Sistema de medición de agua consumida por mascotas

Benearo Semidan Páez

07 de Febrero de 2021

1. Introducción

1.1. Antecedentes

Las mascotas en una familia requieren de una serie de cuidados, variantes según el animal específico que se haya elegido como compañero. No obstante, uno de dichos cuidados que todo animal comparte es la necesidad de hidratrarse de manera diaria.

Un animal que no ingiera agua de manera regular puede ser indicador de la mascota tiene algún problema, sea una enfermedad o, incluso, ansiedad.

Todo dueño de una mascota debe ser cuidadoso con el agua que el animal bebe y esto, en ocasiones, puede ser complicado de realizar.

1.2. Objetivos

Por lo tanto, el objetivo principal de este trabajo es permitir la medición y almacenamiento del peso de el utensilio del que bebe una mascota, así como la visualización de estos datos, con el fin de asegurar que el animal esté hidratándose de manera regular.

1.3. Ubicación del proyecto

Este proyecto se encuentra alojado en la plataforma Github©bajo el siguiente enlace:

`https://github.com/darkhorrow/pet-water-supply-weight-meassures`

En él se encuentra toda la documentación y código generados para este trabajo.

2. Desarrollo

2.1. Planteamiento

Para resolver la problemática introducida previamente, se realiza una aplicación en **MATLAB ©**, más concretamente, con **App Designer**, que permite realizar las mediciones y visualizarlas.

Para las mediciones en sí misma, se hace uso de una placa **Arduino Uno** que, mediante un sensor **FSR**, con el que se obtiene el peso.

2.2. Hardware para la medición

2.2.1. Sensor FSR

Para este proyecto, solo se utiliza un sensor para la medición de el peso del recipiente de agua: un FSR.

Un FSR (Force-Sensing Resistor) son elementos que reaccionan a una presión ejercida alterando su resistencia, disminuyendo esta conforme se aplica mayor fuerza sobre el sensor.

El rango de pesos que suele soportar este elemento suele ser de 100g a 10Kg.

El motivo por el que se ha seleccionado este elemento es principalmente a su disponibildiad y bajo coste, aunque existen opciones más precisas para la medición, como puede ser una celda de carga.

Figura 1: Sensores FSR

Los FSR tienen la forma ilustrada en la Figura 1.

El FSR utilizado para el prototipo es el **RP-C7.6-LT** (también denominado MF01), que cuenta con las siguientes características:

- Rango de inducción de presión: 20 g a 1,5 Kg.
- Trigger: 20 g, resistencia estándar $< 200\text{K}\Omega$.
- Grosor: 0,4 mm.
- Resistencia no disparar: ¿10 MΩ.
- Tiempo de activación: $<$ 0,01 s.
- Tiempo de respuesta: $<$ 10 ms.
- Tamaño: 4,3 x 7,8 mm.
- Peso: 5 g.

2.2.2. Montaje del circuito

El circuito a montar es muy sencillo y fácilmente reproducible, ilustrado en la Figura 2.

En primer lugar, se recomienda probar la conexión de la placa Arduino a usar con el equipo destino que ejecutará la aplicación en MATLAB con, por ejemplo, algunos de los pequeños programas que dispone el IDE de Arduino, garantizando que los controladores de la palca estén instalados (de no ser así, se recomienda usar la guía de Arduino: *Getting Started with Arduino UNO*[1]).

A continuación, dado que la señal que se va a recibir es una variación del voltaje en función de la variación de la resistencia FSR y, por lo tanto, una señal analógica.

Lo más sencillo para la medición es conectar una patilla del sensor al positivo de la fuente de alimentación utilizada, en este caso, los 5V procedentes de la propia placa Arduino UNO. La otra patilla del sensor irá a una resistencia

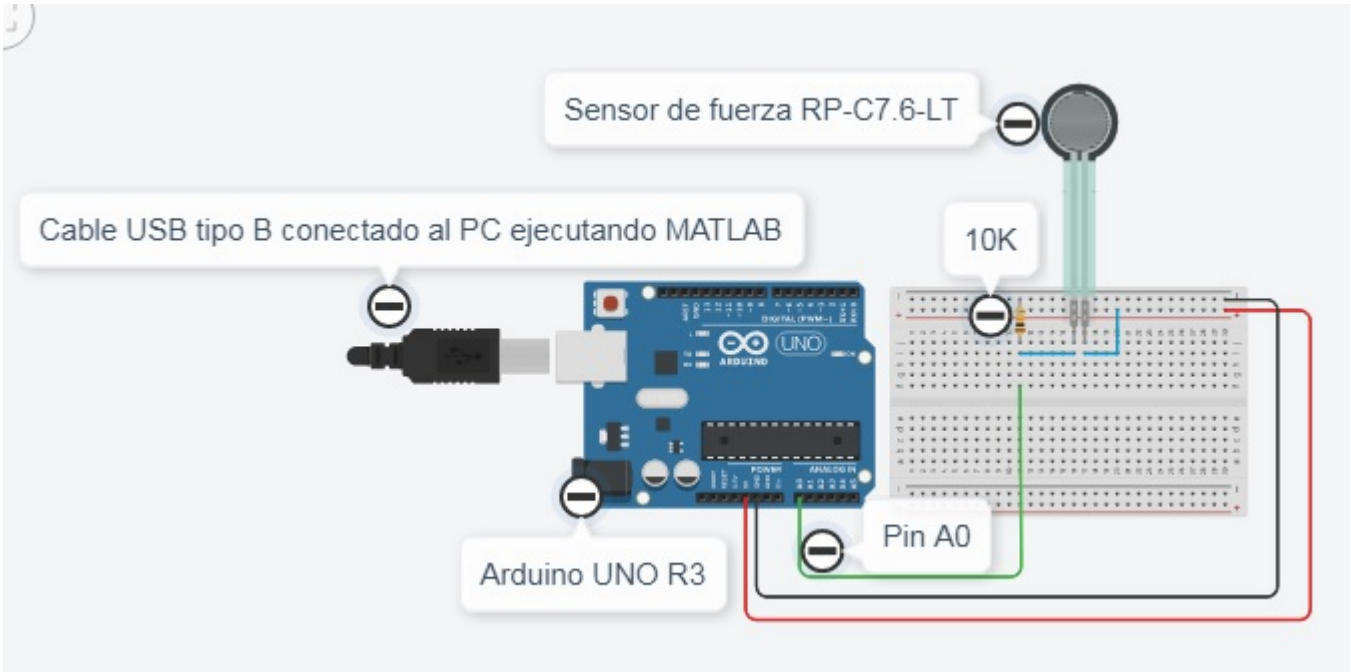

Figura 2: Esquema de montaje

Listado 1: Interpolación del voltaje para obtener el peso

```
function [vq] = FSRValue(xq)
    v = [0 0.1 0.5 1.0 1.1 1.2 1.3 1.4]; % Weight values in Kg
    x = [0 2.2 2.57 3.67 4.02 4.06 4.08 4.09]; % Voltage values in V
    vq = interp1(x, v, xq, 'makima', 'extrap');
end
```

de 10KΩ, ejerciendo de *pull-down* y, a su vez, el otro extremo de la resistencia conectado a tierra (negativo de la fuente de alimentación).

El punto de unión entre la resistencia de 10KΩ y el sensor es el que se conectará al pin analógico del microcontrolador, Arduino UNO, en este caso.

El funcionamiento consiste en que, a medida que la resistencia del sensor FSR disminuye (a causa de la fuerza ejercida sobre él), la resistencia total (FSR + pull-down), al estar ambas en serie, disminuye de aproximadamente 100KΩ a 10KΩ. Eso implica que la corriente que circula por ambas resistencias aumenta, lo que a su vez hace que aumente el voltaje a través de la resistencia fija de 10KΩ (V = IxR).

El esquema eléctrico mencionado es lo que comúnmente se le denomina divisor de tensión [2], ilustrado en la Figura 3.

2.2.3. Transformación de voltaje (V) a peso (Kg)

Como ya se comentó previamente, la desventaja principal de el FSR es que resulta más impreciso que una una celda de carga. Dicho problema surge devido a la respuesta no-lineal existente entre el voltaje obtenido y la fuerza aplicada sobre el sensor.

Por ello, realizamos una interpolación del voltaje obtenido, de manera similar a la respuesta ilustrada en la Figura 3, obteniéndose un peso concreto. Es importante destacar que para este prototipo, se trabaja con randos de peso ente 1Kg y 1.4Kg, ya que es el peso de el recipiente de agua vacío y lleno, respectivamente, por lo que los resultados serán más precisos en dicho rango.

En el Listado 1 se representa la función con la que se realiza la interpolación anteriormente comentada.

El método de interpolación usado es el *Modified Akima*[3], ya que este cuenta con la característica de no generar altas ondulaciones locales y el que obtenía pesos más próximos a los reales.

Voltage Dividerv

The sensor series connection a fixed resistor to measure the output voltage at both ends of the fixed resistance.Vout=Vcc*R0/(R0+Rs)

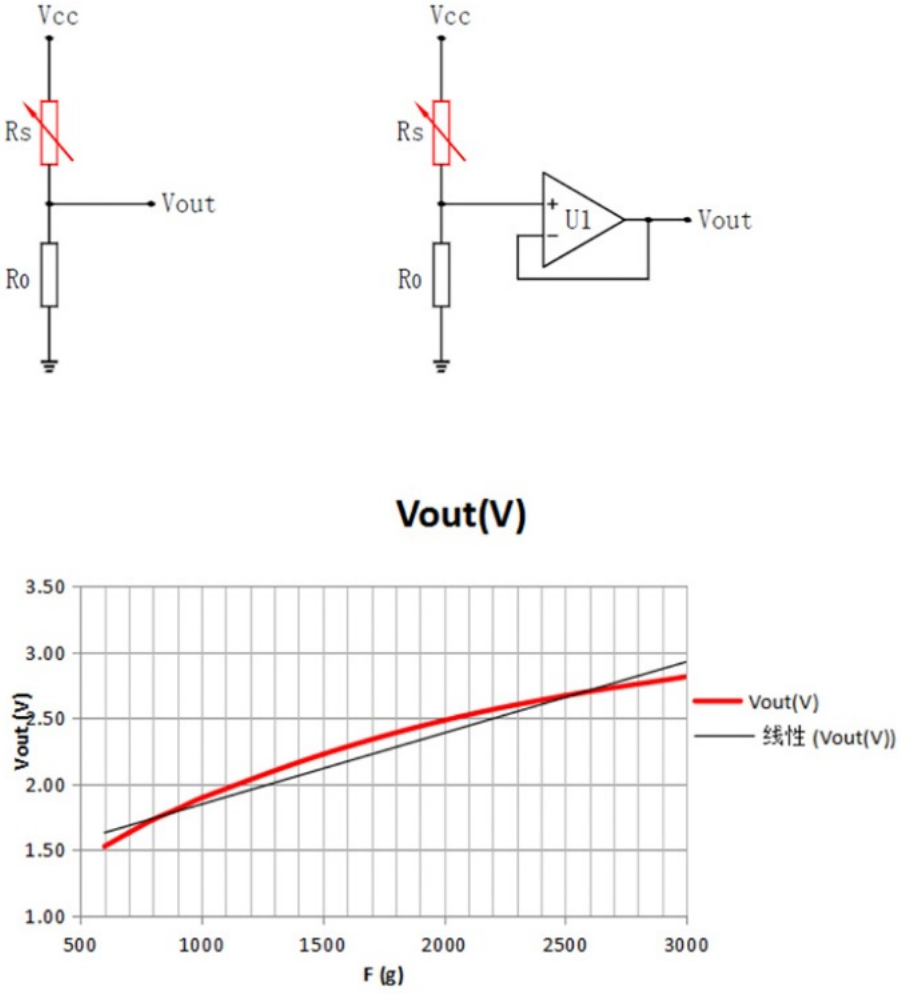

Figura 3: Divisor de tensión y respuesta ante aplicación de fuerza

Listado 2: Timer de la medición diaria

```
app.dailyTimer = timer( ...
        'StartDelay', 0, ...
        'Period', 60, 'TasksToExecute', Inf, ...
        'ExecutionMode', 'fixedRate', ...
        'UserData', app ...
    );

app.dailyTimer.TimerFcn = @app.dailyMeasure;
```

2.2.4. Impresión 3D de piezas de soporte

De manera adicional y con el fin de mejorar la estabilidad y precisión de las mediciones, se realiza una base similar a una báscula simple, que contiene en el centro de esta el sensor FSR.

Los ficheros para la imrpesión se encuentran en el repositorio asociado con anterioridad, junto con el resto de código y este mismo documento.

2.3. Aplicación de manejo

El manejo de la placa Arduino, con el hardware presente en la sección Hardware para la medición, se realiza mendiante MATLABR2020b y una de sus aplicaciones integradas, App Designer.

App Designer permite la realización de interfaces de usuario, integrando lso componenetes clásicos de MATLAB. De esta manera, se ofrece un control más limpio y amigable de las tres tareas principales que se plantean en este prototipo: *Medición en tiempo real del sensor*, *Medición diaria del sensor* e *Informe diario*.

La conexión entre MATLAB y Arduino se realiza mendiante el uso de la librería *Arduino Support from MATLAB*[4], que soporta la comunicación básica entre ambos.

2.3.1. Medición en tiempo real del sensor

La medición del FSR se realiza de manera muy sencilla mediante *polling*, ya que *Arduino Support from MATLAB*[4] no ofrece ninguna manera de realizar interrupciones con Arduino.

Por ello, cuando se realiza esta acción, no se puede realizar la *Medición diaria del sensor*, ya que habría un problema de acceso simultáneo a la lectura del pin del sensor. Antes de realizar esta acción, si ya había una medición diaria en ejecución, se le solicita al usuario que confirme que quiere para dicha medición.

Por otra parte, se puede destacar que la medición muestra las últimas 60 muestras realizadas del sensor.

En la Figura 4 se presenta un ejemplo de visualización de esta parte del programa.

2.3.2. Medición diaria del sensor

Para la medición diaria del sensor, se obtiene una muestra cada minuto mediante un *Timer* de MATLAB. El Timer se encarga de ejecutar de modo asíncrono la llamada, de manera que permita realizar otras acciones (siempre y cuando no entre en conflicto con las llamadas a la palca Arudino). Este timer se puede ejecutar, parar y reanudar en cualquier momento.

El Timer en cuestión usado está presente en el Listado 2.

La función ejecutada por el Timer se encargará de generar un fichero .mat con el nombre *weigths-fecha-actual.mat*, si no existiera ya, añadiendo el voltaje (convertido a peso) y la fecha y hora de la medición en el archivo.

2.3.3. Informe diario

Esta última acción se encarga de graficar los datos del fichero generados de manera diaria, como se explica en la sección Medición diaria del sensor.

Un dato a tener en cuenta es que esta gráfica no se actualiza conforme el fichero cambia su contenido por simplicidad.

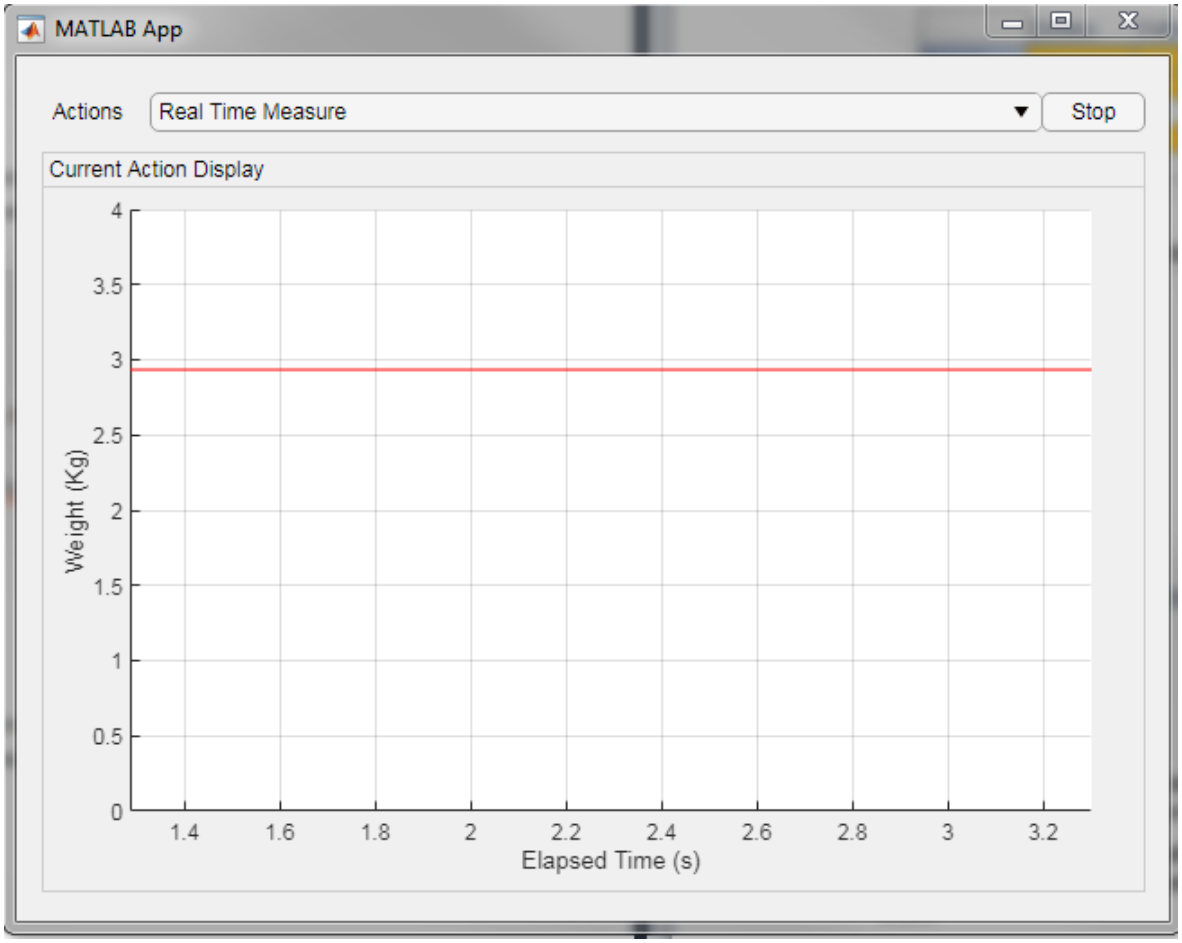

Figura 4: Programa realizando mediciones en tiempo real

Figura 5: Programa visualizando la medición diaria tomada hasta el momento

Con este gráfico, se tiene disponible las mediciones realizadas, permitiendo ver los cambios en el peso en el recipiente que contiene el agua. Un ejemplo se encuentra ilustrado en la Figura 5.

3. Conclusiones

Mediante el prototipado propuesto en este trabajo, se consigue solventar en cierta medida el problema planteado inicialmente: medir la variación de peso a lo largo de un día y graficarlo.

No obstante, los FSR no son los componentes más ideales para este propósito, ya que son muy complicados de obtener el peso para un caso genérico debido a su respuesta no-lineal. Aún así, se obtienen unos resultados aceptables para el rango de valores trabajado.

Referencias

[1] Arduino. Getting started with arduino uno. https://www.arduino.cc/en/Guide/ArduinoUno.

[2] Adafruit. Force sensitive resistor (fsr). https://learn.adafruit.com/force-sensitive-resistor-fsr?view=all.

[3] Arduino. Modified akima interpolation. https://es.mathworks.com/help/matlab/ref/makima.html.

[4] MathWorks. Arduino support from matlab. https://es.mathworks.com/hardware-support/arduino-matlab.html.