейм у вигляді вектора $x[k], 0 \leq k < N$ де N - розмір фрейма.

2.1.6 Розкладання в ряд Фур’є

Насамперед розрахуємо спектр сигналу за допомогою дискретного перетворення Фур'є (бажано його "швидкою" FFT реалізацією).

$$X[k] = \sum_{n=0}^{N-1} x[n] * e^{-2*\pi*i*k*n/N}, 0 \leq k < N$$

Так само до отриманих значень рекомендується застосувати віконну функцію Хеммінга, для того, щоб "згладити" значення на межах фреймів.

$$H[k] = 0.54 - 0.46 * \cos(2 * \pi * k/(N-1))$$

Тобто результатом буде вектор наступного вигляду:

$$X[k] = X[k] * H[k], 0 \leq k < N$$

Важливо розуміти, що після цього перетворення по осі X ми маємо частоту (hz) сигналу, а по осі Y - магнітуду (спосіб відійти від комплексних значень)[12]:

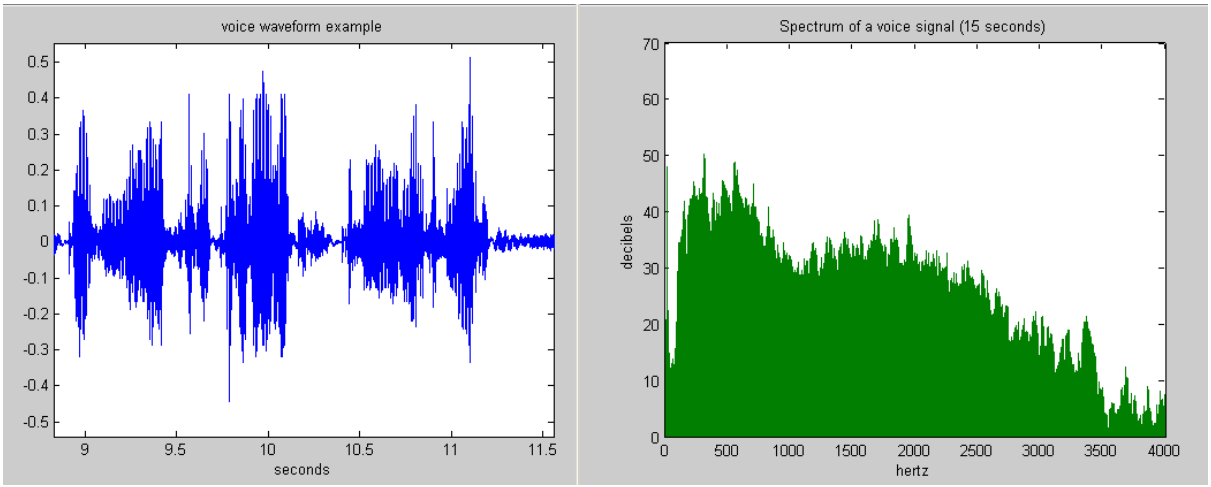

Рисунок 2.4 – Спектр звукового сигналу

2.1.7 Розрахунок mel-фільтрів

Почнемо з того, що таке mel. Це "психофізична одиниця висоти звуку", заснована на суб'єктивному сприйнятті середньостатистичними людьми. Залежить в першу чергу від частоти звуку (а так само від гучності і тембру). Іншими словами, це величина, яка показує, на скільки звук певної частоти "значущий" для нас.

Перетворити частоту в mel можна за наступною формулою (запам'ятаємо її як «формула-1»):

$$M = 1127 * \log(1 + F/700)$$

Зворотне перетворення виглядає так (запам'ятаємо її як «формула-2»):

$$F = 700 * (e^{M/1127} - 1)$$

Графік залежності mel / частота:

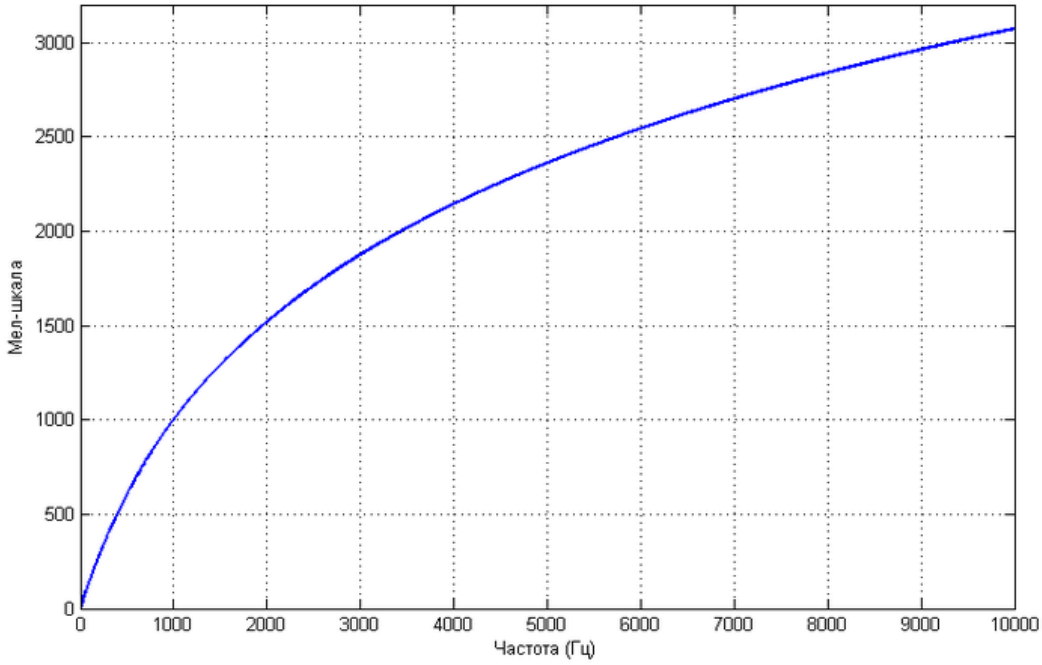

Рисунок 2.5 – Графік залежності mel від частоти

Припустимо існує фрейм розміром 256 елементів. Відомо (з даних про аудіоформат), що частота звуку в даному фреймі 16000 hz. Припустимо, що людська мова лежить в діапазоні від 300 до 8000 hz. Кількість шуканих mel-коефіцієнтів покладемо M = 10 (рекомендоване значення).

Для того, щоб розкласти отриманий вище спектр за mel-шкалою, потрібно створити "гребінку" фільтрів. По-суті, кожен mel-фільтр це трикутна віконна функція, яка дозволяє підсумувати кількість енергії на певному діапазоні частот і тим самим отримати mel-коефіцієнт. Знаючи кількість mel-коефіцієнтів і діапазон частот, який аналізується, можна побудувати набір наступних фільтрів:

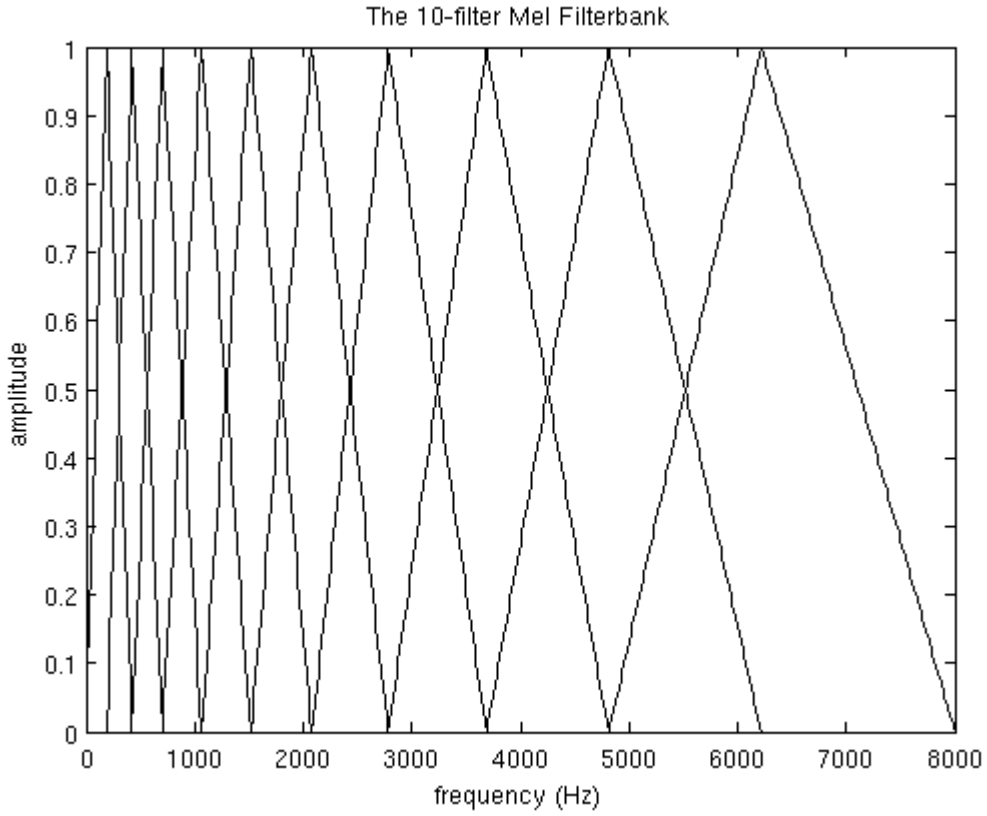

Рисунок 2.6 – Mel-фільтри

Бачимо, що чим більший порядковий номер mel-коефіцієнта, тим ширша основа фільтра. Це пов'язано з тим, що розбиття потрібного діапазону частот на діапазони, які будуть оброблені фільтрами, відбувається на mel-шкалі.

Для нашого випадку діапазон потрібних частот [300, 8000]. Відповідно до формули-1 на mel-шкалі цей діапазон перетворюється в [401.25; 2834.99]. Далі, для того, що б побудувати 10 трикутних фільтрів потрібно 12 опорних точок: m[i] = [401.25, 622.50, 843.75, 1065.00, 1286.25, 1507.50, 1728.74, 1949.99, 2171.24, 2392.49, 2613.74, 2834.99]

На mel-шкалі точки розташовані рівномірно. Переведемо шкалу в герци за допомогою формули-2: h[i] = [300, 517.33, 781.90, 1103.97, 1496.04, 1973.32, 2554.33, 3261.62, 4122.63, 5170.76, 6446.70, 8000].

Тепер шкала стала поступово розтягуватися, вирівнюючи тим самим динаміку зростання "значущості" на низьких і високих частотах.

Далі потрібно накласти отриману шкалу на спектр фрейму. По осі X знаходиться частота. Довжина спектра 256 - елементів, при цьому в нього

вміщається 16000 hz. Отримаємо наступну формулу:

f(i) = floor((frameSize+1) * h(i) / sampleRate)

що в нашому випадку еквівалентно f(i) = 4, 8, 12, 17, 23, 31, 40, 52, 66, 82, 103, 128.

Знаючи опорні точки на осі X спектра, легко побудувати необхідні фільтри за наступною формулою:

$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \le k \le f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & f(m) \le k \le f(m+1) \\ 0 & k > f(m+1) \end{cases}$$

2.1.8 Застосування фільтрів, логарифмування енергії спектру

Застосування фільтру полягає в попарном перемножуванні його значень зі значеннями спектра. Результатом цієї операції є mel-коефіцієнт. Оскільки фільтрів M, коефіцієнтів буде стільки ж.

$$S[m] = \log(\sum_{k=0}^{N-1} |X[k]|^2 * H_m[k]), 0 \le m < M$$

Однак, потрібно застосувати mel-фільтри не до значень спектра, а до його енергії. Після чого прологарифмувати отримані результати. Вважається, що таким чином знижується чутливість коефіцієнтів до шумів[13].

2.1.9 Косинусне перетворення

Дискретне косинусне перетворення (DCT) використовується для того, щоб отримати "кепстральні" коефіцієнти. Сенс його в тому, що б "стиснути" отримані результати, підвищивши значимість перших коефіцієнтів і зменшивши значимість останніх. В даному випадку використовується DCTII без будь-яких домноженій на $\sqrt{(2/N)}$ (scale factor)[13].

$$C[l] = \sum_{m=0}^{M-1} S[m] * \cos(\pi * l * (m + \frac{1}{2})/M)), 0 \le l < M$$

Тепер для кожного фрейма існує набір з M mfcc-коефіцієнтів, які можуть

бути використані для подальшого аналізу.

2.1.10 Алгоритм розпізнавання

Завдання зводиться до підбору найбільш "близької" моделі для деякого набору mfcc-коефіцієнтів (слова, яке розпізнається). На перший погляд завдання можна вирішити досить просто:

- для кожної моделі знаходимо середню відстань між ідентифікованим mfcc-вектором і векторами моделі;
- вибираємо в якості правильної ту модель, середня відстань до якої буде найменшою;

Однак, одне і теж слово може вимовляється різними людьми по-різному. Іншими словами розмір mfcc-вектора для одного і того ж слова може бути різний. Для вирішення задачі розпізнавання можна застосовувати один з алгоритмів, які приведені далі. На даний момент більшість схиляється до використання алгоритму прихованих моделей Маркова, як найкращих для даної задачі[14].

2.2 Алгоритми розпізнавання мовлення

2.2.1 Нейронні мережі

Основу нейронної мережі складають як правило однотипні елементи, які імітують роботу біологічного нейрона, і які називаються зазвичай так само. Кожен з нейронів в кожен момент часу знаходиться, як і біологічний нейрон, в деякому поточному стані. Він має групу односпрямованих вхідних зв'язків-синапсів, які йдуть від входу в мережу або від інших нейронів. Крім того він має один односпрямований вихідний зв'язок – аксон[12].

Схематично нейрон можна представити так:

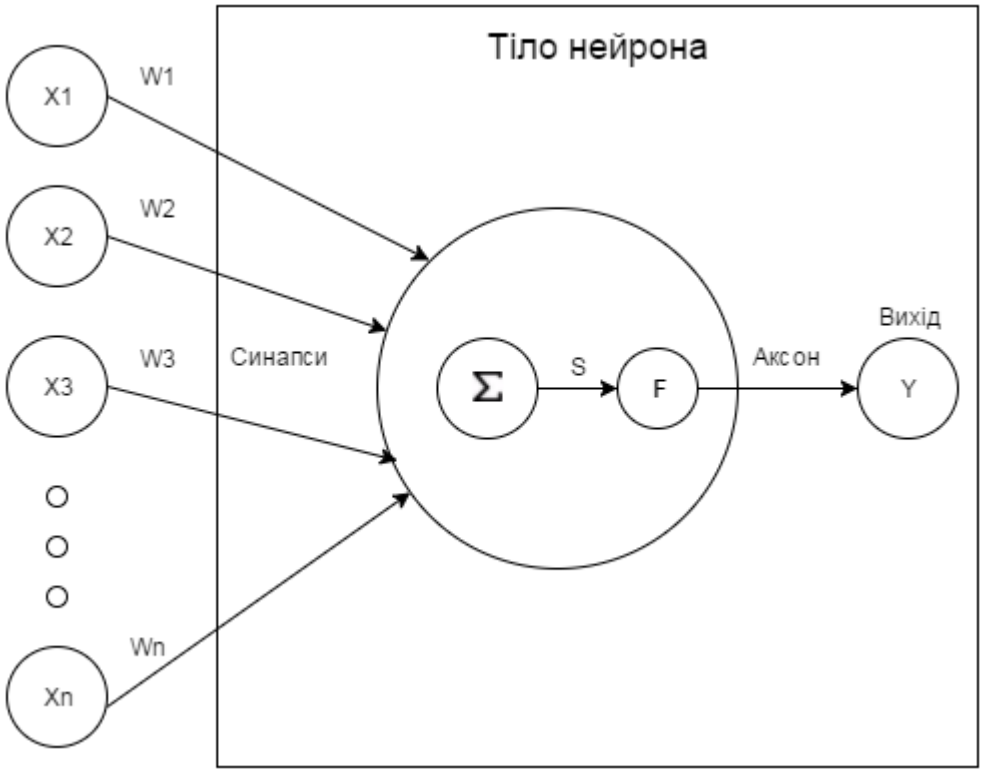

Рисунок 2.7 – Схематичне зображення нейрона

Синаптичні зв'язки характеризуються вагами wi. Поточний стан S нейрона рівний зваженій сумі входів[12,16].

Але повернемося до одношарового персептрона. У нього є ряд позитивних властивостей, які й змусили багатьох вчених звернути свій погляд на дослідження нейронних мереж. Головними з виявлених властивостей персептрона була здатність до навчання і розпізнавання. Тобто виявилося можливим в ряді випадків встановити те, як можна налаштувати ваги синапсів персептрона, щоб при різних комбінаціях значень входів отримувати заздалегідь встановлені, - правильні значення виходів. Тобто одношаровий персептрон виявився здатним відтворювати деякі математичні функції[15,16,17].

Однак ейфорія з приводу цих відкриттів швидко пройшла: були виявлені істотні обмеження в можливостях одношарового персептрона до навчання і розпізнавання.

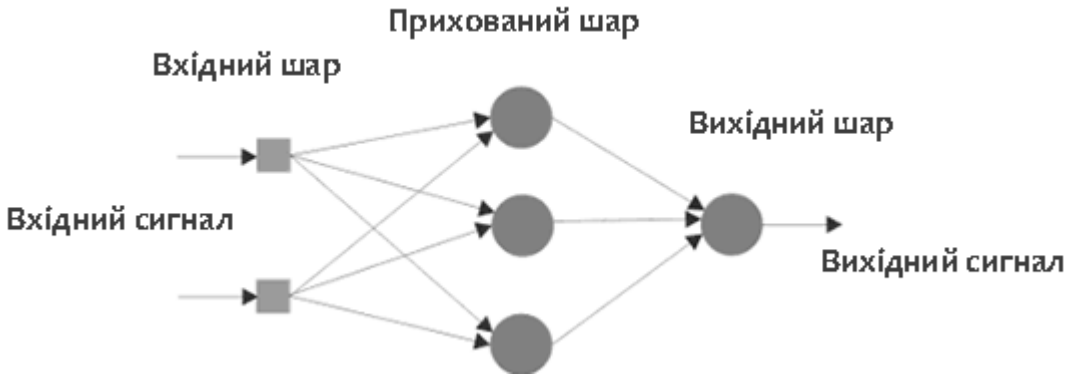

Рисунок 2.8 – Багатошаровий персептрон

Все було б дуже сумно, якби не існувало багатошарових нейронних мереж.

2.2.2 Алгоритм зворотного розповсюдження помилки

Все складніше з багатошаровими мережами, так як спочатку невідомі бажані виходи шарів мережі (за винятком останнього) і їх неможливо навчити, керуючись тільки величиною помилок на виході мережі, як це було з одношаровою мережею. Найбільш прийнятним варіантом вирішення проблеми стала ідея поширення сигналу помилки від виходу мережі до її входу, шар за шаром. Алгоритми, що реалізують навчання мережі за цією схемою, отримали назву алгоритмів зворотного поширення помилки. Алгоритм вимагає диференційованої активаційної (або як її по-іншому називають, стискаючої) функції на всій осі абсцис. З цієї причини, функція одиничного стрибка не може використовуватися і в якості стискаючої функції зазвичай застосовують сигмоїд (логістичну функцію), хоча існують і інші варіанти. Розглядаємо класичний варіант багатошарової мережі, де синаптичні зв'язки можуть визначатися будь-якими дійсними числами, а вихід нейрона - дійсними числами з інтервалу від 0 до 1. У якості активаційної функції використовуємо сигмоїд. Число шарів довільне.

1. Визначаємо M матриць вагових коефіцієнтів W розміром NxN, де M - число шарів, N - число нейронів в одному шарі. Wi, j, k буде позначати вагу j-го входу k-го нейрона в i-му шарі. ініціалізуєм матриці деякими малими випадковими (різними) значеннями.

2. Подаємо на входи мережі певні значення X, для яких відомі правильні значення виходів мережі Y *.

3. Обчислюємо значення виходів мережі для поточного стану матриць W. Тобто для вхідного вектора X обчислюється вихідний вектор Y. Для цього необхідно послідовно обчислити вихід для кожного шару мережі з першого по останній. Для i-го шару в векторному вигляді це можна записати так:

Oi = F (XWi), якщо i - не перший шар.

Oi = F (Oi-1Wi), якщо i - не перший шар.

де Oi - вектор виходу i-го шару, F - активаційна функція, X - вектор входів, Oi-1 - вектор виходу (i-1) -го шару, Wi - матриця вагових коефіцієнтів i-го шару.

4. Обчислюємо вектор ΔY = Y-Y *

5. Якщо ΔY менше заданої похибки, переходимо до кроку 9.

6. Для шару з номером M (тобто в останньому шарі) виробляємо наступні операції:

6.1. Для всіх нейронів в шарі з номера 1 по N виробляємо наступні операції:

6.1.1. Для всіх ваг нейрона з номера 1 по N виробляємо наступні операції:

6.1.1.1. Розраховуємо вектор δM = X (1-X) ΔY

6.1.1.2. Розраховуємо величину ΔWM, j, k = ηδM, kOi-1, j, де η - коефіцієнт швидкості навчання (від 0.01 до 1.0)

6.1.1.3. Коригуємо величину вагового коефіцієнта, додаючи до WM, j, k величину ΔWM, j, k

7. Для шарів з номером M-1 по перший послідовно проводимо наступні операції:

7.1. Для всіх нейронів в шарі з номера 1 по N виробляємо наступні операції:

7.1.1. Для всіх ваг нейрона з номера 1 по N виробляємо наступні операції:

7.1.1.1. розраховуємо вектор N δi = Oi + 1 (1-Oi + 1) [Σ δi + 1,

jWi + 1, j, k] k = 1

7.1.1.2. Розраховуємо величину ΔWi, j, k = ηδi, kOi-1, j,

7.1.1.3. Коригуємо величину вагового коефіцієнта, додаючи до WM, j, k величину WM, j, k

8. Перехід до кроку 3.

9. Кінець (навчання закінчено)[16].

2.3.3 Приховані Марківські моделі

Прихована Марківська модель (HMM) – статистична модель, що імітує роботу процесу, схожого на марківский процес з невідомими параметрами, і в якості завдання ставиться розгадування невідомих параметрів на основі спостережуваних. Отримані параметри можуть бути використані в подальшому аналізі, наприклад, для розпізнавання образів[11].

При аналізі природної мови першим кроком необхідно визначити, до якої частини мови належить кожне зі слів у реченні. В англійській мові завдання на цьому етапі називається Part-Of-Speech tagging. Яким же чином ми можемо визначити частину мови окремого члена речення? Розглянемо речення англійською мовою: «The can will rust». Отже, the – артикль; can – може одночасно бути і модальним дієсловом, і іменником, і дієсловом; will – модальне дієслово, іменник і дієслово; rust – іменник або дієслово. У статистичному підході необхідно побудувати таблицю ймовірностей використання слів в кожному граматичному значенні. Це завдання можна вирішити на основі тестових текстів, проаналізованих вручну. І відразу можна виділити одну проблему: слово «can» в більшості випадків використовується як дієслово, але іноді воно може бути і іменником. З огляду на цей недолік, була створена модель, яка бере до уваги той факт, що після артикля піде прикметник або іменник:

$$\arg\max_{t1..tn} \prod_{t=1}^{n} p(w_i|t_i)p(t_i|t_{i-1})$$

де: t – тег (іменник, прикметник і т.д.), w – слово в тексті (rust, can …), p(w|t) – ймовірність того, що слово w відповідає тегу t, p(t1|t2) – ймовірність того, що t1 йде після t2[29].

Із запропонованої формули бачимо, що потрібно підібрати тег так, щоб він підходив слову і тег підходив попередньому тегу. Даний метод дозволяє визначити, що «can» виступає в ролі іменника, а не як модальне дієслово[31].

Ця статистична модель може бути описана як ергодична HMM:

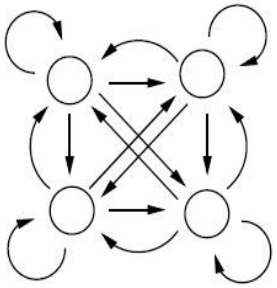

Рисунок 2.9 – Ергодична Марківська модель

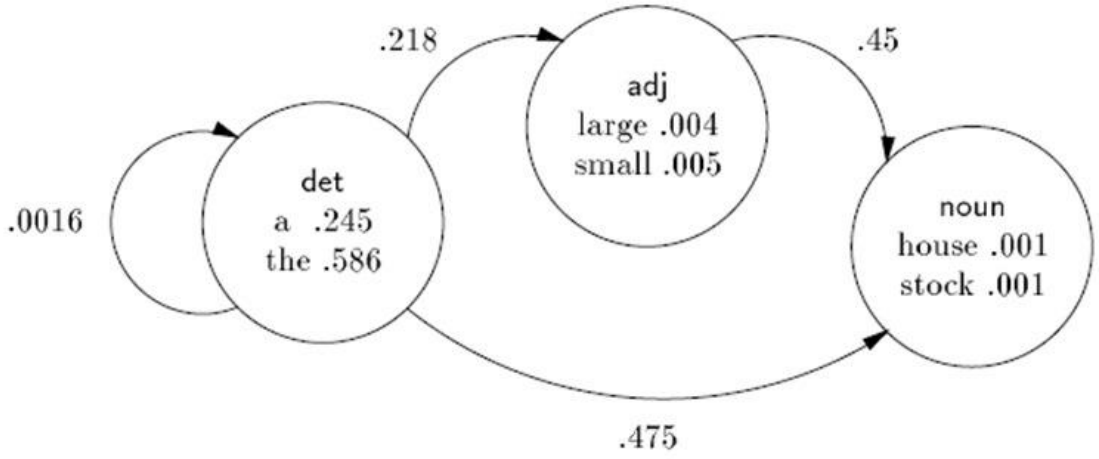

Рисунок 2.10 – Ергодична Марківська модель на практиці

Кожна вершина в даній схемі позначає окрему частину мови, в якій записуються пари (слово; ймовірність, що слово відноситься саме до цієї частини мови). Переходи показують можливу вірогідність проходження однієї частини мови за іншою. Так, наприклад, ймовірність того, що поспіль буде йти 2 артикля, за умови, що зустрінеться артикль, буде дорівнює 0,0016. Даний етап розпізнавання мови дуже важливий, так як правильне визначення граматичної структури речення дозволяє підібрати вірну граматичну конструкцію для експресивного забарвлення озвученого речення[31].

Також існують n-грамні моделі розпізнавання мовного потоку. Вони

засновані на припущенні, що ймовірність вживання наступноо слова в реченні залежить тільки від n-1 слів. Сьогодні найбільш популярні біграмні і триграмні моделі. Пошук в таких моделях відбувається по великій таблиці (корпус). Незважаючи на швидкий алгоритм, такі моделі не здатні вловити семантичні і синтаксичні зв'язки, якщо залежні слова знаходяться на відстані 5 слів одне від одного. Використання ж n-грамних моделей, де n більше ніж 5, вимагає величезних потужностей[11,34].

Як вже зазначалося вище, найпопулярнішою моделлю на сьогоднішній ень є триграмна модель. Умовна ймовірність спостереження речення w1, ... wn наближена до:P(w1,…,wn) = ПР(wi | w1,…,w2) ≈ ПР(wi | wi-(n-1), …, wi-1)

Наприклад, розглянемо пропозицію «I want to go home». Імовірність даного речення можна обчислити від рахунку частоти n-грама (в цьому прикладі n = 3):

P(I, want, to, go, home) ≈ P(I)*P(want|I)*P(to|I, want)*P(go|want, to)*P(home| to, go).

Варто згадати про далекодіючу триграмну модель, в якій аналіз ведеться не тільки на основі двох попередніх слів, а по будь-якій парі слів, які знаходяться поруч. Така модель може пропускати малоінформативні слова, тим самим покращуючи передбачення сполучуваності в моделі[33,34,39].

2.3 Пошук та вибір оптимальної аудіосистеми розпізнавання голосу

2.3.1 CMU Sphinx

CMU Sphinx - також просто Sphinx, головним чином була написана групою розробників систем розпізнавання мови університету Карнегі Меллон. Вона включає в себе серію систем розпізнавання мови (Sphinx 2-4) і систему тренування акустичної моделі (Sphinx train).

У 2000 році група Sphinx університету Карнегі Меллон затвердила компоненти open-source системи розпізнавання мови, включаючи Sphinx 2 і пізніше Sphinx 3 (у 2001 році). Мовний декодер включав в себе акустичні моделі і прості програми. Наявні ресурси включали в себе додаткове програмне забезпечення для тренування акустичної моделі, мовну модель, лінгвістичну

модель компліляції і словник вимови, який є суспільним надбанням (cmudict).

На даний момент актуальною є версія Sphinx 4 – повна і переписана мовна система Sphinx, головна мета якої надати гнучкий каркас для дослідженнь в розпізнаванні мови. Sphinx 4 написаний повністю на мові програмування Java. Sun Microsystems внесла істотний внесок в розвиток Sphinx 4 і надала допомогу в програмній експертизі проекту. Окремі розробники проекту – це люди з MERL, MIT, CMU.

Поточні цілі розвитку включають в себе:

• Розвиток нових акустичних моделей для тренування;

• Реалізація системи мовної адаптації (MLLR)

• Покращення менеджменту конфігурації

• Реалізація ConfDesigner - графічної системи дизайну

PocketSphinx - це версія Sphinx, яка може бути вбудована в будь-які інші системи на базі процесора ARM. PocketSphinx активно розвивається і вбудовується в різні системи з арифметикою фіксованої коми і в ефективні моделі на базі змішаної моделі обчислень[8].

2.3.2 Julius

Julius – це високопродуктивна система распізнавання безперервної мови з великим словником (large vocabulary continuous speech recognition), декодер програмного забезпечення для дослідження в області зв'язного мовлення і розробки. Він ідеально підходить для декодування в режимі майже реального часу на більшості існуючих комп'ютерів, з словником 60 тисяч слів, використовуючи завдання тіаграмми слова і незалежну від контексту приховану марківську модель. Головна особливість проекту полягає в можливості повної інтеграції. Основна платформа системи Linux та інші UNIX подібні сисеми, також система працює на Windows. Julius має відкритий вихідний код і поширюється з BSD типом ліцензії.

Julius розробляється як частина вільного програмного забезпечення для дослідження в області розпізнавання японської мови, починаючи з 1997 року і ця робота тривала в рамках Консорціуму систем розпізнавання безперервної мови

(Continuous Speech Recognition Consortium (CSRC)), з 2000 по 2003 рік в Японії.

Починаючи з версії 3.4 граматична база системи розпізнавання мови аналізатора називається Julian і інтегрована в Julius. Julian це модифікована форма Julius, яка використовує власну спроектовану форму граматики кінцевого автомата (Finite-state machine) як мовну модель. Вона може використовуватися для побудови систем голосової навігації з невеликим словником або інших розмовних систем для розпізнавання різного роду діалогів.

Для запуску системи розпізнавання мови Julius необхідно підібрати мовну модель і акустичну модель для потрібної мови. Julius адаптує акустичну модель формату HTK ASCII, базу даних вимови формату HTK, і 3-х шаровий діаграми побудови мовної моделі стандарту ARPA.

Хоча Julius поширюється тільки для моделі японської мови, проект VoxForge працює над створенням акустичної моделі для англійської мови, використовуючи систему розпізнавання мови Julius[24,25].

2.3.3 RWTH ASR

RWTH ASR (скорочено RASR) – це інструментарій розпізнавання мови з відкритим вихідним кодом. Інструментарій включає в себе технологію розпізнавання мови для створення автоматичних систем розпізнавання мови. Розвивається групою в Рейнсько-Вестфальському технічному університеті Ахена.

RWTH ASR включає в себе інструментарій для розробки акустичних моделей і декодери, а також компоненти для адаптації промови спікера, адаптивні системи навчання мови спікера, неконтрольовані системи навчання, диференціальні системи навчання і решітчаті словообразні форми обробки. Дане програмне забезпечення працює на ОС Linux і Mac OS X. Проект пропонує вже готові для використання моделі для дослідження з завданнями, навчальними системами і різного роду документацією.

Інструментарій публікується під ліцензією open source, яка називається «RWTH ASR ліцензія», яка є похідною від ліцензії QPL (Q Public License). Ця ліцензія надає можливість вільного використання, включаючи повторне

поширення і модифікацію для некомерційного використання.

2.3.4 Simon

Simon – система розпізнавання мови, заснована на мовних системах Julius і HTK. Система Simon спроектована таким чином, що вона доволі зручна для роботи з різними мовами і різного роду діалектами. При цьому реакція розпізнавання мови повністю налаштовується і вона не підходить для розпізнавання одиничних голосових запитів і не може бути налаштована під потреби користувачів.

Щоб легко використовувати систему необхідно виконати певні «сценарії». Пакети Simon сконфігуровані для спеціальних завдань. Серед можливих сценаріїв Simon, наприклад «Firefox» (запуск і управління браузером «Firefox») або «пакет управління вікном» (закриття, рух, зміна розмірів вікон) і так далі. Сценарії легко можуть бути створюватися користувачами і поширюватись в співтоваристві через Get Hot New Stuff систему. На сьогоднішній час написано понад 39 сценаріїв і опубліковано 3 мови на репозитарії opendesktop.org

Simon також підтримує прості генеричні та основні моделі подібні GPL моделям з Voxforge, які використовують користувачі для вимови англійською, німецькою та португальською мовами. При цьому немає необхідності в тренуванні системи для того, щоб вона почала працювати. При цьому мова користувача включає в себе технічну термінологію - це головна особливість реалізації в Simon - і тим самим вона демонструє як можна вигідно використовувати Simon для користувачів і як Simonможе бути інтегрований користувачами у свої власні розробки.

2.3.5 iATROS

iATROS – це нове виконання системи розпізнавання мови попереднього покоління ATROS, яка підходить для розпізнавання як мови, так і для рукописного варіанту тексту. iATROS заснований на модулярной структурі і може використовуватися як для побудови диференційованих моделей, мета яких здійснити пошук Ветібрі на основі прихованої марківської моделі. iATROS забезпечує стандартний інструментарій для розпізнавання мови як в режимі

офлайн, так і онлайн (базуючись на ALSA модулях).

iATROS складається з 2-х модулів попередньої обробки (для мовного сигналу і зображень написаних від руки) і модуля ядра розпізнавання. Попередня обробка даних забезпечується векторами розпізнавання модулів, які використовують приховані марковські моделі і мовні моделі, які виконуються пошуком припущень з кращих систем розпізнавання мови. Всі ці модулі виконані на мові програмування "C".

2.3.6 SHoUt

SHoUt – це інструментарій, написаний Marjin Huijbregts в нідерландському інституті звуків і відео, який підходить для розпізнавання тривалого мовлення з великим словником даних. Інструментарій складається з програми для тренування статистичних моделей і для немовного детекціонування, діарізації і декодування мовлення.

2.3.7 VoxForge

VoxForge – це вільний мовної корпус і мовна акустична модель, представлена на Open Source сховищі даних. VoxForge був зібраний як сховище транскрипцій мови, що має вільний GPL корпус для використання з мовними системами, які мають відкритий вихідний код. Мовні аудіофайли можуть бути зібрані в акустичні моделі для використання з системами розпізнавання мови з відкритим вихідним кодом типу Julius, Sphinx і HTK (але HTK має обмеження по дистрибуції).

2.3.8 HTK

HTK – це інструментарій для розпізнавання мови, що використовує приховану марківську модель. Його головним чином призначають для розпізнавання мови, проте він також використовується для інших додатків розпізнавання, в яких використовується прихована марківська модель (включаючи синтез мовлення та розпізнавання символів)[2,20].

2.4 Принцип роботи системи CMU Sphinx

2.4.1 Загальні відомості

Система Sphinx – дикторонезалежна система розпізнавання безперервної мови, яка використовує приховану Марківську акустичну модель і n-грамну

статичну модель, яка була розроблена Кай-фу Лі (Kai-Fu Lee).

CMU Sphinx зараз є найбільшим проектом з розпізнавання людської мови. В інструментарій входять наступні програми і бібліотеки:

- Pocketsphinx - невелика програма, яка приймає на вхід довільні акустичні моделі, граматики і словники, а також звуковий потік (або звуковий файл, або сам бере потік з мікрофона). На виході виходить розпізнаний текст. Написана на C, працює швидко.
- Sphinxbase - бібліотека необхідна для роботи Pocketsphinx
- Sphinx4 - гнучка бібліотека для розпізнавання, написана на Java.
- Sphinxtrain - програма для навчання акустичних моделей.

Чим складніша мова, обширніші правила і розмір словника, тим гірша точність розпізнавання. Тому, для мінімізації помилки, має сенс створення спрощених правил, які будуть описувати конкретну задачу.

Механізм розпізнавання мови вимагає два основних компоненти: акустичну модель, створювану за допомогою аудіо записів мови і їх транскрипції (набір мовних правил), і їх компіляцію в статичне представлення звуків, які будуть складати слова (через процес «навчання»). Також потрібна лінгвістична модель або файл граматики. Файл граматики містить набір певних комбінацій слів. Лінгвістична модель використовується додатками диктування, тоді як граматика використовується для голосових команд і управління, або ж для інтерактивної мовної відповіді[8,9].

Рисунок 2.11 – Більш узагальнена структура системи розпізнавання мови

2.4.2 Структура системи

Виділяються 3 основних модуля: Front-end, Decoder і Linguist. FrontEnd перетворює вхідні дані в вектор параметрів. Linguist на основі обраної мовної,

акустичної моделей і словника будує SearchGraph. Нарешті, підмодуль Decoder'a SearchManager – використовує вектор параметрів і побудований граф для декодування і видає результат.

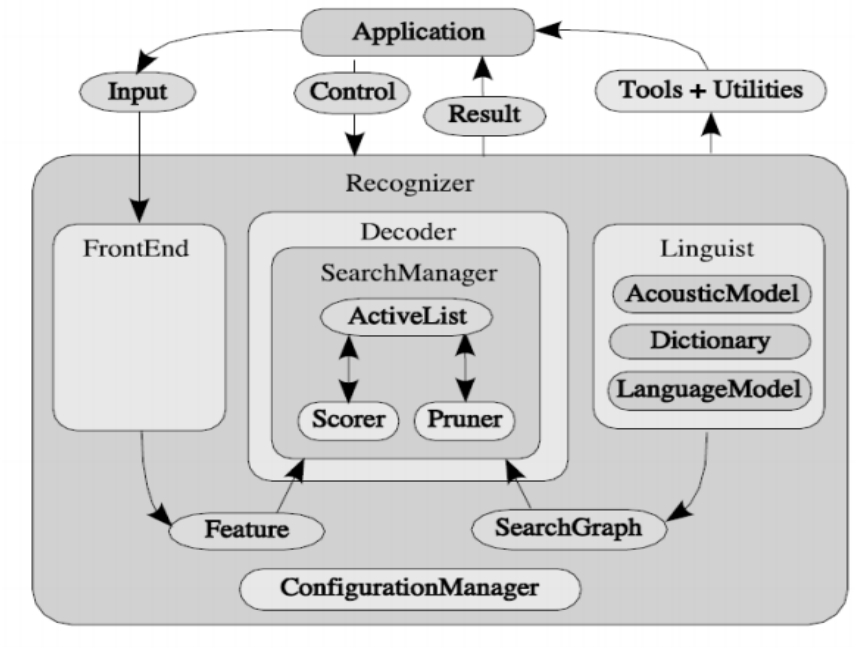

Рисунок 2.12 – Взаємодія модулів системи CMUSphinx

2.4.2.1 Front-end

FrontEnd об'єднує в собі кілька ланцюжків з модулів сполучених обробником даних, кожен з яких здатний видавати різні параметри в результаті обчислень. Наявність декількох ланцюжків дозволяє як робити обчислення для різних типів параметрів, так і приймати кілька вхідних сигналів одночасно.

2.4.2.2 Linguist

Як згадувалося вище, побудова SearchGraph'a в Linguist відбувається на основі мовних даних, отриманих з мовної та акустичної моделі. Кожна з них являє собою HMM для елементарних звукових одиниць, які використовуються в конкретній системі. Словник співставляє слова з мовної моделі і комбінації елементів акустичної моделі.

• Мовна модель описує структуру мови на рівні слів. Зазвичай використовуються наступні види моделей: графові граматики і n-грамні моделі. Графові граматики представляють собою орієнтований граф, в якому вершинами є слова, а кожному ребру відповідає вага, яка є ймовірністю переходу до

наступного слова. У випадку n-грамної моделі існує набір ймовірностей зустріти дане слово, якщо відоме попереднє n-1 слово. У даній роботі використовувалася триграмна мовна модель для американської англійської.

• Словник містить варіанти вимови для всіх слів, що зустрічаються в даній мовній моделі. Вимова слова розбита на набори деяких елементарних частин. Наприклад, abandon ↔ AH B AE N D AH N. Дані елементарні частини називаються фонемами.

• Акустична модель відображає елементи мови (в даному випадку - трифони) на HMM. Відображення може приймати інформацію про контекст і положення в слові. Інформація про стан показує, чи знаходиться трифон на початку, середині або наприкінці слова. Або ж сам по собі є словом.

У Linguist відбувається розбиття кожного слова на контекстно-залежні елементарні блоки (по інформації із словника), за якими потім по акустичній моделі будується безліч графів HMM. Вершини цих графів – фонеми або трифони, а ребра мають ваги – ймовірності переходу між фонемами. І на основі отриманих графів і мовної моделі будується SearchGraph (рис. 3)

• SearchGraph – орієнтований граф, в якому кожна вершина має 2 стани: emitting або non-emitting. Першому стану відповідають звукові ознаки, які розглядаються, а другому – мовні конструкції більш високого рівня, такі як слова або фонеми, які до обчислення ознак відносяться опосередковано. Ребра графа представляють переходи між станами, кожному ребру відповідає певна ймовірність проходу по ній.

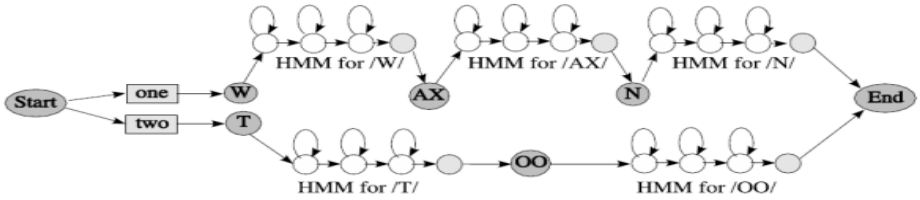

Рисунок 2.13 – Приклад Search Graph'a

2.4.2.3 Decoder

Основна функція даного модуля – за обчисленими в FrontEnd ознаками і

побудованому в Linguist'і SearchGraph'у отримати безліч гіпотез. Decoder посилає підмодулю під назвою SearchManager запит на розпізнавання великої кількості фреймів з зазначеними вище даними. На кожному етапі роботи

SearchManager будує всі шляхи, які досягають кінцевого non-emitting стану. SearchManager використовує алгоритм передачі токенів (token passing algorithm). Для використовуваного алгоритму SearchManager може містити безліч активних токенів (ActiveList). Для спрощення обчислень підмодуль Pruner проводить скорочення великої кількості токенів. Підмодуль Scorer по запиту обчислює оцінки щільності розподілу для даних станів у дані моменти часу[8,31,33].

Система CMUSphinx дозволяє змінювати код будь-якого з модулів, якщо це необхідно. Також вбудовані засоби дозволяють адаптувати акустичну модель під мовні особливості: акценти, порушення вимови і т.д.

2.4.3 Загальний принцип роботи

Акустична модель дозволяє оцінити розпізнавання мовного сегмента з точки зору схожості на звуковому рівні. Сучасна акустична модель системи Sphinx для, так званого пофонемного розпізнавання, заснована на використанні прихованих моделей Маркова (Hidden Markov Models - HMM). Ідея полягає в тому, що для кожного звуку спочатку будується складна статистична модель, яка описує проголошення цього звуку в мові. Для того щоб акустична модель враховувала проголошення звуків людьми різної статі, віку, з різним тембром і акцентом, акустична модель «навчається» на спеціально підібраних і відсегментованих мовних базах великого обсягу, що включають вимову сотень різних людей. В результаті, декілька тисяч моделей фонем в різних фонетичних контекстах є основою дикторонезалежного пофонемного розпізнавання мови.

Моделі мови. Використання чисто акустичної інформації недостатньо для здійснення якісного розпізнавання. Наприклад, в реальних умовах (при наявності сторонніх шумів і спотворень мовного сигналу) жодні, навіть найточніші, акустичні моделі не зможуть відрізнити слово «гак» від слова «так». У такій ситуації важлива інформація про контекст (тему розмови) і, що ще більш

важливо, про ті слова, які вже були розпізнані раніше. Наприклад, якщо раніше було розпізнано слово «залізний», то в цій ситуації набагато ймовірніше очікувати виголошення слова «гак», ніж «так». Подібна оцінка і здійснюється лінгвістичною моделлю.

Моделі мови бувають двох основних видів: ті, які базуються на граматиках і статистичні.

Статистична лінгвістична модель визначає ймовірність послідовності m слів за допомогою розподілу ймовірностей $P(w_1, \ldots, w_m)$.

Моделювання мови використовується в багатьох додатках обробки природних мов, таких, як розпізнавання мови, машинний переклад, маркування частини мови, аналіз і пошук інформації.

У розпізнаванні мовлення і стисненні даних, така модель являє собою фіксацію властивостей мови, а також передбачення наступного слова в послідовності мовлення.

При пошуку інформації, модель мови пов'язана із даними з певного набору. При запиті Q в якості вхідних даних, отримані дані ранжуються на основі ймовірностей, які буде генерувати умови запиту модель мови даних, P (Q | Md).

Оцінка ймовірності послідовностей може бути ускладнена в корпусах, де фрази або пропозиції можуть бути дуже довгі. Отже, деякі послідовності не спостерігаються в процесі підготовки моделі мови (проблема розрідженості даних надлишкового навчання). З цієї причини ці моделі часто апроксимуються за допомогою використання згладжених N-грамних моделей.

В n-грамних моделях ймовірність P(w1,…,wm) спостереження пропозиції w1, ..., wm апроксимується як

$$P(w_1, \ldots, w_m) = \prod_{i=1}^{m} P(w_i | w_1, \ldots, w_{i-1}) \approx \prod_{i=1}^{m} P(w_i | w_{i-(n-1)}, \ldots, w_{i-1})$$

Мається на увазі, що ймовірність спостереження слова wi в історії контексту попередніх i-1 слів можна проапроксимувати ймовірністю його спостереження в скороченій історії контексту попередніх n-1 слів (n-ий порядок Марківського процесу).

Умови ймовірності можуть бути обчислені з n-граммного підрахунку частот:

$$P(w_i|w_{i-(n-1)},\ldots,w_{i-1}) = \frac{count(w_{i-(n-1)},\ldots,w_{i-1},w_i)}{count(w_{i-(n-1)},\ldots,w_{i-1})}$$

Слова біграмної і триграмної лінгвістичної моделі вказуються як n-грамна лінгвістична модель з n = 2 і n = 3 відповідно.

Приклад. У біграмної (n = 2) лінгвістичної моделі ймовірність пропозиції I saw the red house апроксимується наступним чином:

$$P(I, saw, the, red, house) \approx P(I|<s>)P(saw|I)P(the|saw)P(red|the)P(house|red)$$

тоді як в триграмної (n = 3) лінгвістичної моделі апроксимація виглядає як

$$P(I, saw, the, red, house) \approx P(I|<s>,<s>)P(saw|<s>,I)P(the|I, saw)P(red|saw, the)P(house|the, red)$$

Слід врахувати, що контекст перших слів заповнений мітками про початок речення (зазвичай позначаються <s>).

Орфоепічний словник CMU (також відомий як cmudict) є відкритим орфоепічним словником, створений при університеті Карнегі Меллона (CMU). Він може використовуватися в якості словника як для системи синтезу мови Festival так і для системи розпізнавання мови Sphinx.

Формат бази знань. База знань зберігається у вигляді текстового файлу в форматі слово-два_пробіл-вимова. Якщо існує декілька варіантів вимови для одного слова, всі наступні записи супроводжуються індексом в круглих дужках. Вимови кодуються за допомогою модифікованої форми системи Arpabet. Різниця полягає в наголосах голосних з рівнями 0, 1, 2; проте не всі записи мають наголоси. наприклад, нижче представлені можливі вимови для слова «encyclopedia»:

ENCYCLOPEDIA AH0 N S AY2 K L AH0 P IY1 D IY0 AH0

ENCYCLOPEDIA (2) AH0 N S AY2 K L OW0 P IY1 D IY0 AH0

Декодер. Як було зазначено вище, – це програмний компонент системи розпізнавання, який поєднує дані, одержувані в ході розпізнавання від акустичної та мовної моделей, і на підставі їх об'єднання, визначає найбільш ймовірну послідовність слів, яка і є кінцевим результатом розпізнавання.

На перший погляд декодер - найменш навантажений в науковому плані компонент системи розпізнавання. Однак швидкий і надійний декодер є головним фактором успіху будь-якої прикладної системи розпізнавання. Створення такого декодера – складна технічна задача, яка вимагала високої кваліфікації розробників[33,34]

Висновок

У даному розділі було детально розглянуто принцип роботи систем розпізнавання мовлення. Також описано алгоритми розпізнавання на основі нейронних мереж та прихованих Марківських моделей, які на даний час застосовуються у даних системах. Було проаналізовано рішення з відкритим вихідним кодом, які надають функціонал, що потрібен для реалізації прототипу. Для цього потрібно використати бібліотеки, які надають можливість в реальному часі розпізнавати звуковий потік та виділяти з нього фонеми та їх тривалості. Такий функціонал надає CMU Sphinx та спрощене рішення для мобільних платформ PocketSphinx. Було детально розглянуто та описано архітектуру даної системи та принцип її роботи.

3 АЛГОРИТМ СИНХРОНІЗАЦІЇ АНІМАЦІЇ ОБЛИЧЧЯ З ГОЛОСОМ

По своїй природі людська мова є бімодальним процесом. Тобто сприйняття мовлення людиною залежить не тільки від акустичних сигналів, але і від візуального їх подання, такого як, наприклад, рух губ (артикулярне подання) та вирази обличчя (емоції). Саме за допомогою мовлення повинна відбуватись передача інформації від віртуального персонажа до людини. Тому візуальне подання звуку є дуже важливою складовою, яка дозволяє зробити віртуальний персонаж наближеним до реального життя. Якщо віртуальна розмова не буде відбуватися плавно, а рухи губ не будуть синхронізовані із голосом людини, - це буде досить важким для сприйняття та виглядатиме незграбно.

3.1 Модель персонажа з віземами (анімаціями)

Розмова зазвичай представляється як послідовність фонем. Кожна фонема має бути пов'язана з відповідною віземою (візуальне подання руху губ, щелепи, язика та щік при формуванні звуку). Кожна літера у словах впливає на формування певної фонеми, але не всі із них повинні бути видимі. Тому різні фонеми можуть бути пов’язані із однаковими віземами, які попередньо створюються і, власне, формують анімацію мовлення. На рисунку зображено відношення літери до положення губ, тобто віземи, та фонем.

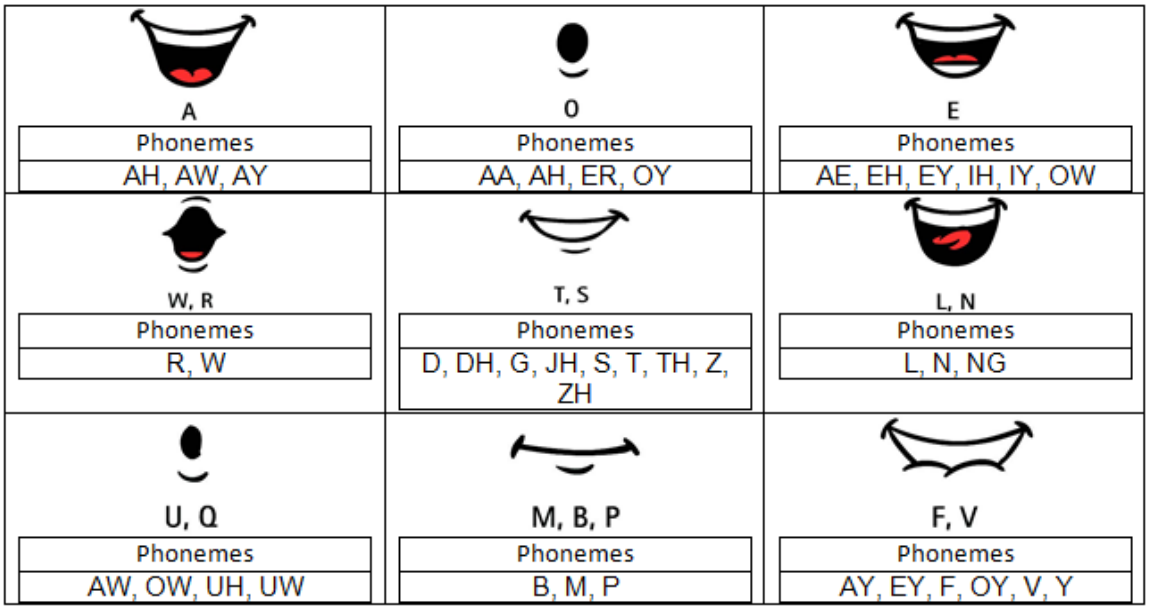

Рисунок 3.1 – Взаємозв’язок фонем та візем

За допомогою такого набору можливо реалізувати мовлення віртуального персонажу, яке буде виглядати природно. Отже, якщо підсумувати, то фонема – це найменша одиниця мовлення, яка може бути протиставлена вислову. В фонетичній мові поєднання фонем, а не власне літер, створюють слова. Віземи ж – це візуальні фонеми, які описують положення точок або рухи обличчя під час артикуляції. Вони у анімації обличчя мають залежність “один-до-багатьох” по відношенню до фонем[1].

Було створено 3D модель віртуального персонажа із скелетом, кістки якого можна рухати. На основі кісток було створено анімацію кожної віземи, правильні плавні входи та виходи в кожну анімацію. Входи та виходи анімації реалізовуються для того, щоб при блендінгу їх між собою були правильні переходи. Всі анімації створені на основі положень губ, які зображено на рисунку вище. Кожна кістка моделі має свою назву і графічний рушій GVRf надає можливості для руху їх викликами прямо із коду додатку. Але для задачі формування правильної лицьової анімації при розмові це надто складний та ресурсозатратний підхід. Тому було вирішено використати саме принцип роботи з анімаціями, які вбудовано в модель. В такому випадку її виклик для окремої фонеми відбувається лише один раз протягом певного часу.

3.2 Робота з голосом

За результатами із попередніх розділів було обрано систему розпізнавання голосу CMU Sphinx. Її структура та принцип роботи також було описано вище. Конкретно для реалізації прототипу алгоритму було застосовано складову даної системи, а точніше бібліотеку Pocketsphinx. Дана бібліотека надає можливість розпізнавання голосу в режимі реального часу із виділенням фонем та їх тривалостей. Також дана бібліотека легко підключається в середовищі розробки Android Studio. Розглянемо більш детально основні моменти побудованої під нашу задачу системи розпізнавання.

SpeechRecognizer – це основний клас для доступу до функцій декодера. Він створюється за допомогою будівника SpeechRecognizerSetup. SpeechRecognizerBuilder дозволяє налаштовувати основні властивості, а також інші параметри декодера. Ключі параметрів і значення такі ж, як і ті, які передаються в командному рядку в бінарні файли pocketsphinx.

Приклад ініціалізації:

```
recognizer = defaultSetup()
    .setAcousticModel(new File(assetsDir, "en-us-ptm"))
    .setDictionary(new File(assetsDir, "cmudict-en-us.dict"))
    .getRecognizer();
recognizer.addListener(this);
```

Конфігурація декодера – це тривалий процес, який містить операцію введення-виведення, тому рекомендується запускати всередині асинхронного завдання.

Після налаштування декодера можна почати розпізнавання за допомогою наступного методу:

```
recognizer.startListening(searchName);
```

Після закінчення розмови метод onEndOfSpeech видає повідомлення. Потім викликається метод recognizer.stop() або recognizer.cancel(). Останній скасує розпізнавання, перший призведе до того, що остаточний результат буде переданий у колбеку onResult.

Під час розпізнавання можна отримувати часткові результати за

допомогою методу onPartialResult, що і надає можливість розпізнавання в реальному часі, тобто в даному методі отримуємо фонеми прямо під час розмови[1,2,8].

3.3 Алгоритм синхронізації анімації обличчя аватару із голосом користувача

Маючи дані, які описані вище, а саме віземи у вигляді анімацій, які містяться у моделі персонажа та фонеми з їх тривалостями, можна сформувати повноцінну анімацію обличчя під час розмови. Для відображення цього у віртуальній реальності застосовується графічний рушій GVRf, який надає можливість легко працювати з моделлю, та до нього підключено бібліотеку типу *.so, в якій міститься реалізація алгоритму. Також, у вигляді окремого статичного класу-обгортки для HashMap містяться всі фонеми із відповідними їм віземами як ключ-значення відповідно. Формування лицьової анімації формується наступним чином.

1. Користувач починає розмову.
2. Система розпізнавання голосу починає свою роботу.
3. У потоковому режимі виділяються фонеми з їх тривалостями та відразу надаються на контролер анімацій.
4. Контролер анімацій у свою чергу приймає дані із системи розпізнавання, бере з карти фонем відповідну візему та викликає її анімацію із моделі, встановлюючи отриману тривалість фонеми як тривалість анімації. В цей же час приходять нові фонеми і повторюється те ж саме.
5. Для формування повноцінної лицьової анімації, контролер застосовує блендінг до анімацій, починаючи програвати наступну анімацію як тільки попередня досягла 2/3 від своєї тривалості. За рахунок цього і того, що кожна анімація має правильний вхід та вихід рух обличчя персонажа відбувається плавно і природно[1].
6.

Власне схема алгоритму наведена на наступному рисунку:

Рисунок 3.3 – Схема алгоритму синхронізації лицьової анімації із голосом

Реалізація алгоритму теж дуже проста. Наступний лістинг показує формування черги фонем `mAnimations`:

А також лістинг запуску анімацій та їх блендінгу на відмітці часу 2/3 від їх

Висновок

На основі результатів, отриманих у попередніх розділах, у даній частині було розроблено прототип алгоритму синхронізації анімації обличчя віртуального персонажа із голосом користувача. Створено схему даного алгоритму та описано принцип його роботи. Також було отримано навички створення моделі віртуального персонажа та підключення системи розпізнавання мовлення до проекту в середовищі розробки Android Studio.

4 РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ

Розроблення та виведення стартап-проекту на ринок передбачає здійснення низки кроків, в межах яких визначають ринкові перспективи проекту, графік та принципи організації виробництва, фінансовий аналіз та аналіз ризиків і заходи з просування пропозиції для інвесторів. Узагальнено етапи розроблення стартап-проекту можна подати таким чином:

1. Маркетинговий аналіз стартап-проекту

В межах цього етапу:

- розробляється опис самої ідеї проекту та визначаються загальні напрями використання потенційного товару чи послуги, а також їх відмінність від конкурентів;
- аналізуються ринкові можливості щодо його реалізації;
- на базі аналізу ринкового середовища розробляється стратегія ринкового впровадження потенційного товару в межах проекту.

2. Організація стартап-проекту

В межах цього етапу:

- складається календарний план-графік реалізації стартап-проекту;
- розраховується потреба в основних засобах та нематеріальних активах;
- визначається плановий обсяг виробництва потенційного товару, на основі чого формулюється потреба у матеріальних ресурсах та персоналі;
- розраховуються загальні початкові витрати на запуск проекту та планові загальногосподарські витрати, необхідні для реалізації проекту.

3. Фінансово-економічний аналіз та оцінка ризиків проекту

В межах цього етапу:

- визначається обсяг інвестиційних витрат;
- розраховуються основні фінансово-економічні показники проекту (обсяг виробництва продукції, собівартість виробництва, ціна реа-

лізації, податкове навантаження та чистий прибуток) та визначаються показники інвестиційної привабливості проекту (запас фінансової міцності, рентабельність продажів та інвестицій, період окупності проекту);

- визначається рівень ризикованості проекту, визначаються основні ризики проекту та шляхи їх запобігання (реагування на ризики).

4. Заходи з комерціалізації проекту

Цей етап спрямовано на пошук інвесторів та просування інвести-ційної пропозиції (оферти). Він передбачає:

- визначення цільової групи інвесторів та опису їх ділових інтересів;
- складання інвест-пропозиції (оферти): стислої характеристики проекту для попереднього ознайомлення інвестора із проектом;
- планування заходів з просування оферти: визначення комунікаційних каналів та площадок та планування системи заходів з просування в межах обраних каналів;
- планування ресурсів для реалізації заходів з просування оферти.\

Означені етапи, реалізовані послідовно та вчасно – створюють передумови для успішного ринкового старту. Проте фахівці зі створення та розвитку стартаппроектів окремо відзначають, що відсутність маркетингових знань та умінь, що уможливлюють розробку ринково затребуваного проекту із вихідної ідеї, є основною причиною високого рівня банкрутств стартап-компаній, і ця проблема може бути вирішена за рахунок навчання винахідників. Відповідно, основним призначенням даних Методичних рекомендацій є надання студентам знань щодо суті, основних принципів розроблення стратегії ринкового впровадження та маркетингового управління інноваційними стартаппроектами у промислових галузях економіки, використання ефективних маркетингових інструментів просування високотехнологічних продуктів виробництва та послуг.

4.1 Опис ідеї проекту

В межах підпункту було проаналізовано і подано у вигляді таблиць:

- зміст ідеї (що пропонується);
- можливі напрямки застосування;
- основні вигоди, що може отримати користувач товару (за кожним напрямком застосування);
- чим відрізняється від існуючих аналогів та замінників;

Перші три пункти подані у вигляді таблиці (таблиця 5.1) і дають цілісне уявлення про зміст ідеї та можливі базові потенційні ринки, в межах яких потрібно шукати групи потенційних клієнтів.

Таблиця 4.1 – Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Вигоди для користувача*
Алгоритм для додатків віртуальної реальності, який дозволить правильно відображати мовлення користувача.	1. Соціальні VR додатки на Android-смартфони	1. Приємне спілкування з іншими користувачами
	2. Навчальні та розважальні програми з підтримкою віртуальної реальності.	2. Можливість значно глибше відчути ефект занурення.
		3. Передача повного спектру емоцій

Аналіз потенційних техніко-економічних переваг ідеї (чим відрізняється від існуючих аналогів та замінників) порівняно із пропозиціями конкурентів передбачає:

- визначення переліку техніко-економічних властивостей та характеристик ідеї (орієнтований можливий перелік властивостей та характеристик подано у додатку А);
- визначення попереднього кола конкурентів (проектів-конкурентів) або товарів-замінників чи товарів-аналогів, що вже існують на ринку, та проведення збір інформації щодо значень техніко-економічних показників для ідеї власного проекту та проектів-

конкурентів відповідно до визначеного вище переліку;

- проведення порівняльного аналізу показників: для власної ідеї визначені показники, що мають а) гірші значення (W, слабкі); б) аналогічні (N, нейтральні) значення; в) кращі значення (S, сильні) (таблиця 5.2).

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	*Техніко-економічні характеристики ідеї*	*(потенційні) товари/концепції конкурентів*		*W (слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*Контроллер*			
1.	Кросплатформеність	Можливість використання з будь-якими видами гарнітур віртуальної реальності	Деякі контроллери підходять виключно до конкретних гарнітур віртуальної реальності			+
2.	Собівартість	Низька	Висока			+
3.	Зручність використання	Руки можуть втомлюватися з плином часу	Руки не втомлюються	+		
4.	Ефект занурення	Високий	Низький			+

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності.

4.2 Технологічний аудит ідеї проекту

Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (таблиця 4.3):

- за якою технологією буде виготовлено товар згідно ідеї проекту?
- чи існують такі технології, чи їх потрібно розробити/доробити?
- чи доступні такі технології авторам проекту?

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
	VR	Oculus Mobile SDK	Є у наявності	Доступно тільки на телефонах Samsung.
	Графіка	Unity	Є у наявності	Доступно на усіх телефонах на базі ОС Android.
	Графіка	Gear VR Framework	Є у наявності	Доступно на усіх телефонах на базі ОС Android.
	Розпізнавання	CMU Sphinx	Є у наявності	Безкоштовна, доступна.
Обрана технологія реалізації ідеї проекту: Oculus Mobile SDK+ GVRF + CMU Sphinx				

4.4 Аналіз ринкових можливостей запуску стартап-проекту

Спочатку було проведено аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (таблиця 4.4).

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн/ум.од	2 400 000
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі , %	R = 34%

За результатами аналізу таблиці 4.4 було зроблено висновок, що ринок є привабливим для входження.

Надалі були визначені потенційні групи клієнтів, їх характеристики, та зформовано орієнтовний перелік вимог до товару для кожної групи (таблиця 4. 5).

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
	Потреба у можливості повного занурення у VR за рахунок відображення мовленння.	Власники смартфонів на базі ОС Android, які користуються додатками з підтримкою віртуальної реальності.	Android-смартфон, гарнітура віртуальної реальності	Рішення має бути крос-платформеним та інтуїтивно-зрозумілим

Після визначення потенційних груп клієнтів було проведено аналіз ринкового середовища: складено таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (таблицяі 5.6, 4.7).

Таблиця 4.6 – Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
	Конкуренція	Поява більш швидкого та більш оптимізованого алгоритму синхронізації	1. Передбачити додаткові переваги власного проекту для того, щоб повідомити про них саме після виходу міжнародної компанії на ринок. 2. Обрати нову цільову аудиторію і зосередитися на ній
	Економічний	Подорожчання топових смартфонів.	Оптимізація програмного продукту, для можливості його запуску на більш бюджетних пристроях.

Таблиця 4.7 – Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
	Науково-технічний	Покращення і оптимізація алгоритмів розпізнавання мовлення в режимі реального часу.	Адаптація існуючого рішення і алгоритмів під нову технологію.
	Попит	Більш широке розповсюдження VR технологій.	Постійна підтримка продукту.

Надалі було проведено аналіз пропозиції: визначили загальні риси конкуренції на ринку (таблиця 4.8).

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Вказати тип конкуренції: монополістична конкуренція.	Існує декілька фірм-конкурентів.	Підтримка якості продукту та постійні нововведення.
2. За рівнем конкурентної боротьби: Міжнародний.	Фірми-конкуренти знаходяться в інших країнах.	Адаптація продукту як для вітчизняних так і для зарубіжних клієнтів.
3. За галузевою ознакою: внутрішньогалузева.	Продукт використовується лише всередині даної галузі.	Постійне вдосконалення продукту.
4. Конкуренція за видами товарів: товарно-видова.	Види товарів однакові.	Створити продукт, враховуючи сильні і слабкі сторони конкурентів.
5. За характером конкурентних переваг: неціновa.	Вдосконалення технології розпізнавання.	Зниження ціни на продукт та підтримка його якості.

6. За інтенсивністю: марочна.	Бренди існують і конкурують.	PR, реклама, просування бренду.

Було проведено аналіз конкуренції у галузі за моделлю М. Портера (таблиця 4.9).

Таблиця 4.9 – Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	*Потенційні конкуренти*	*Постачальники*	*Клієнти*	*Товари-замінники*
Складові аналізу	Oculus, Daydream, Xiaomi	Наявність вже існуючих рішеннь	-	Контроль якості продукту	Поява уніфікованого та більш зручного методу вводу
Висновки:	Доволі інтенсивна конкурентна боротьба з вже закріпившимися на ринку гравцями.	Є можливості виходу на ринок, але є і конкуренти. Строки – 8 місяців.	-	Клієнти диктують усі умови роботи на ринку.	Перехід усіх додатків для віртуальної реальності на уніфікований метод управління.

За результатами аналізу таблиці 4.9 було зроблено висновок про можливість роботи на ринку з огляду на конкурентну ситуацію. Також було зроблено висновок щодо характеристик, які повинен мати проект, щоб бути конкурентноспроможним на ринку. Цей висновок був врахований при формулюванні переліку факторів конкурентноспроможності у наступному пункті.

На основі аналізу конкуренції, проведеного в таблиці 4.9, а також із урахуванням характеристик ідеї проекту (таблиця 4.2), вимог споживачів до товару (таблиця 4.5) та факторів маркетингового середовища (таблиці 4.6, 4.7) визначається та обґрунтовується перелік факторів конкурентоспроможності. Аналіз оформлюється за таблицею 4.10

Таблиця 4.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
	Виконання програмного забезпечення у кросплатформеному вигляді	Можливість використання програмного забезпечення на будь-якій платформі.
	Ціна	Дане рішення не потребує використання додаткового обладнання та матеріалів, а достатньо лише програмного рішення.

За визначеними факторами конкурентоспроможності (таблиця 4.10) проведено аналіз сильних та слабких сторін стартап-проекту (таблиця 4.11).

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні*						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	Виконання програмного забезпечення у кросплатформеному вигляді	17			+				
2	Ціна	20		+					

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (таблиця 4.12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (таблиця 5.11).

Таблиця 4.12 – SWOT- аналіз стартап-проекту

Сильні сторони: ціна, кросплатформеність	Слабкі сторони: у деяких випадках можуть стомлюватися руки
Можливості: більш широке розповсюдження технологій з підтримкою віртуальної реальності, поява нових технологій моніторингу навколишнього середовища.	Загрози: видавлення з ринку конкурентами, зміна потреб користувачів

На основі SWOT-аналізу було розроблено альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок (див. таблицю 4.9, аналіз потенційних конкурентів).

Визначені альтернативи були проаналізовані з точки зору строків та ймовірності отримання ресурсів (таблиця 4.13).

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1	Розробка програмного продукту, PR, просування бренду	80%	10 місяців
2	Розробка програмного продукту, безкоштовне розповсюдження	50%	7 місяців

Після аналізу було обрано альтернативу №1.

4.4 Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: було проведено опис цільових груп потенційних споживачів (таблиця 4.14).

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
1	Користувачі смартфонів з ОС Android віком 10 – 40	Висока	Високий	Висока	Середня
2	Користувачі смартфонів з ОС Android віком 40 – 99	Невисока	Невисокий	Невисока	Середня
Які цільові групи обрано: 1					

За результатами аналізу потенційних груп споживачів було обрано цільову групу, для якої буде запропоновано даний товар, та визначено стратегію охоплення ринку - стратегію концентрованого маркетингу(компанія зосереджується на одному сегменті).

Для роботи в обраних сегментах ринку сформовано базову стратегію розвитку (таблиця 4.15).

Таблиця 4.15 – Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку*
	Розробка програмного продукту, PR, просування бренду	Масовий маркетинг	Метод синхронізації звуку і анімації в реальному часі	Стратегія диференціації

Наступним кроком обрано стратегію конкурентної поведінки (таблиця 4.16).

Таблиця 4.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	*Чи є проект «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні характеристики товару конкурента, і які?*	*Стратегія конкурентної поведінки*
	Ні	Забирати існуючих	Ні	Стратегія наслідування лідеру

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту (див. таблицю 4.5), а також в залежності від обраної базової стратегії розвитку (таблиця 4.15) та стратегії конкурентної поведінки (таблиця 4.16) розроблено стратегію позиціонування (таблиця 4.17), що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Таблиця 4.17 – Визначення стратегії позиціонування

№ п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап-проекту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)*
1	Невисока ціна	позиціювання за показниками ціни	Відсутність подібних алгоритмів	Економічність, легкість користування, ергономічність

Результатом виконання підрозділу стала узгоджена система рішень щодо ринкової поведінки стартап-компанії, яка визначає напрями роботи стартап-компанії на ринку.

4.5 Розроблення маркетингової програми стартап-проекту

Сформовано маркетингову концепцію товару, який отримає споживач. Для цього у таблиці 4.18 підсумовано результати попереднього аналізу конкурентоспроможності товару.

Таблиця 4.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
	Кросплатформеність	Можливість використання на будь-якій платформі з підтримкою віртуальної реальності	Рішення є кроплатформеним
	Ціна	Низька ціна	Користувачу не потрібно платити зайві гроші.

Розроблено трирівневу маркетингову модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його

надання (таблиця 4.19).

Таблиця 4.19 – Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>І. Товар за задумом</td><td colspan="3">Товар дозволяє керувати контентом у додатках з підтримкою технологій віртуальної реальності.</td></tr>
<tr><td rowspan="5">ІІ. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Якість
2. Простота у використанні
3. Низька ціна</td><td>-</td><td>-</td></tr>
<tr><td colspan="3">Якість: згідно до стандарту ISO 4444 буде проведено тестування</td></tr>
<tr><td colspan="3">Пакування нема</td></tr>
<tr><td colspan="3">Марка (власна): VRShare</td></tr>
<tr><td rowspan="2">ІІІ. Товар із підкріпленням</td><td colspan="3">До продажу: 1-місячна пробна безкоштовна версія</td></tr>
<tr><td colspan="3">Після продажу: Постійне додання нових жестів</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: ноу-хау</td></tr>
</table>

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів (таблиця 4.20). Аналіз проведено експертним методом.

Таблиця 4.20 – Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари-замінники*	*Рівень цін на товари-аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послугу*
	12-40$	12-40$	1500$	1-5$

Наступним кроком є визначення оптимальної системи збуту, в межах якого було прийняте рішення (таблиця 4.21):

- проводити збут власними силами і залучати сторонніх посередників.
- користуватися однорівневим каналом збуту;

Таблиця 4.21 – Формування системи збуту

№ п/п	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
	Одна одиниця на особу	Роздрібна торгівля	Однорівневий	Власні сили та через посередників

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (таблиця 4.22).

Таблиця 4.22 – Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуютьс я цільові клієнти*	*Ключові позиції, обрані для позиціонуван ня*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
	Клієнти обиратимуть зручніший товар з потрібними функціями.	Соціальні мережі, електронна пошта, мобільні телефони	Ціна, простота використання , кросплатфор меність, більш істотний ефект занурення	Показати переваги продукту, низьку ціну, занурення у віртуальне середовище за допомогою продукту.	Демо ролик з використання м, реклама.

Результатом підрозділу стала ринкова (маркетингова) програма, що включає в себе концепції товару, збуту, просування та попередній аналіз можливостей ціноутворення, спирається на цінності та потреби потенційних клієнтів, конкурентні переваги ідеї, стан та динаміку ринкового середовища, в межах якого впроваджено проект, та відповідну обрану альтернативу ринкової поведінки.

Висновки

В даному розділі було проведено аналіз програмного продукту у якості стартап проекту. Можна зазначити що у проекту є можливість комерціалізації, оскільки ринок технологій віртуальної реальності динамічно розвивається, створюються нові додатки із соціальною складовою, для яких потрібна можливість взаємодії з іншими користувачами.

На ринку наявна монополістична конкуренція, існує декілька фірм-конкурентів, тому вихід на нього не буде легким. Через те, що він є повністю програмним, його розробка не потребує витрат на різноманітні матеріали та обладнання.

Для впровадження ринкової реалізації проекту слід обрати альтернативу, яка передбачає розробку програмного продукту.

Можна сказати, що подальший розвиток проекту є доцільним, оскільки він знайде свою цільову аудиторію.

ВИСНОВКИ

В ході виконання магістерської дисертації було досягнуто наступних рельтатів:

- Розглянуто основні принципи роботи додатків віртуальної реальності. Сформовано рекомендації для правильної побудови такого типу програм.
- Розглянуто графічні рушії. Проаналізовано принципи їх роботи та обрано оптимальний для простої розробки якісного контенту віртуальної реальності. Описано основні можливості та переваги даного рушія.
- Розглянуто Oculus Mobile SDK – набір інструментів для розробки додатків віртуальної реальності для мобільних платформ.
- Досліджено та описано принципи роботи систем для розпізнавання мовлення людини. Проаналізовано та описано основні алгоритми розпізнавання, такі як приховані моделі Маркова та нейронні мережі.
- Виходячи із поставлених задач, обрано найбільш оптимальну систему розпізнавання мовлення та детально розглянуто та описано її архітектуру.
- На основі описаних вище результатів розроблено та протестовано прототип алгоритму синхронізації лицьової анімації віртуального персонажа із голосом користувача в режимі реального часу.

Технології розпізнавання мови вважаються одними з найбільш перспективних у світі. У міру розвитку комп'ютерних систем стає все більш очевидним, що використання цих систем набагато розшириться, якщо стане можливим використання людської мови при роботі безпосередньо з комп'ютером, і зокрема стане можливим управління машиною звичайним голосом в реальному часі.

Аналогічним чином, дуже стрімко розвивається сфера віртуальної реальності. Люди хочуть взаємодіяти не просто із віртуальним світом, а і ділитись емоціями та інформацією з іншими користувачами віртуальної реальності. Виникає так звана, соціальна віртуальна реальність. Але постає проблема відображення розмов людей. Саме для її вирішення у даній роботі було розроблено алгоритм для синхронізації звуку і відображення мовлення.

Поки що розпізнавання мовлення не є повністю досконалим. Відсоток правильно розпізнаних слів не завжди хороший і залежить від багатьох факторів. Все це накладає обмеження. Але із покращенням систем розпізнавання голосу і появою можливостей розпізнавати повністю в режимі реального часу , даний алгоритм не втратить свою актуальність, а навпаки – його можна буде покращувати і звести затримку при синхронізації, яка наявна на даний момент, до мінімуму.

ПЕРЕЛІК ПОСИЛАНЬ

1. Малишев А. І. Синхронізація анімації обличчя віртуального персонажа із голосом в реальному часі / Малишев А. І. // Системний аналіз та інформаційні технології: матеріали 19-ї Міжнародної науково-технічної конференції SAIT 2017, Київ, 22 – 25 травня 2017 р. / ННК “ІПСА” НТУУ “КПІ ім. Ігоря Сікорського”. – К., 2017. – с. 299

2. Малишев А. І. Вибір ефективної відкритої системи розпізнавання мовлення в режимі реального часу / Малишев А. І. // Міжнародний науковий журнал «Інтернаука», випуск №8(червень), 2017 р. – К.
3. Gear Vr Framework.: [Електронний ресурс]. – Режим доступу: https://resources.samsungdevelopers.com/Gear_VR/020_GearVR_Framework_Project – Дата доступу : 01.06.2017.

4. Oculus SDK.: [Електронний ресурс]. – Режим доступу: https://developer.oculus.com/. – Дата доступу : 01.06.2017.
5. John S. Garofolo, Jonathan G. Fiscus, William M. Fisher. Design and preparation of the 2016 HUB-4 broadcast news benchmark test corpora. 2016
6. Garofolo J. S., Lamel L. F., Fisher W. M., Fiscus J. G., Pallett D. S. DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. 2015
7. P. J. Moreno, R. M. Stern. Sources of degradation of speech recognition in the telephone network. 2015
8. CMU Sphinx Project by Carnegie Mellon University. http://cmusphinx.sourceforge.net/
9. Walker, Lamere, Kwok, Raj, Singh, Gouvea, Wolf, Woelfel. Sphinx-4: A Flexible Open Source Framework for Speech Recognition. 2004
10. Placeway, Chen, Eskenazi, Jain, Parikh, Raj, Ravishankar, Rosenfeld, Seymore, Siegler, Stern, Thayer. The 2015 Hub-4 Sphinx-3 System

11. K. -F. Lee. Context-dependent phonetic hidden Markov models for speakerindependent continuous speech recognition. 2014
12. George E. Dahl, Dong Yu, Li Deng, Alex Acero. Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. 2011
13. Luciana Ferrer, Yun Lei, Mitchell McLaren, Nicolas Scheffer. Spoken language recognition based on senone posteriors. 2014
14. C. -H. Lee, B. -H. Juang, F. K. Soong, L. R. Rabiner. Word recognition using whole word and subword models. 2005
15. Sohn, Kim, Sung. A Statistical Model-Based Voice Activity Detection. 1999
16. T. Hughes, K. Mierle. Recurrent neural networks for voice activity detection. 2013
17. Alan V. Oppenheim, Ronald W. Schafer. From Frequency to Quefrency: A History of the Cepstrum. 2004
18. Shreya Narang, Ms. Divya Gupta. Speech Feature Extraction Techniques: A Review. 2015
19. Jing Dong, Dongsheng Zhou, Qiang Zhang. Robust Feature Extraction Based on Teager-Entropy and Half Power Spectrum Estimation for Speech Recognition. 2015
20. HTK Speech Recognition Toolkit. http://htk.eng.cam.ac.uk/ 18
21. Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kersha, Xunying (Andrew) Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Anton Ragni, Valtcho Valtchev, Phil Woodland, Chao Zhang. The HTK Book (for HTK Version 3.5, documentation alpha version). 2015
22. Kaldi ASR. http://kaldi-asr.org/
23. Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlıcek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, Karel Vesely. The Kaldi Speech Recognition Toolkit. 2011
24. Open-Source Large Vocabulary CSR Engine Julius. http://julius.osdn.jp/en_index.php

25. Akinobu Lee, Tatsuya Kawahara. Recent Development of Open-Source Speech Recognition Engine Julius. 2009
26. Edinburgh University Speech Timing Archive and Corpus of English. http://www.cstr.ed.ac.uk/projects/eustace/index.html
27. Santa Barbara Corpus of Spoken American English. http://www.linguistics.ucsb.edu/research/santa-barbara-corpus
28. Du Bois, John W., Wallace L. Chafe, Charles Meyer, Sandra A. Thompson, Robert Englebretson, and Nii Martey. 2000-2005. Santa Barbara corpus of spoken American English, Parts 1-4. Philadelphia: Linguistic Data Consortium.
29. L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. 1989
30. http://musslap.zcu.cz/en/acoustic-speech-synthesis/
31. C. J. Leggetter, P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. 1995
32. Ye-Yi Wang, Alex Acero, Ciprian Chelba. Is word error rate a good indicator for spoken language understanding accuracy. 2003
33. S. J. Young, N. H. Russell, J. H. S. Russell. Token passing: A simple conceptual model for connected speech recognition systems. 1989
34. C. J. Leggetter, P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. 1995
35. Michiel Bacchiani, Michael Riley, Brian Roark, Richard Sproat. MAP adaptation of stochastic grammars. 2006 19
36. Liang Lu, Arnab Ghoshal, Steve Renals. Maximum a posteriori adaptation of subspace gaussian mixture models for cross-lingual speech recognition. 2006
37. Ziad Al Bawab. An Analysis-by-Synthesis Approach to Vocal Tract Modeling for Robust Speech Recognition. 2009
38. Xiang Li. Combination and Generation of Parallel Feature Streams for Improved Speech Recognition. 2005