UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UN EMULADOR DE SISTEMA DE CONTROL AUTOMÁTICO DE GLUCOSA EN SANGRE SOBRE UN SISTEMA HARDWARE DE BAJO COSTE

AUTOR: FRANCISCO NAVARRO MOYA

TUTOR: JOSÉ LUÍS DIEZ RUANO

COTUTOR: JORGE BONDÍA COMPANY

Curso Académico: 2017-18

RESUMEN

Este documento corresponde al proyecto final de grado del alumno de ingeniería en tecnologías industriales Francisco Navarro Moya. Se trata de un trabajo dentro del dominio de la automatización y el control en el que se busca automatizar el tratamiento para las personas diabéticas que necesiten de inyecciones de insulina para conseguir controlar su nivel de glucosa en sangre a través de un sistema inalámbrico que funcione sin necesidad de la actuación del paciente como bomba de insulina automatizada. Esto ha sido posible gracias a un simulador de Matlab que ofrece medidas de glucosa y todos los datos necesarios sobre una serie pacientes ficticios para poder trabajar en todos los escenarios posibles y poder comprobar el buen funcionamiento de los controladores implementados; y se han empleado dos placas Arduino, una de ellas será la encargada de realizar el control y la otra se encargará de realizar la función de monitor de glucosa y suministrará a la primera el valor de la glucosa en un intervalo de tiempo que correspondería a cinco minutos en la vida real para poder llevar a cabo el control, lo que permitirá emular el funcionamiento de una bomba de insulina inalámbrica, que recibe los valores de glucosa del paciente cada cinco minutos.

Palabras Clave: Controlador, Diabetes, Matlab, insulina, Arduino

RESUM

Aquest document correspon al projecte final de grau de l'alumne d'enginyeria en tecnologies industrials Francisco Navarro Moya. Es tracta d'un treball dins del domini de l'automatització i el control en el qual se cerca automatitzar el tractament per a les persones diabètiques que necessiten d'injeccions d'insulina per a aconseguir controlar el seu nivell de glucosa en sang a través d'un sistema sense fil que funcione sense necessitat de l'actuació del pacient com a bomba d'insulina automatitzada. Açò ha sigut possible gràcies a un simulador de Matlab que ofereix mesures de glucosa i totes les dades necessàries sobre una série pacients ficticis per a poder treballar en tots els escenaris possibles i poder comprovar el bon funcionament dels controladors implementats; i s'han emprat deus plaques Arduino, una d'elles serà l'encarregada de realitzar el control i l'altra s'encarregarà de realitzar la funció de monitor de glucosa i subministrarà a la primera el valor de la glucosa en un interval de temps que correspondria a cinc minuts en la vida real per a poder dur a terme el control, la qual cosa permetrà emular el funcionament d'una bomba d'insulina sense fil, que rep els valors de glucosa del pacient cada cinc minuts.

Paraules Clau: Controlador, Diabetis, Matlab, insulina, Arduino

ABSTRACT

This document corresponds to the final degree project of the student of engineering in industrial technologies Francisco Navarro Moya. This is a work within the domain of automation and control in which it seeks to automate the treatment for diabetics who need insulin injections to achieve control of their blood glucose level through a wireless system that works without the need for the patient's performance as an automated insulin pump. This has been possible thanks to a Matlab simulator that offers glucose measurements and all the necessary data on a series of fictitious patients to be able to work in all possible scenarios and be able to check the proper functioning of the implemented controllers; and two Arduino plates have been used, one of them will be in charge of carrying out the control and the other will be in charge of carrying out the function of glucose monitor and will supply the first with the glucose value in a time interval corresponding to five minutes in real life to be able to carry out the control, which will allow emulating the operation of a wireless insulin pump, which receives the patient's glucose values every five minutes.

Keywords: Controller, Diabetes, Matlab, Insulin, Arduino

ÍNDICE

Memoria

Presupuesto

Anexos

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UN EMULADOR DE SISTEMA DE CONTROL AUTOMÁTICO DE GLUCOSA EN SANGRE SOBRE UN SISTEMA HARDWARE DE BAJO COSTE

Memoria

Curso Académico: 2017-18

CAPÍTULO 1. INTRODUCCIÓN

1.1 OBJETIVO DEL TRABAJO

Este trabajo se enmarca en el proyecto final de grado en ingeniería en tecnologías industriales dentro del ámbito de la automatización y control de procesos, consiste en la automatización del proceso correspondiente a la inyección de insulina por las personas que padecen diabetes para su tratamiento contra la enfermedad de forma que el paciente no tenga que preocuparse sobre cuándo o cuanta dosis de insulina debe inyectarse, para ello se cuenta con un simulador el cual nos proporciona la posibilidad de hacer ensayos con pacientes virtuales antes de probarlo con pacientes reales.

Para hacerlo de una forma más real, se cuenta con dos placas Arduino comunicadas de forma inalámbrica que simulan el funcionamiento que tendría el dispositivo, la primera de las placas Arduino funcionaría como un monitor que mostraría el valor de glucosa en sangre para que el propio paciente pueda ver dichos valores y el segundo Arduino es el encargado de realizar el control a partir de los valores de glucosa y aportando el valor de la acción de control a simulador al simulador.

Otro objetivo es la de informar a los lectores de las posibles causas, síntomas, consecuencias y todos los aspectos de una de las enfermedades más frecuentes en el mundo y que cada vez está más presente en nuestra sociedad como es la diabetes para intentar de esta forma evitar su aparición o por lo menos paliar las consecuencias y dificultades que tienen las personas que lo padecen dando a conocer un posible futuro remedio como es este posible “páncreas artificial” sobre el que se basa todo el proyecto.

1.2. ESTRUCTURA DEL DOCUMENTO

Este documento se va a estructurar dividiéndose en capítulos en los que se trataran las diferentes etapas en las que se ha estructurado su realización, comenzando con la búsqueda de información y acabando con implementación conjunta de todos los procesos desarrollados.

Como se ha comentado en el párrafo anterior la primera parte del documento va a ir orientado a informar al lector sobre la diabetes en todos sus aspectos, consecuencias, tipos, tratamientos… para situarle dentro del ambiente en el que se ha realizado la investigación y dejar claros cuales son los principales objetivos del proyecto.

Después se tratará como se ha llevado a cabo la implementación de la conexión inalámbrica entre las dos placas Arduino y el de estas con el ordenador en el que se encuentra el simulador, construyendo la estructura física que servirá como base para la realización del conjunto final.

El siguiente punto que tratar es la explicación de todas las partes que posee el simulador de Matlab a modo de un breve manual de uso para posibles lectores que puedan ser usuarios potenciales del mismo, dando la información suficiente para su empleo.

Para finalizar se comentará como se ha conectado cada una de las partes para la realización del conjunto final y el funcionamiento del mismo mostrando los resultados obtenidos contrastados con los de un controlador que esta implementado en el interior del simulador, culminando con las conclusiones generales del proyecto.

Además, en la parte final del documento se encontrarán los anexos en los que se encontrará toda la información necesaria para profundizar en cada uno d los aspectos que se han tratado a lo largo del documento.

CAPÍTULO 2. DIABETES

2.1. ¿QUÉ ES LA DIABETES?

La diabetes es una de las enfermedades más comunes que existen, según un estudio realizado por la Organización Mundial de la Salud (OMS), realizado en 2014, había más de 400 millones de personas afectadas por esta patología, lo que supone que 1 de cada 11 personas la padece y se espera que el número de afectados vaya en aumento por lo que es necesario instruir a todas las personas sobre cómo se debe hacer frente a la enfermedad porque aunque no la padezcan seguramente algún familiar o conocido cercano puede padecerla.

La diabetes mellitus, nombre técnico de la diabetes, es una enfermedad basada en la predisposición a tener elevadas concentraciones de glucosas en sangre por diversos motivos entre los cuales destacan la imposibilidad del páncreas del paciente en crear insulina para la absorción de la glucosa o la mala utilización por parte de nuestro organismo de la insulina que se produce.

Normalmente la diabetes suele producirse acompañada por otro tipo de patologías que surgen como consecuencia de la exposición prolongada a niveles de glucosa elevados, como pueden ser la pérdida de visión, problemas en los riñones e incluso, problemas cerebrales o cardiacos los cuales pueden llegar a ser causa de muerte para el paciente.

2.2. ¿QUÉ ES LA GLUCOSA?

La glucosa es un carbohidrato monosacárido que se corresponde con el azúcar que obtenemos de los alimentos que consumimos. Nuestro organismo a través de unas reacciones químicas descompone los alimentos en sustancias más básicas hasta llegar hasta este carbohidrato que es absorbido en el intestino delgado y pasa directamente al torrente sanguíneo.

Desde ahí va recorriendo todo nuestro cuerpo y es absorbida por las células para su utilización como fuente de energía gracias a la acción de la insulina que es la encargada de transportarla al interior de las células.

2.2.1. NIVELES DE LA GLUCOSA

La glucosa en sangre se mide en miligramos de glucosa por decilitro de sangre (mg/dl), dentro de todo su rango de acción se pueden distinguir hasta 5 zonas diferenciadas dependiendo de su valor, ellas son:

- Hipoglucemia severa, cuando la glucosa se encuentra por debajo de 54 mg/dl
- Hipoglucemia, cuando se encuentra entre 54 y 70 mg/dl
- Normo glucemia, rango situado entre 70 y 140 mg/dl
- Hiperglucemia, cunado la medida está entre 140 y 180 mg/dl

- Hiperglucemia severa, por encima de 180 mg/dl

Los dos casos de hipoglucemia normalmente se producen por una mala estimación de la dosis de insulina que debe inyectarse el enfermo antes de las comidas, si esta es muy elevada el valor de la glucosa en sangre disminuirá por debajo de lo ideal.

En caso de hipoglucemia, para contrarrestarla se recomienda tomar algún alimento rico en glucosa como un zumo o, incluso, azúcar de forma directa, y esperar unos 15 minutos para comprobar el efecto, si se sigue con los síntomas repetir el procedimiento.

Si se trata de una hipoglucemia severa el tratamiento debe hacerlo una persona que este con la persona afectada, ofreciéndole algo de comer o inyectándole una dosis de glucagón, pero si el afectado se encuentra inconsciente se inyectaría directamente el glucagón para evitar una posible asfixia.

Cuando nos encontramos en hiperglucemia o hiperglucemia severa es necesario que el valor de glucosa en sangre baje, pero no siempre hay que actuar, ya que después de las comidas es normal entrar en estos rangos, pero eso no significa ni si quiera que padezcamos de diabetes.

Si este valor se prolonga en el tiempo llegaremos a sufrir algunos de los síntomas típicos de la diabetes, lo que puede indicar que tengamos diabetes y tengamos que controlar dichos niveles, para controlarlos si es una diabetes muy leve con una buena alimentación o ejercicio podría bastar, pero si no es así será necesaria la inyección de insulina intercutánea para disminuirla.

2.3. ¿QUÉ ES LA INSULINA?

La insulina es una hormona que se genera en el páncreas cuya labor principal en la de actuar como llave para que la glucosa sea absorbida por las células y éstas puedan utilizarla como fuente de energía para sus funciones vitales. En caso de que no pueda cumplir su función ya sea por falta de hormonas de insulina o por la resistencia a ellas la glucosa se acumulará en la sangre y el organismo entrará en hiperglucemia.

El páncreas genera dos tipos de insulina:

- Insulina de acción lenta, también conocida como insulina basal, esta se encarga de mantener los valores de glucosa entre los valores óptimos a lo largo de todo el día
- Otra de acción rápida la cual se genera cuando los niveles de glucosa son elevados, normalmente después de las comidas por la glucosa obtenida de los alimentos

2.4. ¿QUÉ ES EL GLUCAGON?

El glucagón es otra hormona que se produce en el páncreas, pero ésta al contrario que la insulina sirve para estimular al hígado para que libere las reservas de glucosa que tiene en forma de glucógeno debido a una baja concentración de glucosa en sangre para, de esta forma, mantener la glucemia estable.

2.5. CLASIFICACIÓN DE LA DIABETES

Dentro de la diabetes existen distintos rangos de afección los cuales se diferencian por el nivel de gravedad con la que la diabetes afecta al paciente, por lo que podemos dividirlas en cuatro grandes grupos, diabetes tipo 1, diabetes tipo 2, diabetes gestacional y otros tipos de diabetes cuya presencia es minoritaria.

2.5.1. DIABETES TIPO 1

La diabetes mellitus tipo 1 (DM1), también conocida como diabetes insulinodependiente, supone alrededor del 5% de los casos de diabetes diagnosticados y se caracteriza por la incapacidad del organismo para la creación de la hormona de la insulina que es la encargada de llevar a cabo la función de la glucosa por parte de las células.

Esto es debido a que el sistema inmunológico ha destruido las células encargas de la creación de la insulina las cuales se deberían encontrar en el páncreas imposibilitando de esta forma la generación de insulina de forma natural.

Este tipo de diabetes suele producirse en la infancia o en la adolescencia, por causas genéticas actuando juntamente con algún tipo de infección o una mala dieta, aunque también puede llegar a producirse en personas adultas, aunque es poco probable.

Los pacientes que sufren este tipo de diabetes solo tienen una posibilidad en su tratamiento, que es la inyección de insulina de forma externa a lo largo de toda su vida para poder suplir la deficiencia que presentan.

2.5.2. DIABETES TIPO 2

La diabetes mellitus tipo 2 (DM2) es el tipo de diabetes más extendido y supone entre el 85% y el 90% de los casos estudiados, ella se caracteriza por la generación insuficiente de insulina para la absorción de la glucosa o por una resistencia del cuerpo a la insulina producida por el mismo lo que hace que esta sea inservible y se acumule la glucosa en el torrente sanguíneo.

La resistencia a la insulina provoca que, aunque se produzca insulina los valores de glucosa en sangre sigan siendo elevados, por lo que el organismo comenzará a producir más insulina de forma continua acabando por desgastar la capacidad del páncreas de producir insulina haciendo que la producción de insulina vaya disminuyendo o lo haga más lentamente.

Este tipo de diabetes suele diagnosticarse en personas ya adultas, puesto que es un proceso secuencial en el que se va disminuyendo la insulina producida o aumentado el rechazo hacia la misma producido por el efecto de una mala dieta, la obesidad y el estrés.

Las formas para combatir esta enfermedad dependen del punto de desarrollo en el que se encuentre la enfermedad, si solamente acaba de empezar con un buen programa de ejercicios y dieta puede ser suficiente para su tratamiento e incluso para revertirla, pero si no se detecta a tiempo, se debe empezar el tratamiento con antibióticos que a la larga llevará al tratamiento con insulina como en el tipo 1.

2.5.3. DIABETES GESTACIONAL

La diabetes mellitus gestacional (DMG) es un tipo de diabetes el cual se diagnostica en mujeres embarazadas a las cuales antes no se les había diagnosticado algún tipo de diabetes.

La principal causa es que algunas de las hormonas producidas por el embarazo interfieran en el buen funcionamiento de la insulina produciendo así un incremento en los niveles de glucosa en sangre inferior a los producidos en los otros tipos, pero lo suficiente para estar por encima de los límites establecidos.

Este tipo de diabetes puede ocasionar problemas en el parto por un posible crecimiento excesivo por parte del bebé y también puede propiciar la aparición de la enfermedad en el mismo y la continuidad de ella en la madre, para intentar paliar sus efectos la madre debe controlarse durante el parto haciendo ejercicio y manteniendo una buena alimentación.

2.5.4. OTROS TIPOS

Existen otros tipos de diabetes, pero ellos suponen en su conjunto una minoría que apenas tiene trascendencia, este tipo de diabetes se debe al síndrome MODY, cuyas siglas significan "**Maturity Onset Diabetes of the Young**" (diabetes de edad madura que se presenta en jóvenes"), debido a que es semejante a una DM2, pero en este caso se produce por un tema genético.

Uno de los genes de la persona que lo padece ha sufrido una mutación que afecta a la correcta producción de la hormona de la insulina o a su buen reconocimiento, su tratamiento es idéntico al de una DM2 comenzando con antibióticos para intentar paliar sus efectos pero que más a delante desembocará a la utilización de inyecciones de insulina.

2.6. CAUSAS DE LA DIABETES

Para que se dé lugar a una enfermedad de este tipo puede haber diversas causas, pero en la mayoría de los casos se produce debido a una predisposición genética que se ha heredado de alguna figura paterna acompañado de alguna condición del entorno que la desencadene.

Los factores genéticos pueden ser determinantes, pero no son suficientes para que la enfermedad se reproduzca en los hijos de personas diabéticas.

Por ejemplo, en el caso de gemelos, donde la genética es la misma entre ellos, si uno de ellos padece diabetes tipo 1 el otro tiene una probabilidad del 50% de padecerla, y si se trata de diabetes tipo 2 el porcentaje aumenta hasta el 75%.

Aparte de la predisposición genética deben darse otros factores para que se produzca la enfermedad, en caso de diabetes tipo 1 algunos de estos factores pueden ser la exposición a algún virus, la mala alimentación durante los primeros meses de vida, donde se reduce el riesgo a los bebés que lactaron de pequeños y otro factor tiene relación con el frío, las personas que viven en climas fríos son más propensos a padecer la enfermedad.

En cuanto a la diabetes tipo 2, se puede decir que tiene una mayor influencia de los factores genéticos, pero la principal causa desencadenadora es un mal estilo de vida con malos hábitos los cuales pueden ser inculcados por los padres, aunque se puede hacer frente a esta posibilidad si se produce un cambio en el estilo de vida con la realización de ejercicio y una buena alimentación.

Si se trata de diabetes gestacional como se ha dicho anteriormente es debido a la aparición de hormonas que interfieran en el buen funcionamiento de la hormona de la insulina, propiciando que la diabetes se pueda prolongar después del embarazo o pueda aparecer en el bebé.

Los otros tipos de diabetes tienen un gran factor genético y la probabilidad de que sea hereditaria es aproximadamente del 50%.

2.7. SIMTOMAS DE LA DIABETES

La diabetes como cualquier otra enfermedad tiene una serie de síntomas que indica que la persona que los padece tiene la posibilidad de padecerla, algunos de estos síntomas son lo que se detallan a continuación:

- Aliento afrutado, esto se debe a que frente a la imposibilidad de quemar el azúcar se utilizan grasa generando cetonas que el cuerpo intenta expulsar a través de la respiración
- Debilidad y cansancio, ya que el azúcar produce mucha más energía que las grasas
- Sensación de hambre descontrolada, al no poder utilizarse la glucosa nuestro cuerpo interpreta que necesita una nueva fuente de energía produciendo esa sensación de hambre
- Alta frecuencia al orinar, debido a que el riñón intenta reducir el nivel de azúcar en sangre, en niños se conoce como el fenómeno de la cama mojada
- Problemas estomacales, como ardores de estomago
- Pérdida de peso, puesto que ante la imposibilidad de utilizar la glucosa como una fuente de energía intenta sacarla de los músculos del paciente propiciando esta disminución de peso
- Sed excesiva, este síntoma viene provocado por el exceso de actividad de los riñones antes comentado, ya que dicha actividad deshidrata nuestro cuerpo
- Vista nublada, este efecto se produce frente a una exposición prolongada a niveles elevados de azúcar haciendo que el cristalino se llene de agua, y si no se pone remedio puede desembocar en ceguera
- Entumecimiento de manos y pies, ya que niveles elevados de glucosa puede afectar a terminaciones nerviosas
- Por último, niveles elevados de azúcar en sangre y orina en situaciones poco comunes, por ejemplo, por la mañana cuando todavía se está en ayunas

2.8. COMO DIAGNOSTICAR LA DIABETES

En primer lugar, si una persona padece varios de los síntomas anteriormente comentados sería prudente que visitará a su médico para comprobar si padece diabetes u otro tipo de patología o no tiene ninguna relación con ello, existen varias pruebas para ayudar a saber si se sufre o no de diabetes.

2.8.1. PRUEBA A1C

Una de las pruebas más empleada para el diagnóstico de la diabetes es la A1c, que consiste en obtener el nivel promedio de la glucosa en sangre durante 2 o 3 meses, además de para diagnosticarlas los enfermos de glucosa deben realizarse esta prueba de forma periódica para comprobar que sus valores de glucosa se mantienen dentro de unos valores limite establecidos, para así asegurarse que la están tratando correctamente.

Los resultados se devuelven en porcentajes y un buen resultado no debería de superar más de un 6’5%, que al traducirlo a miligramos de glucosa por decilitro de sangre correspondería con 140 mg/dl.

2.8.1. GLUCOSA PLASMÁTICA EN AYUNAS

Otra prueba que se lleva a cabo es la conocida como obtención de la glucosa plasmática en ayunas, consiste realizar una medición de glucosa en sangre pasadas por lo menos 8 horas desde que el paciente haya realizado su última comida o desde que bebió algún líquido que no fuese agua.

En esta prueba unos valores por encima de 126 mg/dl pueden ser indicadores de que el paciente padece de diabetes.

2.8.1. TOLERANCIA A LA GLUCOSA ORAL

La prueba de la tolerancia a la glucosa oral, esta prueba consiste en que el médico da al paciente una bebida rica en glucosa para ver cómo responde el organismo al tratamiento de la glucosa, en ella se tomará una medida de glucosa en sangre antes de tomar la bebida y después se realizara una medición a cada hora durante las primeras tres horas después de la ingesta, esta prueba puede realizarse tanto estando de ayunas como sin estarlo, pero en este último caso si en la primera medida sale la glucosa algo elevada, sería mejor repetirla estando en ayunas.

El resultado podrá ser considerado como positivo si dos o más de las mediciones dan valores de glucosa por encima de lo esperado.

Esta prueba es la que se hace a las mujeres embarazadas para comprobar si padecen la diabetes gestacional, pero también puede utilizar para otros tipos

2.8.1. MEDICIÓN ALEATORIA DE LA GLUCOSA PLASMÁTICA

Otro tipo de prueba es una medición aleatoria o casual de la glucosa plasmática, es la prueba más simple de todas se trata de una medición de la glucosa cuando se produzcan síntomas de diabetes severa, si los resultados plasman un elevado nivel de glucosa, por encima de los 200 mg/dl aproximadamente, puede deducirse que el paciente tiene diabetes si no es así esos síntomas deben de tener otra causa

Si en este caso o en algunos de los casos anteriores los valores dan cercanos pero un poco inferiores a los límites establecidos puede deducirse que el paciente padece lo que se conoce como prediabetes, que significa que está en riesgo de padecer diabetes de tipo 2 si no se toman las medidas necesarias para normalizar los valores de glucosa en sangre.

2.9. TRATAMIENTO DE LA DIABETES

El tratamiento de la diabetes es un procedimiento largo y en la mayoría de los casos es un tratamiento que va a acompañar al paciente a lo largo del resto de su vida.

En primer lugar, un enfermo de diabetes debe tomarse medidas de glucosa en sangre de forma periódica, como mínimo una vez al día. Esta medición necesita de un elemento punzón, una tira reactiva y un glucómetro el cual nos dirá los resultados de la prueba.

Esta medición debe ser más frecuente en personas con diabetes tipo 1 o con tipo 2 que necesiten la inyección de insulina en su tratamiento, también debe realizarse en aquellos que no la necesitan cuando cambian el método utilizado para el mismo. La frecuencia también se ve afectada cuando la persona diabética sufre alguna enfermedad que conlleve sufrir fiebre de un valor elevado.

A parte de esto, la diabetes se puede tratar de varias formas dependiendo del grado de desarrollo en la que se encuentre y del tipo de diabetes que se padezca.

Por ejemplo, los enfermos de diabetes tipo 1 y los de diabetes tipo más avanzada deben de tratarse con la inyección de insulina, ya que su cuerpo no es capaz de generar la insulina suficiente y ningún otro método puede suministrarle tal cantidad.

Los enfermos de diabetes tipo 2 en un nivel intermedio se tratan con la ingesta de medicinas vía oral, las cuales reducen la producción de glucosa del hígado y ayudan a que mejore la absorción de la glucosa por parte de las células.

Además de todo esto, siempre es aconsejable llevar un estilo de vida saludable con una buena alimentación y con presencia de ejercicio, esto, además, puede ayudar a que si se tiene la conocida prediabetes en algunos casos puede evitar la aparición de la enfermedad y en la mayoría por lo menos retrasarla, ya que unos de los desencadenantes de la diabetes tipo 2 puede ser la obesidad desencadenada por un mal estilo de vida.

2.9.1. TRATAMIENTO CON INSULINA

El tratamiento con insulina se emplea en caso de diabetes tipo 1 o en diabetes tipo 2 avanzada, consiste en inyectar insulina por debajo de la piel, la cual suplirá la que nuestro organismo no puede generar.

La insulina que se utiliza en estas inyecciones se consigue gracias a implantar el gen humano que contiene la información de la creación de la insulina en una bacteria, la cual produce insulina más rápidamente; a esa insulina después se le hará un proceso de purificación para poder ser utilizada.

2.9.1.1. Tipos de insulina

Existen diferentes tipos de insulinas los cuales se utilizan según la situación que se nos presente durante el día.

2.9.1.1.1. Insulina de acción rápida

La insulina de acción rápida se caracteriza por ser absorbida a gran velocidad por el tejido subcutáneo de nuestro cuerpo y suele emplearse justo antes de las comidas cuando se va a

tener un aumento elevado de la glucosa o en aquellos casos que se tenga la glucosa muy elevada.

Su rango de acción se extiende desde los primeros 15 minutos después de realizar la inyección hasta unas 4 o 6 horas después de la misma, dependiendo del valor de la dosis suministrada. Su efecto máximo se produce transcurridas entre 1 y 2 horas después de la suministración de la insulina.

2.9.1.1.2. Insulina de acción prolongada

Esta insulina es totalmente contraria al anterior tipo, se absorbe lentamente y es la encargada de suplir la insulina basal que generaría el páncreas, para controlar la glucosa que va generando el hígado y mantener los niveles de glucosa más o menos estables.

El inicio de su efecto se produce alrededor de una hora después de su inyección y su efecto puede prolongarse durante las 24 horas del día. En este caso no existe un pico como tal, sino que se llega a un valor máximo que se mantiene durante varias horas, el cual comienza sobre las 5 horas desde el suministro.

2.9.1.1.3. Insulina de acción intermedia

Este tipo de insulina se obtiene, normalmente, por la mezcla de los dos tipos anteriores, sus usos son más parecidos a los de la insulina de acción prolongada ya que no tiene un pico demasiado elevado.

Su efecto comienza pasada la primera hora, con una duración de sobre unas 12 horas, teniendo su pico entorno a las 4 o 6 horas de haberse realizado la inyección.

2.9.1.2. Métodos de suministro

La insulina puede suministrarse de diversas formas, la gran mayoría de los usuarios utiliza métodos de inyección, pero también existen métodos en los no se lleva a cabo ninguna inyección como puede ser a través de inhaladores.

2.9.1.2.1. Frasco y jeringuilla

Este método es el más sencillo se dispone de una jeringuilla a la que se le coloca una aguja y se extrae la insulina de un vial, tanto como la dosis que le sea necesaria al paciente, y después se aplica la inyección, esta suele realizarse en el abdomen por ser la mejor zona, pero también se puede hacer en el muslo o en el glúteo, por ejemplo, para evitar de esta forma realiza la inyección siempre en el mismo lugar.

2.9.1.2.2. Autoinyectores tipo bolígrafo

Es el método más cómodo para el paciente, la insulina se encuentra en un dosificador con forma de bolígrafo que en vez de punta tiene una aguja, en la parte posterior hay una ruleta con la que se elige la dosis que se va a suministrar y un botón el cual se pulsa para insuflar la insulina después de haber clavado la aguja.

Figura 1. Autoinyector de insulina tipo bolígrafo

2.9.1.2.3. Inhalador

Consiste en un inhalador que contiene insulina en polvo la cual se absorbe por la boca y va a parar a los pulmones donde pasa a la sangre rápidamente, este método solo se emplea en adultos con diabetes tipo 1 y tipo 2.

2.9.1.2.4. Puerto de inyección

Un puerto de inyección posee un tubo que se pone bajo la piel y que se sujeta con un parche, el suministro de la insulina se lleva a cabo con una jeringuilla o con un autoinyector, pero en este caso no hace falta clavarlos sobre la piel ya que tenemos el puerto. Este puerto es utilizable durante algunos días y después es necesario sustituirlo por otro nuevo.

2.9.1.2.5. Bomba de insulina

Las bombas de insulina son unos dispositivos externos que cuentan con dos partes, un infusor de insulina y un catéter de conexión con la capa subcutánea de nuestro cuerpo.

su funcionamiento se basa en insuflar insulina de forma constante durante todo el día, emulando el comportamiento del páncreas cuando genera la insulina basal, además el usuario puede indicarle antes de las comidas o si fuera necesario un bolo de insulina superior al que se insufla constantemente para controlar los niveles altos de glucemia sin necesidad de realizar ningún pinchazo.

La insulina que se inyecta es de acción rápida, pero en unas cantidades muy pequeñas, y al hacerlo de forma continua tiene un efecto parecido al de una inyección de insulina de acción prolongada.

Algunas bombas vienen acompañadas de un sensor que lee los valores de concentración de la glucosa subcutánea y los muestra en la pantalla que tiene la bomba para así facilitar al usuario saber cuál es su nivel de glucemia en cada momento.

Figura 2. Bomba de insulina con sensor de glucosa continuo

2.10. PROBLEMAS RELACIONADOS CON LA DIABETES

La diabetes es una enfermedad que si no es bien tratada puede conllevar consecuencias muy graves, yendo desde unos leves síntomas hasta producir la muerte de las personas que la padecen. En la actualidad, hay un estudio realizado por la OMS el cual indica que la diabetes es una de las cuatro causas principales de muerte en el mundo, por lo que es necesario realizar un buen tratamiento para evitar estos riesgos.

Esto se debe a que la acumulación de la glucosa en la sangre aparte de los síntomas ya comentados, si se mantiene en el tiempo puede llegar a taponar pequeñas venas, que afectan a zonas como los ojos o los riñones, y si todavía se prolonga más en el tiempo se pueden llegar a obstaculizar grandes vasos sanguíneos pudiendo dar lugar a problemas cardiacos.

Además, los riegos de padecer alguna enfermedad grave aumentan, ya que la mayoría de los diabéticos desarrollan otros tipos de problemas aumentan esa posibilidad, como, por ejemplo, el sobrepeso, la presión arterial elevada o el aumento del colesterol malo.

Entre todas las enfermedades que puede originar las diabetes destacan dos tipos de ellas por encima de todas, los problemas del corazón y los derrames cerebrales, las cuales suponen 2 de cada 3 muertes de personas debido a la diabetes.

2.10.1. PROBLEMAS OCULARES

Los problemas oculares debidos a la diabetes pueden ir desde una visión borrosa hasta la perdida completa de la visión. La principal causa de los problemas oculares es el taponamiento de los vasos sanguíneos que están dentro de la retina lo que puede provocar visión borrosa, dolor o la aparición de manchas en la vista.

Una de las principales enfermedades en relación con la diabetes son las cataratas, las cuales se pueden solucionar a través de cirugía laser, otro problema puede ser el glaucoma el cual se

produce por la aparición de presión al nervio óptico, el cual se puede solucionar de la misma forma.

Estos problemas si no se tratan pueden desembocar en ceguera, por lo que se recomiendo realizar una revisión de la vista al año para las personas diabéticas.

2.10.2. PROBLEMAS RENALES

Los problemas renales se deben a la obstrucción de los pequeños vasos sanguíneos de los riñones, esto provoca que no puedan realizar correctamente su función y no pueden eliminar toda la suciedad que se acumula en la sangre.

Estas enfermedades tienen su comienzo mucho antes de la aparición de los primeros síntomas, por ello es importante que los enfermos de diabetes se hagan controles periódicos de orina y análisis de sangre para comprobar que los riñones funcionan correctamente.

En caso de que no se detecte a tiempo, puede producir problemas muy graves, la diabetes es una de las principales causas de insuficiencia renal, cuyos afectados necesitan de diálisis y en muchos casos un trasplante, por ello controlar la glucemia es muy importante.

2.10.3. PROBLEMAS NERVIOSOS

La acumulación de glucosa en sangre puede producir daños en las terminaciones nerviosas, originando que no se reciban señales nerviosas desde ese punto, que se haga más lentamente o que se envíen impulsos nerviosos sin razón.

Entre los principales síntomas que pueden aparecer destacan entumecimientos de extremidades, dolores punzantes, náuseas y mareos, y problemas urinarios. Con controlar el nivel de glucosa se pueden llegar a prevenir.

2.10.4. PROBLEMAS CARDÍACOS

La diabetes aumenta la disposición de la persona que la padece a contraer enfermedades cardiovasculares, debido a que si se produce la obstrucción de algunos vasos sanguíneos la sangre que circula por nuestro cuerpo se reduce.

Las principales enfermedades cardiacas que se producen son enfermedades coronarias, insuficiencia cardíaca y cardiomiopatía diabética, pero también puede llegar a producirse un paro cardíaco.

Estas enfermedades se tratan con medicamentos para la disminución de la glucosa y la presión arterial, además de unas recomendaciones de vida saludable como una buena alimentación o la realización de ejercicio físico.

2.10.5. PROBLEMAS CEREBRALES

El principal problema cerebral es la posibilidad de sufrir algún derrame cerebral, por causa de la obstrucción de algún vaso sanguíneo cuya funcionalidad es la de suministrar esa sangre al cerebro.

Para intentar prevenir este tipo de problemas se debe llevar un buen control de la glucosa en sangre, de la presión arterial y del colesterol, acompañado de unos hábitos saludables, como una buena alimentación o el ejercicio físico.

Si se llega a padecer, la mejor solución es la cirugía con un posterior tratamiento para ayudar al afectado a recuperar todas las funcionalidades que haya podido perder como consecuencia del derrame.

CAPÍTULO 3. ARDUINO

3.1. ¿QUÉ ES ARDUINO?

Arduino se trata de una plataforma de software libre, que permite a sus usuarios interactuar con el entorno gracias a las posibilidades que presenta. Detrás de esta plataforma se encuentra una gran comunidad que la respalda, la sostiene y permite al usuario avanzar más rápidamente en su investigación.

3.2. HARDWARE DE ARDUINO

Arduino se basa en la utilización de una placa con un microcontrolador, estas placas pueden ser de distintos modelos, yendo desde las más básicas con la Arduino Uno hasta las más completas con un mayor número de entradas y salidas, como es el Arduino MEGA 2560.

Para la realización de este proyecto se ha contado con la disposición de 2 placas Arduino MEGA ADK, las cuales tienen las siguientes características:

- **Microcontrolador:** ATmega2560
- **Voltaje Operativo:** 5V
- **Voltaje de Entrada:** 7-12V
- **Voltaje de Entrada(límites)**: 6-20V
- **Pines digitales de Entrada/Salida:** 54 (de los cuales 15 proveen salida PWM)
- **Pines análogos de entrada:** 16
- **Corriente DC por cada Pin Entrada/Salida:** 40 mA
- **Corriente DC entregada en el Pin 3.3V:** 50 mA
- **Memoria Flash:** 256 KB
- **Velocidad de reloj:** 16 MHz

Figura 3. Placa Arduino MEGA ADK

La alimentación de la placa Arduino se puede realizar mediante conexión USB o mediante una fuente externa, después de pasar por un convertidor AC/DC y reduciendo el voltaje entre los límites establecidos.

Arduino dispone de muchos complementos compatibles con sus placas como pueden ser módulos bluetooth, módulos Wifi, módulos de joystick… lo que aumenta las posibilidades de esta plataforma.

La alimentación de estos complementos se consigue gracias a los pines de 5V y de 3.3V que dispone la placa en conjunto con los pines de toma de tierra (GND) que también dispone. Y la comunicación con los mismos se va a realizar a través de los pines de comunicación serial, los cuales están marcados en la placa bajo el nombre de “communication”.

3.3. SOFTWARE DE ARDUINO

Arduino dispone de un software sencillo que puede ser utilizado por cualquier tipo de usuario, además, posee su propio lenguaje de programación el cual se basa en C++, el lenguaje de programación más extendido, incluso algunos comandos son los mismos.

El código en Arduino se estructura en dos grandes bloques, el primero de ellos, void setup(), se ejecuta una única vez y es en el que se inicializan las variable y se activan los pines que se van a utilizar ya sean analógicos o digitales.

El segundo, void loop(), es una función que se ejecuta en bucle una vez que empieza, dentro de ella es donde se va a colocar los comandos de la ejecución. Y fuera de ellos se pueden definir constantes e incluir librerías que vayan a ser empleadas en el código.

Y en la barra superior de tareas hay funcionalidades importantes como la de compilar o cargar el sketch en el Arduino que se encuentre conectado al ordenador en ese momento, situados en el lado izquierdo en ese orden y el acceso al monitor serial en el cual puedes mostrar valores que se vayan obteniendo en la ejecución, que esta situado a la derecha del todo con el icono de una lupa.

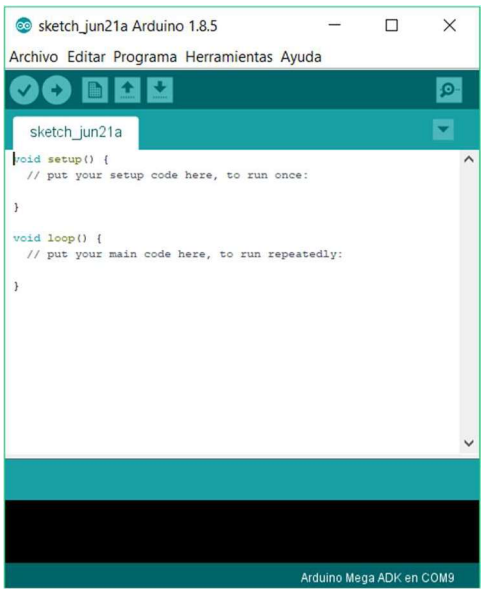

Figura 4. Estructura base código Arduino

CAPÍTULO 4. CONEXIÓN INALÁMBRICA

En el planteamiento del problema se propuso realizar una conexión inalámbrica vía bluetooth entre los dispositivos que van a forma el ciclo, que son dos placas Arduino MEGA ADK y un ordenador en el cual se sitúa el simulador que ha sido suministrado por parte del tutor del proyecto, aunque la transferencia de datos se realizará más lentamente que si se conectan los dispositivos mediante USB.

4.1. DISPOSITIVOS EMPLEADOS

Para llevar a cabo la realización de la conexión bluetooth se han necesitados módulos bluetooth compatibles con las placas Arduino, ya que de forma autónoma no disponen de esta funcionalidad.

Más concretamente se han utilizados 4 módulos, dos de ellos son módulos bluetooth HC-06, el cual se observa en la Figura 5 y se caracteriza por ser un módulo bluetooth modo esclavo, lo cual significa que nunca va a poder establecer una conexión bluetooth por el mismo, sino que necesita de otro dispositivo que la inicie, y por disponer de 4 pines, dos de ellos, llamados “VCC” y “GND”, que sirven para establecer la alimentación del módulo, y los otros dos, llamados “RXD” y “TXD”, los cuales se encargan de la lectura y de la transmisión de datos a la placa respectivamente. Además, cuenta con un led, que indica cuando está conectado a algún dispositivo y cuando no, manteniéndolo fijo en el primer caso y parpadeando rápidamente en el segundo.

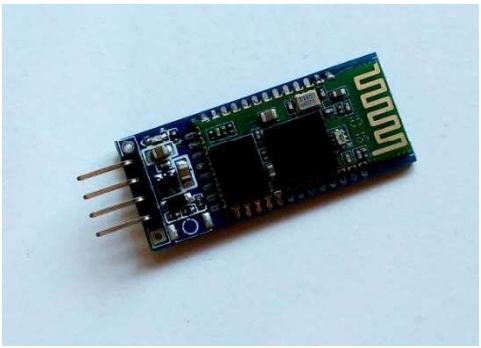

Figura 5. Módulo bluetooth HC-06

Los otros dos módulos empleados en el proyecto son dos módulos bluetooth HC-05 que se muestra en la Figura 6 y que se caracteriza por ser módulo maestro/esclavo, tienen las dos funcionalidades, pero solo pueden estar configurados en una de ellas, cuando se encuentran en modo maestro ellos permanecen constantemente en la búsqueda de un módulo esclavo, que tenga la misma contraseña que él, con el que entablar la conexión y si se encuentran en modo esclavo se comportan como un bluetooth HC-06.

En este caso el módulo dispone de dos pines extra, uno de ellos, llamado “STATE”, solo tiene la funcionalidad de indicar si el módulo se encuentra en conexión con otro o no, y el último pin, llamado “EN”, sirve para cortar la alimentación terminando la conexión con cualquier otro módulo. Al igual que el módulo HC-06, también dispone de un led, pero en este caso tiene 3 estados, parpadea rápidamente cuando no está conectado a ningún dispositivo, parpadea lentamente cuando se encuentra en modo configuración, al cual se entra manteniendo pulsado el botón que hay en el módulo a la vez que se abre el monitor serial, y realiza dos parpadeos de forma periódica cuando se encuentra conectado a otro dispositivo.

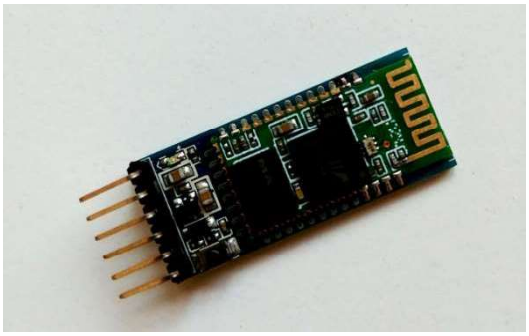

Figura 6. Módulo bluetooth HC-05

4.2. CONFIGURACIÓN DE LOS DISPOSITIVOS

La disposición que se ha escogido a la hora de la conexión ha sido utilizar los dos módulos bluetooth HC-06 para llevar a cabo las conexiones de cada una de las placas Arduino con el ordenador, y utilizar los módulos bluetooth HC-05 para que conecten las dos placas Arduino entre sí.

Para la configuración de los módulos bluetooth HC-06 se ha utilizado un código suministrado por un miembro de la comunidad de Arduino, en uno de sus blogs, el cual se introducirá en la bibliografía. En el que se nos muestra un menú con diversas opciones al abrir el monitor serial como se observa en la Figura 7.

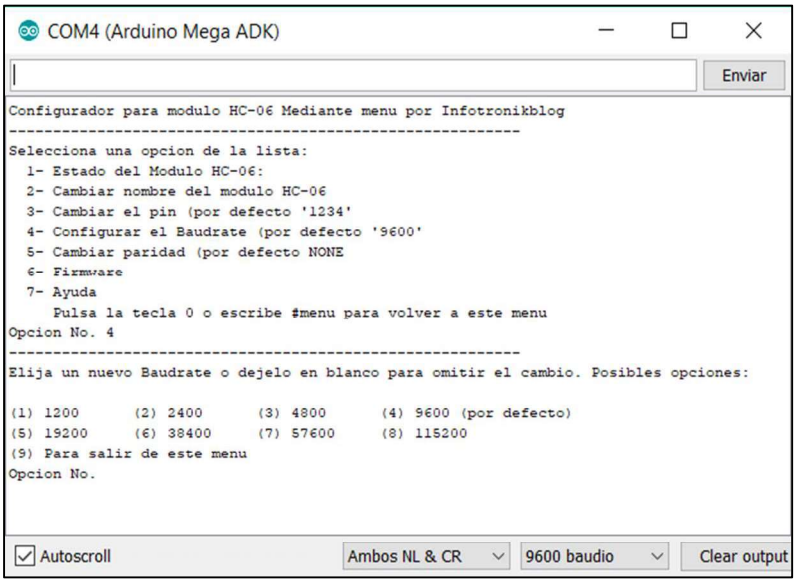

Figura 7. Menú configuración módulos HC-06

Gracias a él se ha cambiado el nombre de los dos módulos a “esclavo1” y “esclavo2” respectivamente a las placas en las que van a trabajar y el baudrate, que es la velocidad con la que se transmiten los datos, cambiando la que viene por defecto que es 9600 baudios, por 115200 baudios que es la velocidad máxima, ya que la gran mayoría de los datos se van a transmitir del ordenador a alguno de las placas o viceversa.

En el caso de la configuración de los módulos bluetooth HC-05 se ha realizado a través de los comandos AT. Estos comandos AT, permiten cambiar el nombre, la contraseña, la velocidad de conexión, elegir a quien se va a conectar el módulo, cambiar entre esclavo y maestro…

Algunos de estos comandos son AT+NAME, que permite conocer y cambiar el nombre del módulo, AT+ADDR, el cual permite conocer la dirección MAC del módulo o AT+BIND, que permite fijar la dirección MAC a la que nuestro módulo se va a conectar.

En este caso se ha establecido que cada uno de los módulos HC-05 tenga preferencia a unirse al otro sin posibilidad de conectarse a ningún otro. En este caso, al contrario que con los módulos HC-06, se ha decidido por mantener la velocidad de transmisión y los nombres por defecto, ya que al no tener la posibilidad de conectarse a ellos los nombre no afectan y, en cuanto a la velocidad se ha mantenido puesto que son pocos los valores a transmitir y la velocidad que nos ofrece es suficiente para obtener un buen resultado. Además, se ha configurado uno de los módulos como esclavo y el otro como maestro, para que se pueda establecer esta conexión entre ellos.

De esta forma tenemos establecida la conexión ordenador-Arduino con el primero de los módulos HC-06, Arduino-Arduino con los dos módulos HC-05 y Arduino-ordenador con el otro módulo HC-06, generando un ciclo semejante al creado con una bomba de insulina que dispone de un sensor de glucosa, que el proceso al se intenta emular con este proyecto.

4.3. COMUNICACIÓN MATLAB-BLUETOOTH

Realmente, la comunicación con los módulos bluetooth se va a realizar desde el software Matlab, que es la plataforma en la que está desarrollado el controlador. Matlab es un software matemático que permite resolver problemas con funciones y ecuaciones de un elevado grado de dificultad rápidamente, por eso ha sido el software escogido para el proyecto.

Dispone de su propio lenguaje de programación (lenguaje M), bastante parecido al lenguaje C++, que se caracteriza por no tener que definir con antelación el tipo de ninguna variable ni el tamaño de las mismas en caso de vectores y matrices.

Para llevar a cabo la comunicación del ordenador con los módulos a través de Matlab es necesario la utilización de algunos comandos de específicos. Estos comandos son:

- B=Bluetooth (‘Remotename’, channel), crea una variable en la que se guradan los datos del módulo bluetooth
- fopen (filename), establece la conexión
- fclose (filename), cierra la conexión
- fprintf (obj, ‘format’, ‘cmd’), escribe un String en el módulo
- fscanf (obj, ‘format’), lee la información que pasa el módulo
- fwrite (obj, A), envía un entero al módulo

- fread (obj), lee un entero del módulo

Como puede verse en la Figura 8, en la que se crea la variable que va a hacer referencia al módulo bluetooth y se realiza su apertura, además en este caso se utilizan variables globales para no tener que abrirlo en cada iteración de nuestro proceso, agilizando el mismo.

```
global A;
global B;
A=Bluetooth('esclavo1', 1);
fopen(A);
B=Bluetooth('esclavo2', 1);
fopen(B);
```

Figura 8. Conexión módulos bluetooth

CAPÍTULO 5. SIMULADOR

El simulador ha sido suministrado por los tutores del proyecto, y como ya he comentado anteriormente se trata de un simulador desarrollado en Matlab debido a la gran facilidad de este software para resolver problemas matemáticos de gran dificultad.

Este simulador es capaz de emular la vida de una persona diabética real con bastante fidelidad, programándole una serie de comidas, una serie de ejercicio, un cierto fallo a la hora de medir la insulina a suministrarse… todo ello dota de una gran realidad a este simulador, por ello es bastante utilizado, su principal objetivo es la de conseguir un control automático de la glucosa en sangre del paciente, facilitándole de esta forma un correcto tratamiento sin necesidad de su intervención, como el funcionamiento de una bomba de insulina pero programándole los horarios de las comidas, para que él introduzca el bolo de insulina.

La estructura de este simulador se caracteriza por tener dos programas principales, los cuales son “set_simulation.m”, en el cual se establece todos supuestos de comidas, ejercicios… que el paciente virtual realizaría, y “run_UVAv32Parallel.m”, que es el programa que lleva el grueso de la ejecución y realiza la simulación de como avanzaría en el tiempo este paciente ficticio.

Además, a parte de estos dos programas, en la Figura 7 podéis ver que el simulador también cuanta con diversas carpetas. En ellas se encuentra información sobre los controladores empleados, pacientes virtuales, comidas…

Y también cuenta con tres variables, todas ellas se tratan de librerías de comidas de las que se van a obtener las diversas comidas que se van a programar en el simulador, dentro de ellas se diferencia entre comidas para niños, comidas para adultos y comidas para adolescentes, ya que la alimentación de las personas varía en función del grupo de edad en el que se encuentre.

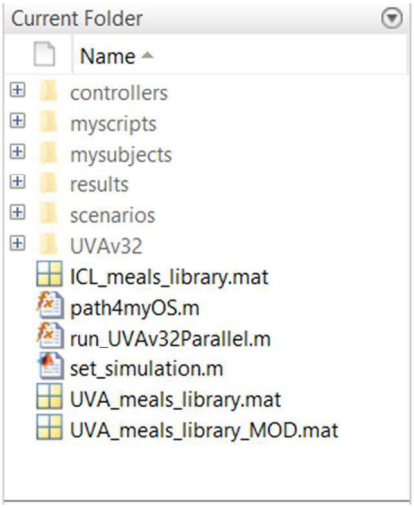

Figura 9. Estructura del simulador

5.1. CARPETAS DEL SIMULADOR

En la Figura 9 se pueden observar como dentro del simulador hay carpetas las cuales albergan la información necesaria para llevar a cabo la simulación, por ello se va a explicar que contienen.

5.1.1. CARPETA CONTROLLERS

Dentro de esta carpeta se encuentra la información de los controladores que existen para llevar a cabo el control de la glucosa en los pacientes. La mayoría son controladores PID (Proporcional-Integral-Derivativo), que es un tipo de controlador que trabaja con retroalimentación para regular la variable sobre la que trabaja comparando con el valor deseado.

Dentro de cada uno de los controladores hay dos funciones, "init_mycontroller.m" y "run_mycontroller.m", en los cuales se definen todos los aspectos relativos al controlador.

En la primera se establecen todas las variables y condiciones iniciales y en el segundo se encuentran las ecuaciones del controlador, devolviendo las acciones de control sobre las variables controladas.

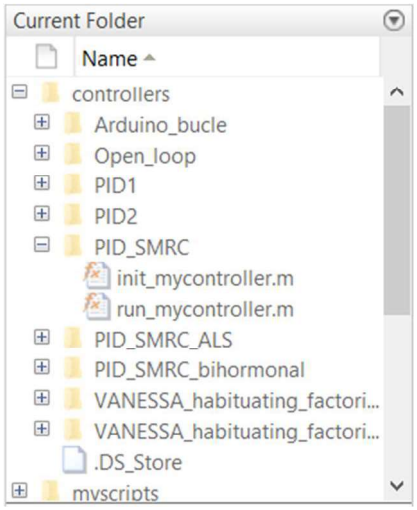

Figura 10. Estructura carpeta controllers

5.1.2. CARPETA MYSCRIPTS

Dentro de esta carpeta se encuentran un conjunto de programas muy variado, como se ve en la Figura 11, entre ellos hay tres tipos de programas según su finalidad. Los programas "classify_meal.m", "exercicse_model.m" y "generate_meal_library.m" se encargan de definir cómo van a afectar las comidas y el ejercicio según el tipo de paciente que se estudie o la cantidad de ejercicio que se vaya a realizar.

Por otro lado, encontramos "CVGA_all.m" y "plotpopfromfile.m" que son programas que contienen las funciones encargadas de la realización de las gráficas que se van a mostrar al usuario para obtener una referencia visual de los resultados.

Y el resto de los programas van orientados a la estructuración de los datos arrojados por la simulación, por ejemplo, el programa "hypos_counter.m" contiene una función la cual cuenta las veces que se ha estado en hipoglucemia y recoge el intervalo de tiempo en el que se ha producido, el programa "time_in_target.m" se encarga de calcular el porcentaje del tiempo que ha estado en cada uno de los rangos de glucosa establecidos, y el programa

"compute_popmetrics.m" estructura todos los resultados obtenidos de la forma en la que se le van a mostrar al usuario.

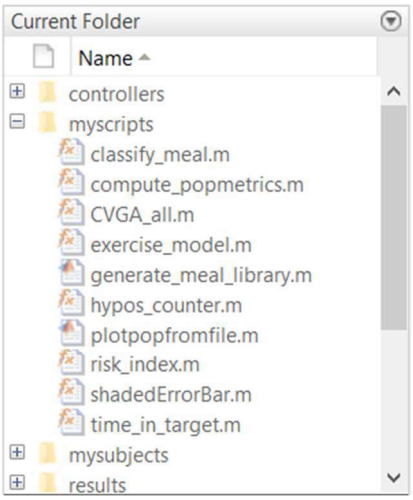

Figura 11. Estructura carpeta myscripts

5.1.3. CARPETA MYSUBJECTS

Dentro de esta carpeta se encuentran los datos de los pacientes virtuales que se van a utilizar durante la simulación. Se disponen de tres tipos de pacientes, adultos, adolescentes y niños, para tener mayor variedad de casos.

Y dentro de cada tipo, hay 10 pacientes diferentes entre ellos, ya que es necesario comprobar que a los controladores no les afecta demasiado las pequeñas variaciones que puede haber entre los integrantes de cada grupo, además se cuenta con un paciente número 11 en cada uno de ellos, que representa a la media de cada grupo. Para el proyecto se ha utilizado este último paciente por el hecho de que así se podían agilizar bastante los cálculos.

Cada paciente tiene forma de estructura en la cual se encuentran los datos más importantes e influyentes a la hora del tratamiento con insulina en la vida real, como por ejemplo puede ser el caso de la edad, el peso…

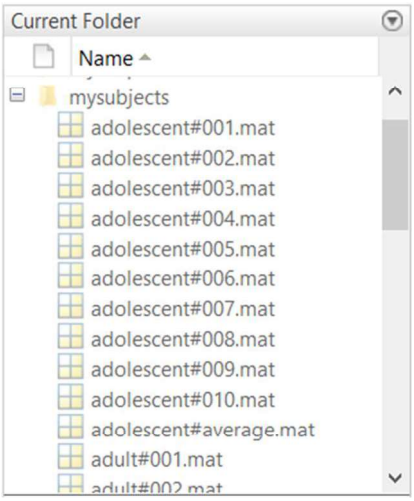

Figura 12. Estructura carpeta mysubjects

5.1.4. CARPETA RESULTS

Dentro de esta carpeta se sitúan las estructuras en las que se van a guardar los resultados de la simulación, en ellas se guarda información muy variada.

En una de las estructuras se guarda la información más relevante de la simulación, entre la que destaca el valor medio de glucosa y el porcentaje de tiempo que ha estado en cada una de las zonas de glucemia, en otras se guardan la información del paciente, del controlador y de las condiciones utilizadas para la simulación que se acaba de realizar y en la última se guardan todos los valores de glucosa en cada instante de tiempo.

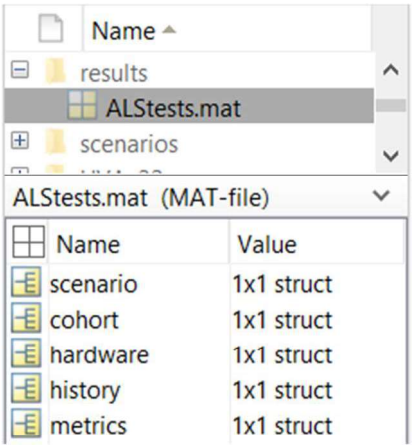

Figura 13. Estructura carpeta results

5.1.5. CARPETA SCENARIOS

La función de esta carpeta es guardar las condiciones iniciales que se hayan configurado en el "set_simulation.m", para tener esta información guardada y poder acceder a ella en caso de necesitarla..

Dentro de la carpeta se encuentran las variables creadas para guardar la información, estas son tipo estructura y pueden ser actualizadas o también se pueden crear otras nuevas con cada simulación.

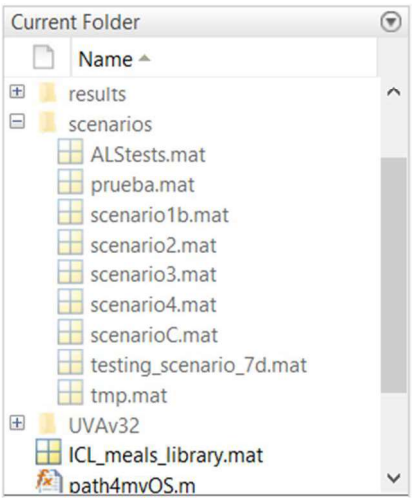

Figura 14. Estructura carpeta scenarios

5.1.6. CARPETA UVAV32

Esta es la última carpeta del simulador, en ella se encuentran 4 programas, como puede verse en la Figura 15, entre ellos los programas “load_subject.m” y “load_quest.m” se encargan de obtener la información del paciente y la información necesaria para la obtención de las constantes del controlador.

Y los otros dos programas contienen funciones en las cuales están implementados los códigos de los cálculos que se utilizarán durante la simulación.

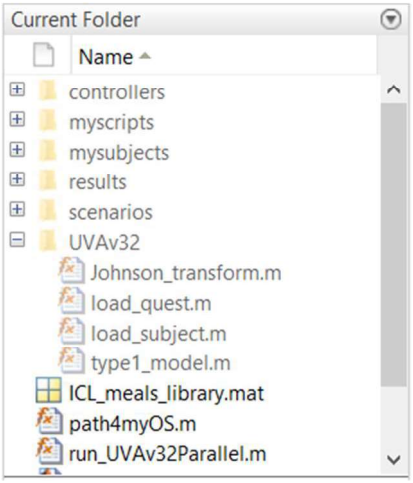

Figura 14. Estructura carpeta UVAv32

5.2. FUNCIONES PRINCIPALES DEL SIMULADOR

Como se ha comentado anteriormente, el simulador posee dos funciones principales, “set_simulation.m” y “run_UVAv32Parallel.m”, el primero de ellos se encarga de establecer las condiciones iniciales de la simulación y algunos aspectos a tener en cuenta; y el segundo es el encargado de realizar la simulación utilizando los datos ofrecidos tanto por el primer programa como por las carpetas anteriormente comentadas.

5.2.1. SET_SIMULATION.M

La funcion “set_simulation.m” se encarga de establecer cuáles van a ser las condiciones iniciales que van a marcar el devenir de la simulación, está estructurado por bloque dentro de los cuales se establecen algunas características comunes.

En esta función por ejemplo se establecen las gráficas que se van a mostrar tras la simulación, el controlador que se va a emplear, la estructura que van a seguir las comidas y el ejercicio físico o cuales van a ser las características de las herramientas empleadas; como también la posibilidad de randomizar las mayoría de factores, como las comidas, las cantidad de insulina que se va a inyectar, para simular posibles fallos de cálculos por parte del paciente dando mayor veracidad a la simulación

Para la realización de este proyecto se ha evitado cualquier posibilidad de randomizar la simulación para que la simulación fuera lo más sencilla posible y no haya ningún elemento aleatorio que imposibilite la comparación entre resultados.

5.2.2. RUN_UVAV32PARALLEL.M

El programa “run_UVAv32Parallel” es el segundo de los programas principales, su ejecución se debe realizar con posterioridad a la del programa anterior, puesto que va a emplear los datos que se han introducido en él.

En el interior de este programa está el código encargado de llevar a cabo la ejecución de la simulación, este código se puede estructurar en varias partes diferenciadas, dentro de las cuales se encuentran diversas funciones, estas partes son:

1. En primer lugar, se sitúa una leyenda, para que el usuario puede tener una buena comprensión sobre los que se realiza con cada comando, y también está la función principal de la simulación, la cual realiza la llamada al resto de funciones que hay dentro de este programa y de otras carpetas, para la realización de la simulación y el guardado de los datos obtenidos
2. En segundo lugar, se sitúan las funciones referentes a la generación de comidas y de ejercicios, estas funciones son llamadas desde la principal y utilizando los valores que se han definido en el “set_simulation.m” nos aportan información sobre el efecto que estas va a tener en los valores de glucosa
3. En tercer lugar, se lleva a cabo la configuración del hardware que se ha escogido, el sensor continuo de glucosa, la bomba de insulina y la de glucagón elegidas en el “set_simulation.m”, el valor del ruido que va a tener nuestro sistema y como va a influir en los valores de glucosa
4. En cuarto y último lugar, se lleva a cabo la configuración de las funciones relativas al controlador, las cuales van a utilizar las funciones de las funciones “init_mycontroller.m” y “run_mycontroller.m” del controlador que se haya seleccionado en el programa “set_simulation.m”

CAPÍTULO 6. CONTROLADOR DISEÑADO

Una vez llevada a cabo la conexión inalámbrica de los módulos bluetooth con el ordenador y entre ellos, otra parte fundamental del proyecto es la adaptación de uno de los controladores internos de Matlab para su utilización en mediante el bucle creado con las placas Arduino, para ello se va a escoger el controlador "PID2" como modelo para realizar esta adaptación, teniendo que conseguir resultados similares.

De entre todos los controladores de los que dispone el simulador, se ha optado por realizar la adaptación del controlador denominado "PID2" debido a que es uno de los más sencillos y otorga unos buenos resultados en las simulaciones, haciendo más asequible su adaptación.

A continuación, se explicará cuáles son las principales características de un controlador de Matlab y posteriormente se enseñará la adaptación realizada para Arduino.

6.1. CARACTERÍSTICAS CONTROLADOR MATLAB

Todos los controladores implementados en Matlab están formados por dos funciones, la función "init_mycontroller.m" y la función "run_mycontroller.m", dentro de las cuales se registran todos los valores, condiciones iniciales y ecuaciones necesarios para el buen funcionamiento del controlador.

6.1.1. INIT_MYCONTROLLER.M

En esta función se realizan las asignaciones para los valores de las constantes del controlador, que en este caso se trata de un controlador PD (Proporcional-Derivativo), también las referencias a seguir y las condiciones iniciales que van a marcar el funcionamiento del controlador.

Para ello, recibe como entrada la información del paciente que se va a usar en la simulación, puesto que algunos de los valores que se van a calcular dependen de estos datos.

Algunas de las condiciones iniciales que se establecen son la posibilidad de realizar un aviso antes de realizar una comida o realizar ejercicio, la posibilidad del cambio de unidades a milimoles por litro (mmol/l) o la fijación de cuáles van a ser los límites que demarquen las zonas de glucemia entre otras.

Y después se calculas las constantes y los valores de referencia que van a caracterizar al controlador diseñado. Con todo esto definido el controlador está totalmente preparado para pasar a la ejecución de la simulación.

6.1.2. RUN_MYCONTROLLER.M

En el interior de esta función residen las ecuaciones que van a realizar el control de las variables escogidas, devolviendo los valores para la acción de control que se les debe aplicar para buscar esa referencia que se ha estipulado en la otra función.

Para poder realizar dichas ecuaciones necesita los datos generados por la función "init_mycontroller.m" y el valor de la glucosa medida por el sensor que se ha escogido, este valor se lo pasará la función principal de la simulación.

Y devuelve como resultado el valor de las acciones de control que calcula con las ecuaciones a parte de los parámetros actualizados.

6.2. CARACTERÍSTICAS CONTROLADOR ARDUINO

Una vez explicadas las características de un controlador de Matlab se va a pasar a comentar cual ha sido esa adaptación que se ha hecho y que cosas se han cambiado para conseguirlo. Este controlador también dispone de las funciones "init_mycontroller.m" y "run_mycontroller.m", pero además hay que añadir los códigos que se introducen en las placas.

Una de las placas funcionará como un monitor de la glucosa recogida por el sensor, mientras que la otra será la encargada de realizar las ecuaciones que anteriormente se situaban en el "run_mycontroller.m".

6.2.1. INIT_MYCONTROLLER.M

En el caso de esta función su contenido se va a ampliar, además de todas las condiciones iniciales y variables que se calculaban en un controlador típico de Matlab, hay que llevar a cabo la creación de los objetos bluetooth y la conexión con los módulos bluetooth.

Además de todo ellos se debe realizar el envío de todas las variables al módulo bluetooth de la placa encargada de realizar las ecuaciones del controlador, ya que les harán falta. Esta transmisión se ha realizado utilizando los comandos fprintf y fwrite, puesto que fwrite solamente puede realizar envíos de números enteros menores de 255, puesto que envía 1 byte de información.

La transmisión se ha realizado separando la parte entera de la decimal, la parte entera se va a enviar mediante el comando fprintf, puesto que su valor podría sobrepasar el rango del fwrite. La parte decimal, se va a seccionar de 2 en 2 decimales, para que puedan ser enviados mediante el comando fwrite, hasta un total de 4 envíos, para evitar que se produzca overflow dentro de las placas Arduino. Aparte, se envía un aviso a la placa que lo va a recibir para que esté preparada para ello, como se ve en la Figura 16.

Hay que aclarar que el comando fprintf envía las viables como si fueran cadenas de caracteres, por lo que después deberán ser transformadas para su utilización numérica dentro de las placas Arduino.

```
intKp = floor(controller.specific.Kp);
fwrite(B,1);
fprintf(B,'%f\n',intKp);
pause(0.1);
a=controller.specific.Kp-intKp;
a=a*100;
for j=0:1:2
decKp=floor(a);
fwrite(B,12);
fwrite(B,decKp);
pause(0.1);
a=a-decKp;
a=a*100;
end
decKp4=floor(a);
fwrite(B,15);
fwrite(B,decKp4);
pause(0.1);
```

Figura 15. Estructura envío Matlab - Arduino

6.2.2. RUN_MYCONTROLLER.M

La función "run_mycontroller.m" ha sufrido muchos cambios con respecto a la de un controlador normal de Matlab, puesto que las ecuaciones del controlador ya no residen en esta función, sino que ahora se van a encontrar en una de las placas Arduino.

Dentro de ellas se realiza el envío de la glucosa, al Arduino el que cumple la función de monitor de glucosa continua, con la misma estructura que el envío de las constantes del controlador.

Para terminar, se leen las acciones de control resultantes y de la glucosa del paso precedente, que es el parámetro del controlador que se va actualizando, ya que la función devuelve como solución estos valores a la función principal.

La lectura de los datos se realiza con el comando fscanf, puesto que los datos que se van a leer no son enteros sino doubles y después se les realiza una corrección por la estructura que se les ha dado a las variables dentro de las placas Arduino.

```
fwrite(B,6);
pause(0.1);
IIRt=fscanf(B,'%lf');
pause(0.3);
IIRt=IIRt/1000000.0;
```

Figura 16. Estructura recepción datos Matlab

6.2.3. ARDUINO ENCARGADO DEL MONITOR DE GLUCOSA CONTINUA

Una de las placas Arduino va a ser el monitor de glucosa proveniente del sensor de glucosa continua, esta placa recibirá los valores de glucosa desde la función "run_mycontroller.m" como se ha explicado anteriormente.

Esta placa Arduino solo sirve como paso intermedio para la glucosa, puesto que esta va a ser enviada a la otra placa para poder utilizar este valor en las ecuaciones.

En esta placa se utilizan los comandos Serialx.read(), Serialx.write(y) y Serialx.println(y), para realizar la lectura de datos, y el envío de ellos en función de si se trata de valores enteros o decimales, respectivamente. En este caso, estas operaciones se realizan de forma consecutiva, como se observa en la Figura 18, porque como se ha dicho esta placa solo funciona de paso intermedio en el camino de la glucosa en sangre.

```
i=Serial2.read();
if(i==1){
  delay(15);
  Glucoses=Serial2.readStringUntil('\n');
  intGlucose=Glucoses.toDouble();
  Serial1.write(1);
  Serial1.println(intGlucose);
  delay(100);
  }
```

Figura 17. Estructura recepción y envío de datos entre placas

6.2.4. ARDUINO ENCARGADO DEL CONTROLADOR

La segunda de las placas va a ser la encargada de realizar los cálculos de las ecuaciones del controlador, que han sido traducidas desde Matlab sin apenas variaciones en sus estructuras.

El grueso del código lo constituyen las recepciones de todas las variables provenientes de Matlab y de la primera placa Arduino, y el envío de los resultados obtenidos nuevamente a Matlab.

En esta placa también se da forma a las variables para estructurarla como se ha comentado anteriormente, con los 6 primeros decimales por delante de la coma y los 2 siguientes por detrás de ella. El envío de datos se realiza de la misma forma que en la otra placa, pero aquí únicamente se usara el comando Serialx.println(y), ya que las variables van a ser todas con parte decimal.

```
 j=Serial1.read();
 if(j==1){
   delay(3);
   Glucoses=Serial1.readStringUntil('\n');
   Glucose=Glucoses.toDouble();
 }if(j==2){
   delay(20);
   decGlucose=Serial1.read();
   Glucose=Glucose*pow(10,2)+decGlucose;
 }
if(j==5){
   delay(20);
   decGlucose4=Serial1.read();
   Glucose=Glucose+decGlucose4/pow(10,2);
}
```

Figura 18. Estructura recepción de datos y creación de las variables

CAPÍTULO 7. RESULTADOS

Para corroborar el buen funcionamiento de la adaptación que se ha realizado para que sea utilizada en las placas Arduino se han realizado una serie de experiencias en las que se ha sometido al simulador a diversas condiciones para comprobar que el resultado es aceptable en cualquier situación sobre la que tenga que actuar.

Para ello, primero se ha realizado dichos ensayos con el controlador interno de Matlab "PID2", para poder tener una referencia sobre la que contrastar los resultados obtenidos las soluciones arrojadas por el controlador de Arduino.

En total se han llevado a cabo 10 ensayos con cada simulador, los 9 primeros son ensayos de una sola comida y se han realizado para comprobar su funcionamiento ante las variaciones de la cantidad de comida ingerida y el posible error por parte del paciente en el suministro de la insulina; y la última es un ensayo de un día completo con el que se quiere comprobar su buena respuesta ante la realización de diversas comidas en una misma simulación.

7.1. ENSAYOS

Antes de mostrar las gráficas y los datos obtenidos es necesario aclarar los valores en los que se miden los ejes de las gráficas, ya que al mostrarse 6 gráficas en una misma figura los títulos de los ejes se solapan entre ellos. Por ellos los títulos y las unidades de las gráficas en orden descendente son:

1. Glucose (mg/dl), muestra el valor de la glucosa en cada instante
2. Insulin delivery (U/min), muestra las unidades por minuto suministradas de la infusión continua de insulina
3. Insulin delivery (U/min), hace referencia al bolo de insulina que se inyecta antes de cada comida
4. Glucagon delivery (mg/Kg/min), se refiere al glucagón suministrado de forma continua en cada iteración
5. Glucagon delivery (mg/Kg/min), muestra el glucagón suministrado debido a una bajada brusca de azúcar en sangre
6. Rescue carbs (g), hace referencia a los gramos de carbohidratos de rescate utilizados en caso de sufrir alguna hipoglucemia

Esta estructura se repite para todas las gráficas y todas ellas se muestran enfrentadas al tiempo medido en horas.

Los tres primeros ensayos vas a mostrar la respuesta ante una comida correspondiente a 40 gramos de carbohidratos variando el bolo de insulina que se inyecta entre la mitad del óptimo, el óptimo y el doble, respectivamente.

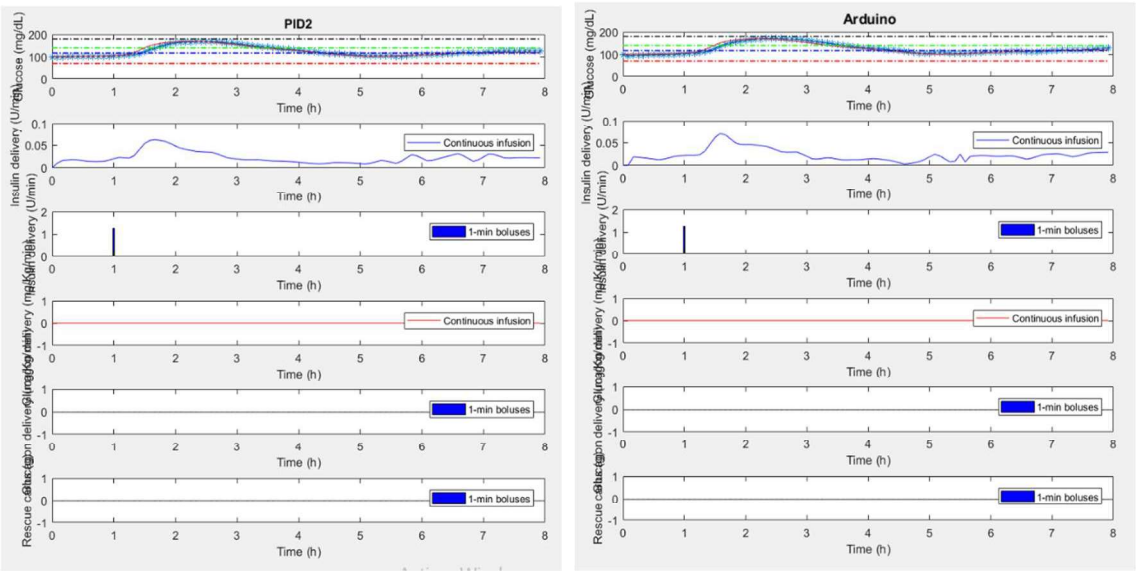

Figura 20. Ensayo con 40 gramos de carbohidratos y mitad del bolo de insulina óptimo con controlador PID2 y Arduino

Entre la diversidad de datos que se han obtenido de las dos simulaciones se van a destacar los más relevantes para poder mostrar el funcionamiento del controlador, por parte del controlador "PID2", se ha obtenido:

- Valor medio de glucosa monitorizada: 124.3477 mg/dl
- Total de insulina suministrada: 11.8431 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 76.04%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 100%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Y los resultados obtenidos por el controlador Arduino son los siguientes:

- Valor medio de glucosa monitorizada: 124.0405 mg/dl
- Total de insulina suministrada: 11.9328 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 75%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 100%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Calculando los errores en tanto por cien del valor medio de glucosa y del total de insulina suministrada obtenemos:

- Error glucosa media: 0.247%
- Error total insulina liberada: 0.757%

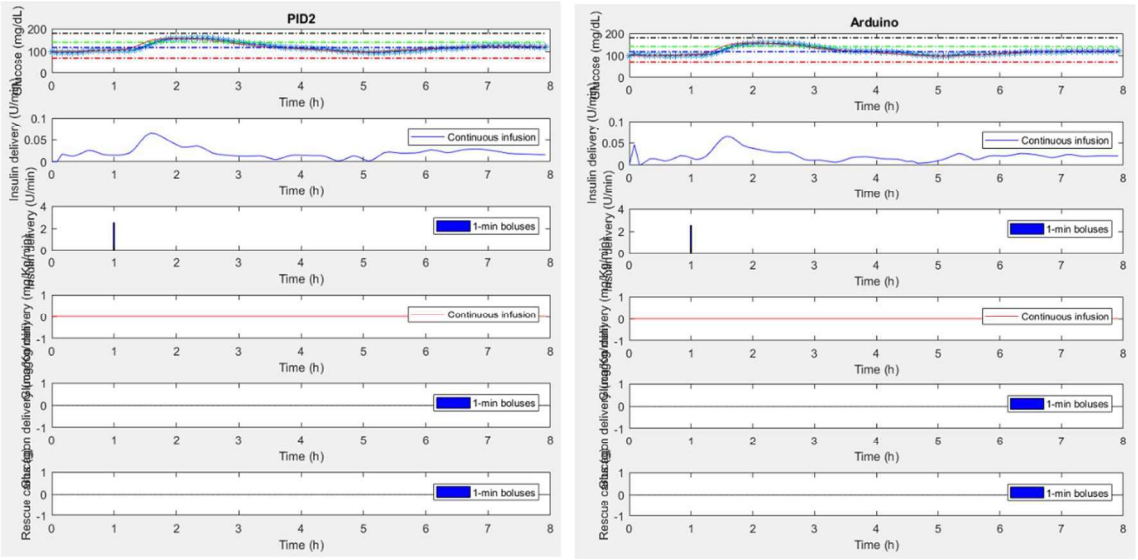

Figura 21. Ensayo con 40 gramos de carbohidratos y bolo de insulina óptimo con controlador PID2 y Arduino

Los datos obtenidos por la simulación con el controlador "PID2" son los siguiente:

- Valor medio de glucosa monitorizada: 118.7273 mg/dl
- Total de insulina suministrada: 12.2156 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 82.29%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 100%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

En el caso de la utilización del controlador diseñado en Arduino son:

- Valor medio de glucosa monitorizada: 118.8991 mg/dl
- Total de insulina suministrada: 12.2475 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 83.33%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 100%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

En este caso los errores serán:

- Error glucosa media: 0.145%
- Error total insulina liberada: 0.261%

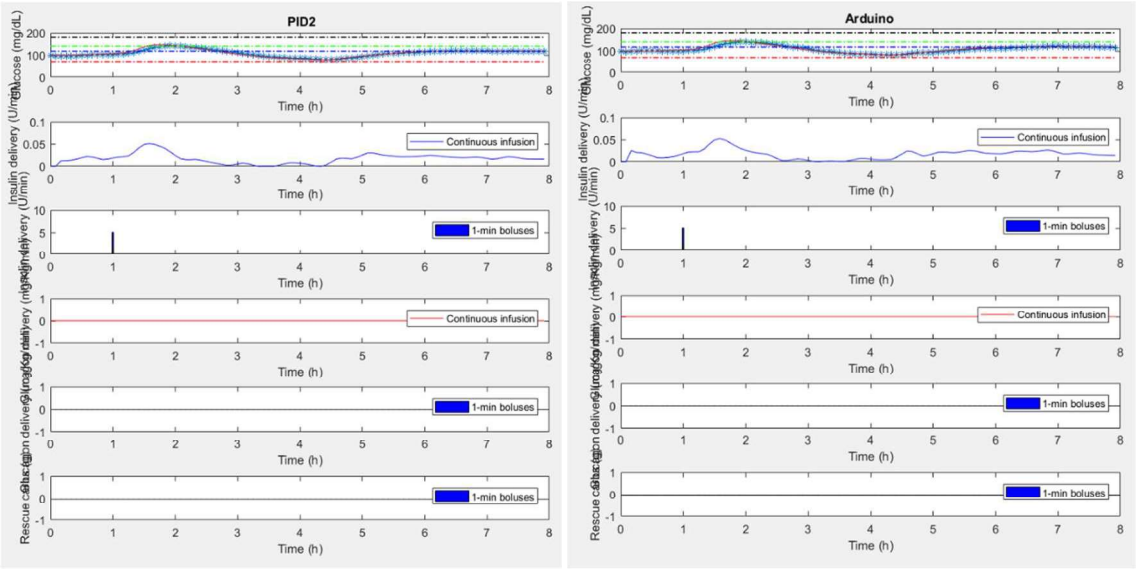

Figura 22. Ensayo con 40 gramos de carbohidratos y doble del bolo de insulina óptimo con controlador PID2 y Arduino

En este caso los datos proporcionados por el controlador "PID2" son:

- Valor medio de glucosa monitorizada: 108.5579 mg/dl
- Total de insulina suministrada: 13.2128 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 95.83%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 100%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Por otro lado al utilizar el controlado de Arduino obtenemos:

- Valor medio de glucosa monitorizada: 108.1608 mg/dl
- Total de insulina suministrada: 13.1281 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 100%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 100%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Para este último caso con 40 gramos de carbohidratos obtenemos unos errores de:

- Error glucosa media: 0.366%
- Error total insulina liberada: 0.641%

El siguiente paso que se ha tomado ha sido incrementar un poco la ingesta de carbohidratos, realizando los mismos experimentos en cuanto al bolo de insulina, para los tres siguiente se ha escogido una ingesta de carbohidratos de 70 gramos.

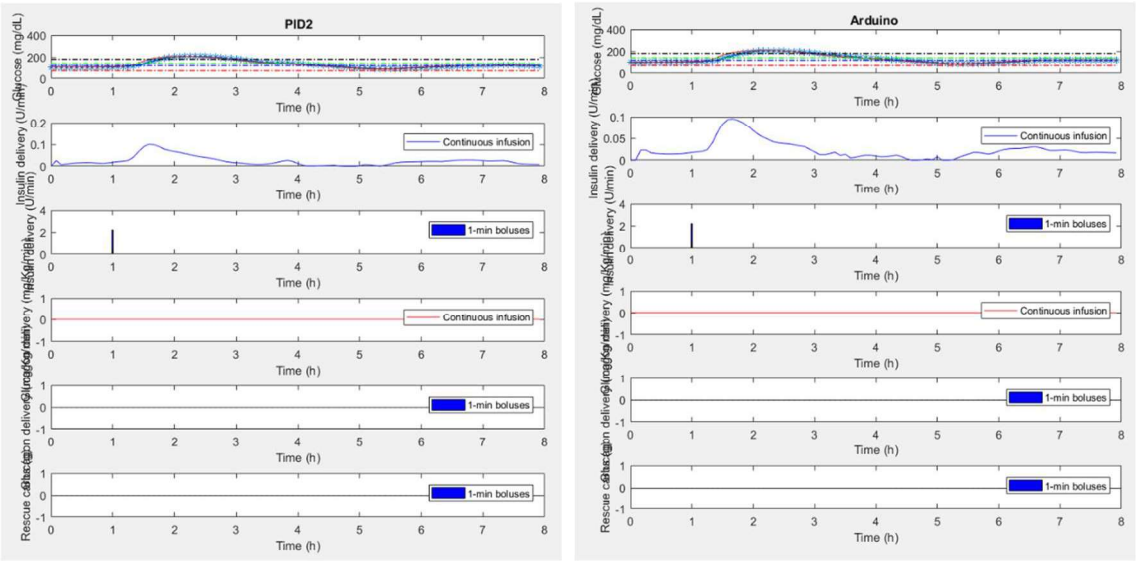

Figura 23. Ensayo con 70 gramos de carbohidratos y mitad del bolo de insulina óptimo con controlador PID2 y Arduino

Los resultados ofrecidos por el controlado "PID2" son los siguientes:

- Valor medio de glucosa monitorizada: 129.5285 mg/dl
- Total de insulina suministrada: 13.2859 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 68.75%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 84.38%
- Porcentaje del tiempo por encima de 180 mg/dl: 15.63%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Y el controlador de Arduino nos aporta los siguiente:

- Valor medio de glucosa monitorizada: 128.7491 mg/dl
- Total de insulina suministrada: 13.3961 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 70.83%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 84.38%
- Porcentaje del tiempo por encima de 180 mg/dl: 15.63%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

En dicho caso los errores que se obtienen en tanto por cien son:

- Error glucosa media: 0.602%
- Error total insulina liberada: 0.829%

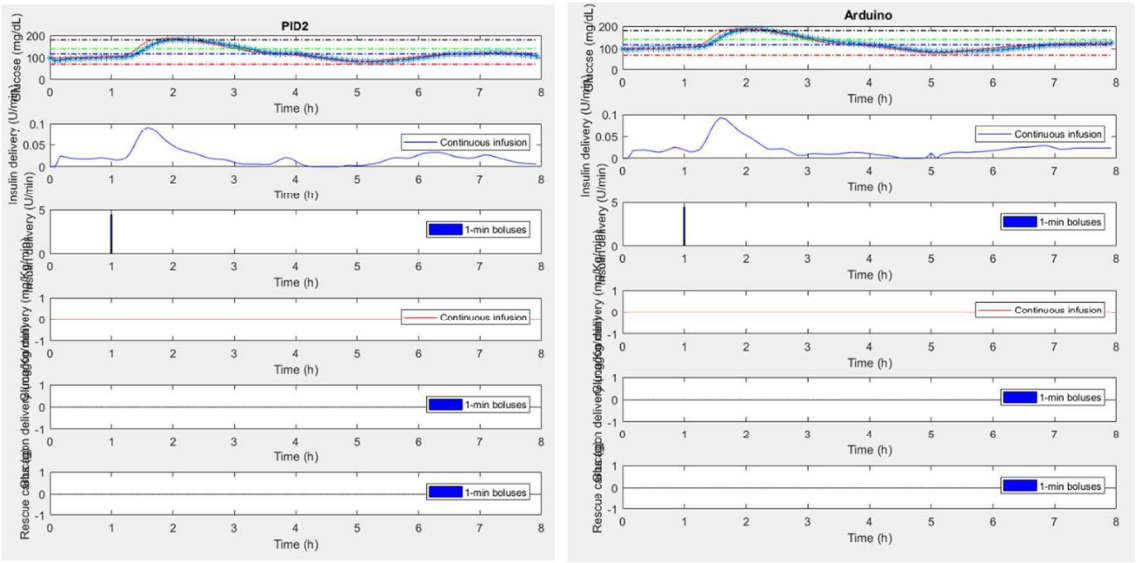

Figura 24. Ensayo con 70 gramos de carbohidratos y bolo de insulina óptimo con controlador PID2 y Arduino

En este ensayo los resultados que arroja el controlador integrado en Matlab son:

- Valor medio de glucosa monitorizada: 118.6623 mg/dl
- Total de insulina suministrada: 14.0690 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 79.17%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 95.83%
- Porcentaje del tiempo por encima de 180 mg/dl: 4.17%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Y los que se obtienen de la simulación con las placas Arduino son:

- Valor medio de glucosa monitorizada: 120.1890 mg/dl
- Total de insulina suministrada: 14.4109 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 79.17%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 93.75%
- Porcentaje del tiempo por encima de 180 mg/dl: 6.25%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Obteniendo unos errores de:

- Error glucosa media: 1.287%
- Error total insulina liberada: 2.430%

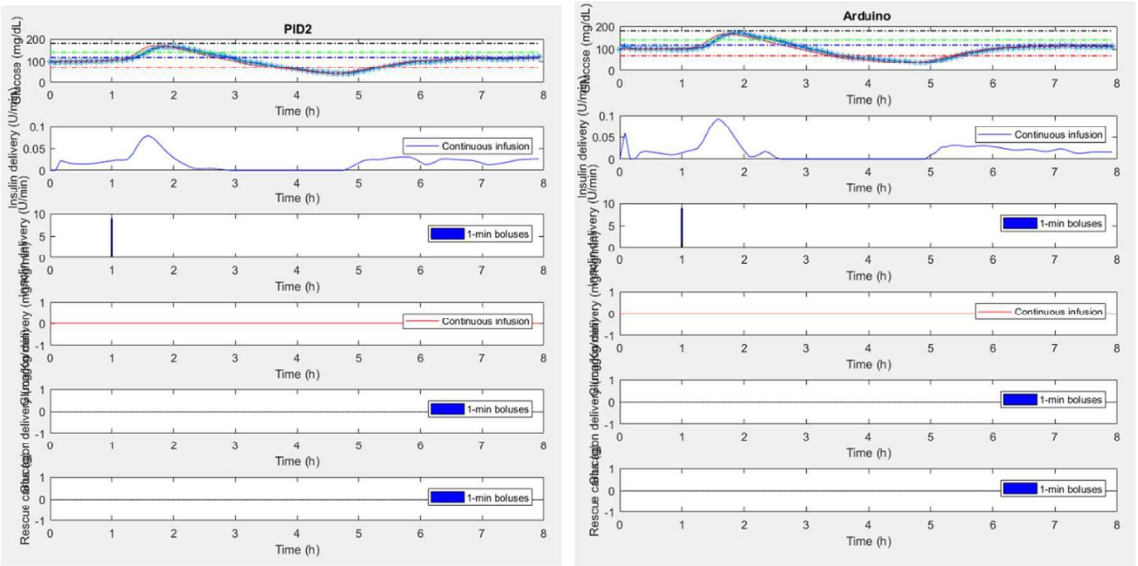

Figura 25. Ensayo con 70 gramos de carbohidratos y doble del bolo de insulina óptimo con controlador PID2 y Arduino

El controlador "PID2" ofrece los siguientes resultados:

- Valor medio de glucosa monitorizada: 100.4547 mg/dl
- Total de insulina suministrada: 17.0736 unidades
- Porcentaje del tiempo por debajo de 50 mg/dl: 9.38%
- Porcentaje del tiempo por debajo de 54 mg/dl: 10.42%
- Porcentaje del tiempo por debajo de 60 mg/dl: 14.58%
- Porcentaje del tiempo por debajo de 70 mg/dl: 19.79%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 68.75%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 80.21%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 1 caso

Y Arduino no aporta los siguientes:

- Valor medio de glucosa monitorizada: 98.0384 mg/dl
- Total de insulina suministrada: 16.7999 unidades
- Porcentaje del tiempo por debajo de 50 mg/dl: 10.42%
- Porcentaje del tiempo por debajo de 54 mg/dl: 13.54%
- Porcentaje del tiempo por debajo de 60 mg/dl: 17.71%
- Porcentaje del tiempo por debajo de 70 mg/dl: 23.96%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 63.54%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 76.04%
- Porcentaje del tiempo por encima de 180 mg/dl: 0%
- Casos de hipoglucemia: 1 caso

Dando lugar a unos errores de:

- Error glucosa media: 2.306%
- Error total insulina liberada: 1.603%

Para completar los casos de una sola comida se ha decidido por incrementar un poco más los gramos de carbohidratos de las comidas hasta un valor 100 gramos, realizando los mismos ensayos en cuanto al bolo de insulina.

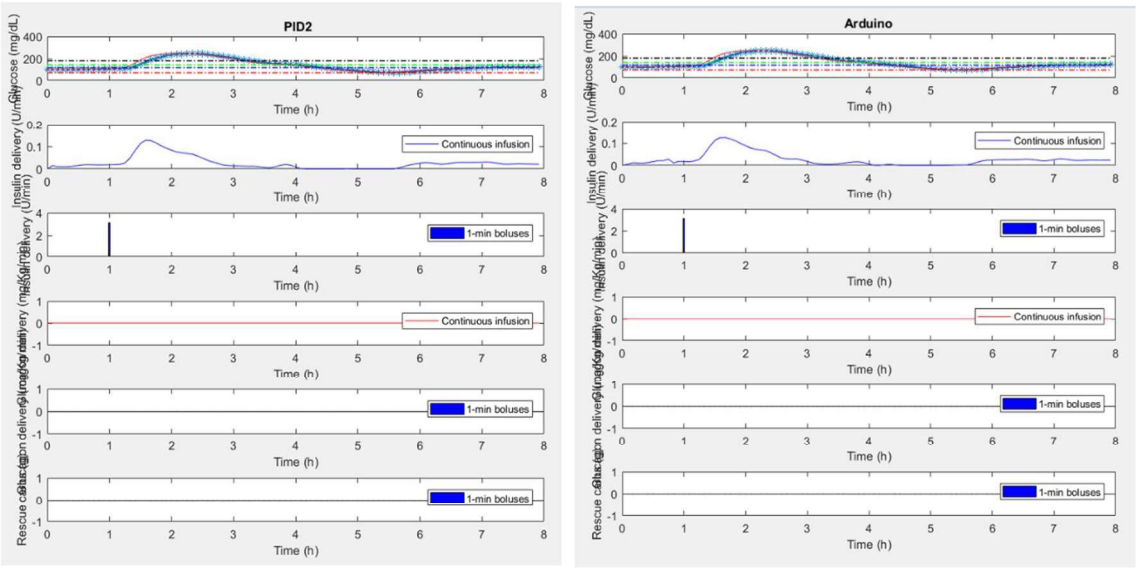

Figura 26. Ensayo con 100 gramos de carbohidratos y mitad del bolo de insulina óptimo con controlador PID2 y Arduino

Los resultados obtenidos en la utilización del controlador "PID2" son:

- Valor medio de glucosa monitorizada: 134.5722 mg/dl
- Total de insulina suministrada: 15.4489 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 66.67%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 78.13%
- Porcentaje del tiempo por encima de 180 mg/dl: 21.88%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

En el caso de la simulación con las placas Arduino se obtienen los siguientes datos:

- Valor medio de glucosa monitorizada: 134.3777 mg/dl
- Total de insulina suministrada: 15.4717 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 67.71%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 78.13%
- Porcentaje del tiempo por encima de 180 mg/dl: 21.88%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Calculando los errores con los datos mencionados se obtienen unos errores de:

- Error glucosa media: 0.145%
- Error total insulina liberada: 0.148%

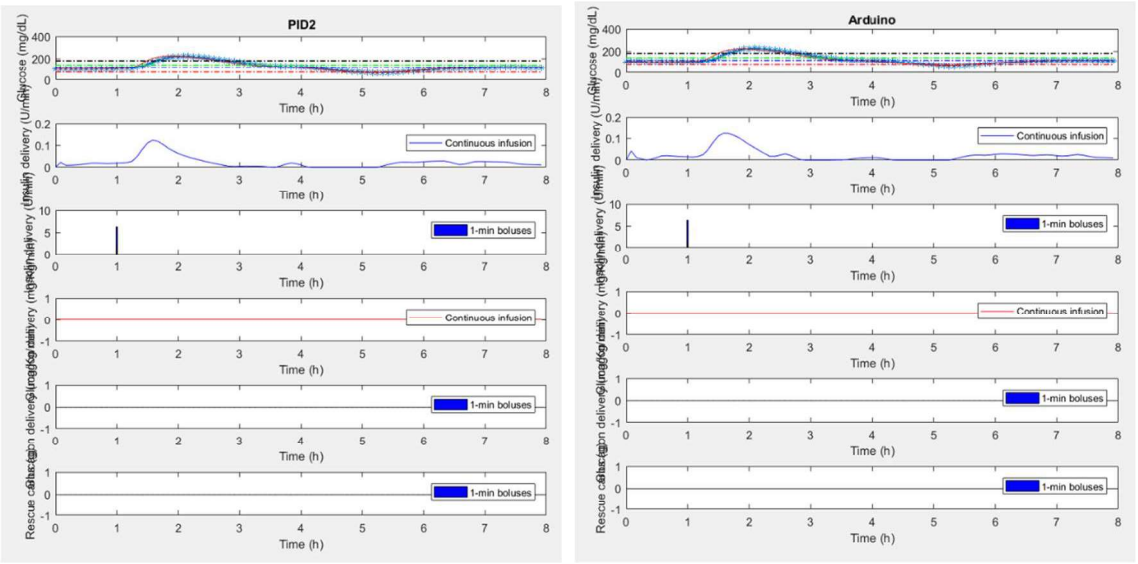

Figura 27. Ensayo con 100 gramos de carbohidratos y bolo de insulina óptimo con controlador PID2 y Arduino

Del ensayo con el controlador de Matlab se obtienen los siguientes resultados:

- Valor medio de glucosa monitorizada: 121.6937 mg/dl
- Total de insulina suministrada: 16.6419 unidades
- Porcentaje del tiempo por debajo de 60 mg/dl: 0%
- Porcentaje del tiempo por debajo de 70 mg/dl: 6.25%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 69.79%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 79.17%
- Porcentaje del tiempo por encima de 180 mg/dl: 14.58%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 1 caso

Y cuando se realiza con el controlador de Arduino obtenemos:

- Valor medio de glucosa monitorizada: 120.5064 mg/dl
- Total de insulina suministrada: 16.6151 unidades
- Porcentaje del tiempo por debajo de 54 mg/dl: 0%
- Porcentaje del tiempo por debajo de 60 mg/dl: 1.04%
- Porcentaje del tiempo por debajo de 70 mg/dl: 8.33%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 68.75%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 77.08%
- Porcentaje del tiempo por encima de 180 mg/dl: 14.58%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 1 caso

Datos de los cuales obtenemos unos errores de:

- Error glucosa media: 0.976%
- Error total insulina liberada: 0.161%

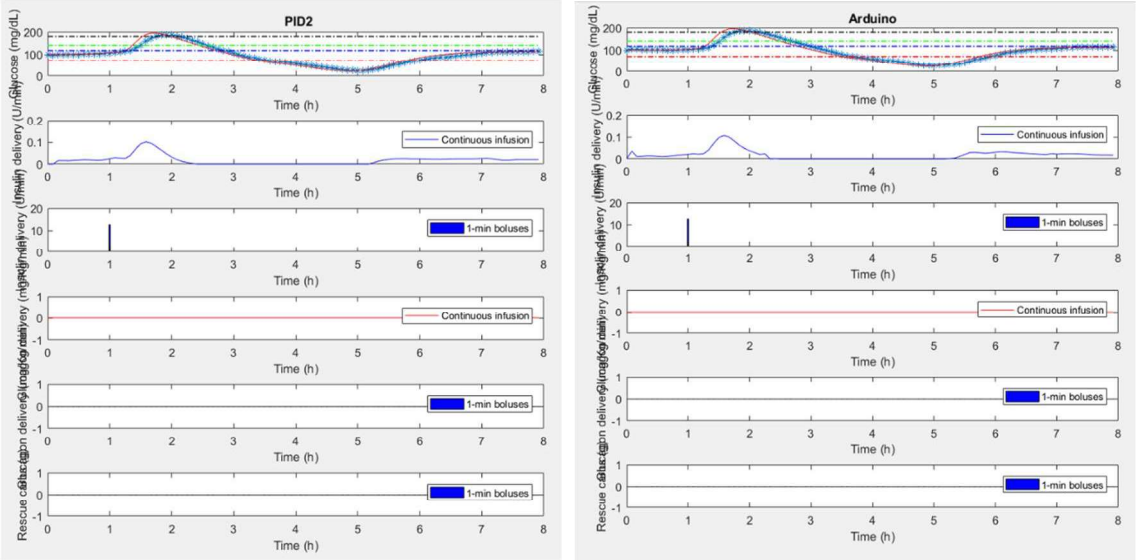

Figura 28. Ensayo con 100 gramos de carbohidratos y doble del bolo de insulina óptimo con controlador PID2 y Arduino

Los datos ofrecidos por el controlador "PID2" son:

- Valor medio de glucosa monitorizada: 93.8601 mg/dl
- Total de insulina suministrada: 20.7284 unidades
- Porcentaje del tiempo por debajo de 50 mg/dl: 17.71%
- Porcentaje del tiempo por debajo de 54 mg/dl: 19.79%
- Porcentaje del tiempo por debajo de 60 mg/dl: 23.96%
- Porcentaje del tiempo por debajo de 70 mg/dl: 30.21%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 56.25%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 64.58%
- Porcentaje del tiempo por encima de 180 mg/dl: 5.21%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 1 caso

Y los que se consiguen con la utilización de las placas Arduino son:

- Valor medio de glucosa monitorizada: 97.2602 mg/dl
- Total de insulina suministrada: 21.0160 unidades
- Porcentaje del tiempo por debajo de 50 mg/dl: 16.67%
- Porcentaje del tiempo por debajo de 54 mg/dl: 18.75%
- Porcentaje del tiempo por debajo de 60 mg/dl: 21.88%
- Porcentaje del tiempo por debajo de 70 mg/dl: 27.08%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 58.33%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 66.67%
- Porcentaje del tiempo por encima de 180 mg/dl: 6.25%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 1 caso

Los errores que se obtiene con estos datos son:

- Error glucosa media: 3.623%
- Error total insulina liberada: 1.387%

Para finalizar se ha realizado un ensayo de un día de duración en la que intervienen varias comidas, siendo una de 40 gramos a las 7 horas, una de 70 a las 14 horas y otra de 60 a las 21.

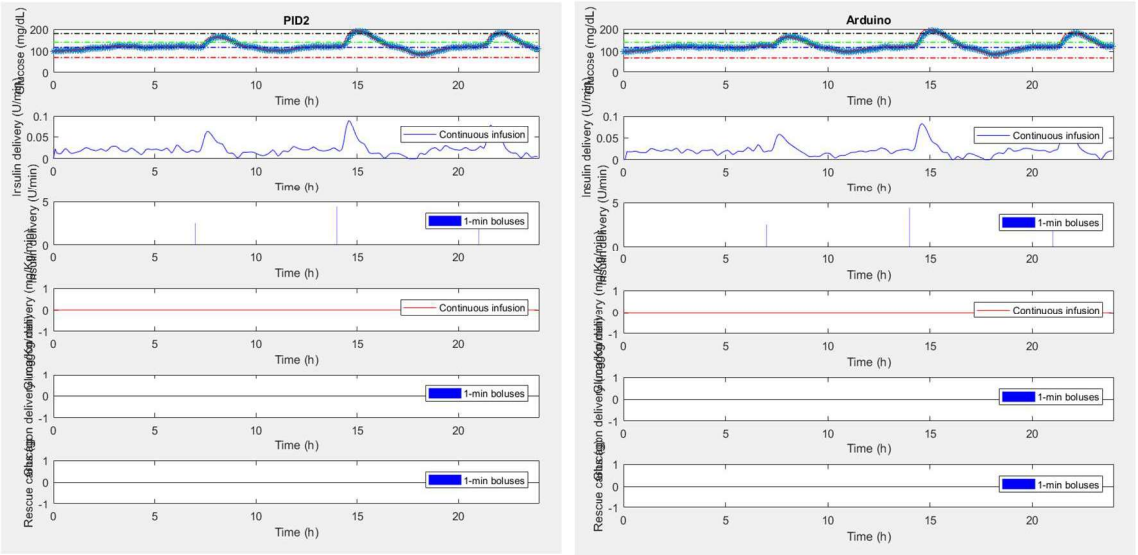

Figura 19. Ensayo de un día completo y bolo de insulina óptimo con controlador PID2 y Arduino

En este caso, los datos que nos ha aportado el simulador con el controlador "PID2" son:

- Valor medio de glucosa monitorizada: 124.9258 mg/dl
- Total de insulina suministrada: 41.4080 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 79.86%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 95.49%
- Porcentaje del tiempo por encima de 180 mg/dl: 4.51%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Y los que se han obtenido de la utilización del bucle con los Arduino son:

- Valor medio de glucosa monitorizada: 124.9098 mg/dl
- Total de insulina suministrada: 41.6516 unidades
- Porcentaje del tiempo por debajo de 70 mg/dl: 0%
- Porcentaje de tiempo entre 70 y 140 mg/dl: 79.17%
- Porcentaje de tiempo entre 70 y 180 mg/dl: 95.83%
- Porcentaje del tiempo por encima de 180 mg/dl: 4.17%
- Porcentaje del tiempo por encima de 250 mg/dl: 0%
- Casos de hipoglucemia: 0 casos

Por lo que los errores que se obtienen de estos datos son:

- Error glucosa media: 0.013%
- Error total insulina liberada: 0.588%

7.2. RESUMEN

Los datos que nos han arrojado todas las experiencias comentadas anteriormente pueden resumirse en la siguiente tabla:

Ensayos	PID2 glucosa (mg/dl)	Arduino glucosa (mg/dl)	PID2 insulina (ud.)	Arduino insulina (ud.)	% entre 70 y 140 mg/dl PID2	% entre 70 y 140 mg/dl Arduino	Error glucosa (%)	Error insulina (%)
$40g + 1/2\ bolo$	124.3477	124.0405	11.8431	11.9328	76.04	75.00	0.247	0.757
$40g + \ bolo$	118.7273	118.8991	12.2156	12.2475	82.29	83.33	0.145	0.261
$40g + 2\ bolos$	108.5579	108.1608	13.2128	13.1281	95.83	100.00	0.366	0.641
$70g + 1/2\ bolo$	129.5285	128.7491	13.2859	13.3961	68.75	70.83	0.602	0.829
$70g + bolo$	118.6623	120.1890	14.0690	14.4109	79.17	79.17	1.287	2.430
$70g + 2\ bolos$	100.4547	98.0384	17.0736	16.7999	68.75	63.54	2.306	1.603
$100g + 1/2\ bolo$	134.5722	134.3777	15.4489	15.4717	66.67	67.71	0.145	0.148
$100g + \ bolo$	121.6937	120.5064	16.6419	16.6151	69.79	68.75	0.976	0.161
$100g + 2\ bolos$	93.8601	97.2602	20.7284	21.0160	56.25	58.33	3.623	1.387
Día completo	124.9258	124.9098	41.4080	41.6516	79.86	79.17	0.013	0.588

Tabla 1. Resumen resultados obtenidos

Como se puede observar en la Tabla 1 los valores de glucosa disminuyen cuando se aplica mayor cantidad de bolo de insulina, dándose el mínimo cuando se da la comida con mayor cantidad de carbohidratos, ya que el exceso de insulina es el máximo en el miso caso.

En cuanto a los porcentajes de tiempo en normoglucemia, se puede ver cuándo se puede ver como los mejores resultados se dan para el bolo de insulina óptimo, como cabría esperar, a excepción de cuando se da la comida de 40g, ya que ese pequeño exceso del bolo impide que suba por encima de 14, pero en el caso de usar el óptimo no es preocupante, ya que después de una comida es normal superar dicho valor.

En cuanto a los errores se pueden ver unos errores no muy elevados con un máximo de un 3.623% esta divergencia entre las soluciones se debe a que aunque se haya querido evitar toda posibilidad de introducir elementos randomizados, el propio sensor que recoge los datos de glucosa introduce un ruido dentro de esta, haciendo que los resultados puedan variar, y, también se percibe que los errores aumentan cunado el valor de la glucosa baja, ya que este ruido tiene mayor influencia en proporción a cuando se dan valores más elevados.

CAPÍTULO 8. CONCLUSIONES

Una vez llevados a cabo los ensayos ya se puede determinar si el funcionamiento del controlador que se ha traducido para ser integrado en las placas Arduino es el correcto o por el contrario es erróneo. Como se dijo para la realización de estos ensayos se eliminó cualquier posibilidad de entrada de valores random, ya que si se permitiera dicha entrada no se podrían igualar en su totalidad los resultados, puesto que estos valores aleatorios variarían con cada experiencia.

En cuanto a los ensayos realizados, se puede observar que la gráfica correspondiente al valor continuo de la glucosa obtenida por el sensor, situada en primer lugar, no sufre grandes variaciones entre los ensayos realizados por el controlador "PID2" integrado en Matlab y los llevados a cabo con el controlador de las placas Arduino, incluso en aquellos casos en los que se produce una mayor divergencia en los datos resultantes, como son el ensayo con 100 g de carbohidratos y el doble de la insulina necesaria, y el de 70 g de y el doble de la insulina necesaria, dicha divergencia apenas tiene efecto sobre la gráfica.

No es así con las gráficas que relacionan la insulina suministrada continuamente frente al tiempo, donde en todos los casos se produce alguna variación entre las gráficas correspondientes a los dos controladores, esto se debe a que aunque como se ha dicho se han eliminado todos los factores aleatorios posibles para poder tener una comparación más limpia, todavía existe una pequeña parte de aleatoriedad introducida por el sensor empleado, que introduce un poco de ruido en su señal, y debido a ello es prácticamente imposible obtener las mismas soluciones en ambos controladores.

También hay que decir que este ruido también afecta a la primera gráfica pero al tener un orden de magnitud más elevado su efecto no es tan notable como con el caso de la insulina suministrada de forma continua, aunque por ello también se observa que cuando los valores de glucosa son más reducidos su efecto es mayor, de ahí que los máximos errores se produzcan en esos casos.

En cuanto a los valores de los porcentajes de tiempo en las distintas zonas, valor medio de glucosa y el total de insulina, los resultado entran dentro de los valores lógicos que cabría esperar, incrementado la insulina con el tamaño de la comida y la dosis de esta, incrementando la glucosa con los carbohidratos y disminuyendo cuando incrementa la dosis de insulina, estando más tiempo en valores elevados cuando la comida aumenta y la dosis decrementa, e incrementando el tiempo en valores bajos cuando aumenta la comida y la dosis de insulina es la mitad de la óptima, ya que la dosis de insulina incrementa con la comida y se produciría un mayor déficit cuando se suministra la mitad.

En definitiva, puede considerarse que la adaptación del código de Matlab a las placas Arduino está bien realizado, el funcionamiento de los módulos bluetooth es el correcto y la implementación del conjunto es la adecuada para la finalidad que se busca.

CAPÍTULO 9. TRABAJOS FUTUROS

En este apartado se va a comentar una serie de puntos con mejoras que se pueden tener en cuenta para un posible trabajo futuro que mejore o vaya un paso más allá que el actual, estos puntos son:

- La investigación de otros métodos de envió de datos que mejoren el actual realizando el mismo de una forma más rápida y sencilla, agilizando de esta forma el proceso, ya que actualmente el tiempo requerido por las placas Arduino es bastante elevado tardando aproximadamente 14 minutos en realizar un ensayo de un día completo el cual los controladores de Matlab realizar en menos de 15 segundos
- La implementación de una interfaz de usuario dentro de Matlab de forma que el simulador se accesible a un mayor número de usuarios y su experiencia con él sea más satisfactoria y sencilla
- Posibilidad de implementación utilizando módulos wifi los cuales permiten conexión con más de un dispositivo al mismo tiempo, no como los módulos bluetooth que necesitan emparejamientos dos a dos, por lo que se podría realizar con únicamente dos módulos, aunque se necesitaría la posibilidad de conexión wifi, que sería un punto en contra de este método
- Implementación de un controlador de mayor complejidad dentro de las placas Arduino para comprobar que su funcionamiento también puede ser válido mejorando de esta forma el control realizado
- Posibilidad de utilización de algún otro tipo de software que tenga la misma funcionalidad de Arduino, pero que trabaje a 64 bits como lo realiza Matlab, para hacer que la comunicación mejore

CAPÍTULO 10. BIBLIOGRAFÍA

- Arduino and programming. *http://crackeconcept.blogspot.com/2014/03/arduino-and-matlab-interfacing-via.html*
- Configuración de módulos Bluetooth HC-05. *http://tienda.rmjelectronics.com/index.php?controller=attachment&id_attachment=1*
- Comunicación entre dos módulos bluetooth HC-05 paso a paso con Arduino. *https://pgcarduino.blogspot.com/p/comunicacion-entre-modulos-bluetooth-hc.html*
- Menú de comandos AT para configurar HC-06. *http://www.infotronikblog.com/2017/09/arduino-menu-de-comandos-at-para.html*
- Conexión para la vida. *http://web.diabetes.org/link/link_for_life_sp/main_sp.html*
- Glucosa en la sangre: niveles. *http://www.netdoctor.es/articulo/niveles-glucosa-en-sangre*
- 1 de cada 11 personas en el mundo ya tienen diabetes, advierte la OMS. *http://www.bbc.com/mundo/noticias/2016/04/160406_salud_diabetes_oms_lb*
- ¿Cómo saber si tengo diabetes? Principales síntomas que nos advierten. Bertrand Redager. *https://psicologiaymente.net/salud/como-saber-si-tengo-diabetes-sintomas*
- Bomba de insulina. *http://www.fundaciondiabetes.org/infantil/185/bomba-de-insulina-ninos*
- Imagen autoinyector. *https://es.freeimages.com/photo/diabetic-needle-1312865*
- Insulina, medicamentos y otros tratamientos para la diabetes. *https://www.niddk.nih.gov/health-information/informacion-de-la-salud/diabetes/informacion-general/insulina-medicamentos-tratamientos*
- Imagen bomba de insulina. *http://flaviabonifazi.com.br/artigos/bomba-infusao-insulina/*
- Tipos de diabetes: riesgos, características y tratamientos. Miguel Zahonero Bermejo. *https://psicologiaymente.net/salud/tipos-diabetes*
- Aspectos genéticos de la diabetes. *http://www.diabetes.org/es/informacion-basica-de-la-diabetes/aspectos-genticos-de-la-diabetes.html?loc=db-es-slabnav*
- Diabetes. *https://cuidateplus.marca.com/enfermedades/digestivas/diabetes.html*
- Pruebas y diagnóstico de la diabetes. *https://www.niddk.nih.gov/health-information/informacion-de-la-salud/diabetes/informacion-general/pruebas-diagnostico*
- Diabetes. *http://www.novonordisk.com.ar/pacientes/diabetes.html*
- Glucosa: ¿Qué es? *https://quierocuidarme.dkvsalud.es/salud-para-todos/glucosa-que-es*
- ¿qué es la insulina? *http://fmdiabetes.org/que-es-la-insulina/*
- El glucagón, ¿para qué sirve? *https://www.enbuenasmanos.com/el-glucagon*

- Sintomas y causas de la diabetes. *https://www.niddk.nih.gov/health-information/informacion-de-la-salud/diabetes/informacion-general/sintomas-causas*
- ¿Qué es Arduino? *http://arduino.cl/que-es-arduino/*
- Conplicaciones. *http://www.diabetes.org/es/vivir-con-diabetes/complicaciones/*
- Hiperglucemia – cuidados personales. *https://medlineplus.gov/spanish/ency/patientinstructions /000332.htm*
- Println() *https://www.arduino.cc/en/Serial/Println*

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UN EMULADOR DE SISTEMA DE CONTROL AUTOMÁTICO DE GLUCOSA EN SANGRE SOBRE UN SISTEMA HARDWARE DE BAJO COSTE

Presupuesto

Curso Académico: 2017-18

CUADRO Nº1: PRECIOS DE LA MANO DE OBRA

En este caso la mano de obra empleada en la realización del proyecto se un alumno de ingeniería industrial, por lo que para calcular el coste de la mano de obra cogeremos el salario medio de una persona recientemente graduada en ingeniería industrial, el cual varía entre los 1500€ y los 1800€, en nuestro caso escogeremos 1500€ por el hecho de que todavía no se ha adquirido el título de ingeniero industrial.

Por lo tanto el sueldo anual que incluye las 14 pagas que se realizan, 12 mensuales y dos pagas extra, asciende hasta los 21.000€ anuales.

Lo que hace que cogiendo el calendario laboral del año 2018 en la comunidad valenciana trabajando de lunes a viernes, la cantidad de días laborales es de 251, cantidad a la que hay que restar los días de vacaciones propios del trabajador los cuales son 31 días menos 8 días que corresponden a los fines de semana durante las vacaciones. Por todo ello la cantidad de días trabajados es de 228 días.

Por tanto con estos dos datos podemos calcular el valor de la mano de obra por cada hora de trabajo realizada, esto es:

$$Precio\ de\ la\ mano\ de\ obra = \frac{21000\frac{€}{año}}{228\frac{días}{año} * 8\frac{h}{día}} = 11.51\frac{€}{h}$$

CUADRO Nº2: PRECIO DE LOS MATERIALES PUESTOS EN OBRA

Los materiales utilizados en la realización del proyecto son:

• Ordenador de sobremesa	895€
• Placa Arduino MEGA ADK	35.34€
• Cable USB (A-B)	3€
• Placa protoboard 830 pines	7.95€
• Juego de cables jumper	3€
• Módulo Bluetooth HC-06	7.99€
• Módulo Bluetooth HC-05	8.99€
• Software Matlab R2017a versión estudiante	241€
• Sistema operativo Windows 10	145€
• Software Arduino 1.8.5	Gratuito

CUADRO Nº3: PRECIOS UNITARIOS

- Unidad de obra 1

Instalación del sistema operativo Windows 10 en el ordenador	116.81€

- Unidad de obra 2

Instalación de los softwares requeridos para el desarrollo del proyecto	289.47€

- Unidad de obra 3

Documentación y desarrollo de la configuración de las placas Arduino y de los módulos bluetooth	1587.18€

- Unidad de obra 4

Documentación y desarrollo sobre el simulador y el controlador para el mismo	1209.24€

CUADRO Nº4: PRECIOS UNITARIOS DESCOMPUESTOS

- Unidad de obra 1:
  Instalación del sistema operativo Windows 10 en el ordenador

	Importe (€)	**Total (€)**
1 Ordenador de sobremesa	895	895
1 Sistema operativo de Windows 10	145	145
2h Ingeniero industrial	11.51	23.02
0.02 medios auxiliares	1063.02	21.26
0.03 costes indirectos	1084.28	32.53
Total:		**1116.81€**

- Unidad de obra 2:
  Instalación de los softwares requeridos para el desarrollo del proyecto

	Importe (€)	**Total (€)**
1 Software Matlab R2017a	241	241
1 Software Arduino 1.8.5	0	0
3h Ingeniero industrial	11.51	34.53
0.02 medios auxiliares	275.53	5.51
0.03 costes indirectos	281.04	8.43
Total:		**289.47€**

- Unidad de obra 3:
  Documentación y desarrollo de la configuración de las placas Arduino y de los módulos bluetooth

	Importe (€)	**Total (€)**
2 Placa Arduino MEGA ADK	35.34	70.68
2 Módulo bluetooth HC-06	7.99	15.98
2 Módulo bluetooth HC-05	8.99	17.98
2 Placa protoboard 830 pines	7.95	15.90
1 Juego de cable jumper	3	3
2 Cable USB (A-B)	3	6
120h Ingeniero industrial	11.51	1381.20
0.02 medios auxiliares	1510.74	30.21
0.03 costes indirectos	1540.95	46.23
Total:		**1587.18€**

- Unidad de obra 4:
  Documentación y desarrollo de la configuración de las placas Arduino y de los módulos bluetooth

	Importe (€)	**Total (€)**
100h Ingeniero industrial	11.51	1151
0.02 medios auxiliares	1151	23.02
0.03 costes indirectos	1174.02	35.22
Total:		**1209.24€**

PRESUPUESTO DE EJECUCIÓN Y POR CONTRATA

Descripción	**Importe**
Ordenador con sistema operativo	1116.81€
Instalación de softwares	289.47€
Documentación y desarrollo Arduino	1587.18€
Documentación y desarrollo simulador y controlador	1209.24€
Presupuesto de Ejecución Material (PEM)	**4202.70€**
Gastos Generales 13%	546.35€
Beneficio Industrial 6%	252.16€
Presupuesto de Ejecución por Contrata	**5001.21€**
I.V.A. 21%	1050.25€
Presupuesto base de licitación	**6051.46€**

Asciende el presente presupuesto a la expresada cantidad de:

SEIS MIL CINCUENTA Y UN EUROS CON CUARENTA Y SEIS CENTIMOS

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UN EMULADOR DE SISTEMA DE CONTROL AUTOMÁTICO DE GLUCOSA EN SANGRE SOBRE UN SISTEMA HARDWARE DE BAJO COSTE

Anexos

Curso Académico: 2017-18

ANEXO I: MANUAL DEL USUARIO

OBJETIVO DEL MANUAL

El objetivo principal de este manual es mostrar al posible usuario cuales son los conocimientos previos que debe tener o adquirir para poder hacer que su experiencia con el simulador sea óptima, los elementos necesarios para su funcionamiento y, además, enseñar la serie de pasos necesarios para llevar a cabo una buena utilización del mismo.

ESTRUCTURA DEL MANUAL

Este manual va a tener tres grandes apartado, los dos primeros son más bien apartados introductorios que aconsejan al usuario en potencia una serie de aspectos que pueden facilitar su uso del simulador.

El manual comenzará con una exposición de cuáles son los conocimientos previos que pueden a ayudar al usuario en su experiencia y sería buenos tenerlos o conseguirlos, se busca que a través de la lectura completa del documento el usuario sea capaz de adquirir dichos conocimientos.

La segunda parte del manual hará referencia al hardware o dispositivos y elementos necesarios para la puesta en funcionamiento del simulador y de los códigos implementados para la adaptación en Arduino de uno de los controladores.

Por último, se realizará el manual del usuario en sí, apartado en el que se comentarán cuáles son los pasos básicos y necesarios que realizar para usar correctamente el simulador y tener la mejor experiencia posible durante su uso.

CONOCIMIENTOS PREVIOS

Para que la experiencia con el simulador sea óptima es aconsejable tener una serie de conocimientos mínimos, dichos conocimientos pueden ser:

- Al ser un simulador que emula el comportamiento de una persona diabética sería interesante que el usuario tuviera un mínimo de conocimiento sobre la enfermedad, causas, síntomas, consecuencias…
- También sería positivo conocer los distintos rangos de glucemia que existen y la interpretación que estos tienen
- Como se corrigen valores extremos de glucemia, con insulina, glucagón, carbohidratos de rescate…, para tener una mejor comprensión de los resultados ofrecidos

- Tener un conocimiento básico de la herramienta en la que se sitúa el simulador, que es Matlab, sin necesidad de saber programar, pero si el conocimiento de ejecutar funciones, mostrar variables…
- Y conocimientos básicos de informática, como puede ser el manejo de carpetas, de diferentes pestañas, de ventanas emergentes…

Si se disponen de todos estos conocimientos se estaría en disposición de realizar un buen uso del simulador y poder tener una buena interpretación de los resultados que se obtienen tras cada ejecución, consiguiendo así el objetivo principal del simulador.

HARDWARE Y SOFTWARE

Para poder obtener unos buenos resultados de la utilización del simulador es necesario disponer de una serie de dispositivos y software sin los cuales no se podría realizar ningún ensayo pese a disponer del simulador.

Los requisitos indispensables, en cuanto a hardware se refiere, para el uso del simulador son los siguientes:

- Ordenador dispuesto de:
  - Sistema operativo Windows 10
  - Bluetooth
  - Entre 3 y 4GB de espacio libre en el disco duro para la instalación de los softwares

A parte de esto con lo que ya se podría utilizar el simulador, si se quisiera utilizar el controlador implementado en Arduino serían necesarios los siguientes dispositivos:

- 2 placas Arduino MEGA ADK con:
  - Microcontrolador: ATmega2560
  - Voltaje Operativo: 5V
  - Pines digitales de Entrada/Salida: 54 (de los cuales 15 proveen salida PWM)
  - Pines análogos de entrada: 16
- 2 módulos bluetooth HC-06, módulos esclavo
- 2 módulos bluetooth HC-05, módulos maestro
- 2 placas protoboard de 830 pines, para establecer las conexiones
- 2 cables USB para la alimentación de las placas Arduino
- 1 juego de cables jumper

En cuanto al software para la realización de los ensayos es indispensable:

- Software Matlab R2017a
- Sistema operativo Windows XP o superior

Y si se va a utilizar el controlador de Arduino, también es necesario:

- Software Arduino 1.8.5.

MANUAL

El usuario que va a trabajar con el simulador solamente debe conocer cómo utilizar el programa "set_simulation.m", la carpeta "results" donde se almacenan todos los resultados, la carpeta "scenarios" para ver configuraciones antiguas, la carpeta "controllers" para observar que controlador se adapta mejor a sus exigencias y la ventana de comandos y de variables propias de Matlab, puesto que el resto de carpetas y programas contienen información referente a ámbitos de creación de variables y utilización de funciones que al potencial usuario del simulador no le son de gran interés.

Primero se comentará el funcionamiento del programa "set_simulation.m" y después se realizará la explicación detallada de lo que se puede encontrar dentro de las carpetas "results", "controllers" y "scenarios".

SET_SIMULATION.M

El programa "set_simulation.m" es el encargado de realizar la asignación de los valores de los estados iniciales o de aquellos valores que van a estar programados desde el principio, así como de otorgar la posibilidad de que se tengan o no en cuenta multiplicidad de factores.

Dicho programa está estructurado en varias partes bien diferenciadas unas de otras, puesto que se encuentran separadas físicamente por unas líneas horizontales de un lado al otro de la ventana. Estas zonas en las que se separa el programa son:

1. Se estipulan algunas condiciones y premisas generales, como la búsqueda de archivos pertenecientes a versiones antiguas del simulador, el fichero en el que se va a llevar a cabo la escritura de los resultados, las gráficas que se van a mostrar por pantalla y las condiciones necesarias para que se dé una ejecución en paralelo
2. Se seleccionan cuáles van a ser los pacientes que va a ser utilizados para la simulación, pudiendo elegir entre adultos, adolescentes y niños y dentro de ellos hay una variedad de entre 11 pacientes, se puede escoger solo uno de ellos, varios o incluso los 33 paciente que se disponen
3. Se realiza la elección de que controlador de la carpeta "controllers" va a ser el escogido para la simulación que se está programando
4. Se escoge cual va a ser el escenario en el que se va a guardar esta configuración, puede ser un escenario ya existente en la carpeta "scenarios", el cual se actualizará, o uno nuevo generando de esta forma un nuevo escenario dentro de dicha carpeta
5. Se estipulan cuáles van a ser las condiciones iniciales para la simulación, se fijó el valor inicial de la glucosa en 100 mg/dl, el tiempo entre mediciones, el cual se corresponderá a unos 5 minutos de la vida real, y la duración total de la simulación expresada en minutos
6. Se lleva a cabo la asignación de todos los valores relacionados con las comidas, se estipula una duración de 15 minutos en cada comida y se ofrecen varios ejemplos de casuísticas que puede realizar una persona, como no realizar ninguna comida al día, realizar 1, realizar múltiples comidas o realizar múltiples comidas múltiples días, en todas las casuísticas se da información sobre la cantidad de comida que se va a ingerir y el momento en el que se va a hacer, además, cabe la posibilidad de programar que se

realicen comidas aleatorias entre las comidas que nosotros hemos programada para dotar de más realidad al simulador

7. Se permite al usuario del simulador la posibilidad de programar que el bolo de insulina que se inyecte el paciente no sea exactamente el ideal para la comida que se va a realizar, como puede ocurrir en la vida real, además también te permite marcar la proporción entre las que se va a desplazar el valor de dicho bolo en relación con el ideal
8. Se encuentran los valores de variabilidad de cómo afecta al paciente una comida, una dosis de insulina o de glucagón, también se da la posibilidad de permitir o no dicha variabilidad
9. Se establece cual va a ser la librería de comidas que se va a utilizar para las comidas anteriormente programadas, también permite la posibilidad de que a veces se elija una comida distinta a la que se había programado de forma aleatoria, y la posibilidad de añadir un nuevo tipo de comida que no se encuentre dentro de la librería escogida
10. Se establecen todos los aspectos relacionados con la realización de ejercicio físico, pudiendo elegir si seguir un modelo que se ofrece o generar uno propio y, también, elegir cuando, la duración y la intensidad del ejercicio que se va a programar, además, al igual que en el caso de las comidas se da la posibilidad de elegir si se desea que el propio simulador genere ejercicios de forma aleatoria para dar mayor credibilidad a la simulación
11. Se establece la posibilidad de introducir un ruido a la señal generada por el sensor de glucosa
12. En último lugar, se da la posibilidad de elegir la bomba de insulina y la bomba de glucagón que se van a utilizar, en función del error que estas van a tener, en este caso se escogerá que no tengan error, y, también, se puede se puede escoger la calidad del sensor continuo de glucosa, ofreciendo un gran número de alternativas

Cuando se hayan establecido todas las condiciones que el usuario ha interpretado oportunas, se lleva a cabo la ejecución de la simulación, para ello se debe cargar el programa "set_simulation.m" a través de la ventana de comandos y posteriormente el programa "run_UVAv32Parallel.m" el cual contiene el código que realiza la simulación. Cuando se ejecuta este comando puede aparecer un mensaje en pantalla de si se quiere actualizar o no el escenario elegido, si este ya existe, si no existiese no saldría el mensaje y, después, aparecerá otro en el que hay que al pulsar el botón "enter" se comenzará a realizar la simulación o "ctrl+c" si no se desea realizarla.

Una vez realizada la simulación, la configuración que se ha realizado en el "set_simulation.m" se guarda en la carpeta "scenarios" con el nombre que se le ha dado en el programa principal y los resultados finales que arroja la simulación pueden encontrarse en la carpeta "results", carpeta que el usuario debe manejar para poder comprobar que los resultados están dentro de las expectativas.

CARPETA RESULTS

En el interior de esta carpeta se encuentran varias variables tipo estructura, ellas se llaman "metrics", "scenario", "cohort", "hardware" e "history", la primera de ellas reúne los resultados más importantes obtenidos por la simulación.

Dentro de esta estructura podemos encontrar variables tan relevantes como el valor medio del seguimiento continuo de glucosa, el porcentaje del tiempo que ha estado en hipoglucemia, en hipoglucemia severa, en hiperglucemia…, el riesgo de entrar en hipo o hiperglucemia, el total de la insulina suministrada, los casos de hipoglucemia, si se han tenido que utilizar carbohidratos de rescate o glucagón y la cantidad de que se ha tenido que utilizar, a parte de otros valores de importancia.

En la siguiente de las estructuras están registrados todos los valores iniciales que se han establecido para esa simulación en el programa "set_simulation.m" y en la siguiente todos los datos relativos al paciente o los pacientes sobre los que se ha decidido realizar el estudio.

La cuarta de las estructuras está formada por otras estructuras, estas últimas contienen la información referente a los valores máximos y mínimos posibles de ciertas variables como el bolo de insulina o la insulina basal, la información relativa al controlador utilizado y la precisión del sensor ficticio utilizado.

Y por otro lado la última estructura recoge valores importantes en cada intervalo de tiempo lo que completaría la información ofrecida por la primera estructura pudiendo obtener más información de los momentos más relevantes de la simulación, como pueden ser los picos más altos y los más bajos.

CARPETA SCENARIOS

La funcionalidad de esta carpeta es guardar las configuraciones que se realicen en el programa "set_simulation.m", para poder saber la configuración que se empleó en cualquier ejecución y poder así configurar de nuevo las condiciones iniciales sin posibilidad de error en caso de querer repetir alguno de los escenarios.

Las variables que hay dentro de la carpeta, las cuales corresponden a al escenario inicial diseñado, son creadas por el programa "set_simulation.m", pudiendo generar tantas como se quiera o actualizando algunos de los escenarios ya existentes.

CARPETA CONTROLLERS

En el interior de esta carpeta se sitúa toda la información en relación con los controladores creados para mantener los valores de glucosa en sangre dentro de los valores límites pasando el menor tiempo posible en hipoglucemia y en hiperglucemia.

Dentro de esta carpeta pueden ir desde controladores muy sencillos como puede ser el controlador "Open_loop", el cual es como si trabajara en circuito abierto, sin ninguna realimentación por parte del controlador, hasta otros controladores tipo PID (Proporcional-Integral-Derivativo) en los que ya se produce una situación de bucle cerrado.

En el interior de cada controlador se encuentran dos programas .m, "init_mycontroller.m" y "run_mycontroller.m", en los cuales se definen todos los aspectos relativos al controlador.

En el primero de ellos se establecen los valores de las condiciones iniciales, el nombre del controlador, los valores límites entre los que se va a trabajar, el valor de las constantes del controlador, la referencia a seguir y algunos constantes necesarias para la ejecución del controlador, como puede ser la insulina basal, este programa se ejecuta una única vez al comienzo de la simulación.

El segundo de los programas se ejecuta en cada una de las iteraciones de la simulación, es el programa que lleva a cabo el control de la glucosa y devuelve la acción de control que se le debe aplicar para mantenerla en los valores correctos. Dentro de ella se encuentran todas las ecuaciones que se realizan en un controlador calculando la acción de control, el bolo de insulina que se debe aplicar o los carbohidratos de recuperación.

También se puede observar en esta carpeta la adaptación que yo he realizado para el Arduino, la cual se comentará más adelante.

ANEXO II: MANUAL DEL PROGRAMADOR

OBJETIVO DEL MANUAL

Este es un documento formativo en el cual se busca aportar a un posible futuro programador los conocimientos y las herramientas necesarias para poder configurar por si mismo un nuevo controlador con o sin conexiones bluetooth a placas Arduino u otro tipo de ellas y a poder realizar dichas conexiones entre el ordenador y las placas Arduino mediante los módulos HC-06 y HC-05.

ESTRUCTURA DEL MANUAL

Este manual se va a estructurar en tres grandes vertientes, al igual que en el manual del usuario se realizará un apartado para conocimientos previos y otro para hardware y software, para después pasar al manual.

Dentro de este se encuentran varios apartados, como las configuraciones de los módulos bluetooth y como se realiza la conexión de estos para conseguir un buen resultado, cual es el código que se debe implementar en Matlab y cuál es el código necesario en Arduino para conseguir los resultados que hemos obtenido.

Primero hay que diferenciar entre los dos distintos tipos de módulos bluetooth de los que se dispone, módulos HC-06 y módulos HC-05, ya que su configuración no realiza de la misma manera, y también se comentara como se realiza la conexión entre ellos y el software de Matlab, para que estos puedan ser utilizados.

Después se debe comentar cuales son las partes que se deben programar en Matlab y cuál va a ser la estructura que están deben tener para conseguir un buen funcionamiento del controlador que se va a programar.

Por último se comentará cuáles son los códigos que se deben realizar en Arduino para poder conseguir una buena comunicación entre las dos placas y entre ellas y el ordenador.

CONOCIMIENTOS PREVIOS

Para poder hacer frente a este manual, es necesario tener algunos de los conocimientos que se les exigían a los usuarios y otros distintos, estos son los conocimientos que sería bueno tener:

- Conocimientos básicos de informática, como el manejo de carpetas, ventanas múltiples… como se les pedía a los usuarios
- Conocimientos sobre la enfermedad de la que se trata, la diabetes, los distintos rangos en los que puede estar la glucemia y cuáles son sus consecuencias, y que hay que hacer para corregir o evitar esas consecuencias

- Conocimientos de algunos lenguajes de programación como el lenguaje M, el utilizado en Matlab o el de Arduino, también podría servir con tener conocimientos del lenguaje C++, puesto que ambos lenguajes lo toman como base para el suyo

Con todos estos conocimientos el potencial programador debería estar listo para llevar a cabo la programación de un controlador PID dentro de las placas Arduino, como se ha realizado en este proyecto.

HARDWARE Y SOFTWARE NECESARIOS

En este caso sí que las exigencias coinciden completamente con las necesarios para tener una experiencia de usuario. Por tanto los requisitos que se necesitan en cuanto a hardware son los siguientes:

- Ordenador dispuesto de:
  - Sistema operativo Windows 10
  - Bluetooth
  - Entre 3 y 4GB de espacio libre en el disco duro para la instalación de los softwares
- 2 placas Arduino MEGA ADK con:
  - Microcontrolador: ATmega2560
  - Voltaje Operativo: 5V
  - Pines digitales de Entrada/Salida: 54 (de los cuales 15 proveen salida PWM)
  - Pines análogos de entrada: 16
- 2 módulos bluetooth HC-06, módulos esclavo
- 2 módulos bluetooth HC-05, módulos maestro
- 2 placas protoboard de 830 pines, para establecer las conexiones
- 2 cables USB para la alimentación de las placas Arduino
- 1 juego de cables jumper

Y en cuanto a software serán:

- Software Matlab R2017a
- Sistema operativo Windows XP o superior
- Software Arduino 1.8.5.

Una vez cumplidos todos estos requisitos se puede comenzar a plantear los pasos necesarios para la programación de un PID dentro de una placa Arduino.

MANUAL

INTRODUCCIÓN DE CÓDIGOS EN LAS PLACAS ARDUINO

Como para este proyecto se va a emplear dos placas Arduino para realizar la introducción de códigos en ellas se recomienda conectar de forma individual las dos placas Arduino, para evitar confusiones entre ellas, y para llevarlo a cabo será necesario tener descargado el software de Arduino.

Se va a explicar el procedimiento general para la introducción de sketches en las placas Arduino para lo que será necesario realizar los siguientes:

Paso 1. Apertura del fichero Arduino el cual alberga el código que se quiere introducir en una de las placas

Paso 2. Seleccionar el modelo de placa que se va a utilizar, empleando la pestaña de herramientas, después la pestaña "Placa:" dentro de esta y seleccionar la placa que va a usar, en este caso sería Arduino MEGA 2650, como puede observarse en la Figura 30

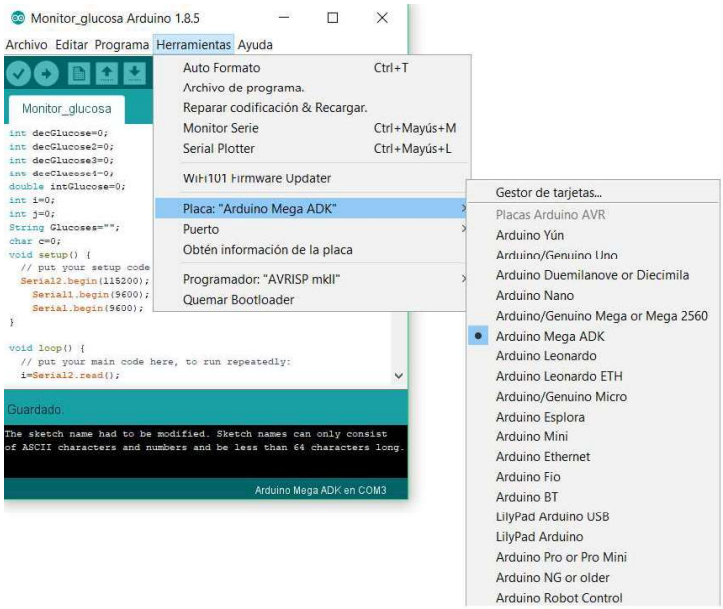

Figura 20. Selección del tipo de placa Arduino

Paso 3. Seleccionar el puerto COM al que se encuentra conectada la placa Arduino, la pestaña de herramientas, entrando en la pestaña "Puerto" y seleccionando la que te indica donde está conectada la placa, si no te lo indica se puede encontrar en el administrador de dispositivos de la sección de hardware y sonido dentro del panel de control, en el apartado de puertos (COM Y LPT)

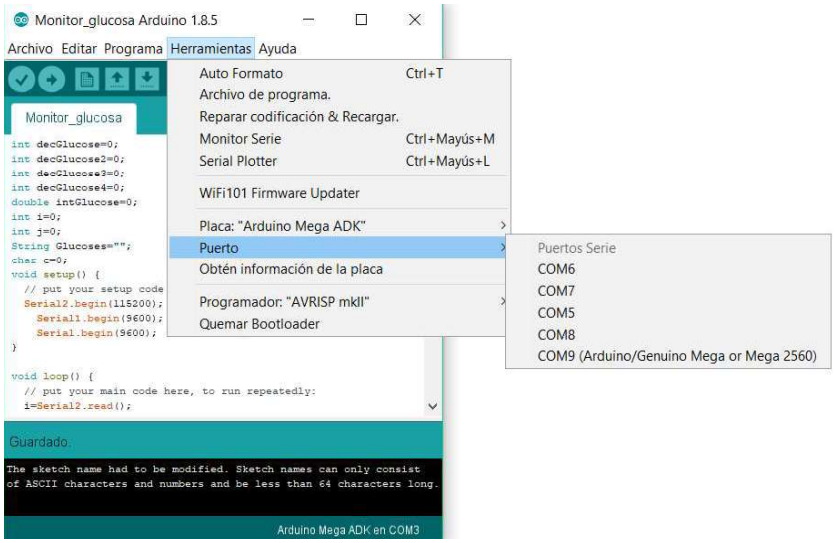

Figura 21. Selección del puerto COM

Paso 4. Cargar el sketch en la placa Arduino, con el botón en la barra de herramientas con forma de flecha señalando a la derecha, como se puede observar en cualquiera de las dos figuras anteriores

Repitiendo estos pasos para la segunda de las placas Arduino, ya se habría conseguido el objetivo para poder disponer de las dos placas para la consecución del proyecto.

CONFIGURACIÓN DE LA CONEXIÓN BLUETOOTH

Los módulos bluetooth se han organizado generando una especie de ciclo cerrado con el ordenador en el que la información va a circular desde el ordenador hasta el módulo HC-06 de la placa que va a realizar la función de monitor, ahí se comunican los dos módulos HC-05 de las dos placas y para cerrar el ciclo se realiza la conexión entre el módulo bluetooth HC-06 de la placa del controlador y el ordenador. El único momento en el que no se cumple el ciclo es el envío de las constantes del controlador que van del ordenador a la placa del controlador de forma directa, sin intermediarios.

Módulo bluetooth HC-06

Para su configuración se ha empleado el código suministrado por un miembro de la comunidad de Arduino, el cual se puede encontrar en uno de los enlaces de la bibliografía. Este código comprende las siguientes opciones:

- Estado del módulo.
- Cambiar el nombre del módulo.
- Cambiar el pin del módulo.
- Cambiar el baudrate (desde 1200 hasta 115200).
- Ver la versión del firmware.
- Información sobre el programa.

Dentro de cada una de estas opciones nos ofrecen varias posibilidades dependiendo de la opción que hayamos escogido en el menú, por ejemplo, en la opción cambiar nombre se la posibilidad de escribir un nuevo nombre o volver al menú principal, y en la opción cambiar baudrate, que es la velocidad con la que se transmiten los datos, nos da a elegir entre unos valores predeterminados para esta función o volver al menú principal, de esta forma podemos configurar nuestro módulo bluetooth sin mayores dificultades.

Módulo bluetooth HC-05

En el caso de los módulos bluetooth HC-05 se realiza a través de los comandos AT introducidos por el monitor serial. Para ello se ha utilizado un programa de ejemplo que esta introducido dentro del software de Arduino, llamado “SoftwareSerialExample” y poniendo el módulo bluetooth en modo configuración, a este modo se entra manteniendo pulsado el botón del que dispone el módulo bluetooth a la vez que se abre el monitor serial. Para la configuración de estos módulos existen varios comandos AT, que son los siguientes:

- **AT:** comprueba que el módulo está bien conectado y ha entrado en modo configuración
- **AT+RESET:** devuelve el módulo al modo normal manteniendo los cambios realizados
- **AT+ORGL:** al ejecutar este comando se vuelve a programar el módulo con sus valores de fábrica

- **AT+ROLE?:** te devuelve el estado en el que esta, pudiendo ser en valor 0, 1 o 2 en función de si está en modo esclavo, maestro o esclavo-bucle, respectivamente; por defecto se encuentra en modo esclavo
- **AT+ROLE=<param>:** este comando nos permite cambiar el rol que cumple nuestro módulo, poniendo los valores 0, 1 o 2 después del igual
- **AT+PSWD?:** muestra por pantalla la contraseña de acceso para establecer la conexión, por defecto sería "1234"
- **AT+PSWD=<param>:** permite personalizar la contraseña del módulo
- **AT+UART?:** indica a qué velocidad se están transmitiendo y recibiendo los datos, por defecto sería 9600
- **AT+UART=<param>:** este comando nos da la posibilidad de cambiar la velocidad a la que queremos que se transmitan y reciban los datos
- **AT+NAME?:** nos muestra el nombre del módulo
- **AT+NAME=<param>:** otorga la posibilidad de cambiar el nombre del módulo para poder reconocerlo más fácilmente
- **AT+ADDR?:** enseña cual es la dirección MAC del módulo
- **AT+BIND?:** muestra la dirección MAC del dispositivo al cual se va a conectar
- **AT+BIND=<param>:** permite establecer cuál va a ser la dirección MAC del dispositivo al cual queremos que se vincule
- **AT+CMODE?:** nos permite averiguar a que dispositivos se va a enlazar, si devuelve un valor 0 se conectará únicamente a los dispositivos que posean la dirección MAC establecida en el comando precedente, si devuelve un 1 se puede conectar a cualquier dispositivo y si devuelve un 2 se conectará a un esclavo-bucle
- **AT+CMODE=<param>:** permite configurar cual va a ser el tipo de dispositivos a los que se va a enlazar

En este caso se recomienda configurar uno de ellos como esclavo y otro como maestro, para poder entablar la conexión, y utilizar los comandos AT+ADDR, AT+BIND y AT+CMODE para configurarlos de forma que los dos módulos bluetooth se conecten únicamente entre ellos para así evitar problemas de que el módulo configurado en maestro se conecte a algún módulo HC-06 o confundirnos y conectar el configurado como esclavo al ordenador.

De esta forma tenemos establecida la conexión ordenador-Arduino con el primero de los módulos HC-06, Arduino-Arduino con los dos módulos HC-05 y Arduino-ordenador con el otro módulo HC-06, generando un ciclo semejante al creado con una bomba de insulina que dispone de un sensor de glucosa.

Comunicación matlab-bluetooth

Realmente, la comunicación con los módulos bluetooth se va a realizar desde el software Matlab, que es la plataforma en la que está desarrollado el simulador. Para llevar a cabo esta comunicación del ordenador con los módulos a través de Matlab es necesario la utilización de algunos comandos específicos que se usan para ello, estos son:

- **B=Bluetooth ('Remotename', channel):** con este comando creamos un objeto bluetooth que después se utilizará en el resto de los comandos, es necesario que le módulo este alimentado cuando se ejecuta este comando. El "Remotename" se trata

del nombre del módulo bluetooth por ello la importancia de personalizarlos y el "channel" es el canal en el cual se va a realizar esa conexión, que debe ser el mismo para los dos módulos bluetooth que van vinculados al ordenador

- **fopen (filename):** con este comando se establece la conexión con el módulo bluetooth y se está preparado para empezar a enviar valores
- **fclose (filename):** con él se cierra la conexión con el módulo bluetooth en cuestión
- **fwrite (obj, A):** este comando sirve para enviar un entero desde Matlab hasta el módulo bluetooth y que este se lo comunique al Arduino, "obj" es el objeto bluetooth creado con el primer comando y "A" es el entero que se le va a transmitir, que puede de tener un valor máximo de 255
- **fread (obj):** este comando permite leer el valor que tiene el bluetooth en su memoria, este solo puede ser un valor entero
- **fprintf (obj, 'format', 'cmd'):** con este comando puedes transmitir al módulo y posteriormente a la placa Arduino un valor de cualquier formato, indicado en la llamada al mismo, desde un float, hasta un String, por ejemplo, en él, "cmd" es la variable que se quiere traspasar
- **fscanf (obj, 'format'):** este comando al igual que fread permite leer el valor que el módulo tiene en memoria, pero en este caso puede ser cualquier tipo de variable, float, double, String…

Con la utilización de estos comandos se puede establecer la conexión, interrumpirla y enviar y recibir datos siempre que se necesite, por lo que la conexión de los módulos bluetooth ya está completamente realizada.

CONEXIONES PLACAS ARDUINO

Para poder utilizar el controlador introducido en las placas Arduino es necesario realizar la conexión de los pines de las placas con los módulos bluetooth a través de las protoboard de una forma específica que se explica a continuación. Esta conexión se repite en ambas placas Arduino por lo que la explicación es aplicable a ambas, estos son los pasos a seguir:

Paso 1. Agrupar una placa Arduino MEGA ADK, un módulo bluetooth HC-05, un módulo bluetooth HC-06, una placa protoboard y 8 cables jumper, que serán los elementos necesarios para realizarla

Paso 2. Insertar los módulos bluetooth en la placa protoboard con todos los pines de cada módulo insertados en una columna señaladas por una letra mayúscula, las filas se encuentran numeradas y los pines de cada fila están interconectados por eso se insertan en una columna

Paso 3. Alimentar los módulos, uniendo con un cable jumper el pin donde pone "5V" en la placa Arduino con la fila en donde se encuentra el pin del módulo llamado "VCC", hacer lo mismo con el pin de la placa llamado "GND" y el pin del módulo con el mismo nombre, después unir las filas de la placa protoboard donde se encuentran los dos "VCC" de los dos módulos y unir, también los dos pines "GND"

Paso 4. Establecer los pines para la comunicación serial, para ello se conecta el pin "RXD" del módulo HC-06 con el pin 16 de la placa Arduino, que corresponde al Serial2, unir el pin "TXD" del módulo HC-06, con el pin 17 de la placa Arduino, repetir lo mismo

con el módulo HC-05, uniendo los pines “RXD” y “TXD” con los pines 18 y 19 de la placa Arduino respectivamente

Después de tener realizada la conexión entre las placas y los módulos ya se pueden conectar los cables USB al ordenador para realizar la alimentación de las placas Arduino y se podría proseguir con la introducción de los códigos que se han desarrollado para que ejerzan como controlador en un caso y como monitor de glucosa en el otro.

CONTROLADOR DISEÑADO EN MATLAB

A parte de la configuración de la conexión inalámbrica mediante los módulos bluetooth, anteriormente explicada, para la consecución de la programación de un controlador utilizando placas Arduino, es bueno tomar como referencia un controlador interno de Matlab, para ver las similitudes y diferencia que estos tienen, para facilitar la programación del primero.

Los controladores dentro de Matlab se caracterizan por estar divididos en dos grandes funciones, una de ellas es la encargada de establecer las condiciones iniciales y las variables del controlador y la segunda se encarga de albergar la formulas con las que se realiza el control de las distintas variables, estas se llaman “init_mycontroller.m” y “run_mycontroller.m” respectivamente.

Por todo ello se va a seccionar la explicación del controlador en el contenido de cada una de estas dos funciones y se va a escoger como ejemplo para esta el controlador “PID2”, que ha sido el elegido en el proyecto para realizar su adaptación a las placas Arduino.

init_mycontroller.m

Dentro de esta función, como ya se ha comentado, se llevan a cabo las asignaciones de los valores de las contantes del controlador, también se establecen los valores de las variables de referencia y de algunos datos con gran relevancia como la insulina basal, aparte de definir cuáles van a ser las condiciones iniciales con las que va a trabajar.

Para poder realizar las asignaciones de todas las condiciones y valores iniciales el programa principal va a pasarle a la función los valores del paciente al que se le esté realizando la simulación, como puede verse en la cabecera de la función, representada en la Figura 32.

El objetivo de esta función en la de crear una estructura con toda la información anteriormente dicha. Dentro del código, en primer lugar, se establecen las condiciones iniciales con las que se va a trabajar, en el caso de este controlador se establece:

- El controlador genera un anuncio de que se va a realizar una comida cuando el tiempo de la simulación incide con el que se ha definido para realizar alguna comida
- Se va a generar también un aviso de la realización de ejercicio
- Se define que el bolo de insulina se va a realizar de forma manual, por lo que se tendrá en cuenta el posible error introducido desde el apartado del “set_simulation.m”
- El tiempo de anterioridad con el que se avisa de la realización de un ejercicio, expresado en minutos, en el caso de este controlador se define como 20 minutos
- Y los valores límites de los rangos entre los que puede estar la glucosa (50, 54, 60, 70, 140, 180, 250, 300)

Además de todas estas condiciones también se pueden añadir unas cuantas más, como, por ejemplo, cambiar la unidad de medida de la glucosa a la de milimoles por litro (mmol/l).

Y después de todo esta se define una estructura dentro de esta en la que se introducen los valores de las constantes del controlador y de las referencias. En el caso en el que nos encontramos, se establece:

- Glucosa de referencia igual a 120 mg/dl
- La glucosa del paso precedente se igual al valor de la de referencia
- La constante Td se define como 75 minutos
- El valor de la constante Kp se obtiene a partir de un valor dl paciente con el que se está trabajando
- La insulina basal se iguala a un valor propio de cada paciente
- Se guarda el peso del paciente en una variable denominada BW

Con todo esto definido el controlador "PID2" está totalmente preparado para pasar a la ejecución de la simulación, en caso de realizar un controlador con mayor complejidad se debe incluir todas las variables y condiciones iniciales necesarias para su correcta implementación.

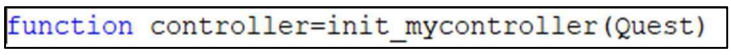

Figura 32. Cabecera función init_mycontroller.m

run_mycontroller.m

En el interior de esta función nos encontramos con las ecuaciones que va a definir funcionamiento del controlador. Al comienzo de la función se encuentra una leyenda que explica el significado de cada variable que se va a emplear para conseguir una buena comprensión por parte del usuario.

Después se realizan una serie de operaciones de donde se obtienen las acciones de control sobre distintos factores, las ecuaciones empleadas en este caso son:

- Primero se calcula el incremento o decremento de la glucosa en función del tiempo, expresando en minutos
- Después la respuesta que se obtiene del controlador en la variable uk
- Después se realiza la asignación de la glucosa de esta iteración a la glucosa del paso precedente
- Se calcula la infusión de insulina continua que se va a realizar
- Se establece que el valor de bolo de insulina que se debe inyectar antes de las comidas es siempre nulo, debido a la condición inicial que se ha utilizado de que los bolos de insulina se suministran de forma manual
- El valor de los carbohidratos de rescate también será nulo siempre
- Y que la cantidad de glucagón que se va a inyectar el paciente también va a ser nula siempre

Para poder realizar todas estas operaciones es necesario que a la función se le transmitan los valores de las contantes del controlador, de la medida de glucosa realizada por el sensor, de los

anuncios de comidas y de ejercicio y de la iteración en la que se encuentra y el tiempo transcurrido, como puede verse en la Figura 33.

Y devolverá como resultado los valores de los últimos cuatro puntos que se han expuesto anteriormente que corresponden con los cuatro valores de las acciones de control, más los parámetros del controlador, puesto que el valor de la glucosa del paso precedente va variando en cada iteración.

Al igual que con la función "init_mycontroller.m" si se desea emplear un controlador de mayor complejidad que necesitara de un mayor número de funciones estas deberían de ser añadidas al código.

```
function [IIRt,GnIRt,insulin_bolus,rescueCHO,specificparams]=
run_mycontroller(k,t,Glucose,Meal_announcement,Exercise_announcement,specificparams)
```

Figura 33. cabecera función run_mycontroller.m

CONTROLADOR DISEÑADO PARA ARDUINO

Una vez explicadas todas las características del controlador que se ha escogido para llevar a cabo la adaptación para ser utilizada con las placas Arduino se va a pasar a comentar cual ha sido esa adaptación que se ha hecho y que cosas se han cambiado para que dicha adaptación pueda ser emulada por otro programador.

Cabe destacar que al igual que el controlador "PID2" este controlador también dispone de una función "init_mycontroller.m" y otra "run_mycontroller.m", pero a estas dos funciones hay que añadirle el código desarrollado en Arduino, para el funcionamiento de las dos placas.

La función de las placas de Arduino no va a ser la misma en ambas, la primera de ellas funcionará como un monitor de la glucosa continua recogida por el sensor, mientras que la segunda será la encargada de albergar en su interior el código que constituye al controlador.

Por lo que será necesario explicar el funcionamiento de cada una de las partes, tanto de las funciones de Matlab como de los códigos introducidos en las placas Arduino.

init_mycontroller.m

En el caso de la función "init_mycontroller.m", el contenido que se encontraba en la correspondiente al controlador "PID2", se han mantenido de forma íntegra, pero se han añadido unas cuantas líneas de código.

Al principio de la función, se ha utilizado el comando para la creación de los objetos bluetooth correspondientes a los módulos de las dos placas (B=Bluetooth ('Remotename', channel)) descrito en el apartado de comunicación Matlab-bluetooth, en este caso con una peculiaridad añadida, las variables en las que se va a introducir dicho objeto han sido definidas como variables globales para que puedan ser utilizadas por otras funciones sin necesidad de realizar los mismos comandos cada vez que se quiera enviar o recibir un dato.

Además, justamente después de crear estos objetos bluetooth se ha ejecutado el comando fopen, también explicado en el apartado de comunicación Matlab-bluetooth, con el cual se realiza la conexión con los módulos bluetooth de las dos placas.

Después se establecen los valores de las condiciones iniciales y se realizan los cálculos de las constantes y de los valores de referencia para el controlador, al igual que se hace en el controlador que se ha tomado como referencia.

Para terminar, se realiza la transmisión de estos valores al módulo bluetooth de la placa Arduino en la que se va a introducir las ecuaciones el cálculo de las acciones de control. Esta transmisión se ha realizado utilizando los comandos fprintf y fwrite, puesto que fwrite solamente puede realizar envíos de números enteros menores de 255, puesto que envía 1 byte de información.

La transmisión se ha realizado separando la parte entera de la decimal, con la función interna de Matlab floor, la que te devuelve el valor de la parte entera de la variable sobre la que se aplica, esta parte entera se va a enviar mediante el comando fprintf, puesto que su valor podría sobrepasar el rango del fwrite. La parte decimal, se va a seccionar de 2 en 2 decimales, para que puedan ser enviados mediante el comando fwrite, puesto que es más fiable que el comando fprintf, hasta un total de 4 envíos, para evitar que se produzca overflow dentro de las placas Arduino.

La estructura del envío, que se puede observar en la Figura 16, consta de la ya citada separación de la parte entera y la decimal, posteriormente se envía a la placa Arduino un aviso de que se le va a enviar la parte entera, se le envía dicha parte y se realiza una pausa para que la placa pueda leer el dato, después se obtienen los dos primeros decimales de la parte decimal y se entra en un bucle for donde se enviarán los 6 primeros valores decimales con la misma estructura que la parte entera, aviso, envío y pausa, al salir del for se enviarán otros dos decimales más, es necesaria esta diferenciación, puesto que los seis primeros decimales en Arduino irán por delante de la coma los dos siguiente por detrás.

El envío se ha realizado con esta estructura por diversas razones:

- La primera de ellas es que Matlab trabaja a 64 bits, mientras Arduino lo hace a 32, por lo que al enviar valores tipo float hay ciertas incompatibilidades que hacen que al comunicar un float con un gran número de cifras algunas de ellas las permute de forma aleatoria
- La elección de desplazar la coma hacia la derecha haciendo que algunas cifras decimales se conviertan en enteras de forma momentánea se ha tomado debido a que Arduino internamente si trabaja con todas estas cifras decimales, pero a la hora de realizar el envío por bluetooth o al monitor serie, realiza un truncamiento en la segunda cifra decimal, perdiendo mucha información, de esta forma conseguimos conservar información que si no se hubiese realizado ser hubiese perdido

run_mycontroller.m

La función “run_mycontroller.m” si ha sufrido una mayor variación con respecto a la función del controlador “PID2”, puesto que las ecuaciones del controlador ya no residen en esta función, sino que ahora se van a encontrar en una de las placas Arduino.

Al principio de la función se realiza la llamada a las variables globales que contienen la información sobre los dos objetos bluetooth, para poder ser empleados, agilizando el proceso al no tener que crearlos en cada iteración.

Después se realiza el envío de la glucosa, pero esta vez al otro módulo bluetooth, el correspondiente a la placa que cumple la función de monitor de glucosa continua, este envío cuenta con exactamente la misma estructura que el envío de las constantes del controlador.

Para finalizar, se lleva a cabo las lecturas de la acción de control sobre el bolo de insulina continuo procedente de un módulo bluetooth y de la glucosa del paso precedente, ya que estas son las únicas dos variables que se actualizan y la función devuelve como solución de la actuación del controlador.

La lectura de los datos se realiza con el comando fscanf, puesto que los datos que se van a leer no son enteros y es necesario establecerle el formato en el que se va a leer, que este caso corresponde a un double, como se observa en la Figura 17.

La estructura de recepción cuenta con un aviso para la lectura, que se envía a la placa que contiene el controlador, seguido de una pausa para que Arduino tenga tiempo suficiente para escribir el dato en el pin de transmisión, posteriormente se realiza la lectura seguida de una pausa, y se debe corregir el efecto producido por haber introducido en Arduino esos 6 decimales por delante de la coma dividiendo por un millón el valor leído, tanto en el caso de la acción de control como el de la glucosa del paso anterior se debe realizar esta corrección, pero podría haberse dado alguna variable que no necesitase de esta corrección o que necesitara una corrección mayor, en función de las ecuaciones que actuasen en la obtención de dicha variable.

Arduino encargado del monitor de glucosa continua

Una de las placas Arduino va a ser la encargada de realizar la función de monitorización de glucosa proveniente del sensor de glucosa continua, esta placa recibirá esta información desde la función “run_mycontroller.m” como se ha explicado anteriormente.

Esta placa Arduino solo sirve como paso intermedio para la glucosa, ya que esta deberá ser enviada con posterioridad a la otra placa para poder utilizar este valor en las ecuaciones del controlador.

Esta placa lo primero que se hace es leer el aviso mandado por Matlab para identificar el valor que se le está enviando y después se leen los valores que le llegan utilizando la función Serialx.read(), con la que lee el valor que se encuentra en el puerto serial “x” donde están conectados los módulos bluetooth, y automáticamente se los transmite a la segunda placa, después de enviarle de nuevo un aviso, con la función Serialx.println(y) o Serialx.write(y) en función de si la variable “y” se trata de un número decimal o uno entero respectivamente, en cuanto a la temporización se sigue la misma estructura que en Matlab, una pequeña pausa antes de leer, para que reciba el dato y otra después de enviar para que el dato llegue a la otra placa Arduino.

Cabe destacar también que los valores que se han enviado con el comando fprintf, van a ser recibidos en las placas Arduino como un String, por lo que es necesario, transformarlos a una variable numérica, en este cado la elegida ha sido doublé, como se muestra en la Figura 18.

Arduino encargado del controlador

La segunda de las placas Arduino va a ser en este caso la encargada de realizar todos los cálculos de las ecuaciones del controlador, que han sido traducidas desde Matlab sin apenas variaciones en sus estructuras.

En esta placa el grueso del código lo constituyen las recepciones de todas las variables provenientes de Matlab y de la primera placa Arduino, y el envío de los resultados obtenidos nuevamente a Matlab.

En esta placa al mismo tiempo que se realiza la lectura de los datos que le llegan se realiza la configuración de las variables, como se ha comentado anteriormente, con los 3 primeros envíos antes de la coma y el último por detrás de ella, gracias a la función pow, como se puede observar en la Figura 19.

El envío de variables a Matlab se realiza de la misma forma que en el primer Arduino, pero en este caso la placa no es la que envía el aviso, sino que es quien lo recibe como ya hemos explicado anteriormente y no es necesario un delay después del envío ya que este código no realiza ninguna operación si no le llega ningún aviso desde Matlab o desde la otra placa Arduino.