**РОЕКТ: 237**

**1. ТЕМА: Ръкавици за превод на български жестомимичен език**

**2. АВТОРИ:**

**Автор 1**

**Име**: Екатерина Ангелова Радева

**ЕГН**: 0651286330

**Адрес**: гр. Хасково, ул. “Тутракан” 7 -9

**GSM**: 0889145284

**ЕMAIL**: ekaterina.a.radeva@gmail.com

**УЧИЛИЩЕ**: Природо-математическа гимназия "Акад. Боян Петканчин"

**Автор 2**

**Име**: Исмет Метин Исмет

**ЕГН**: 0548148440

**Адрес**: гр. Хасково, ул. "Персенк” 27

**GSM**: 0877379894

**ЕMAIL**: Iscometbg@gmail.com

**УЧИЛИЩЕ**: Природо-математическа гимназия "Акад. Боян Петканчин"

**3. РЪКОВОДИТЕЛ:**

**ИМЕ**: Коля Петрова

**GSM**: 0885827390

**ЕMAIL**: k\_\_petrova@abv.bg

**ДЛЪЖНОСТ**: Старши учител по информатика и ИТ

**4. РЕЗЮМЕ:**

**4.1. Цели (предназначение, кратък анализ на потребностите и на съществуващите**

**решения):**

В България има над 25 000 души, ползващи българския жестомимичен език, а комуникацията между хора, които не владеят езика, и глухонемите става чрез лист и химикал или чрез жестомимичен преводач. За съжаление, броят на преводачите у нас достига едва 200. Нашият проект цели да реши проблема с недостига на преводачи, като позволява на потребителя да превежда в реално време жестовете в реч и по този начин осигурява лесна комуникация между участниците в общуването.

**4.2. Основни етапи в реализирането на проекта (основни дейности, роли на авторите):**

1. Разработване на електронни схеми (Работил/и: автор 1)
2. Набавяне на необходимите компоненти и материали (Работил/и: автор 1 и автор 2)
3. Сглобяване на електрониката и ръкавицата (Работил/ли: автор 1)
4. Интегриране на необходимите библиотеки в кода (Работил/и: автор 2)
5. Изготвяне и сегментиране на код в три различни класа:
   * Клас I: Определяне на положението на пръстите (Работил/и: автор 2)
   * Клас II: Определяне на положението на китката (Работил/и: автор 2)
   * Клас III: Определяне на посоката на ускорението на ръката (Работил/и: автор 1)
6. Създаване на MP3 аудио файлове, включващи превод на думи и изрази, и тяхното безпроблемно интегриране в кодовата база (Работил/и: автор 1 и автор 2)
7. Тестове и дебъгване (Работил/и: автор 1 и автор 2)

**4.3. Ниво на сложност на проекта − основни проблеми при реализация на поставените**

**цели:**

Сложността на проекта е от високо ниво поради съчетаването на хардуер и софтуер, както и създаването на алгоритми за изчисляването на състоянието на ръката. Някои от предизвикателствата, които трябваше да преодолеем са:

1. Превръщането на най-обикновени ръкавици за спорт в жестомимичен преводач, чрез свързването на отделните компоненти и сензори и намирането на начин за прикрепянето им към ръкавицaтa.
2. Проучването на библиотеките (MPU6050.cpp/.h, MPU6050\_6Axis\_MotionApps20.cpp/.h, I2Cdev.cpp/.h, helper\_3dmath.h, mp3tf16p.h, SoftwareSerial.h и Wire.h), нужни за използването на електронните компоненти.
3. Калибриране на сензорите.
4. Обработването на данните, получени от MPU6050 (жироскоп и акселерометър), за позицията и придвижването на ръката.
5. Оптимизиране на времето за работа на програмата.
6. Дебъгване и тестване на проекта.

**4.4. Логическо и функционално описание на решението – архитектура, от какви модули е изградено, какви са функциите на всеки модул, какви са взаимодействията помежду им и т.н.:**

Електронни компоненти:

1. 1 x Arduino Uno (Rev3)

2. 5 x Flex Sensor

• Отчитат съпротивлението на пръстите при свиване

3. 1 x MPU6050 сензор

• Засича посоката на движение на ръката, както и ъгъла на завъртане на ръката по осите X, Y и Z

4. 1 x DFPlayer Mini

• Съхранява mp3 файлове с думи и изрази

5. 1 x Спийкър

Структура на кода:

1. Клас Fingers:

• Получава данните от Flex сензорите и класифицира състоянието на всеки пръст според увеличението на съпротивлението при

свиването на пръста:

Състояние 1 – пръстът е изправен или в отпуснато положение Състояние 2- пръстът е леко присвит

Състояние 3- пръстът напълно свит

2. Клас AccelerationDirection:

• Cпоред увеличаването или намалянето на стойности по осите X, Y и Z се открива посоката на движение на ръката (ляво-дясно, горе- долу, напред-назад)

3. Клас Position:

• Засича ъгъла на наклон по осите X, Y и Z при завъртане на китката

4. Main.ino:

• Получава стойностите от всички класове и проверява за съвпадение с условието за състоянието на ръката при определен жест, след което активира спийкъра

**4.5. Реализация − обосновка за използвани технологични средства, алгоритми, литература, програмни приложения и др.:**

1. Софтуер за електронна схема:
   * Използвахме Fritzing, защото ни позволява да направим лесна за разбиране схема, както и автоматично да изведем списък с нужните компоненти.

2. Среда за програмиране:

* + Използвахме Arduino Uno платка и съответния Arduino софтуер, защото те ни позволяват обединяването на електронните компоненти и библиотеките нужни за употребата им.

3. Използвани библиотеки:

* + MPU6050.h - позволи ни да изведем ускорението без да взимаме предвид гравитацията.
  + SoftwareSerial.h - позволява серийна комуникация с цифров щифт, различен от серийния порт.
  + Wire.h - позволява комуникацията с I2C устройства.
  + mp3tf16p.h - позволява съставянето на речник от mp3 файлове.

4. Алгоритми:

* + Ръкавиците разчитат жестовете на ръката като ги разделят на три основни категории (степен на съпротивление на сензорите на пръстите при свиване на ръката, завъртане на китката и посока на движение на ръката).
* Алгоритъм за класификация на позицията на пръстите (от 1 до 3) за по улеснена работа с данните

Класифициране на състоянието на палеца:

int Fingers::classifyThumbFlexValue(int sensorValue) {

if (sensorValue < 5) {

return 1;

} else if (sensorValue >= 5 && sensorValue < 25) {

return 2;

} else {

return 3;

}

}

* Алгоритъм за получаване на посоката на движение на ръката:

Проверяваме стоиността по осите X, Y и Z за промяна на всеки 100 милисекунди и при такава се отчита пососка, като се взима на предвид шума(стоиността е по малка от *directionThreshold*), при който дори и да има промяна в стойността не се оттчита движение

Пример за движение по остта X:

if (abs(deltaX) > directionThreshold) {

if (deltaX > 0) {

//Serial.println("Backward");

detectedXDirection="Backward";

} else {

//Serial.println("Forward");

detectedXDirection="Forward";

}

}

* Алгоритъм за изчисляване на завъртане на китката в градуси

* Алгоритъм за проверка за съвпадение между получените данни и спецификациите на даден жест

Литература:

* <https://www.youtube.com/watch?v=PBdqgHj_AkU&t=340s>
* https://dev.azure.com/overlording/The%20Last%20Outpost%20Workshop/\_git/MP3-TF-16P
* <https://github.com/CarbonAeronautics>
* <https://github.com/ElectronicCats/mpu6050>
* <https://www.spreadthesign.com/bg.bg/search/by-category/>

**4.6. Кадри от работа с ръкавицата на този** [**линк**](https://pmghaskovo-my.sharepoint.com/:v:/g/personal/ismetismet05_pmg-haskovo_org/EVjAvpZyL1FCrHGYK-630MEBUbCxjJTvDxrnSQ4_csVDmw) **или този QR код:**



**4.7. Описание на приложението – как се стартира и/или инсталира, как се използва, как се поддържа:**

Софтуерът е предварително качен на ръкавицата, което прави работата с нея изключително лесна за потребителя. Единствено кабелът на батерията се слага в подходящия порт на ръкавицата и те са готови за употреба.

**4.8. Заключение – какъв е основният резултат, дали има приложения до момента, какви възможности съществуват за развитие и усъвършенстване:**

До момента речникът на ръкавицата се състои от азбуката, както и основни думи и изрази. За в бъдеще планираме да добавим втора ръкавица за още разнообразни жестове, както и създаването на приложение за телефон или компютър, чрез което ще могат да се измислят и добавят жестове от потребителя. Това ще позволи персонализацията на ръкавиците (ще бъде възможно създаването на жест за собствено име на човек, град, училище и др.). Други подобрения които желаем до имплементираме са преработка на дизайн и създаването на възможност за превод и на чужди езици.