Guión video Bikey.

Introducción y justificación:

Los accidentes viales representan la octava causa de muerte mundial en personas de todas las edades, y la primera causa de muerte en niños y adultos jóvenes entre 5 y 29 años de edad. Las muertes en una motocicleta como vehículo representan el 28% de todos los accidentes viales en el mundo. Y esta cifra ha permanecido constante desde 2013. Hay algo en las motocicletas o en su manejo que no ha cambiado con los años y que sigue haciendo de estas un vehículo inseguro.

Uno de los factores más importantes con el cual es posible reducir la tasa de muertes en accidentes viales y sus consecuencias, es la seguridad y la calidad del vehículo. Características como sistemas de estabilidad y control electrónico, y frenos avanzados son ejemplos de estándares de seguridad vehicular que pueden prevenir un choque y reducir sus lesiones. A pesar de los beneficios potenciales, no todos los vehículos requieren estar equipados con estándares internacionales. En México solamente es requisito que un vehículo cumpla con uno de los ocho estándares internacionales de seguridad vehicular.

Es posible hacer un vehículo que sea más seguro si le proporcionamos al motociclista toda la información relevante acerca de su vehículo, confirmando que está en buena condición para salir al camino. Esto se puede hacer mejorando la forma en la que la motocicleta se comunica consigo misma y con el motociclista, proporcionando información clara de su estado y mejorando la calidad de vida del motociclista.

Esto fue lo que tuvimos en mente cuando se diseñó Bikey, mejorar las motocicletas para que se reduzcan los accidentes viales en las cuales están involucradas y que se disminuyan las consecuencias posteriores.

Hardware y Software:

La principal idea en nuestra mente a la hora de elegir los componentes y los entornos de desarrollo para desarrollar la idea de Bickey que se tenía en mente, fue asegurar una buena experiencia de usuario, completa compatibilidad y la capacidad de trabajar remotamente debido a la pandemia que se sigue viviendo hasta este momento.

Para el cerebro de todo el proyecto se utilizó una Raspberry Pi 4B, ya que era la forma más cómoda y pequeña de controlar al resto del sistema, así como por su gran flexibilidad y variedad de programas. Adicionalmente se utilizará una pantalla táctil de 7´´ compatible con la Raspberry Pi la cual nos permita mostrar la información recolectada de una forma intuitiva.

Para obtener el valor de los sensores y enviar la información a la Raspberry Pi 4B se usaron Microcontroladores AtMega328p, ya que es muy compatible con los protocolos de comunicación utilizados y tiene capacidad de procesamiento de sobra. Se intentó utilizar un AtTiny85, pero hubo problemas a la hora de comunicarlo con la Raspberry, por lo que al final se utilizó el AtMega328p.

Para comunicar a la Raspberry Pi 4B con los microcontroladores se utilizó I2C como protocolo de comunicaciones. Este protocolo fue el elegido debido a su gran simplicidad en la conexión eléctrica, la cual solamente requiere de dos líneas y una resistencia de pull-up en cada una de las líneas, así como por la simplicidad en su trama de datos y su compatibilidad natural con la computadora principal y los microcontroladores. Las desventajas de este protocolo son su baja velocidad de transmisión de solamente 400Kb/s y la corta longitud de las líneas permitida, pero esto no es perjudicial para este proyecto debido a la baja cantidad de datos a utilizar y el tamaño corto de las conexiones.

Para el diseño de la interfaz gráfica se utilizó PyQt5, ya que este tiene una implementación muy natural del protocolo I2C, así como por la posibilidad de diseñar la interfaz utilizando las herramientas de Qt Designer y programarla con Python.

Para el diseño de la interfaz se utilizó el entorno gráfico de Qt Designer para simplificar el proceso. Para hacer el programa de Python que controle a la interfaz se utilizó el IDE PyCharm por ser el más ampliamente utilizado.

Para diseñar los programas de los microcontroladores, se utilizó el nuevo IDE de Arduino 2.0, el cual actualmente se encuentra en fase Beta 7. Esto para analizar sus nuevas características y darlas a conocer.

Por último, se utilizó el sistema de control de versiones Git para tener un completo control en el desarrollo del proyecto, así como los servicios de almacenamiento de repositorios de Git: GitHub, el cual facilitaba enormemente el trabajo remoto.

Al final de este video se podrán encontrar nuestros comentarios acerca de por qué no funcionó el AtTiny85 y el análisis del nuevo entorno de desarrollo para Arduino.

Metodología:

Bikey consta de una computadora principal, en este caso una Raspberry 4B, la cual está conectada a dos placas de circuito impreso las cuales incluyen un microcontrolador ATmega328P cada una, encargados de leer los datos de los sensores de la motocicleta y mandarlos a la computadora principal para que esta pueda procesarlos y mostrarlos en una pantalla táctil de 7”. Todo esto para sustituir al tablero principal por una pantalla HMI la cual muestra los indicadores de manera más clara y permite un enorme rango de mejora y actualización.

Se conectarán las dos placas con microcontroladores a los diferentes sensores de la motocicleta, en un principio para sustituir las funciones que ya vienen establecidas en el tablero. La iformación será enviada posteriormente a la Raspberry Pi por medio de la comunicación I2C. La Raspberry Pi leerá las tramas enviadas por los dos microcontroladores y las procesará la información para que pueda ser manipulada y mostrada a partir de la pantalla táctil.

Resultados:

El objetivo del proyecto está cumplido, se puede observas cómo los indicadores en la nueva pantalla cambian conforma al estado de sus respectivos sensores. A continuación, se muestra la prueba de varios botones y demás indicadores.

Conclusiones:

Al implementar una pantalla grande con imágenes claras, una interfaz intuitiva y una alta velocidad de reacción, se mejora enormemente la experiencia de usuario del motociclista y se abre la puerta a una infinidad de mejoras como un botón de pánico, un sistema de localización por GPS, conexión al celular por Bluetooth, detección de accidentes y cuidados post-accidentes y un sistema antirrobos por mencionar algunas.

Estamos conscientes que un vehículo seguro no va a acabar completamente con los accidentes y las muertes vinculadas a las motocicletas, hay que hacer mucho más. En cuanto al ámbito legislativo, es necesario implementar leyes en el uso adecuado de una motocicleta como obligar el uso de un casco certificado, leyes reguladoras de velocidad y de mejora de los caminos. También es importante hacer que el usuario de una motocicleta sea más responsable. En México casi nadie sigue las leyes establecidas. Solamente el 83% de los conductores y el 55% de los pasajeros utilizan un caco de seguridad el cual no requiere de ninguna certificación. También es posible utilizar a Bikey para generar cultura en los usuarios de motocicletas y dar solución a esta problemática.

¿Por qué no funcionó el AtTiny85?

En un inicio, se tenía la intención de utilizar un microcontrolador más pequeño y económico que fuese justo lo necesario para los requerimientos de velocidad y procesamiento del proyecto, ese microcontrolador era el AtTiny85. Este microcontrolador fue descartado en las etapas tempranas del desarrollo del proyecto ya que era imposible establecer una comunicación estable con la Raspberry Pi. Hay muchas discusiones con respecto a este tema, pero se llegó a la conclusión de que no funcionó debido a que el esclavo, el AtTiny85, no podía alcanzar la velocidad del reloj que determina el maestro, la Raspberry Pi, creando un fenómeno llamado clock stretching. Específicamente, el módulo que controla la comunicación I2C de la Raspberry Pi no se lleva bien con el clock stretching, haciendo imposible la comunicación. Al final se tomó la decisión de cambiar de microcontrolador y utilizar el AtMega328p.

Análisis Arduino IDE 2.0:

Después de muchos años teniendo el mismo entorno de desarrollo integrado anticuado para Arduino, por fin se está actualizando a una nueva versión que tiene las mismas características que la versión clásica, pero con todas las características de desarrollo que tienen los IDEs modernos. Este nuevo IDE cuenta con una nueva interfaz muy mejorada y más intuitiva, personalización del entorno de trabajo, modo de depuración, nuevos y mejorados administradores de placas y bibliotecas y autocompletado de código. Todas son características que mejoran el desarrollo de aplicaciones para la plataforma de Arduino. Desafortunadamente, actualmente se encuentra en una fase de desarrollo muy temprana, específicamente la Beta 7, por lo que está repleto de bugs y errores que entorpecen la codificación, llegando a ser casi imposible utilizar el entorno. Algunos errores era la incorrecta detección de las bibliotecas, de las definiciones de las funciones, la detección de placas, entre muchos otros, por lo que no es recomendable utilizar este entorno de desarrollo por el momento. Si buscas alguna alternativa al el clásico IDE de Arduino, es recomendable utilizar el editor de código Visual Studio Code e instalar las extensiones de Arduino o de Platform IO, las cuales funcionan correctamente y son muy recomendables, con la única desventaja de tener que hacer configuraciones con las que no suelen estar familiarizadas los desarrolladores principiantes. El nuevo IDE de Arduino tiene futuro, pero todavía le falta mucho camino por recorrer para solucionar todos sus problemas.