СОКРАЩЕНИЯ

ЦНС – центральная нервная система

ЗАО – закрытое акционерное общество

АПК – аппаратно-программный комплекс

ПО – программное обеспечение

ТС – техническое средство

ЦД – центр давления

МПС – микропроцессорная система

ПК – персональный компьютер

МПГУ – Московский педагогический государственный университет

COM – Component object model

ФИО – фамилия, имя, отчество

ВВЕДЕНИЕ

Современная медицина во многом опирается на точную и быструю диагностику. Поскольку легче предотвратить заболевание, чем лечить уже прогрессирующую патологию, возникает потребность в средствах и методах, позволяющих оценить состояние человека достаточно детально, но быстро и желательно без использования дорогостоящих инструментов.

С другой стороны, лабораторная диагностика – совокупность различных методов исследования состава тканей и жидкостей человеческого организма – хоть применяется повсеместно и имеет важнейшее значение в исследовании состояния здоровья человека, но в ряде случаев не дает полной картины и является довольно дорогой процедурой в силу необходимости использовать множество химических реактивов и лабораторного оборудования. В таких областях, как ортопедия, неврология и некоторых других предпочтение следует отдать функциональной диагностике – оценке качества выполнения системами организма своих функций.

Одним из активно развивающихся направлений функциональной диагностики сегодня является стабилометрия или стабилография – неинвазивный метод исследования функций поддержания равновесия тела на основе анализа изменения координат проекции общего центра масс тела на плоскость и колебаний этого центра масс. На основании данных, полученных стабилометрическими методами, специалист может оценить работу нервной системы пациента, дать рекомендации относительно дальнейшего медицинского обследования и даже охарактеризовать психическое состояние человека.

Наиболее распространенным стабилометрическим инструментом является стабилоплатформа – устройство, фиксирующее изменения координат центров масс во времени. Однако их использование сопряжено с рядом ограничений, как то возможность оценки лишь общего функционального состояния ЦНС человека ввиду взаимодействия стабилоплатформы с относительно небольшими участками нижних конечностей человеческого тела.

Как развитие данной технологии, кафедрой анатомии и физиологии человека Московского педагогического государственного университета совместно с ЗАО "ОКБ "РИТМ" был разработан аппаратно-программный комплекс «Многофункциональное кресло», позволяющий детектировать мышечный тремор во всех крупных мышечных группах тела человека в отдельности. В настоящее время, данный АПК эксплуатируется в связке с произведенными тем же закрытым акционерным обществом программным обеспечением «StabMed».

Работа в связке «Многофункционального кресла» и упомянутых устройств и ПО имеет ряд недостатков: «StabMed» разработан для использования прежде всего с классическими стабилоплатформами, что является причиной невозможности использования технических возможностей АПК в полной мере, а анализ получаемых при помощи «кресла» данных, исходя из специфики их источника, приходится осуществлять сторонними средствами, для этого не предназначенными.

Целью данной работы является выявление потребностей специалистов-физиологов в области интерпретации стабилографических измерений, определение требований к инструменту, который бы позволил осуществлять эту интерпретацию и разработка такого инструмента в виде автоматизированной системы и графическим интерфейсом, ориентированным на работу с указанным выше АПК.

В соответствии с целями работа над автоматизированной системой была разделена на несколько этапов, были решены следующие задачи:

- анализ библиографических источников по теме стабилометрии для теоретической проработки методов обработки и визуализации данных;

- сбор и обработка данных стабилографических измерений при помощи АПК «Многофункциональное кресло»;

- сбор и анализ требований, предъявляемых специалистами, использующими АПК «Многофункциональное кресло» к системе обработки стабилограмм;

- определение методов получения, хранения, обработки и визуализации данных, получаемых с АПК «многофункциональное кресло»;

- разработка технического задания на создание описанной автоматизированной системы и тестов для проверки реализации требований к ней;

- разработка программной реализации автоматизированной системы с графическим интерфейсом;

- тестирование разработанной системы с использованием АПК «Многофункциональное кресло».

В настоящий момент указанный аппаратно-программный комплекс активно используется в исследованиях, проводимых в Московском педагогическом государственном университете, а также реализуется на розничном рынке ЗАО «ОКБ «РИТМ», что обусловило актуальность работы. Здесь же стоит отметить, что не так давно стабилометрия как вид медицинской услуги была включена в российский стандарт оказания медицинской помощи [1]. В то же время новизна данной работы обусловлена отсутствием, как уже указывалось ранее, специализированных инструментов обработки стабилографических измерений для АПК «Многофункциональное кресло».

ГЛАВА 1

Биологическая природа стабилометрических показателей

Для понимания специфики данных, с которыми будет работать разрабатываемая автоматизированная система, следует рассмотреть биологическую природу явлений, связанных с поддержанием равновесия.

Явления, связанные с удержанием человеком равновесия (постуральные явления), связаны с так называемыми тоническими рефлексами [3]. Данные рефлексы осуществляются как ответная реакция на рецепторные сигналы, возбуждающиеся в мышцах, глазах и некоторых других системах человеческого организма. Эти рефлексы принято разделять на несколько групп: выпрямительные, рефлексы позы и статокинетические. Выпрямительные рефлексы срабатывают при отклонении тела от положения «стоя» и служат защитой от внезапных падений, рефлексы позы – при угрозе потери равновесия во время изменения положения тела, главным образом при изменении положения головы, статокинетические рефлексы – во время изменения положения тела в пространстве даже в том случае, когда человек не совершает никаких движений.

Однако в контексте работы с аппаратно-программным комплексом «Многофункциональное кресло» больший интерес представляют явления, происходящие именно в случае нахождения человека в покое в сидячем положении. Основным наблюдаемым в этом случае постуральным явлением является мышечный тонус. Данное явление заключается в постоянном пребывании всех мышц тела в напряжении, даже в состоянии, которое человек охарактеризовал бы как «полное расслабление». Этот рефлекс на растяжение мышц является начальным состоянием для совершения какого-либо движения, как бы подготавливая мышцы к грядущему сокращению, и позволяет телу сохранять любую позу. В явлении мышечного тонуса участвует множество биологических систем: спинной мозг, ствол, мозжечок, мышечные рецепторы [3].

К проявлению мышечного тонуса как к результату автоколебательных процессов в цепи управления мышечной активностью можно, по-видимому, отнести тремор – непроизвольные ритмичные колебательные движения тела или его отдельных частей [5]. Выделяют физиологический тремор и патологический тремор. Первый представляет собой низкоамплитудное высокочастотное (8-12 Гц) дрожание, наблюдаемое в любой части тела человека и им не ощущаемое. Данный вид тремора может усиливаться при определенных состояниях и воздействиях: волнение, алкогольное опьянение, интоксикация, прием психостимуляторов [4]. Также данный вид тремора во многом носит индивидуальный характер и зависит от состояния нервной системы конкретного человека [5]. Паталогический тремор характеризуется более высокой амплитудой при частоте ниже, чем частота физиологического тремора, и подразделяется на дрожание покоя, постуральный и кинетический тремор. Постуральный тремор проявляется в процессе удержания определенной позы (например, удержание вытянутых перед собой рук); кинетический характеризуется усилением дрожания при движении; тремор покоя, исходя из названия, наблюдается при нахождении человека в расслабленном состоянии. С практической точки зрения имеет значение, что тот или иной вид паталогического тремора характерен для определенной группы нервных расстройств [4] – данная зависимость и обуславливает ценность стабилометрии как способа медицинской диагностики.

Технические аспекты стабилометрии

Стабилометрия (стабиллометрия, стабилография, постурография) – это методика оценки характеристик контроля человеком позы, основанная на измерении координат центра давления на чувствительной поверхности. В качестве такой чувствительной поверхности обычно выступает стабилоплатформа – устройство, оснащенное множеством датчиков, сигналы от которых интерпретируются компьютером для построения траектории перемещения центра давления, формируемого участком тела человека. Под центром давления в данном случае понимается точка, к которой приложена равнодействующая сил, порожденных взаимодействием человека с опорой. Здесь стоит отметить принятые Московским консенсусом по применению стабилометрии [2] термины для обозначения системы координат, в рамках которой происходят измерения:

- координатами центра давления называют числовые характеристики положения центра давления в прямоугольной координатной системе на плоскости стабилоплатформы;

- ось абсцисс Ox в стабилографии принято называть «фронтальная ось» или «фронталь»;

- ось ординат Oy в стабилографии принято называть «саггитальная ось» или «саггиталь».

Термином «стабилограмма» обозначают график зависимости той или иной координаты центра давления в плоскости стабилоплатформы от времени (рис. 1). Следует различать «стабилограмму» и «статокинезеграмму», несмотря на схожий физический смысл. Статокинезеграмма представляет собой графическое отображение траектории движения ЦД в рамках системы координат стабилоплатформы.

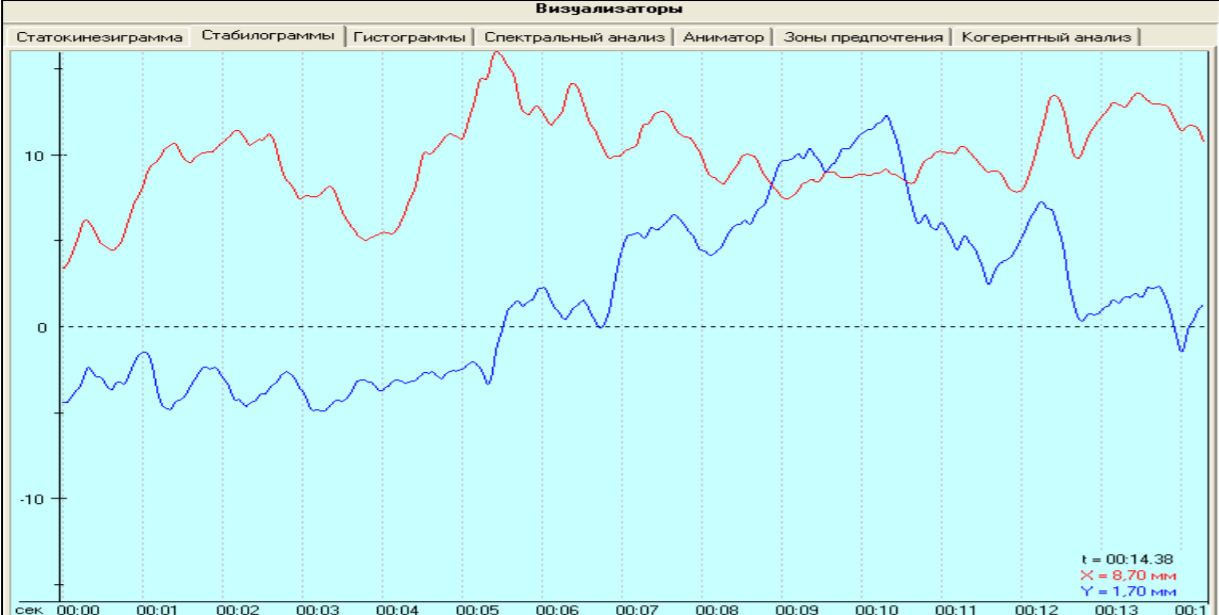


Рис. 1. Стабилограмма.

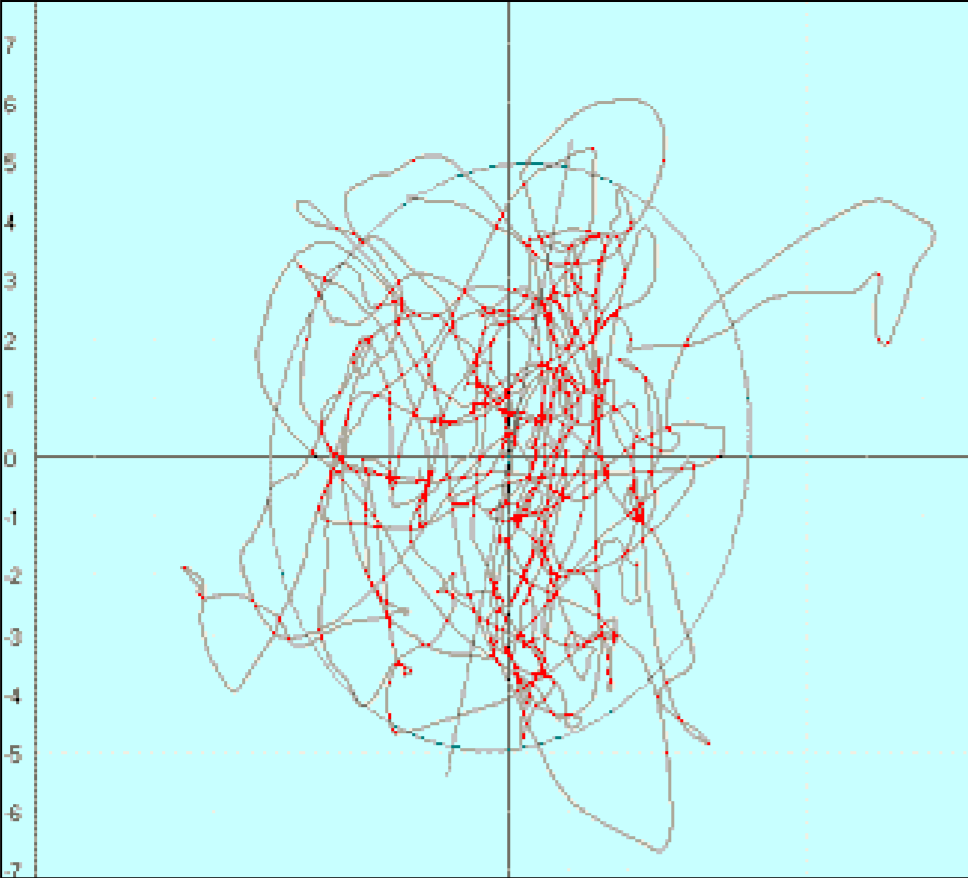


Рис. 2. Статокинезеграмма.

Стабилометрические показатели принято классифицировать на прямые и расчетные. К первым относят [2] координаты центра давления по фронтали и саггитали, а также массу. Расчетными показателями называют показатели, вычисленные на основе прямых. Прежде всего это:

- L – длина статокинезеграммы в миллиметрах;

- V – скорость перемещения ЦД в миллиметрах в секунду;

- Fx – частота колебаний ЦД по фронтали в герцах;

- Fy – частота колебаний ЦД по саггитали в герцах;

- S – площадь статокинезеграммы в миллиметрах квадратных;

- A – механическая работа, совершенная в результате перемещения, в джоулях.

Основным инструментом для осуществления стабилометрических измерений является стабилограф (стабилометрическая платформа или стабилоплатформа) – устройство, обычно представляющее собой «очувствленную» при помощи датчиков поверхность, позволяющую фиксировать координаты ЦД в отдельный момент времени. Типичная стабилоплатформа представлена на рисунке 3 – данная модель, к примеру, используется на кафедре анатомии и физиологии человека и животных МПГУ.

Принцип работы стабилоплатформы заключается в следующем: пациент каким-либо образом соприкасается с рабочей («очувствленной») поверхностью стабилографа (большинство методик подразумевает стояние человека на платформе обеими ногами); датчики, в большом количестве вмонтированные в платформу, реагируют на вес человека, посылая на встроенную микропроцессорную систему соответствующие сигналы. МПС, обработав поступившие сигналы, преобразует их, отправляя на дальнейшую обработку в персональный компьютер. На ПК специализированная программа интерпретирует полученный сигнал, вычисляя центр давления и траекторию его перемещений, отрисовывает ее и рассчитывает другие необходимые параметры.

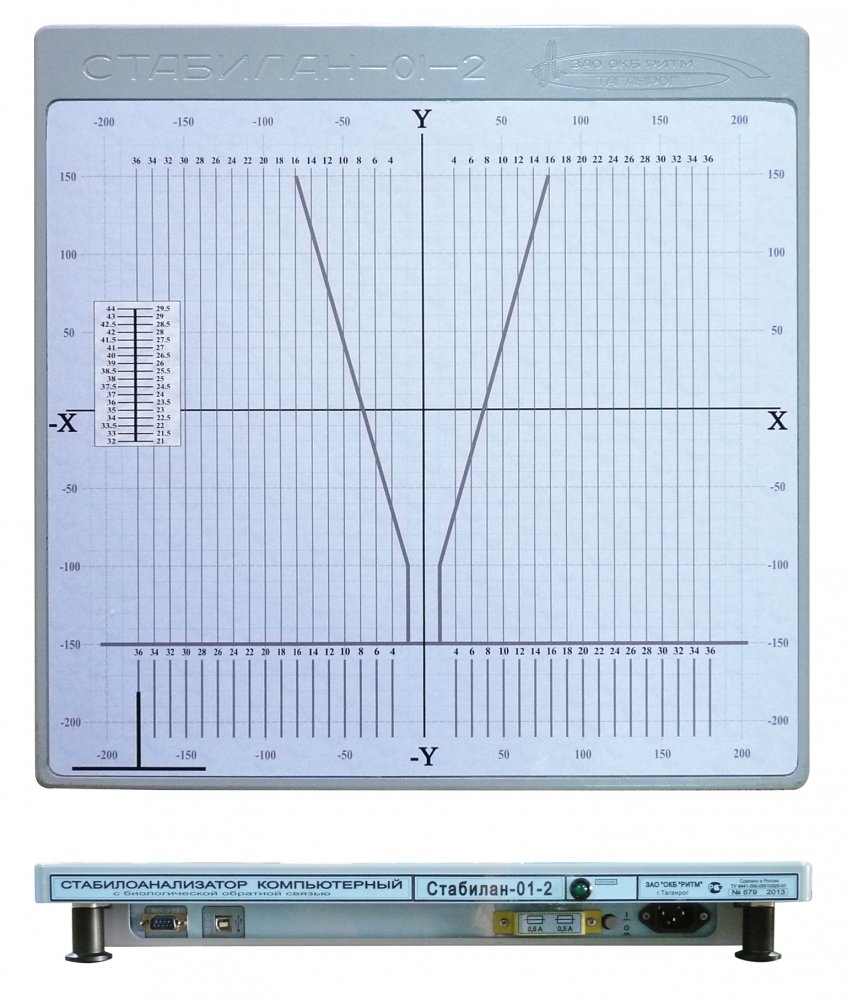


Рис. 3. Стабилоплатформа «Стабилан-01-2»

Обзор ПО «StabMed» и его возможностей

Управление и съем данных со стабилометрических приборов, в частности с аппаратно-программного комплекса «Многофункциональное кресло», осуществляется при помощи специализированного ПО. На кафедре анатомии и физиологии человека и животных Московского педагогического государственного университета в рамках проводимых исследований используется программное обеспечение «StabMed», разработанное ЗАО «ОКБ «РИТМ». Данная программа представляет собой систему для управления стабилометрическим оборудованием (главным образом, произведенного данным предприятием), снятия и визуализации стабилографических данных, а именно построения стабилограмм и статокинезеграмм, создания и управления базой данных пациентов и выполнения ряда других функций (рис. 4).

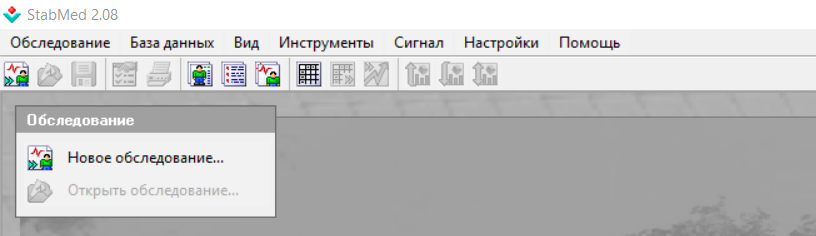


Рис. 4. Меню в главном окне программы «StabMed».

Следует отметить, что данное программное обеспечение поддерживает технологический стандарт Microsoft Component Object Model. Программа StabMed в рамках интерфейса COM выступает в роли сервера COM, то есть приложения, поставляющего данные, а проектируемая автоматизированная система обработки стабилометрических измерений – в роли COM-клиента, то есть приложения, использующего эти данные.

При запуске на ПК StabMed автоматически регистрируется в качестве COM-сервера, и последующие обращения к нему идут через этот зарегистрированный COM-сервер.

Если при обращении приложения-пользователя (автоматизированной системы) к серверу COM StabMed не загружен, то он автоматически загружается, а после использования автоматически выгружается. Если же при обращении приложения пользователя к серверу COM StabMed загружен, то статус StabMed не меняется.

С помощью предусмотренного в ПО COM-интерфейса сторонняя программа имеет возможность получать данные из базы пациентов (ФИО, дата рождения, биометрические параметры), техническую информацию о пробах и измерениях, непосредственно сигналы, фиксируемые стабилометрическими приборами. В полной мере данный интерфейс описан в Приложении 1.

Аппаратно-программный комплекс «Многофункциональное кресло»

Принцип работы созданного в МПГУ совместно с ЗАО «ОКБ «РИТМ» аппаратно-программного комплекса «Многофункциональное кресло» аналогичен принципу работы типичного стабилографа, описанного выше. Главным отличием от традиционных стабилоплатформ является возможность производить измерения для нескольких мышечных групп по отдельности. Такую возможность обеспечивает специфическая конструкция «кресла» (рис. 4). АПК состоит из одной трехкомпонентной силомоментной платформы D0(3k), которая служит опорой всего «кресла», и семи шестикомпонентных силомоментных платформ D1-D4(6k): 2 подлокотника, 2 сиденья, спинка и 2 опоры под ноги. Датчики в каждой платформе (кроме платформы D0) имеют по 6 осей чувствительности (рис. 5), что позволяет оценивать мышечный тремор соответственно по 6 сигналам: трем сигналам, отражающим изменение координат вдоль осей X, Y и Z во времени, и трем сигналам моментов сил, порождаемых колебаниями мышц.

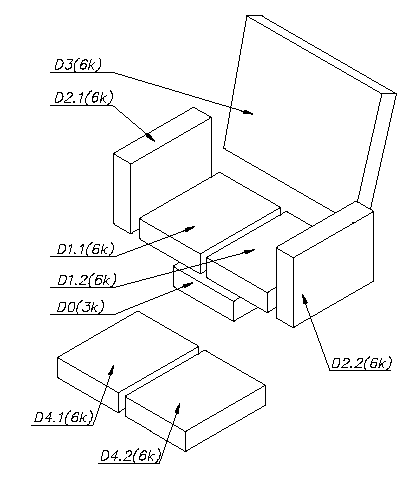


Рис. 5. Структурная схема «Многофункционального кресла».

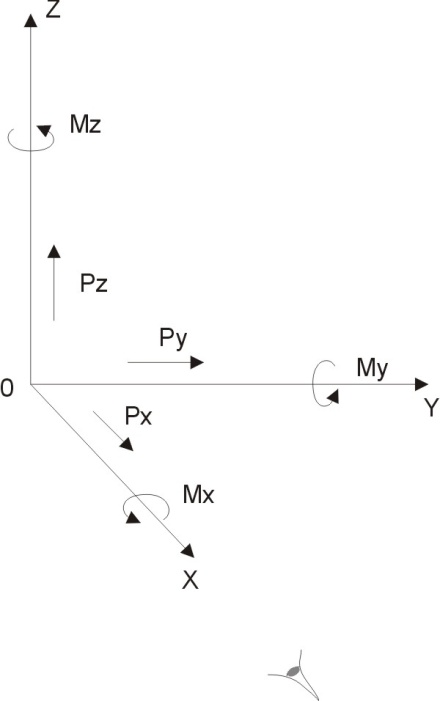


Рис. 6. Расположение осей чувствительности многокомпонентного датчика.

Таким образом, данный аппаратно-программный комплекс служит инструментом для снятия показаний, характеризующих физиологический тремор и тремор покоя. Это позволяет использовать «Многофункциональное кресло» для оценки психоэмоционального состояния пациента, а также ранней диагностики заболеваний нервной системы, связанных с наличием выраженного тремора покоя – например, болезнью Паркинсона [5].

Однако это связано с рядом проблем, главная из которых – интегративный характер фиксируемых устройством колебаний. Помимо мышечного тремора, зависимость которого от времени сама по себе носит сложный характер, вклад в фиксируемый устройством колебательный сигнал вносят также дыхание и сердцебиение [5]. То есть на выходе из АПК мы имеем сигнал, являющийся композицией квазипериодических и непериодических колебаний различных частот. Следовательно, основной задачей при обработке получаемых с «Многофункционального кресла» данных является анализ сигналов.

Анализ требований к автоматизированной системе

В процессе проектирования автоматизированной системы были изучены методики использования АПК «Многофункциональное кресло», было проведено наблюдение за работой специалистов-физиологов с данным аппаратно-программным комплексом, а также принято непосредственно участие в стабилографическом исследовании (рис. 7)



Рис. 7. Проведение стабилографического исследования

В числе прочего, было проведено интервьюирование специалистов, ведущих работу с АПК «Многофункциональное кресло». Они отметили, что автоматизированная система обработки результатов стабилографических измерений, которая упростила бы им работу и сделала ее более удобной и эффективной, должна обладать следующими свойствами:

- иметь вид «окна»

- выводить на экран данные сразу со всех чувствительных поверхностей

- проводить анализ сигнала, который бы позволил оценить частотный и амплитудный спектр

- давать оценку «нормальности» данных

- сравнивать различные пробы по одному и тому же сигналу

- осуществлять вышеприведенные действия без ручной обработки выходных данных и без ручного ввода данных в обработчик.

Таким образом, для проектируемой автоматизированной системы можно выделить следующие требования:

- наличие графического пользовательского интерфейса

- загрузка результатов стабилометричсеких измерений непосредственно из ПО «StabMed»

- отображение стабилограмм и статокинезеграмм в рамках одного программного окна

- отображение стабилограмм и статокинезеграмм для всех чувствительных поверхностей одновременно

- возможность интерактивного выбора канала, который в данный момент отображается на всех стабилограммах

- декомпозиция сигналов, поступающих от соответствующих компонент силомоментных датчиков

- двоичная классификация сигналов по признаку возможного наличия патологии у пациента.

Выводы и результаты

В главе 1 был проведен обзор стабилометрии как предметной области, рассмотрен биологический аспект стабилометрических измерений. Также проведен анализ существующих подходов инструментов, которые, в частности, используются на кафедре анатомии и физиологии человека и животных Московского педагогического государственного университета. В процессе этого анализа были выявлены пользовательские требования к проектируемой автоматизированной системе. Вкупе со спецификой предметной области были сформированы требования к автоматизированной системе обработки результатов стабилографических измерений.

ГЛАВА 2

Структура автоматизированной системы

Для понимания дальнейших задач проектирования опишем структуру системы стабилографических измерений при помощи концептуальной схемы **(рис. 8).** На уровне концепции система включает три составляющих: аппаратно-программный комплекс «Многофункциональное кресло», программное обеспечение «StabMed» и проектируемую автоматизированную систему обработки результатов стабилографических измерений, которая здесь обозначена как «Автоматизированная система».

Исходя из требований к автоматизированной системе, сформированных в главе 1, ее следует проектировать как совокупность следующих подсистем:

- COM-интерфейс;

- программный модуль анализа сигнала;

- база данных;

- программный модуль классификации сигнала;

- программный модуль графического представления сигнала;

- программный модуль пользовательского интерфейса.

Структура автоматизированной системы представлена на **рисунке 9.**

Для программной реализации автоматизированной системы был выбран язык программирования Python версии 3.7, так как на базе данной технологии возможно реализовать все указанные требования.

COM-интерфейс

COM-архитектура реализована на языке Python в пакете «win32com». Данный пакет предоставляет возможность создания как приложений-серверов, так и приложений-клиентов. Проектируемая система является получателем данных с АПК, то есть COM-клиентом. Необходимые для создания COM-клиента функции реализованы в пакете «win32com.client» [6].

Проектирование осуществляется на базе объектно-ориентированного подхода. В рамках реализации COM-интерфейса было решено создать только один класс, в котором будут реализованы все необходимые геттеры, в соответствии со спецификациями, приведенными в Приложении 1. Реализация данной части автоматизированной системы в виде листинга «COM.py» представлена в Приложении 2.

Модуль анализа сигнала

В главе 1 была рассмотрена природа сигналов, получаемых с аппаратно-программного комплекса. Данные сигналы представляют собой композицию квазипериодических и непериодических колебаний различных частот. Получаемый сигнал является дискретной функцией от времени. Принимая во внимание результаты исследований, проводившихся при помощи АПК «Многофункциональное кресло» [5], задача обработки сигналов сводится к задаче спектрального анализа этих сигналов.

На основе требований, предъявляемых пользователями к системе, и опыте анализа стабилографических измерений [5], в качестве основных методик спектрального анализа сигналов, поступающих с АПК, были выбраны преобразование Фурье и Вейвлет-преобразование.

Реализация данных математических преобразований была выполнена на базе пакета

СПИСОК ИСТОЧНИКОВ

1. Приказ Минздрава России от 28.12.2012 N 1583н "Об утверждении стандарта специализированной медицинском помощи при болезни Паркинсона, требующей стационарного лечения в связи с нестабильной реакцией на противопаркинсонические средства" (Зарегистрировано в Минюсте России 11.02.2013 N 26971)
2. Московский консенсус по применению стабилометрии и биоуправления по опорной реакции в практическом здравоохранении и исследованиях / НИИ нормальной физиологии имени П.К. Анохина. – М., 2017 – 10 с.
3. Миловзорова М.С. Анатомия и физиология человека / Миловзорова М.С. – М.: Книга по Требованию, 2012. – 215 с.
4. Штульман Д. Р., Левин О. С. Нервные болезни: Учебник. – М.: Медицина, 2000. – 464 с.: ил. – (Учеб. лит. Для учащихся мед. училищ и колледжей). – ISBN 5-225-04587-1
5. *Кручинин П. А., Лебедев А. В., Холмогорова Н. В.* ОСОБЕННОСТИ ЧАСТОТНОГО АНАЛИЗА СИГНАЛОВ СИЛОМОМЕНТНЫХ ДАТЧИКОВ В ЗАДАЧЕ ИССЛЕДОВАНИЯ ФИЗИОЛОГИЧЕСКОГО ТРЕМОРА // *Российский журнал биомеханики*. — 2013. — Т. 17, № 1. — С. 64–77. Силомоментные датчики используют для оценки функционального состояния, неврологических и ортопедических патологий человека. Сигнал, измеренный силомоментным датчиком, взаимодействующим с телом человека или его сегментами, является интегративным. Он включает механические составляющие, обусловленные системой управления движением, а также, дыханием, кардиоритмом и т.п., что находит свое отражение в его частотном спектре. В работе обсуждается задача выделения треморных составляющих, порожденных последовательным сокращением скелетных мышц.
6. Mark Hammond. Python Programming on Win32 / Mark Hammond, Andy Robinson. - O'Reilly & Associates, Inc., 2000. – P. 652
7. Диагностика ранних неврологических нарушений с помощью силомоментных аппаратно-программных комплексов / Н. В. Холмогорова, П. А. Кручинин, Ю. С. Левик и др. // Известия ЮФУ. Технические науки. — 2012. — Т. 134, № 9. — С. 256–261. В работе обсуждается технология неинвазивной диагностики ранних неврологических нарушений. Рассматриваются приемы диагностики, основанные на регистрации и последующем спектральном анализе колебательных движений тела человека и его частей, вызванных ритмическими биениями сердца, дыханием, сокращением отдельных двигательных единиц и т.д. Приводятся результаты апробации отдельных элементов технологии на примере анализа особенностей моторной регуляции пациентов с паркинсоническими нарушениями. Показано, что наиболее полную информацию для подобной диагностики предоставляет аппаратно-программный комплекс с распределенной системой силомоментных датчиков. http://www.rista.ru/production/stabila/docs/mis12/diagn\_nevrol.pdf.

ПРИЛОЖЕНИЕ 1

Доступ к данным StabMed

Интерфейс доступа к данным характеризуется глобальным уникальным идентификатором GUID и доступ к нему осуществляется через этот GUID. Интерфейс доступа к данным StabMed2 имеет следующий GUID:

{45809734-A07A-4718-9C54-D62188776B87}

Интерфейс доступа к данным содержит следующие функции:

1. Возвращает количество проведенных обследований

function GetTestCount(out Count: Integer): HResult; stdcall;

2. Возвращает TestID проведенного обследования по порядку

function GetTestID(Index: Integer; out TestID: Integer): HResult; stdcall;

Index - проядковый номер обследования в неотсортированной БД

Результат COM\_NOT\_FOUND, если обследования с таким номером нет

3. Возвращает TestID последнего проведенного обследования

Результат COM\_NOT\_FOUND, если обследования в БД нет

function GetLastTestID(out TestID: Integer): HResult; stdcall;

4. Возвращает TestID обследования, обрабатываемого в настоящий момент

Результат COM\_NOT\_FOUND, если обследования в обработке нет

function GetActiveTestID(out TestID: Integer): HResult; stdcall;

5. Возвращает количество открытых в настоящий момент обследований

function GetOpenedTestCount(out Count: Integer): HResult; stdcall;

6. Возвращает TestID проведенного обследования по порядку открытия

function GetOpenedTestID(Index: Integer; out TestID: Integer): HResult; stdcall;

Index - проядковый номер обследования среди открытых

Результат COM\_NOT\_FOUND, если обследования с таким номером нет

7. Возвращает количество проб в обследовании или COM\_NOT\_FOUND, если обследования TestID нет

function GetProbesCount(TestID: Integer; out Count: Integer): HResult; stdcall;

TestID - идентификатор обследования, который вернула одна из функций Get...TestID

8. Возвращает ProbeID пробы в обследовании по TestID и порядковому номеру пробы

function GetProbeID(TestID, Index: Integer; out ProbeID: Integer): HResult; stdcall;

TestID - идентификатор обследования, который вернула одна из функций Get...TestID

Index - проядковый номер пробы в обследовании

Результат COM\_NOT\_FOUND, если обследования с таким идентификатором или пробы с таким номером нет

9. Возвращает количество каналов, записанных в пробе ProbeID или COM\_NOT\_FOUND, если пробы ProbeID нет

function GetChannelCount(ProbeID: Integer; out Count: Integer): HResult; stdcall;

ProbeID - идентификатор пробы, который вернула функция GetProbeID

10. Возвращает ChannelID канала в пробе ProbeID и порядковому номеру канала

function GetChannelID(ProbeID, Index: Integer; out ChannelID: Integer): HResult; stdcall;

ProbeID - идентификатор пробы, который вернула функция GetProbeID

Index - проядковый номер канала в пробе

Результат COM\_NOT\_FOUND, если пробы с таким идентификатором или канала с таким номером нет

11. Возвращает тип канала по его идентификатору

function GetChannelType(ChannelID: Integer; out ChannelType: Integer): HResult; stdcall;

ChannelID - идентификатор канала, который вернула функция GetChannelID

12. Возвращает сигнал по идентификатору пробы и канала

function GetSignal(ProbeID, ChannelID: Integer; out Signal: PSafeArray): HResult; stdcall;

Сигнал возвращается в виде двухмерного защищенного массива PSafeArray,

первое измерение - подканалы, второе измерение - отсчеты по времени

13. Возвращает ФИО пациента по идентификатору обследования

function GetPatientName(TestID: Integer; out Name: PAnsiChar): HResult; stdcall;

TestID - идентификатор обследования, который вернула одна из функций Get...TestID

14. Возвращает дату рождения пациента по идентификатору обследования

function GetPatientBirthday(TestID: Integer; out Birthday: TDateTime): HResult; stdcall;

TestID - идентификатор обследования, который вернула одна из функций Get...TestID

15. Возвращает название методики по идентификатору обследования

function GetTestName(TestID: Integer; out Name: PAnsiChar): HResult; stdcall;

TestID - идентификатор обследования, который вернула одна из функций Get...TestID

16. Возвращает дату и время проведения обследования по идентификатору обследования

function GetTestDateTime(TestID: Integer; out DateTime: TDateTime): HResult; stdcall;

TestID - идентификатор обследования, который вернула одна из функций Get...TestID

17. Возвращает название пробы по идентификатору пробы

function GetProbeName(ProbeID: Integer; out Name: PAnsiChar): HResult; stdcall;

ProbeID - идентификатор пробы, который вернула функция GetProbeID

Работа с интерфейсом

Приложение пользователя работает с интерфейсом доступа к данным следующим образом:

Описывает интерфейс для доступа к данным StabMed2

type

ISignalExporter = interface(IUnknown)

['{5F5CF2EC-9F6E-453C-8EE5-C7003BE498A7}']

function GetTestCount(out Count: Integer): HResult; stdcall;

function GetTestID(Index: Integer; out TestID: Integer): HResult; stdcall;

function GetLastTestID(out TestID: Integer): HResult; stdcall;

function GetActiveTestID(out TestID: Integer): HResult; stdcall;

function GetOpenedTestCount(out Count: Integer): HResult; stdcall;

function GetOpenedTestID(Index: Integer; out TestID: Integer): HResult; stdcall;

function GetProbesCount(TestID: Integer; out Count: Integer): HResult; stdcall;

function GetProbeID(TestID: Integer; Index: Integer; out ProbeID: Integer): HResult; stdcall;

function GetChannelCount(ProbeID: Integer; out Count: Integer): HResult; stdcall;

function GetChannelID(ProbeID: Integer; Index: Integer; out ChannelID: Integer): HResult; stdcall;

function GetChannelType(ChannelID: Integer; out ChannelType: Integer): HResult; stdcall;

function GetSignal(ProbeID: Integer; ChannelID: Integer; out Signal: PSafeArray): HResult; stdcall;

function GetPatientName(TestID: Integer; out Name: PAnsiChar): HResult; stdcall;

function GetPatientBirthday(TestID: Integer; out Birthday: TDateTime): HResult; stdcall;

function GetTestName(TestID: Integer; out Name: PAnsiChar): HResult; stdcall;

function GetTestDateTime(TestID: Integer; out DateTime: TDateTime): HResult; stdcall;

function GetProbeName(ProbeID: Integer; out Name: PAnsiChar): HResult; stdcall;

end;

Указанный в первой строчке описания интерфейса GUID является идентификатором интерфейса и не служит для доступа к нему через технологию COM. Для доступа используется GUID, приведенный ниже.

Описывает GUID для доступа к интерфейсу

const

CLASS\_SignalExporter : TGUID = '{45809734-A07A-4718-9C54-D62188776B87}';

Получает экземпляр интерфейса

uses

ComObj, ActiveX;

var

StabMed2 : ISignalExporter;

begin

StabMed2 := CreateComObject(CLASS\_SignalExporter) as ISignalExporter;

Вызывает методы интерфейса для получения данных

var

Born : TDateTime;

TestID, ProbeID, ChanId : Integer;

NamePac : PAnsiChar;

begin

StabMed2.GetActiveTestID(TestID);

if TestID >= 0 then begin

StabMed2.GetPatientBirthday(TestID, Born);

StabMed2.GetPatientName(TestID, NamePac);

end;

Освобождает интерфейс

StabMed2 := nil;

Получение сигналов из StabMed2

Получить сигналы можно с помощью функции GetSignal, реализованной в интерфейсе. Эта функция возвращает сигнал в виде защищенного массива PSafeArray. Пример вызова приведен ниже.

const

// Константы каналов. Жестко прошиты в StabMed2. От версии не меняются.

chanSKG = 10;

// Эта функция получает на вход ссылку на COM сервер, идентификаторы пробы и канала

// На выходе

procedure TForm1.GetSignalArray ( Server : ISignalExporter;

ProbeID, ChanID : Integer;

out Sign : array of TDoubleDynArray);

type

TBounds = array [0..1] of

record

L, U : Integer;

end;

TIndexes = packed record

Row, Col : DWord;

end;

var

Signal : PSafeArray;

I, J : Integer;

Bnd : TBounds;

Idxs : TIndexes;

Value : OleVariant;

St : String;

begin

// Получение сигнала из StabMed2

Server.GetSignal ( ProbeID, ChanID, Signal );

// Если данные вернули

if Signal <> nil then begin

// Получим диапазон измерений массива

for I:=0 to 1 do begin

SafeArrayGetLBound ( Signal, I+1, Bnd[I].L );

SafeArrayGetUBound ( Signal, I+1, Bnd[I].U );

end;

// Создадим временный массив

SetLength ( Sign, Bnd[0].U - Bnd[0].L + 1);

for I:=0 to Length(Sign) - 1 do

SetLength ( Sign[I], Bnd[1].U - Bnd[1].L + 1 );

// Разбираем массив на составляющие

for I := Bnd[0].L to Bnd[0].U do begin

for J := Bnd[1].L to Bnd[1].U do begin

Idxs.Row := I;

Idxs.Col := J;

// Получение элемента массива

if SafeArrayGetElement ( Signal, Idxs, Value ) = S\_OK then

Sign[I][J] := Value

else

Break;

end;

end;

// Удалим PSafeArray

SafeArrayDestroy(Signal);

end;

end; // of TForm1.GetSignalArray

//--------------------------------------------------------

// А здесь вызовем Form1.GetSignal

procedure TForm1.btnGetDataClick(Sender: TObject);

var

StabMed2 : ISignalExporter;

NamePac : PAnsiChar;

Born : TDateTime;

TestID, ProbeID, ChanId : Integer;

I, J, PrC, ChC : Integer;

Sign : array of TDoubleDynArray;

begin

// Создаем COM-объект и запрашиваем у него интерфейс

StabMed2 := CreateComObject(CLASS\_SignalExporter) as ISignalExporter;

// Получим идентификатор теста

StabMed2.GetActiveTestID(TestID);

// Получим кол-во проб в тесте

StabMed2.GetProbesCount ( TestID, PrC );

// Цикл по пробам в тесте

for I := 0 to PrC - 1 do begin

// Получим идентификатор пробы

StabMed2.GetProbeID ( TestID, I, ProbeID );

// Получим кол-во каналов в пробе ProbeID

StabMed2.GetChannelCount ( ProbeID, ChC );

// Цикл по каналам пробы ProbeID

for J := 0 to ChC - 1 do begin

// Получим идентификатор канала

StabMed2.GetChannelID ( ProbeID, J, ChanId );

// Если это статокинезиграмма, то прочтем сигнал

if ChanID = chanSKG then begin

GetSignalArray ( StabMed2, ProbeID, ChanID );

meSignals.Lines.Add ( 'Начало сигнала пробы ' + IntToStr(ProbeID) );

for I:=0 to Length(Sign[0]) - 1 do begin

St := '';

for J:=0 to Length(Sign) - 1 do

St := St + FloatToStr(Sign[J][I]) + ' ';

meSignals.Lines.Add ( St );

end;

meSignals.Lines.Add ( '' ); // Добавим пустую строку между сигналами

// Удалим временный массив

for I:=0 to Length(Sign) - 1 do

Sign[I] := nil;

Sign := nil;

end;

end;

end;

StabMed2 := nil;

Идентификаторы каналов

Каждому каналу в StabMed2 ставится в соответствие уникальный идентификатор канала, реализованный в виде целого числа. Эти идентификаторы присваиваются один раз при создании канала и не меняются со временем. В таблице приведены основные каналы и их идентификаторы.

|  |  |
| --- | --- |
| Название канала | Идентификатор |
| Стабилографические сигнал | 10 |
| Стабилографические сигнал 1 | 11 |
| Стабилографические сигнал 2 | 12 |
| Стабилографические сигнал 3 | 13 |
| Стабилографические сигнал 4 | 14 |
| Стабилографические сигнал спинки | 15 |
| Стабилографические сигнал стопы | 16 |
| Стабилографические сигнал рук | 17 |
| Стабилографические сигнал левого подлокотника | 18 |
| Стабилографические сигнал правого подлокотника | 19 |
| Баллистограмма | 20 |
| Баллистограмма 1 | 21 |
| Баллистограмма 2 | 22 |
| Баллистограмма 3 | 23 |
| Баллистограмма 4 | 24 |
| Баллистограмма спинки | 25 |
| Баллистограмма подножки | 26 |
| Баллистограмма рук | 27 |
| Баллистограмма левого подлокотника | 28 |
| Баллистограмма правого подлокотника | 29 |
| Пульс | 41 |
| Пульс 2 | 42 |
| Миограмма | 51 |
| Миограмма 2 | 52 |
| Дыхание | 61 |
| Дыхание 2 | 62 |
| Дыхание 3 | 63 |
| Дыхание второй платформы | 91 |
| Дыхание второй платформы 2 | 92 |
| Дыхание второй платформы 3 | 93 |
| Кистевой силомер | 101 |
| Кистевой силомер 2 | 102 |
| Кистевой силомер 3 | 103 |
| Кистевой силомер второй платформы | 111 |
| Кистевой силомер второй платформы 2 | 112 |
| Кистевой силомер второй платформы 3 | 113 |
| Становой силомер | 121 |
| Становой силомер 2 | 122 |
| Становой силомер 3 | 123 |
| Становой силомер второй платформы | 131 |
| Становой силомер второй платформы 2 | 132 |
| Становой силомер второй платформы 3 | 133 |
| Напряжение шестикомпонентного датчика U1 | 370 |
| Напряжение шестикомпонентного датчика U2 | 371 |
| Напряжение шестикомпонентного датчика U3 | 372 |
| Напряжение шестикомпонентного датчика U4 | 373 |
| Напряжение шестикомпонентного датчика U5 | 374 |
| Напряжение шестикомпонентного датчика U6 | 375 |
| Напряжение шестикомпонентного датчика U7 | 376 |
| Напряжение шестикомпонентного датчика U8 | 377 |
| Главный вектор сил и моментов | 380 |
| Главный вектор сил и моментов 2 | 381 |

ПРИЛОЖЕНИЕ 2

Листинг программного модуля «gui.py»

# -\*- coding: utf-8 -\*-

from PyQt5 import QtCore, QtGui, QtWidgets

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.setWindowModality(QtCore.Qt.ApplicationModal)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.radioButton = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton.setGeometry(QtCore.QRect(950, 310, 95, 20))

self.radioButton.setObjectName("radioButton")

self.radioButton\_2 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton\_2.setGeometry(QtCore.QRect(950, 350, 95, 20))

self.radioButton\_2.setObjectName("radioButton\_2")

self.radioButton\_3 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton\_3.setGeometry(QtCore.QRect(950, 390, 95, 20))

self.radioButton\_3.setObjectName("radioButton\_3")

self.radioButton\_4 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton\_4.setGeometry(QtCore.QRect(950, 430, 95, 20))

self.radioButton\_4.setObjectName("radioButton\_4")

self.radioButton\_5 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton\_5.setGeometry(QtCore.QRect(950, 470, 95, 20))

self.radioButton\_5.setObjectName("radioButton\_5")

self.radioButton\_6 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton\_6.setGeometry(QtCore.QRect(950, 510, 95, 20))

self.radioButton\_6.setObjectName("radioButton\_6")

self.graphicsView = QtWidgets.QGraphicsView(self.centralwidget)

self.graphicsView.setGeometry(QtCore.QRect(450, 30, 256, 192))

self.graphicsView.setObjectName("graphicsView")

self.graphicsView\_2 = QtWidgets.QGraphicsView(self.centralwidget)

self.graphicsView\_2.setGeometry(QtCore.QRect(130, 30, 256, 192))

self.graphicsView\_2.setObjectName("graphicsView\_2")

self.graphicsView\_3 = QtWidgets.QGraphicsView(self.centralwidget)

self.graphicsView\_3.setGeometry(QtCore.QRect(770, 30, 256, 192))

self.graphicsView\_3.setObjectName("graphicsView\_3")

self.graphicsView\_4 = QtWidgets.QGraphicsView(self.centralwidget)

self.graphicsView\_4.setGeometry(QtCore.QRect(280, 270, 256, 192))

self.graphicsView\_4.setObjectName("graphicsView\_4")

self.graphicsView\_5 = QtWidgets.QGraphicsView(self.centralwidget)

self.graphicsView\_5.setGeometry(QtCore.QRect(590, 270, 256, 192))

self.graphicsView\_5.setObjectName("graphicsView\_5")

self.graphicsView\_6 = QtWidgets.QGraphicsView(self.centralwidget)

self.graphicsView\_6.setGeometry(QtCore.QRect(280, 500, 256, 192))

self.graphicsView\_6.setObjectName("graphicsView\_6")

self.graphicsView\_7 = QtWidgets.QGraphicsView(self.centralwidget)

self.graphicsView\_7.setGeometry(QtCore.QRect(590, 500, 256, 192))

self.graphicsView\_7.setObjectName("graphicsView\_7")

self.label = QtWidgets.QLabel(self.centralwidget)

self.label.setGeometry(QtCore.QRect(890, 270, 161, 20))

self.label.setObjectName("label")

self.label\_2 = QtWidgets.QLabel(self.centralwidget)

self.label\_2.setGeometry(QtCore.QRect(550, 230, 55, 16))

self.label\_2.setObjectName("label\_2")

self.label\_3 = QtWidgets.QLabel(self.centralwidget)

self.label\_3.setGeometry(QtCore.QRect(190, 230, 131, 16))

self.label\_3.setObjectName("label\_3")

self.label\_4 = QtWidgets.QLabel(self.centralwidget)

self.label\_4.setGeometry(QtCore.QRect(840, 230, 131, 16))

self.label\_4.setObjectName("label\_4")

self.label\_5 = QtWidgets.QLabel(self.centralwidget)

self.label\_5.setGeometry(QtCore.QRect(350, 470, 131, 16))

self.label\_5.setObjectName("label\_5")

self.label\_6 = QtWidgets.QLabel(self.centralwidget)

self.label\_6.setGeometry(QtCore.QRect(670, 470, 131, 16))

self.label\_6.setObjectName("label\_6")

self.label\_7 = QtWidgets.QLabel(self.centralwidget)

self.label\_7.setGeometry(QtCore.QRect(360, 700, 131, 16))

self.label\_7.setObjectName("label\_7")

self.label\_8 = QtWidgets.QLabel(self.centralwidget)

self.label\_8.setGeometry(QtCore.QRect(680, 700, 131, 16))

self.label\_8.setObjectName("label\_8")

self.checkBox = QtWidgets.QCheckBox(self.centralwidget)

self.checkBox.setGeometry(QtCore.QRect(80, 450, 81, 20))

self.checkBox.setObjectName("checkBox")

self.checkBox\_2 = QtWidgets.QCheckBox(self.centralwidget)

self.checkBox\_2.setGeometry(QtCore.QRect(80, 490, 81, 20))

self.checkBox\_2.setObjectName("checkBox\_2")

self.label\_9 = QtWidgets.QLabel(self.centralwidget)

self.label\_9.setGeometry(QtCore.QRect(80, 420, 81, 16))

self.label\_9.setObjectName("label\_9")

self.radioButton\_7 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton\_7.setGeometry(QtCore.QRect(950, 630, 95, 20))

self.radioButton\_7.setObjectName("radioButton\_7")

self.radioButton\_8 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton\_8.setGeometry(QtCore.QRect(950, 590, 95, 20))

self.radioButton\_8.setObjectName("radioButton\_8")

self.label\_10 = QtWidgets.QLabel(self.centralwidget)

self.label\_10.setGeometry(QtCore.QRect(890, 550, 161, 20))

self.label\_10.setObjectName("label\_10")

MainWindow.setCentralWidget(self.centralwidget)

self.statusbar = QtWidgets.QStatusBar(MainWindow)

self.statusbar.setSizeGripEnabled(True)

self.statusbar.setObjectName("statusbar")

MainWindow.setStatusBar(self.statusbar)

self.menubar = QtWidgets.QMenuBar(MainWindow)

self.menubar.setGeometry(QtCore.QRect(0, 0, 1125, 26))

sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Maximum)

sizePolicy.setHorizontalStretch(0)

sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.menubar.sizePolicy().hasHeightForWidth())

self.menubar.setSizePolicy(sizePolicy)

self.menubar.setNativeMenuBar(True)

self.menubar.setObjectName("menubar")

self.menu = QtWidgets.QMenu(self.menubar)

self.menu.setObjectName("menu")

MainWindow.setMenuBar(self.menubar)

self.menubar.addAction(self.menu.menuAction())

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow","Обработка стабилометрических измерений"))

self.radioButton.setText(\_translate("MainWindow", "Ox"))

self.radioButton\_2.setText(\_translate("MainWindow", "Oy"))

self.radioButton\_3.setText(\_translate("MainWindow", "Oz"))

self.radioButton\_4.setText(\_translate("MainWindow", "Px"))

self.radioButton\_5.setText(\_translate("MainWindow", "Py"))

self.radioButton\_6.setText(\_translate("MainWindow", "Pz"))

self.label.setText(\_translate("MainWindow", "Выбор источника сигнала"))

self.label\_2.setText(\_translate("MainWindow", "Спинка"))

self.label\_3.setText(\_translate("MainWindow", "Правый подлокотник"))

self.label\_4.setText(\_translate("MainWindow", "Левый подлокотник"))

self.label\_5.setText(\_translate("MainWindow", "Правое сиденье"))

self.label\_6.setText(\_translate("MainWindow", "Левое сиденье"))

self.label\_7.setText(\_translate("MainWindow", "Правая ступня"))

self.label\_8.setText(\_translate("MainWindow", "Левая ступня"))

self.checkBox.setText(\_translate("MainWindow", "Проба 1"))

self.checkBox\_2.setText(\_translate("MainWindow", "Проба 2"))

self.label\_9.setText(\_translate("MainWindow", "Выбор пробы"))

self.radioButton\_7.setText(\_translate("MainWindow", "Вейвлет"))

self.radioButton\_8.setText(\_translate("MainWindow", "Фурье"))

self.label\_10.setText(\_translate("MainWindow", "Выбор метода обработки"))

self.menu.setTitle(\_translate("MainWindow", "Загрузить пробу"))