ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

УДК 51-76

Отчет о программном проекте на тему:

Создание алгоритма вычисления параметров движения для проведения графомоторного тренинга с помощью планшета

Выполнил:
студент группы БПМИ219
Сидоров Дмитрий Алексеевич


(подпись)

18.05.2023
(дата)

Принял руководитель проекта:
Чернышев Всеволод Леонидович
Доцент департамента больших данных
и информационного поиска
Факультета компьютерных наук НИУ ВШЭ


(подпись)

18.05.2023
(дата)

Москва 2023

Содержание

Аннотация

В данной работе описывается создание приложения, которое будет измерять отклонение от идеальной траектории при проведении графомоторного тренинга с помощью планшета. Приложение необходимо для автоматизирования обработки результатов тренинга, в котором ребенку нужно переводить карандаш от одного кружка на листе бумаги к другому под ритм.

Ключевые слова

Графомоторный тренинг, объектно ориентированное программирование, Unity.

1 Введение

Известно, что у детей после перенесенных онкозаболеваний нарушается перифирическая нервная система, в том числе способность управлять рукой, и ухудшается почерк. Например, после перенесенного заболевания у ребенка уменьшается скорость написания текста, ухудшаются ровность и точность строчки, ширина и пропорциональность букв. При этом для того, чтобы вернуть ребенка в школу, необходимо восстановить способность писать.

В настоящее время у нейробиологов есть тренинг, который позволяет тренировать руку с помощью бумажных листов и ручек. Врачи расставляют на листе бумаги кружочки, и ребенку нужно переводить карандаш от одного кружка к другому под ритм, при этом не выходя за границу очерченной области. Например, на Рисунке 1.1 изображен успешно выполненный тренинг, так как ребёнок почти не выходит за границы области, а на Рисунке 1.2 можно увидеть, что ребёнок не попадает в очерченную область на протяжении всего тренинга, поэтому данный тест считается неуспешным.

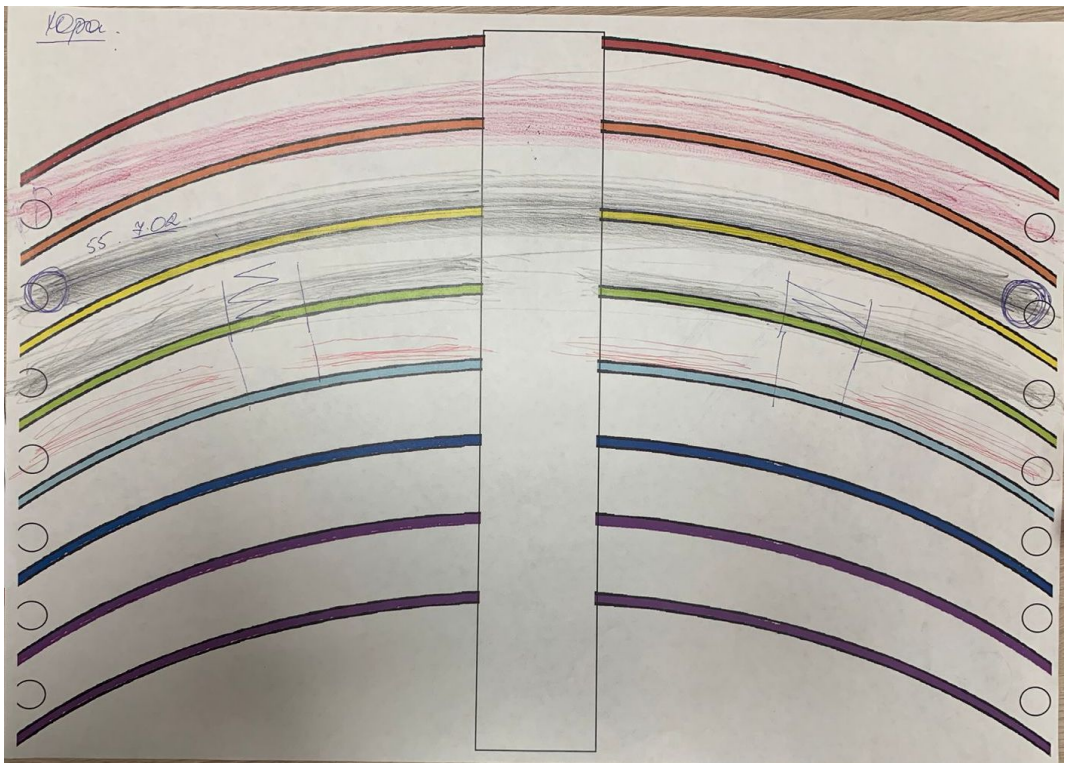

Рис. 1.1: Пример успешно выполненного тренинга.

Этот тренинг обладает рядом недостатков. Во-первых, бумажный тренинг не позволяет точно определить успешность выполнения задания, и врачам приходится измерять точность попадания ребёнка в область "на глаз". Во-вторых, при таком подходе у нейробиологов нет возможности подстраивать тест под конкретного ребёнка, например, уменьшать область, в которой нужно проводить линии, если ребёнок хорошо справляется с тестом. В-третьих,

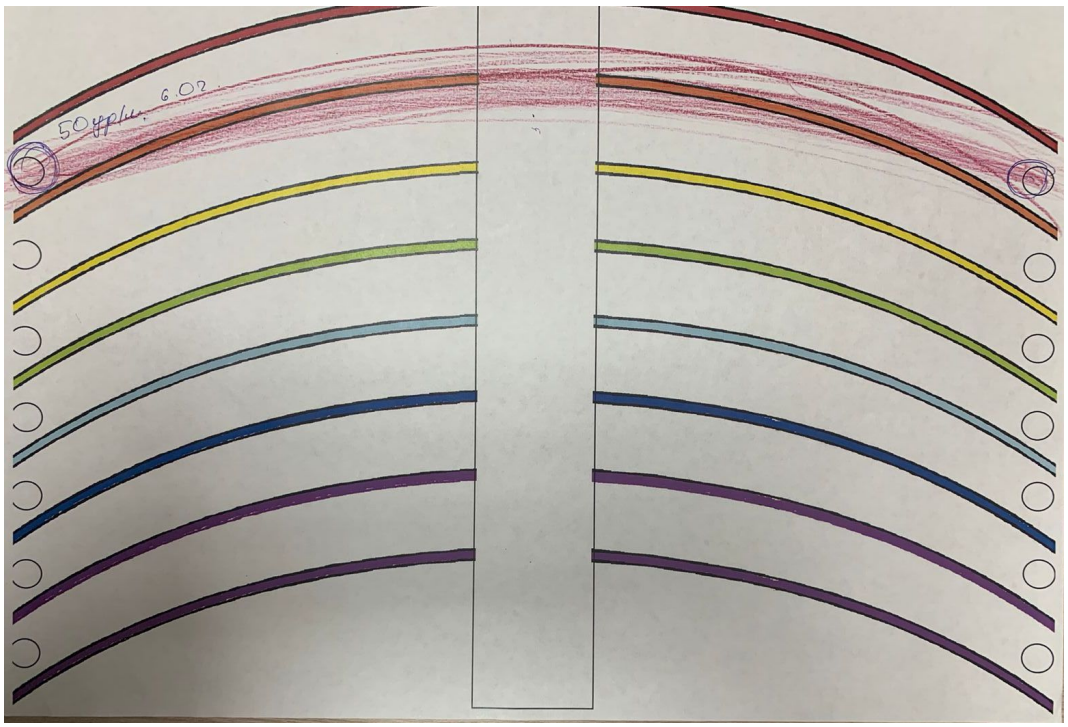

Рис. 1.2: Пример неуспешного тренинга, ребёнок выходит за пределы выделенной области.

дети не могут удалённо выполнять тренинг, и им приходится приезжать из далеких городов России и жить вдали от родных. Кроме того, они должный выполнять тренинг почти каждый день, и интерактивный тренинг может повысить их интерес к упражнению.

Для проведения графомоторного тренинга у нейробиологов есть планшеты на платформе Windows. Поскольку планируется, что дети смогут пройти новый тренинг удалённо, а большинство планшетов имеют операционную систему Android или IPadOS, мной была выбрана среда разработки Unity, так как она позволяет писать приложение на языке C#, которое можно импортировать сразу для всех операционных систем, указанных выше.

2 Постановка задачи

Из существующих недостатков бумажного тренинга можно определить основной функционал, которым должно обладать приложение. Во-первых, приложение должно считать отклонение проведенных ребенком линий от идеальной программной траектории. Более формально, во время начала тренинга счётчик ошибки равен нулю. Далее во время каждого кадра приложение должно получать текущую координату касания, находить расстояние от этой координаты до соответствующей координаты идеальной траектории и прибавлять полученное расстояние к счётчику.

Во-вторых, приложение должно корректировать положение границ во время выпол-

нения тренинга: приложение должно понимать, что ребёнок не пересекает границы заданное время, которое определяет врач, и уменьшать расстояние между границами. Аналогично приложение должно увеличивать расстояние между границами, если ребенок пересек их. Кроме того, приложение должно иметь возможность вручную настроить такие параметры как скорость сужения/расширения границ, начальное расстояние между границами, максимальное/минимальное расстояние между границами и другие.

В-третьих, поскольку скорость перемещения ребёнка между терминальными точками (началом и концом движения) регулируется с помощью метронома, необходимо реализовать метроном в приложении, причем необходимо добавить возможность врачам менять частоту ударов метронома.

Так же приложение должно иметь историю и файловый вывод с результатами теста, о формате которых я напишу далее.

3 Проектирование приложения

3.1 Основные термины, определения и сокращения

Двигательный контроль – способность управлять двигательной активностью, реализуемая на основе взаимодействия нервной системы и скелетной мускулатуры.

Дегенерация – процесс упрощения организации, связанный с исчезновением органов и функций, а также целых систем органов.

Дисграфия – нарушение письма, не связанное с интеллектом. Каллиграфические ошибки – изменение формы, размера, пространственного положения буквы.

Когнитивные функции – способность понимать, познавать, изучать, осознавать, воспринимать и перерабатывать внешнюю информацию.

Нейродегенеративные заболевания – группа в основном медленно прогрессирующих, наследственных или приобретённых заболеваний нервной системы.

Онкология – раздел медицины, изучающий доброкачественные и злокачественные опухоли, механизмы и закономерности их возникновения и развития, методы их профилактики, диагностики и лечения.

Периферическая нервная система – условно выделяемая часть нервной системы, находящаяся за пределами головного и спинного мозга, которая состоит из черепных и спинальных нервов, а также нервов и сплетений вегетативной нервной системы, соединяя центральную нервную систему с органами тела.

3.2 Обзор литературы

В статье [**1**] выделяются следующие характеристики качеств письма:

– четкость и разборчивость письма (правильное и четкое начертание всех букв в соответствии с прописями);

– равнонаклонность (соблюдение наклона), параллельность одинаково направленных штрихов;

– линейность (соблюдение строки снизу и сверху);

– правильность соединения элементов при написании букв, букв в словах;

– плавность письма при достаточной скорости;

– равное расстояние между элементами в букве, между буквами в слове, между словами в строке (равномерное, пропорциональное расстояние, равное одной букве и);

– одинаковая высота букв.

В статье [**2**] эти характеристики используются для сравнения почерка здоровых людей и людей, имеющих болезнь Альцгеймера или болезнь Паркинсона. Для анализа исследователи используют графический планшет, который позволяет не только изобразить ход движений пера, но так же определить угол наклона относительно поверхности, силу нажатия, скорость и ускорение пера. Согласно этому анализу тонкой моторики у здоровых и нездоровых людей был сделан вывод, что у пациентов было больше трудностей с выполнением плавных движений, чем с контролем длины. Результаты показали, что данная закономерность сохраняется при выполнении любого задания. Однако в данном исследовании принимали участие только взрослые пациенты, поэтому нет гарантии, что такие результаты сохранятся и в случае дисграфии у детей.

В исследовании [**3**] сравнивались 3 группы: дети с дисграфией, дети без дисграфии и взрослые без дисграфии. В ходе экспериментов была предложена новая переменная для более точной оценки аномальных колебаний профиля скорости движения почерка – разность пиков скорости, которую использовали для оценки нарушения беглости движений при написании текста. Ученые пришли к выводу, что новая переменная показывает плавность движения, независимую от изменения траектории пера, то есть избегает шум, возникающий из-за неспособности двигательной системы ограничивать естественную степень изменчивости движений.

Из выше описанных статей следует, что для анализа тренинга важен не столько факт пересечения границ, сколько расстояние и время, на которое ребенок пересек эти границы. Таким образом, при пересечении границ в счетчику отклонения будет прибавляться расстоя-

ние до идеальной траектории, а так же в приложение должна быть добавлена опция выбора дополнительного штрафа – числа, на которое нужно увеличить счетчик каждые, например, 0.1 секунду, которую ребенок проводит вне границ. Кроме того, необходимо добавить возможность врачам самим регулировать дополнительный штраф, в том числе установить его нулем, то есть выключить его.

Так же, суммируя информацию, полученную из статей, и запросы нейробиологов, в приложение стоит добавить историю тренингов, хранящую время начала, продолжительность и счётчик для конкретного теста, а так же для удобной выгрузки результатов, необходимо добавить создание текстовых файлов (то есть формата .txt), хранящих информацию о каждом касании ребенка (координата, расстояние до идеальной траектории, время, проведшее с первого касания, дата теста).

3.3 Проектирование сцен

По запросу нейробиологов, приложение должно иметь страницу с настройками, на которой можно вручную изменять и сохранять параметры, а так же удалять историю тренингов. Поэтому было решено, что у приложения не будет начального экрана, и сразу при входе пользователь будет попадать на главный экран с тренингом. На этом экране пользователь сможет проводить линии между границами, видеть текущую продолжительность теста и текущую сумму расстояний до идеальной траектории, включать/выключать метроном и открывать меню паузы, в которое можно вынести вспомогательный функцционал приложения, чтобы не нагружать интерфейс главной страницы. В меню паузы нужно было добавить кнопки перехода в меню настроек и открытия истории тренингов, кнопку сохранения несохраненных тренингов (дата, продолжительность, сумма расстояний), а так же кнопки для выхода из приложения и возврата на главный экран с тренингом.

3.4 Проектирование границ с помощью кривой Безье

Одной из важных составляющих приложения являются дуги-границы области, внутри которой ребенок должен проводить линии. В Unity нет встроенной реализации дуг, поэтому их будет необходимо реализовать самостоятельно. Границы должны обладать следующими свойствами: во-первых, они должны представлять собой плавные и непрерывные дуги и быть визуально приятными для пользователей и способствовать естественному движению. Во-вторых, их надо реализовать так, чтобы потом можно было быстро и эффективно найти расстояние от любой точки плоскости до кривой. В-третьих, чтобы определить положение

кривой – идеальной траектории и положение терминальных точек, нужно уметь вычислять расстояние между границами.

Из статьи [4] я узнал, что этим свойствам удовлетворяет кривая Безье. Кроме того, кривая Безье позволяет легко настраивать форму и кривизну дуги, используя контрольные точки. Это позволяет создавать разнообразные формы и эффекты, адаптируясь к различным дизайнам и требованиям приложения. Таким образом, при необходимости, достаточно изменить несколько параметров и получить новые кривые, что может сделать даже человек, незнакомый с программированием, в том числе врачи.

Для программирования кривой Безье я буду использовать 3 точки: 2 точки над терминальными в случае верхней границы (2 под терминальными на том же расстоянии от них для нижней) и точку между двумя предыдущими, поднятую (опущенную) так, чтобы между тремя точками образовался тупой угол (расстояние от точки в вершине угла до прямой, образованной двумя другими точками, равно $\frac{1}{4}$ расстояния между этими двумя точками), см. пример построенной таким образом кривой Безье на Рисунке 3.1.

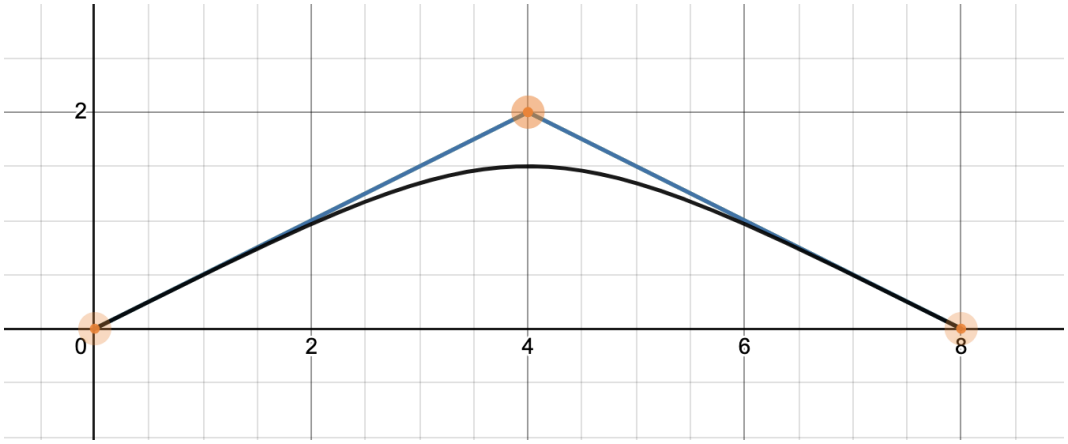

Рис. 3.1: Пример кривой Безье с тупым углом

Точку B кривой Безье можно построить, используя формулу (1)

$$B = (1-t)\cdot(1-t)\cdot point_0 + 2\cdot(1-t)\cdot t\cdot point_1 + t\cdot t\cdot point_2, \tag{1}$$

где $point_0$, $point_2$ – левая и правая терминальные точки соответственно, $point_1$ – точка – вершина угла, t – коэффициент, значение которого будет описано ниже.

Тогда псевдокод формирования границ будет выглядеть следующим образом:

1) фиксируем параметр $t = 0$ типа float

2) генерируем новую точку границы по формуле (1)

3) прибавляем к параметру t число $\frac{1}{\text{номер итерации}}$

4) повторяем шаги выше n раз (например, можно взять $n = 200$)

Таким образом, используя написанный выше алгоритм, получим n точек, последовательность которых образует требуемую границу.

4 Реализация приложения

4.1 Реализация страницы с тренингом

4.1.1 Реализация границ и идеальной траектории

Поскольку точки кривой Безье приходится часто генирировать, для оптимизации приложения было принято решено хранить только 200 точек. Кроме того, приложение работает с частотой 60 кадров/сек, и во время каждого кадра необходимо считать расстояние от точки текущего касания до идеальной траектории, что является ресурсоемким процессом. Посоветовавшись с нейробиологами и проанализировав поведение участников тренинга, я решил, что в качестве идеальной траектории можно брать не кривую Безье, лежащую между границами, а верхнюю часть эллипса, приближающую кривую Безье так, чтобы расстояние между эллипсом и кривой было не больше 0.0001 относительных единиц (так как я делал масштаб подстраивающимся под размер экрана, то это число одинаково на всех поддерживаемых устройствах). Таким образом, если приложение работает с частотой 60 кадров/сек, в среднем тренинг длится не более 1 мин, то отличие сумм расстояний до эллипса и до честной идеальной траектории составляет не более 0.36 условных единиц. За 1 мин в среднем здоровый человек набирает сумму расстояний до идеальной траектории, полученной с помощью кривой Безье, равную 10 (пользователь, имеющий проблемы с двигательным контролем, набирает больше), значит общая погрешность равна менее 5%, что приемлемо для нейробиологов, но даёт значительный выигрыш в производительности приложения, что особенно актуально при необходимости сделать приложение доступным для большинства устройств.

4.1.2 Реализация нахождения расстояния от точки касания до идеальной траектории

С учетом предыдущего пункта, далее под "идеальной траекторией" подразумевается приближение кривой Безье верхней частью эллипса. Теперь необходимо реализовать подсчет расстояния от точки касания до идеальной траектории. Более формально, в каждый момент касания экрана приложение получает координату (x, y) касания (обозначим точку касания как A), нужно найти такую точку верхней части эллипса с координатами (x, y'), что ее

абсцисса совпадает с абсциссой точки касания. Обозначим точку пересечения вертикальной прямой, проходящей через точку A, и идеальной траектории как точку B. Тогда искомым расстоянием до эллипса является модуль разности ординат полученных точек $|y - y'|$. На Рисунке 4.1 точка касания находится над идеальной траекторией, а на Рисунке 4.2 точка касания находится под идеальной траекторией. В обоих случаях искомое расстояние равно длине отрезка AB.

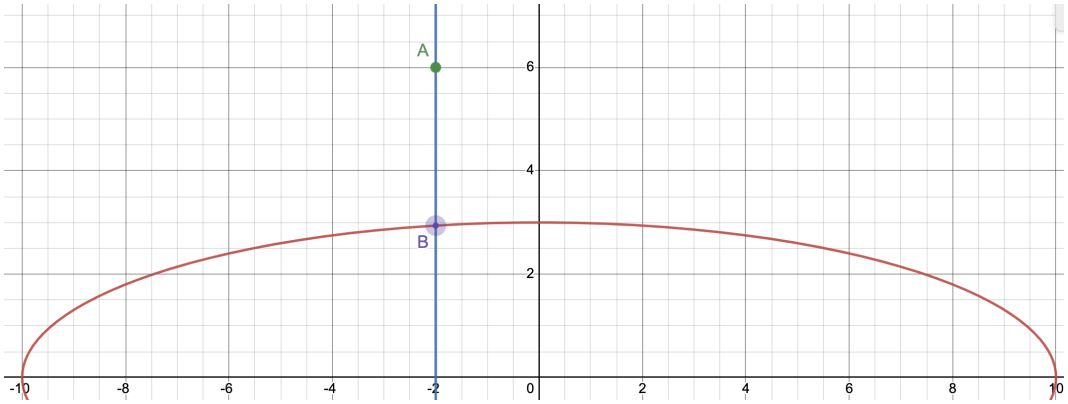

Рис. 4.1: Расстояние от точки касания до идеальной траектории (касание над верхней частью эллипса)

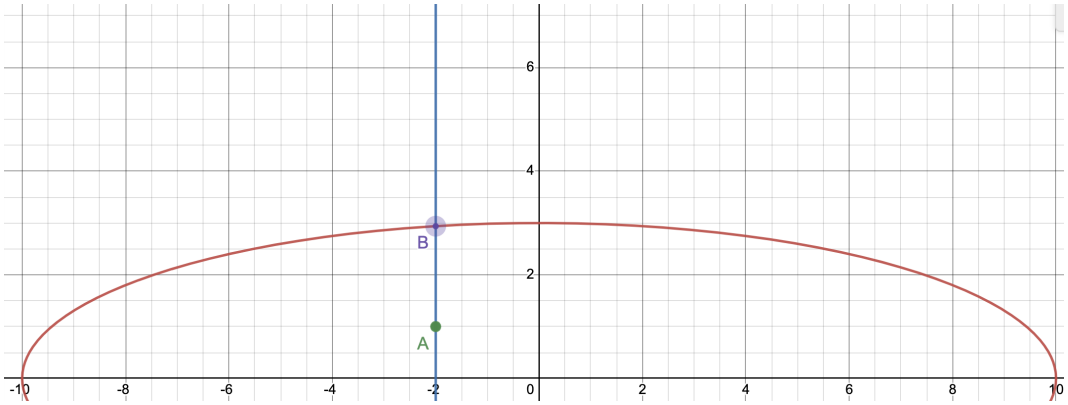

Рис. 4.2: Расстояние от точки касания до идеальной траектории (касание под верхней частью эллипса)

Известно, что любой эллипс с центром в точке $(0, 0)$ можно задать с помощью следующей формулы

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 = r^2 \tag{2}$$

В моём случае в формуле (2) я использовал $r = 1$. Из статьи [**5**] я узнал следующий эффективный алгоритм нахождения расстояния от любой точки плоскости до эллипса:

1) Считаем, что параметры a, b и координаты точки на плоскости, для которой ищем расстояние, (x_0, y_0) известны. Задаём $x = a\sqrt{2}$, $y = b\sqrt{2}$. Инициализируем вспомогательные параметы $t = 0$, $t_x = \sqrt{2}$, $t_y = \sqrt{2}$, $\Delta x = 0$, $\Delta y = 0$.

2) Обновляем параметры следующим образом:

$$\Delta x = \frac{(a^2 - b^2) \cdot t_x^3}{a} \tag{3}$$

$$\Delta y = \frac{(b^2 - a^2) \cdot t_y^3}{b} \tag{4}$$

$$t_x = \min\left(1,\ \max\left(0,\ \frac{\frac{(|x_0| - \Delta x) \cdot \sqrt{(x-\Delta x)^2 + (y-\Delta y)^2}}{\sqrt{(|x_0| - \Delta x)^2 + (|y_0| - \Delta y)^2}} + \Delta x}{a}\right)\right) \tag{5}$$

$$t_y = \min\left(1,\ \max\left(0,\ \frac{\frac{(|y_0| - \Delta y) \cdot \sqrt{(x-\Delta x)^2 + (y-\Delta y)^2}}{\sqrt{(|x_0| - \Delta x)^2 + (|y_0| - \Delta y)^2}} + \Delta y}{b}\right)\right) \tag{6}$$

$$t = \sqrt{t_x^2 + t_y^2} \tag{7}$$

$$t_x = \frac{t_x}{t},\ t_y = \frac{t_y}{t} \tag{8}$$

3) Повторяем шаг 2 $n = 3$ раз,

4) Искомая точка эллипса будет иметь координату $(\text{sign}(x_0) \cdot t_x \cdot a,\ \text{sign}(y_0) \cdot t_y \cdot b)$, где $\text{sign}(\alpha)$ – знак числа α.

Однако данный метод имеет существенный недостаток: он находит расстояние до любой ближайшей точки всего эллипса, а не только до его верхней части. Поэтому я модифицировал описанный выше алгоритм. Так как я использую эллипс с центром в точке $(0, 0)$, для точек касания с неотрицательной ординатой алгоритм можно не менять, тк для таких точек ближайшая точка эллипса лежит на его верхней части. Для точек с отрицательной ординатой (обозначим такую точку A) алгоритм вернет координату $(x, y),\ y < 0$ точки, лежащей на нижней части эллипса (обозначим такую точку B). Заметим, что в силу симметрии эллипса относительно оси абсцисс, ближайшая к точке A точка B', лежащая на верхней части эллипса, будет иметь координату $(x, -y)$ (см. Рисунок 4.3).

Таким образом, я реализовал алгоритм нахождения расстояния от точки касания экрана до идеальной траектории.

4.1.3 Реализация движения границ

Одной из задач, поставленных нейробиологами, было создание возможности адаптировать приложение для конкретного ребёнка. В ходе обсуждения с врачами мной был

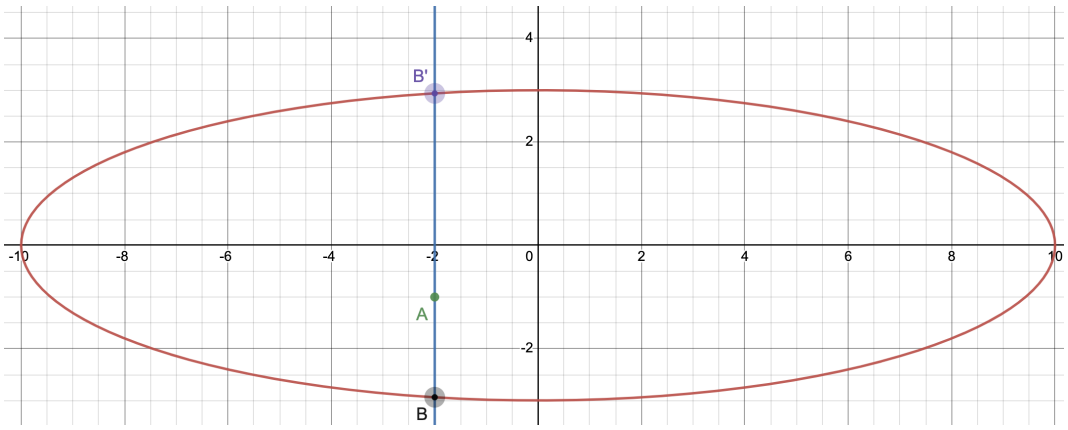

Рис. 4.3: Нахождение точки на идеальной траектории для точек касания с отрицательной ординатой

предложен следующий алгоритм: каждый кадр программа сначала получает точку касания экрана и находит расстояние от этой точки до идеальной траектории. Затем, если это расстояние меньше или равно расстоянию от идеальной траектории до каждой из границ, то есть пользователь не вышел за границы, то программа уменьшает и увеличивает ординаты точек, по которым строятся верхняя и нижняя граница (кривые Безье) соответственно, границы сужаются. Если пользователь вышел за границы, то программа увеличивает и уменьшает ординаты точек, по которым строятся верхняя и нижняя граница (кривые Безье) соответственно, границы расширяются. Величины, на которые увеличиваются/уменьшаются ординаты, можно регулировать на странице с настройками, в том числе сделать их различными, например, сделать скорость расширения границ быстрее скорости их сужения. Так же было добавлено минимальное и макимальное расстояние между границами, которое можно регулировать в настройках.

В ходе тестирования было выявлено, что обновление положения границ во время каждого кадра может сильно сказаться на производительности на некоторых устройствах, поэтому было решено сделать обновление границ каждые 0.1 сек, а не во время каждого кадра. С учётом того, что приложение работает с частотой 60 кадров/сек, это изменение дало ощутимый выигрыш в производительности, но при этом визуально почти не ухудшило плавность движения границ.

4.1.4 Реализация движения терминальных точек

Еще одной из задач, поставленных нейробиологами, было создание терминальных точек, то есть точек, с помощью которых ребенок должен начинать и заканчивать проводить линии. Поскольку было реализовано сужение и расширение границ, было решено так же сделать уменьшение и увеличение радиуса терминальных точек. По аналогии с границами,

я ограничил величину радиуса минимальным и макимальным значением, которое можно регулировать в настройках. По умолчанию радиус точек изменяется с той же скоростью, с которой изменяется расстояние между границами (в том числе скорость увеличения радиуса больше, чем скорость уменьшения). Примеры положения границ и размера терминальных точек можно увидеть на Рисунках 4.4 и 4.5.

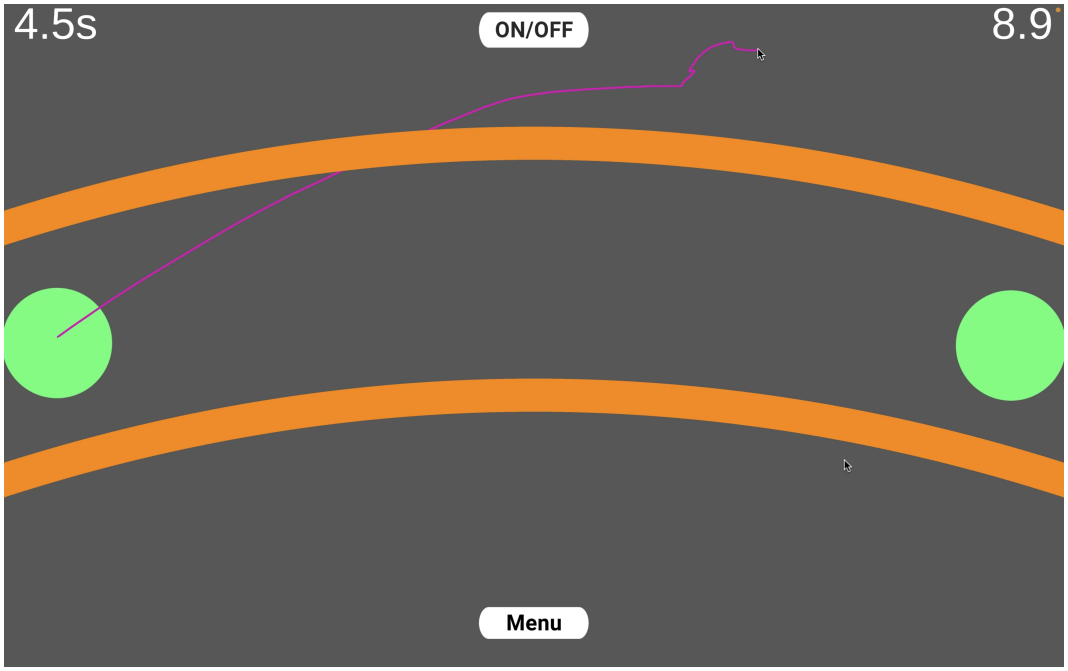

Рис. 4.4: Положение границ при длительном непопадании в выделенную область

4.1.5 Реализация интерфейса

Помимо границ теста и терминальных точек, на главный экран я добавил таймер, показывающий продолжительность теста, счетчик, показывающий сумму отклонение, кнопки для включения метронома и перехода в меню паузы. В итоге получился интерфейс приложения, который можно увидеть на Рисунке 4.6.

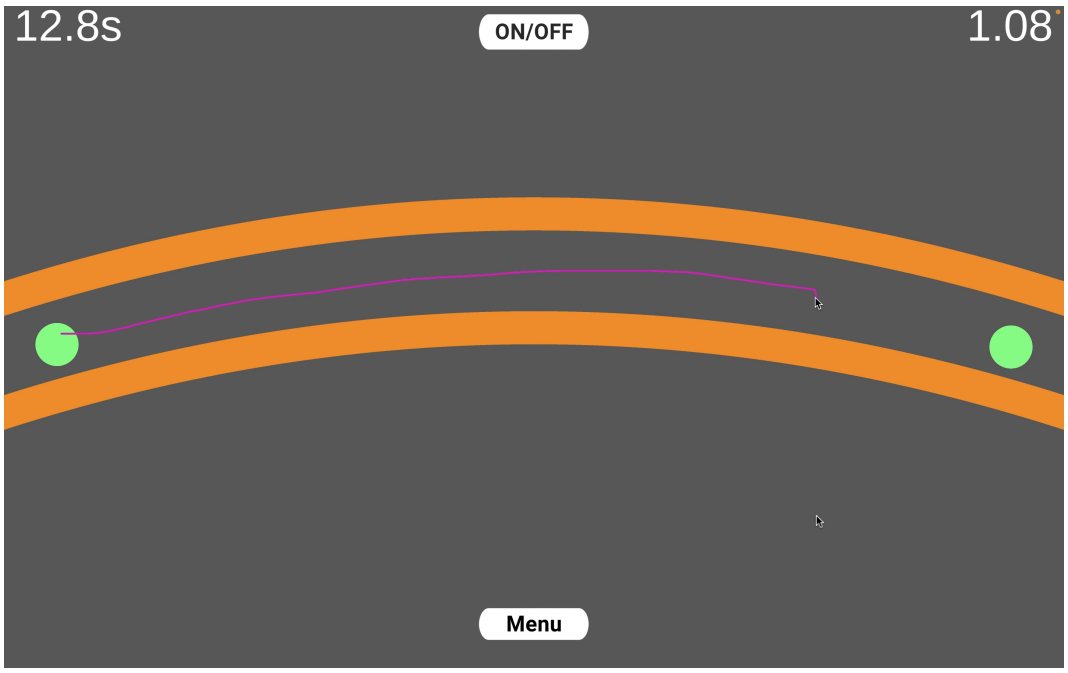

Рис. 4.5: Положение границ при длительном попадании в выделенную область

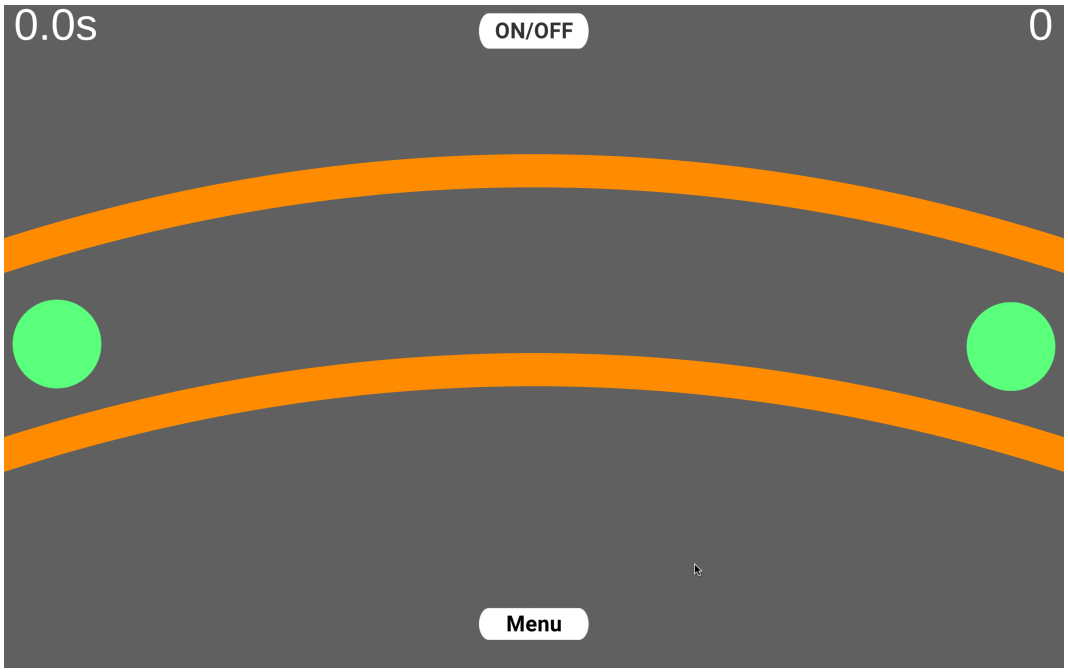

Рис. 4.6: Интерфейс страницы с тренингом

4.1.6 Реализация отображения трека касаний

Во время выполнения тренинга ребенок должен видеть линии, которые он проводит (см. Рисунок 4.7).

Одним из условий успешного прохождения теста является непрерывность линий. Поэтому, если пользователь оторвет руку, то есть приложение не зафиксирует касание во время текущего кадра, нарисованный трек сотрется, при этом таймер и счётчик остановятся (см.

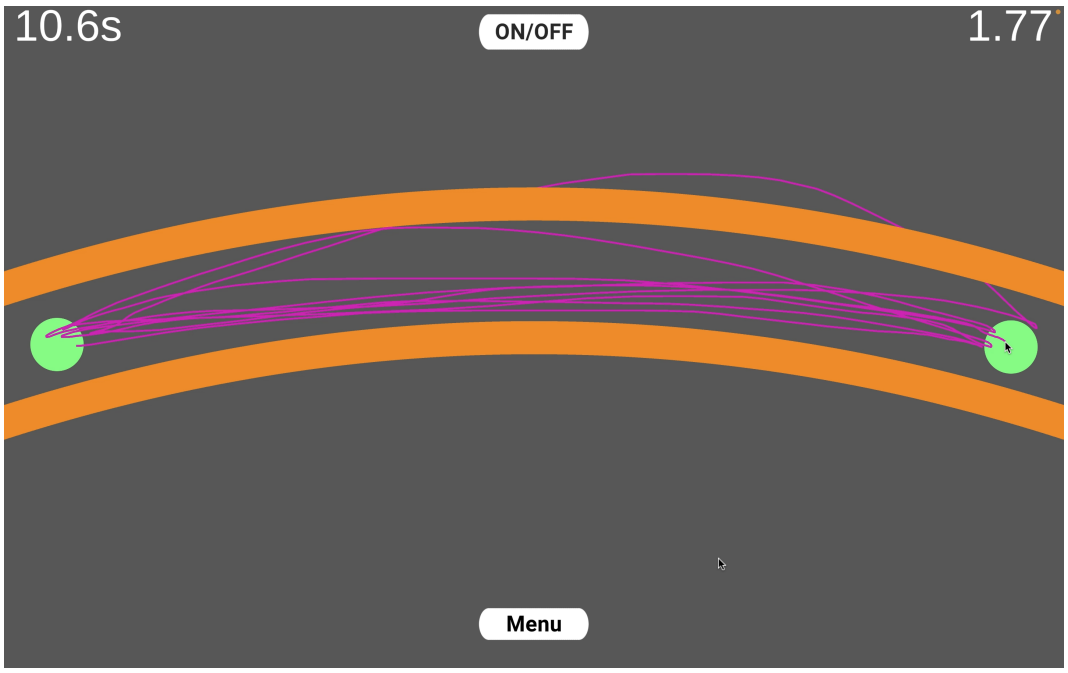

Рис. 4.7: Интерфейс при выполнении тренинга

Рисунок 4.8). Так как при следующем касании экрана (в том числе и случайном) таймер и счетчик обновятся, я сделал историю касаний и тренингов, которую опишу ниже.

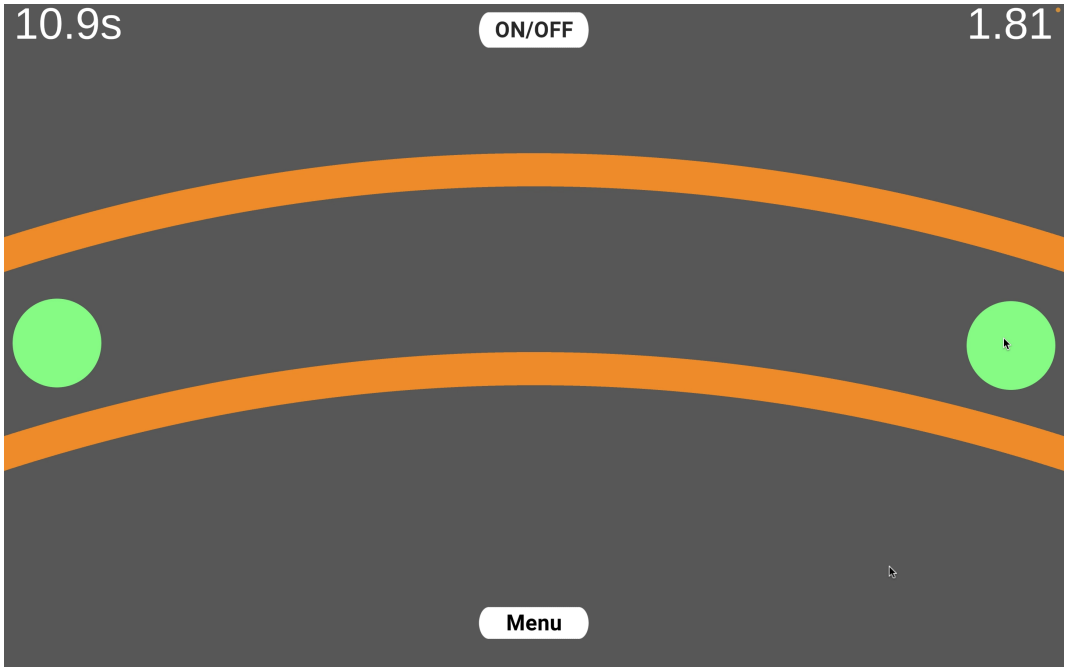

Рис. 4.8: Интерфейс после выполнения тренинга (таймер и счетчик не равны 0)

4.2 Реализация меню паузы

Чтобы упростить интерфейс страницы с тренингом, я добавил кнопку «Menu», по нажатию на которую пропадут границы тренинга, терминальные точки и кнопка включения

метронома, метроном, таймер и счетчик остановятся, фон станет темнее, и откроется меню, которое можно увидеть на Рисунке 4.9.

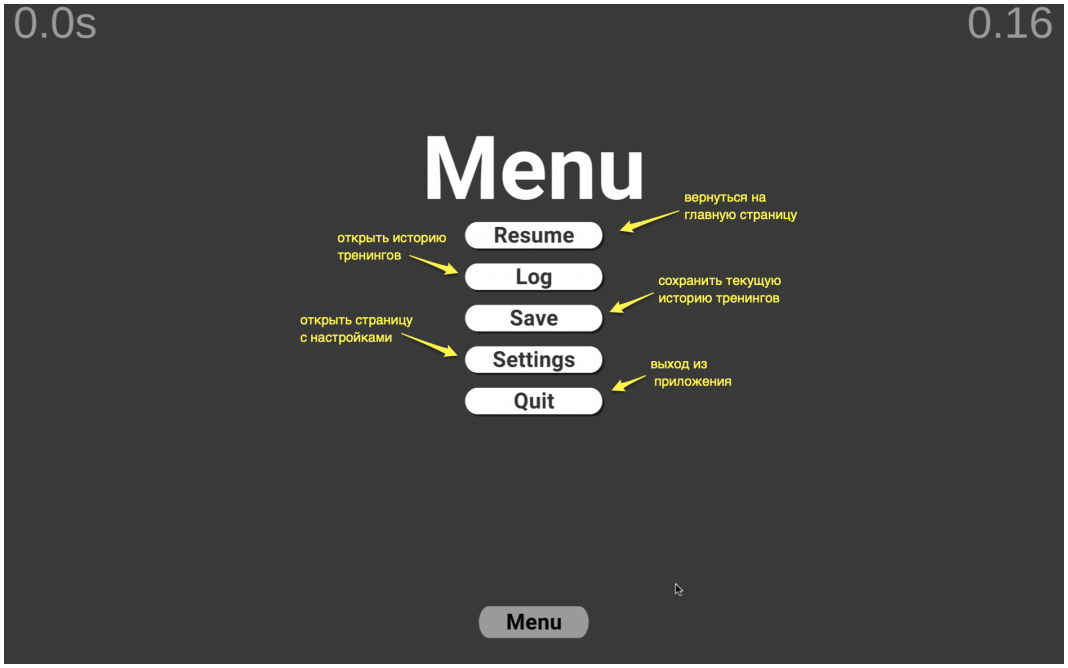

Рис. 4.9: Меню паузы

Из этого меню можно открыть страницы с историей тренингов и настройками, сохранить текущую историю тренингов, выйти из приложения или вернуться на главную страницу.

4.3 Реализация страницы с историей тренингов

4.3.1 Сохранение истории

Следующей задачей было реализование сохранения истории тренингов. Для этого я завел 3 массива, которые хранили следующую информацию: дата и время начала теста, продолжительность теста, сумма отклонений за тест. Я добавил следующий функционал. Когда пользователь впервые касается экрана, приложение сохраняет текущее время и дату, которые можно получить с помощью встроенных в Unity функций, запускает таймер и обнуляет счётчик. В момент, когда пользователь перестает касаться экрана, приложение добавляет 3 полученные величины в соответствующие массивы.

При этом при нажатии на кнопку на главном экране возникает следующая проблема: приложение воспринимает нажатие на кнопку как нажатие на экран и сохраняет новую запись в историю с нулевой длительностью и нулевым счётчиком. Чтобы не сохранять пустые записи, я реализовал проверку на нажатие кнопки и, если какая-либо кнопка на главном экране была нажата, не сохранял последнюю запись в историю.

Для удобства пользователя я сделал сохранение истории при выходе из приложения. Для этого нужно нажать на кнопку «Save» в меня паузы. При этом всю историю можно очистить с помощью соответствующей кнопки на странице с настройками, которую я опишу далее в соответствующем разделе. Для сохранения данных я пользовался функционалом PlayerPrefs, используя официальную документацию Unity [**6**]. PlayerPrefs позволяет сохранить данные в память программы и проверить их наличие во время запуска приложения. Данный функционал не подходит для хранения секретных данных, так как доступ к нему имеет любой пользователь приложения, но его можно использовать в случае с хранением информации о тренингах.

4.3.2 Отображение истории

Страницу с историей я реализовал следующим образом: пользователь видит 2 колонки с не более чем 5 записями в каждой. Если записей больше 10, то есть они не помещаются на одной странице, в правом верхнем углу появляются кнопки для перемещения между страницами. В левом верхнем углу пользователь может видеть номер текущей страницы, а сверху расположена кнопка для возврата на главную страницу с тренингом. Пример можно увидеть на Рисунке 4.10. Функционал Unity позволяет реализовать расположение записей так, чтобы на разных устройствах расстояние между объектами было пропорционально размеру экрана.

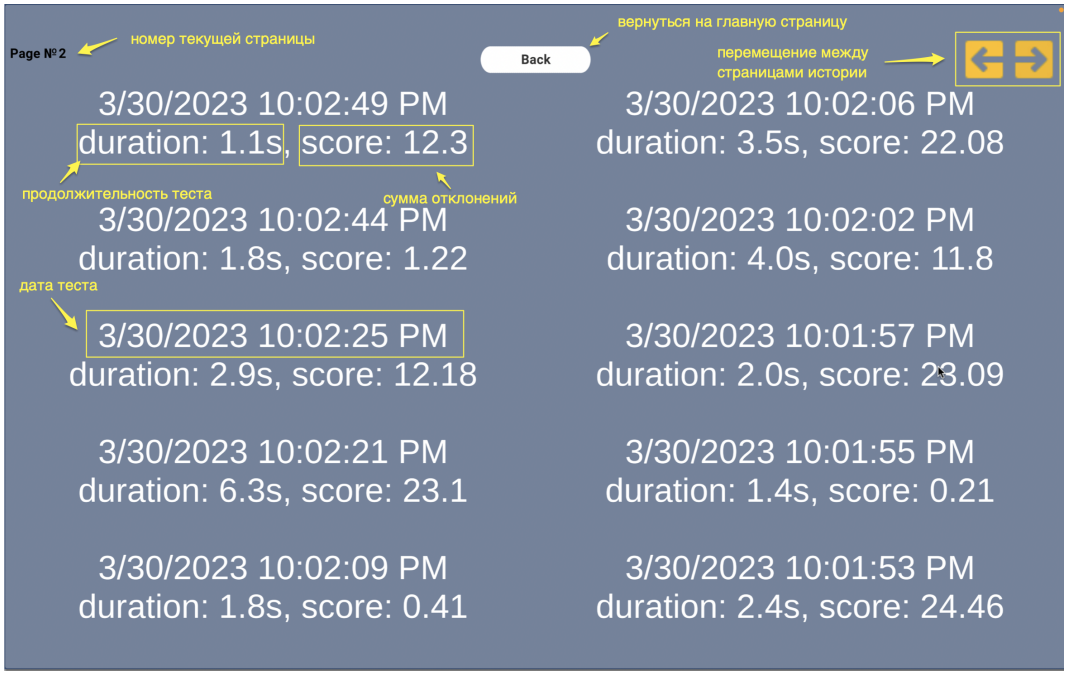

Рис. 4.10: Интерфейс страницы с историей тренингов

4.4 Реализация выгрузки результатов (положение в зависимости от времени)

Для дальнейшего анализа двигательного контроля нейробиологи поставили задачу реализовать файловую выгрузку результатов с координатами касания в зависимости от времени. На официальном сайте Unity [7] я узнал о возможности сделать создание файлов и запись в них с помощью встроенного в C# класса StreamWriter. Таким образом, я добавил создание файлов с координатой касания во время каждого кадра. В папке с приложением во время каждого открытия главной страницы будет создаваться текстовый файл (то есть, если открыть приложение, то создастся первый файл, если потом открыть настройки и затем вернуться на главную страницу, то создастся еще один файл и так далее. Если не закрывать главную страницу, то в один текущий файл будет записываться информация сразу о нескольких подходах. При закрытии приложения все файлы автоматически сохраняются. При этом во время первого открытия приложения оно может попросить разрешение на запись). Все файлы имеют название в формате «data_[год][месяц][день][час][минута]_[секунда создания].txt». С видом записей в файлах можно ознакомиться на Рисунках 4.11 и 4.12. Файлы содержат абсциссу и ординату касания, расстояние до идеальной траектории, продолжительность тренинга во время каждого кадра и время и дату начала/конца текущего тренинга в формате [год]_[месяц]_[день]_[час]_[минута]_[секунда].

```
-7.425926 // x
1.953704 // y
0.05267689 // distance to ellipse
0 // duration (milisec)
-7.425926 // x
1.953704 // y
0.05267689 // distance to ellipse
1 // duration (milisec)
```

Рис. 4.11: Содержание файла с координатами касания в зависимости от времени

4.5 Реализация метронома

Для регулирования скорости выполнения тренинга врачи используют метроном, поэтому мне поступил запрос добавить метроном в приложение. Так как частота ударов мет-

```
2023_5_8_23_4_52 # start time
-7.666667 # x
1.777778 # y
0.1395736 # distance to ellipse
0 # duration (milisec)
```

```
1.972222 # x
2.768518 # y
0.1801924 # distance to ellipse
117 # duration (milisec)
2023_5_8_23_5_4 # end time
```

Рис. 4.12: Начало (слева) и конец (справа) файла с координатами касания в зависимости от времени

ронома может достигать 200 ударов/мин и промежутки между ударами должны быть строго равны во время всего использования программы, я решил добавить удары метронома с помощью техники многопоточного программирования. Для этого в отдельном процессе исполнения я создал отдельный таймер, который высчитывает время работы приложения и в зависимости от заданного параметра, который можно изменять в настройках от 30 ударов/-мин до 200 ударов/мин, отправляет звуковой сигнал на динамик устройства.

Для удобства пользователя звук каждого четвертого удара я сделал отличным от остальных звуков. Аудио я взял из бесплатного ассета из магазина Unity [8]. Так же, чтобы пользователь не выключал звук устройства, на главную страницу с тренингом я добавил кнопку включения/выключения метронома.

4.6 Реализация страницы с настройками

Для смены параметров приложения я сделал отдельную страницу с настройками, интерефейс которой можно увидеть на Рисунке 4.13.

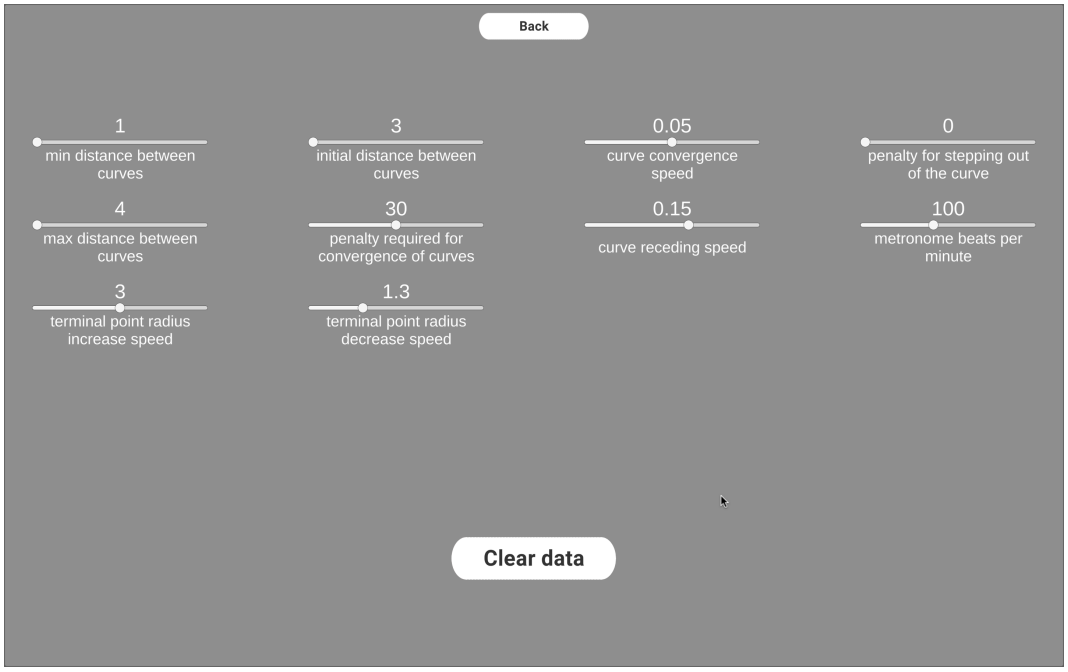

Рис. 4.13: Интерефейс страницы с настройками приложения

На данной странице пользователь может поменять следующие параметры:

– минимальное и максимальное расстояния между границами;

– начальное расстояние между границами;

– необходимый штраф для начала расширения границ;

– скорости сближения/расширения границ;

– дополнительный штраф за выход за пределы границы;

– количество ударов в минуту метронома;

– скорости расширения/сужения терминальных точек.

Так же на странице с настройками пользователь может удалить всю историю с тренингами из памяти приложения. Чтобы предотвратить случайное удаление данных, я добавил подтверждающее окно (см. Рисунок 4.14).

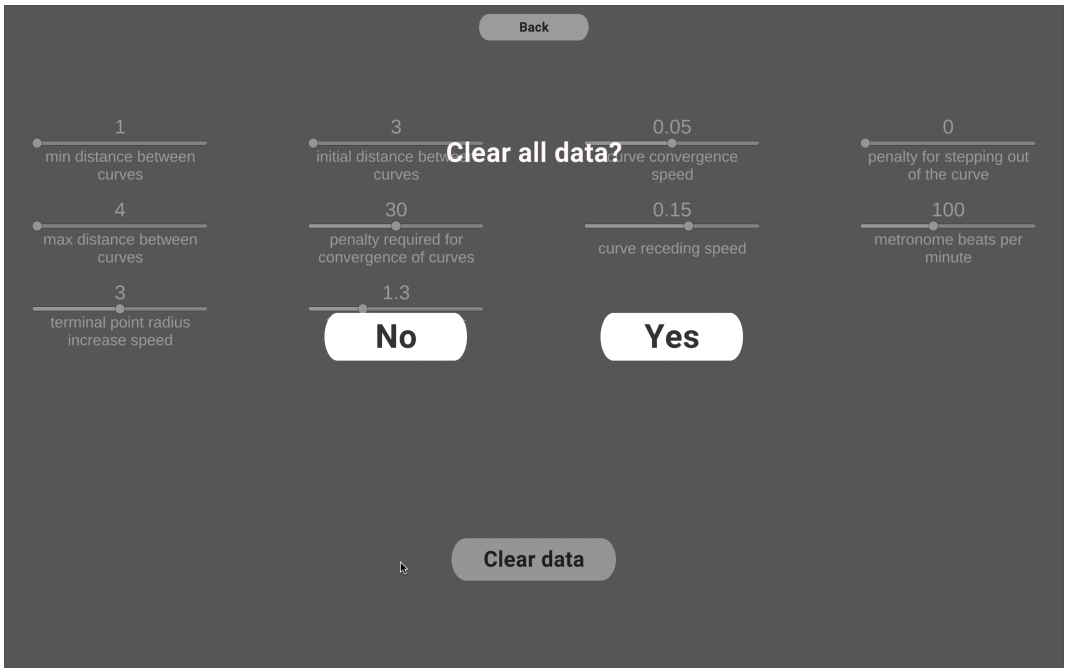

Рис. 4.14: Интерефейс окна подтверждения удаления данных

Ко всем элементам страницы я добавил атрибуты, позволяющие подстраивать расположение элементов в зависимости от размера экрана устройства.

5 Установка приложения

Для установки приложения я выложил его на онлайн хранилище [1], с помощью которого нейробиологи смогли скачать и установить приложение на свои планшеты.

[1] Ссылка на приложение: https://disk.yandex.ru/d/GDAqEOqyGQdW8w, дата обр. 17.05.2023

Список литературы

[1] Ивкина Юлия Михайловна, Жайнагуль Саргазыевна Искужиева. "Основы каллиграфического письма: Теория и методика." (2019): с. 25-47.

[2] Impedovo, Donato, and Giuseppe Pirlo. "Dynamic handwriting analysis for the assessment of neurodegenerative diseases: a pattern recognition perspective."IEEE reviews in biomedical engineering 12" (2018): p. 209-220.

[3] Danna, J., Paz-Villagrán, V. and Velay, J.L. Signal-to-Noise velocity peaks difference: A new method for evaluating the handwriting movement fluency in children with dysgraphia. Research in developmental disabilities (2013), 34(12): p. 4375-4384.

[4] Farin, Gerald. "Algorithms for rational Bézier curves." Computer-aided design 15.2 (1983): p. 73-77.

[5] Nürnberg, Robert. "Distance from a point to an ellipse." (2006) URl: https://www.ma.imperial.ac.uk/~rn/distance2ellipse.pdf (дата обр. 17.05.2023).

[6] PlayerPrefs class in UnityEngine. URl: https://docs.unity3d.com/ScriptReference/PlayerPrefs.html (дата обр. 17.05.2023).

[7] StreamWriter class in UnityEngine. URL: https://support.unity.com/hc/en-us/articles/115000341143-How-do-I-read-and-write-data-from-a-text-file- (дата обр. 17.05.2023).

[8] Unity asset store. URL: https://assetstore.unity.com/packages/audio/sound-fx/free-casual-game-sfx-pack-54116 (дата обр. 17.05.2023).