



# PECERA INTELIGENTE

Beltrán Barrera V., León Chávez J. H., Sandoval Cuellar H. J., López Silva E. R., Quintana Quintana O.J., García Amaya I.A., Espinosa Guerrero J.L., Lara Ramos J. A. y Soni Urquiza F.

Universidad Autónoma de Querétaro, Facultad de Ingeniería  
[victor.beltran.barrera@gmail.com](mailto:victor.beltran.barrera@gmail.com)

## Resumen

*En este trabajo se presenta un sistema inteligente para una pecera con capacidad de 40 litros. El sistema está compuesto por una tarjeta de desarrollo (RaspBerry Pi 3B), un dispensador automático de comida para peces, una fuente de alimentación con entrada de 120 AC y una cámara montada en la pecera para visualizar el estado del pez. La tarjeta de desarrollo se conecta a internet de manera inalámbrica e intercambia información en tiempo real con una base de datos generada en Firebase, esta base de datos se conecta a la vez con la aplicación "SmartFish" donde el usuario puede revisar el estado de su pez además de programar los horarios y el número de comidas para su pez.*

**Palabras clave:** Pecera inteligente, RaspBerry, Inteligencia Artificial, Android, Firebase.

## Abstract

*In this work we presents an intelligent system for a fish tank with a capacity of 40 liters. The system consists of a development board (RaspBerry Pi 3B), an automatic fish feed dispenser, a 120 AC input power supply, and a fish tank mounted camera to view fish status. The development card connects to internet wirelessly and exchanges information in real time with a database generated in Firebase, this database connects at the same time with the "SmartFish" application where the user can check the status of their fish in addition to programming the times and number of meals for its fish.*

**Keywords:** Smart FishTank, RaspBerry, Artificial Inteligent, Android, Firebase.

## 1. Introducción

En los últimos años se ha observado una evolución en la complejidad de las funcionalidades que los aparatos que acompañan la vida diaria ofrecen. La tendencia ha sido y seguirá siendo el de automatizar tantas actividades cotidianas en la medida de lo posible, bajo este contexto al paso del avance tecnológico cada vez existe mayor capacidad de delegar actividades a algún dispositivo por lo general electrónico. De manera puntual, más allá de los avances electrónicos y mecánicos, el mayor crecimiento reciente y del que se espera aún más en las próximas décadas es el del software, hoy en día existen dispositivos electrónicos cuyas capacidades van desde contar con puertos de entrada y salida generales, hasta módulos de conexión inalámbrica bluetooth o Wifi. Este crecimiento en cuanto a funcionalidad y conectividad, así como el incremento en cuanto a computadores ha llevado a destacar términos como el de Internet of Things o Internet de las Cosas, o bien, un sistema en el cual exista una interconexión y control entre diferentes elementos. Asimismo, el presente proyecto busca brindar funcionalidades a un bajo costo, ya que en su mayoría se han hecho uso de herramientas de software de acceso libre y uso gratuito, lo que abarata el costo total del proyecto.



En este documento se describe detalladamente un proyecto que consiste en una pecera inteligente, la idea general es agregar funcionalidades inteligentes a una pecera común, de manera que el usuario no tenga que preocuparse de ejecutar manualmente cada actividad o bien tenga la opción de realizarlas en cualquier lugar en donde se encuentre en tiempo real. Se presentan varias secciones, tales como Materiales y Métodos, que explica los materiales utilizados en el diseño y elaboración del proyecto, al igual que la metodología que se siguió. También se incluye una sección de Resultados en donde se exponen los efectos o consecuencias de la elaboración del proyecto.

## 2. Materiales y Métodos.

### 2.1 Especificaciones

Este proyecto consistió en desarrollar (diseñar, modelar, simular, construir, instalar y probar) una pecera inteligente, de forma que por diseño se brinde al menos una cualidad “inteligente” al usuario. Las especificaciones del proyecto fueron las siguientes:

Dimensiones máximas: 270 mm de altura, 420 mm de anchura y 220 mm de profundidad.

- Material base: Acrílico. (Pudiendo combinar con aluminio, nylamid y plástico, principalmente).
- Estilo: Minimalista.
- Alimentación: Voltaje de 120 [V] AC.
- Consumo máximo: 50 [W].
- Inversión máxima: 5,000.00 M.N. (Cinco mil pesos).
- Control: Temporal por control remoto mediante teléfono celular.

### 2.2 Metodología

En la Figura 1 se muestra un diagrama general del proyecto realizado, el proyecto se divide de manera general en dos bloques, uno consta de la parte del sistema físico que corresponde al sistema de alimentación, la tarjeta de desarrollo Raspberry, y la pecera inteligente, el segundo bloque incluye la parte de la aplicación móvil desarrollada en la plataforma de Android y la base de datos generada en FireBase.

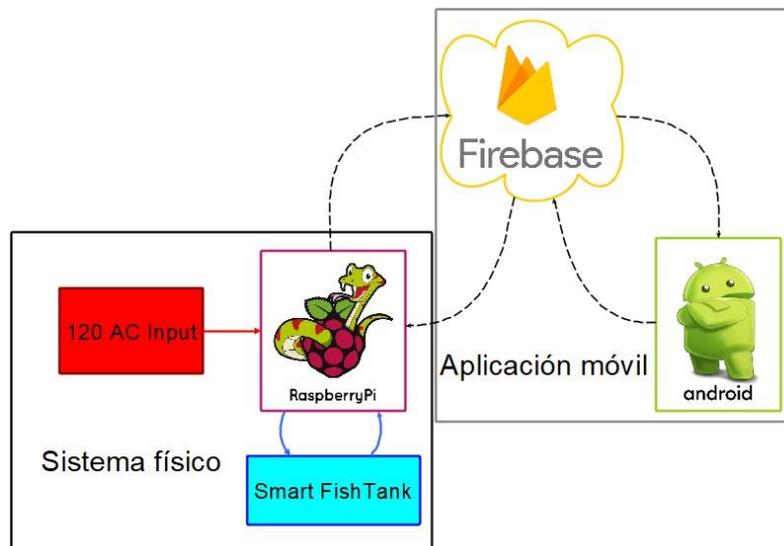


Figura 1. Diagrama general del proyecto “Pecera inteligente”.

La parte del sistema físico se conecta a internet para transferir información con la base de datos y ésta a su vez con la aplicación móvil.

En la Figura 2 se muestra el diagrama del sistema físico, está integrado por la tarjeta de desarrollo Raspberry Pi 3B que se conecta a una cámara web montada en la pecera para monitorear al pez y a un dispensador automático que consta de un sistema integrado por un motor a pasos para proporcionar alimento al pez y un driver para controlar su movimiento.

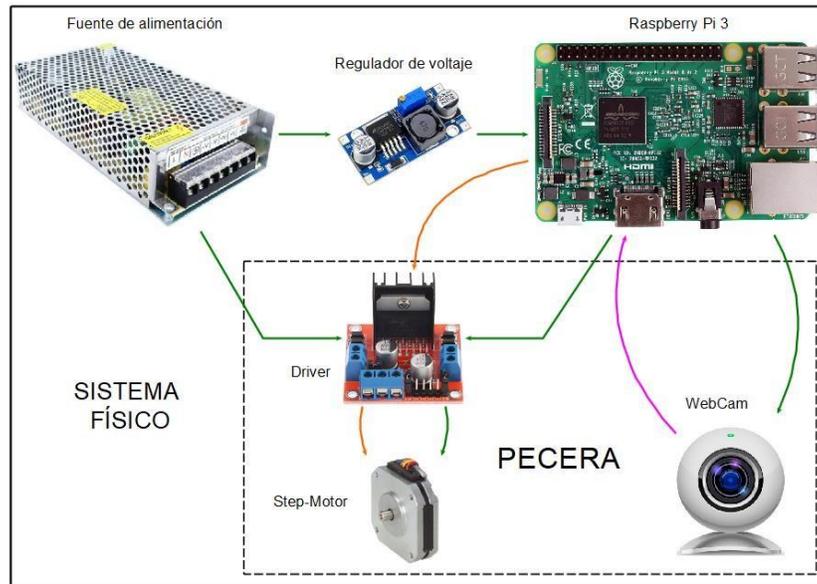


Figura 2. Diagrama del sistema físico.

Para su realización, el proyecto fue dividido en 4 partes:

- Modelo de alimentación.
- Visión.
- Modelado 3D.
- Aplicación móvil y base de datos.

### 2.2.1 Modelo de alimentación.

#### 2.2.1.1 Pseudocódigo.

---

**Algorithm 1:** Probability of occurrence

---

```

input : x
output: f(x)
1  m ← (1/n) * (sum(data_f$distancia)).
2  s ← sqrt(sum((data_f$distancia - m)2)/(n)).
3  for i → range do
4  |   f(x)
   |   ← function(x)1/sqrt(2 * pi * σ2) * exp(-((x - μ)2/(2 * σ2)))
5  end
    
```

---

Figura 3. Pseudocódigo.



### 2.2.1.2 Repositorio.

Para el presente trabajo, se generó el repositorio que se muestra en la figura 4, como se observa, este está basado en la distancia que recorre un pez y la cantidad de comida que ingiere, en un determinado tiempo, el siguiente análisis, se basa en estos datos.

	distancia	comida	tiempo
1	585	5	539
2	569	6	392
3	567	3	522
4	608	6	536
5	594	5	399
6	587	5	527
7	582	8	439
8	614	5	480
9	575	2	397
10	585	4	429
11	558	6	536

Figura 4. Repositorio.

Donde la distancia se mide en metros, la comida en piezas y el tiempo en minutos. El enfoque para el análisis de los datos se plantea partiendo de la media de la distribución que se representa en la ecuación 1, que es la forma estándar de la distribución Gaussiana [1].

$$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (1)$$

Donde P(x) es la la probabilidad de ocurrencia de la variable n,  $\sigma$  es la desviación estándar  $\sigma^2$  es la varianza,  $\mu$  es la media. Una vez obtenidos los valores, se genera el procedimiento definido por el siguiente pseudocódigo, con la finalidad de encontrar los valores con mayor probabilidad.

### 2.2.1.3 Máxima probabilidad de distribución normal.

En el segundo enfoque es aplicado a la segunda lista de datos, en este caso la comida, este enfoque se denomina Máxima probabilidad de distribución normal.

El método se basa en el postulado de que los valores de los parámetros desconocidos son aquellos que producen una probabilidad máxima de observar los datos medidos. Suponiendo que las mediciones son independientes entre si, la cantidad:

$$P = \prod_{i=1}^N P(x_i) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} \quad (2)$$

La ecuación 2, también se puede expresar de la siguiente manera:

$$L(\mu, \sigma | x_1, x_2, \dots, x_n) = L(\mu, \sigma | x_1) x_1 \dots x_n L(\mu, \sigma | x_n) \quad (3)$$

Sustituyendo la ecuación 1 se obtiene:

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} x, \dots, x \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (4)$$



Aplicando  $\ln$  en la ecuación 3 y 4

$$\ln(L(\mu, \sigma|x_1, \dots, x_n)) = \ln\left[\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_1-\mu)^2}{2\sigma^2}} \dots \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_n-\mu)^2}{2\sigma^2}}\right] \quad (5)$$

Una vez aplicado  $\ln$ , podemos expresar la ecuación 5, de la siguiente manera.

$$-\frac{1}{2}\ln(2\pi) - \ln(\sigma) - \frac{(x_i - \mu)^2}{2\sigma^2} \quad (6)$$

Entonces.

$$-\frac{1}{2}\ln(2\pi) - \ln(\sigma) - \frac{(x_i - \mu)^2}{2\sigma^2} - \dots - \frac{1}{2}\ln(2\pi) - \ln(\sigma) - \frac{(x_n - \mu)^2}{2\sigma^2} \quad (7)$$

Simplificando obtenemos:

$$-\frac{n}{2}\ln(2\pi) - n\ln(\sigma) - \frac{(x_1 - \mu)^2}{2\sigma^2} - \dots - \frac{(x_n - \mu)^2}{2\sigma^2} \quad (8)$$

Para encontrar la máxima probabilidad se determina el valor de la derivada con condición cero.

$$-\frac{\partial}{\partial \mu} \ln[L(\mu, \sigma|x_1, \dots, x_n)] \quad (9)$$

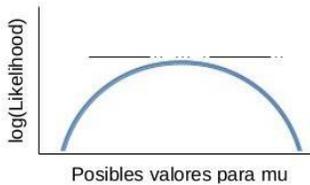


Figura 5. Likelihood respecto  $\mu$ .

Entonces aplicando la derivada a la ecuación 9 obtenemos:

$$\frac{1}{\sigma^2} [(x_1 + \dots + x_n) - n\mu] \quad (10)$$

Ahora es necesario obtener la derivada de la función de la probabilidad con respecto a sigma.

$$-\frac{\partial}{\partial \sigma} \ln[L(\mu, \sigma|x_1, \dots, x_n)] \quad (11)$$

Aplicando la deriva y simplificando obtenemos.

$$\frac{n}{\sigma} + \frac{1}{\sigma^3} [(x_1 - \mu) + \dots + (x_n - \mu)^2] \quad (12)$$

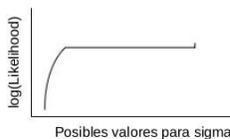


Figura 6. Likelihood respecto  $\sigma$ .



Entonces sabiendo que el máxima probabilidad es cuando la pendiente es cero, igualamos la ecuación 10,12 tenemos:

$$0 = \frac{1}{\sigma^2}[(x_1 + \dots + x_n) - n\mu] \quad (13)$$

$$0 = \frac{n}{\sigma} + \frac{1}{\sigma^3}[(x_1 - \mu) + \dots + (x_n - \mu)^2] \quad (14)$$

Entonces despejando la ecuación, tenemos que la máxima probabilidad estimada por  $\mu$  es la media de los datos.

$$\mu = \frac{x_1 + \dots + x_n}{n} \quad (15)$$

Despejando la ecuación, tenemos la estimación máxima probabilidad dada por  $\sigma$  es la derivación estándar de los datos.

$$\sigma = \sqrt{\frac{(x_1 - \mu)^2 + \dots + (x_n - \mu)^2}{n}} \quad (16)$$

## 2.2.2 Visión.

El sistema de visión consta de una cámara que monitorea el movimiento del pez y calcula la distancia en pixeles que recorre. Dado que la cámara se asume fija y el entorno del pez también es estático, es posible asumir que lo único que cambia entre dos cuadros tomados por la cámara en distintos momentos es la posición del pez. Dicho esto, podemos calcular cuánto se movió entre un cuadro y otro mediante la diferencia de los dos cuadros. Esta diferencia eliminará el fondo dejando visibles los pixeles que son distintos entre ambos cuadros, que en nuestra aplicación en particular corresponderá al pez. El sistema de visión es implementado en Python sobre el sistema embebido Raspberry Pi y el seguimiento del pez es procesado en cada fotograma obtenido por la cámara.

La cámara se posiciona en la parte superior de la pecera, con el fin de requerir una menor área junto con la pecera y evitar generar sombras que podrían afectar la precisión con que se detecta al pez. Para realizar la etapa de visión se utilizó una metodología de desarrollo de software en cascada que consta de las etapas de requerimientos, diseño, implementación y resultados.

### 2.2.2.1 Requerimientos

El sistema real funcionará con una cámara fija conectada al sistema de visión mediante un cable USB.

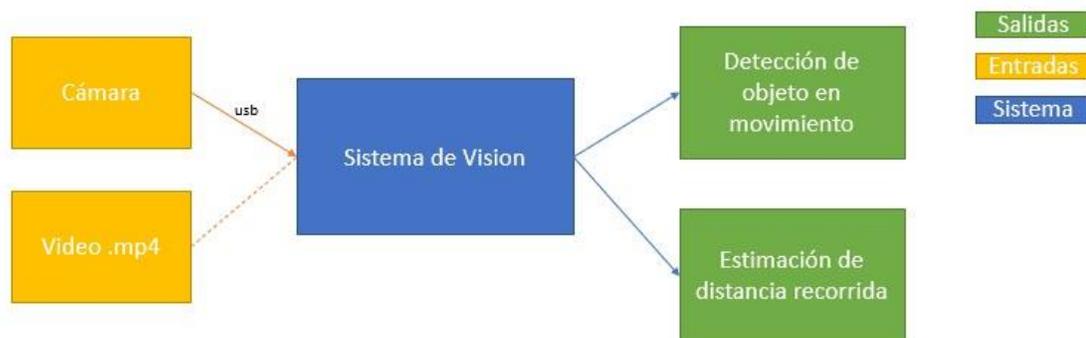


Figura 7. Diagrama de bloques del sistema de visión

Los requisitos de entrada al sistema son:

1. Cámara que soporte el protocolo de comunicación USB
2. Cámara que tome imágenes a color RGB a al menos 30 FPS
3. Soporte para posicionar la cámara y minimizar las vibraciones



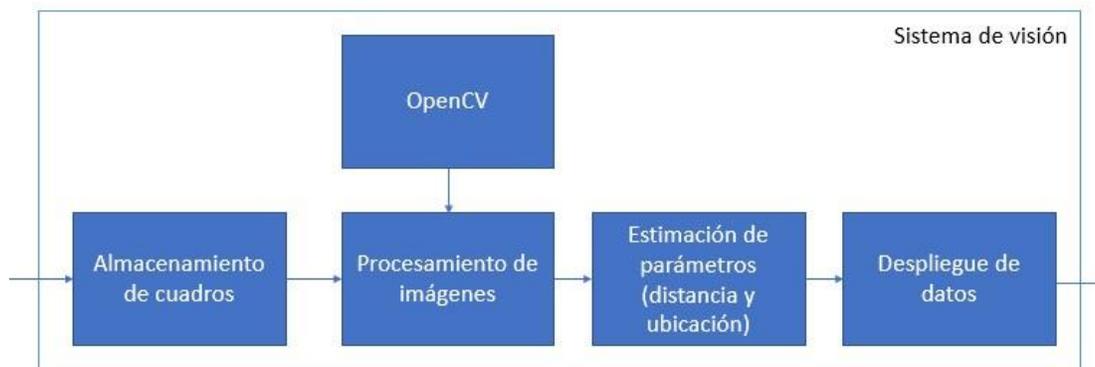
**Figura 8. Ejemplo de soporte para cámara**

Los requisitos de implementación y salida del sistema son:

1. El sistema de visión debe reconocer objetos en movimiento en un fondo estático
2. El sistema de visión debe estimar la distancia en pixeles que recorrieron los objetos entre dos cuadros
3. El sistema de visión debe aceptar como formato de entrada cualquier “stream” de una cámara de video o archivos con formato mp4
4. El sistema debe aceptar cuadros a color RGB o escala de grises

### 2.2.2.2 Diseño

Para cumplir los requisitos de implementación se seleccionó Python como lenguaje de programación ya que incluye bibliotecas de fácil acceso y libre uso para obtener datos del “stream” de video. De igual forma es compatible con OpenCV que provee interfaces para acceder a funciones de tratamiento de imágenes. El diagrama a bloques del sistema se muestra a continuación



**Figura 9. Diagrama a bloques de los componentes del sistema de visión.**



### 2.2.2.3 Implementación

La adquisición de las imágenes se realiza a partir de una cámara digital con resolución VGA (800x600 píxeles) a una tasa de refresco de 30 Hz, cuya secuencia de fotogramas es procesada, convirtiéndose primeramente cada fotograma de la paleta de color RGB a escala de grises. El fotograma actual se resta con el previo para obtener la máscara de la diferencia y así identificar al objeto en caso de haber movimiento del mismo.

Para mejorar la calidad de la máscara obtenida se aplica un filtro gaussiano y se dilata, con el fin de obtener bordes mejor definidos del objeto. Posteriormente, se realiza el etiquetado del objeto en el que se detecta el movimiento y así obtener sus coordenadas en función de su centroide. La coordenada se compara con la previa o se asigna como inicial en caso de ser el primer fotograma procesado para obtener la diferencia en posición del objeto y agregarla al acumulador. Así mismo, a partir de las coordenadas del objeto, se crea un *bounding box* para encerrar y resaltar al objeto. Finalmente, el resultado tanto de la distancia como del *bounding box* es mostrado en pantalla. A continuación, se muestra el diagrama de flujo del algoritmo implementado.

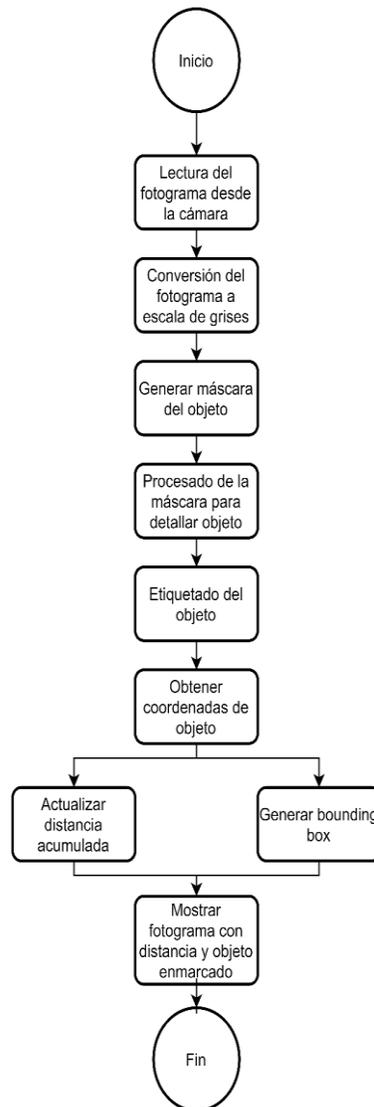
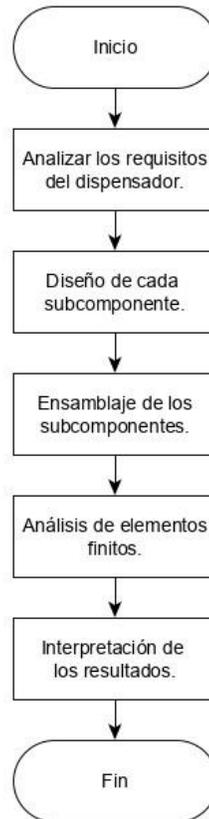


Figura 10. Diagrama de flujo del algoritmo de detección de movimiento



### 2.2.3 Modelado 3D.

A continuación, se muestra el diagrama de flujo correspondiente al diseño mecánico del dispensador automático de comida.



**Figura 11. Diagrama de flujo correspondiente al diseño mecánico.**

La descripción detallada de cada una de las etapas mostradas en el diagrama de flujo se presenta en la sección de resultados correspondiente.

### 2.2.4 Aplicación móvil y base de datos.

Como un requisito para este proyecto se pidió implementar al menos una característica inteligente en la pecera, para esto se optó por desarrollar una aplicación móvil en donde el usuario pudiera monitorear a su pez desde fuera de casa gracias a la webcam implementada, además de poder alimentarlo a la hora que desee gracias al dispensador automático y el control temporal incorporado en la aplicación móvil, para esto, la tarjeta de desarrollo se conecta a internet mediante cable ethernet o por medio del Wifi y manda la información recopilada por la webcam a una base de datos, la aplicación se conecta a la misma base de datos para poder intercambiar información con la tarjeta Raspberry, como se puede observar en la Figura 12.

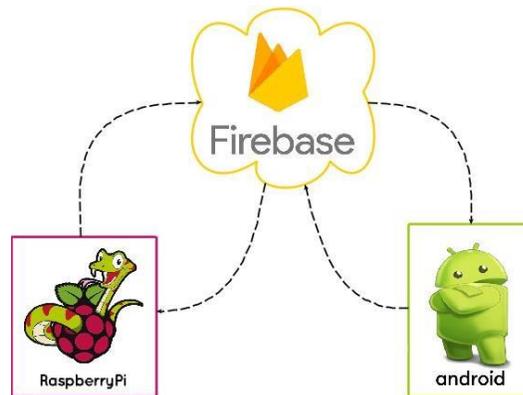


Figura 12. Conexión Raspberry con la aplicación móvil por medio de FireBase.

### 3. Resultados

#### 3.1 Modelo de alimentación.

Aplicando el enfoque de Máxima probabilidad de distribución normal, el resultado con mayor probabilidad de que el pez ingiera algún tipo de comida es aproximadamente en 580 metros, dentro de las diferentes cantidades, de comida la cantidad con mayor probabilidad de que ingiera alimentos es 5 piezas en 447 minutos o 7.45 horas en la imagen, se muestra la distribución normal de los datos.

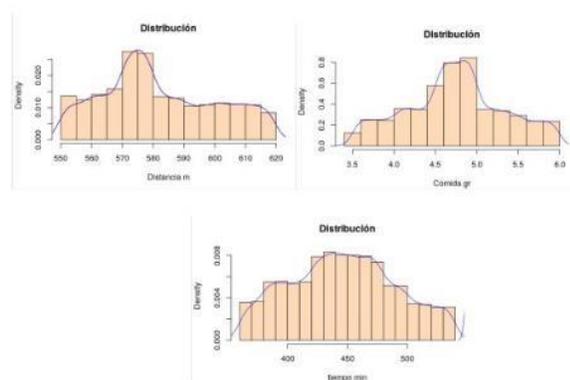


Figura 13. Distribución Distancia-Comida-Tiempo.

En la siguiente figura, se observa la distribución de la probabilidad de ocurrencia de las diferentes columnas de los datos.

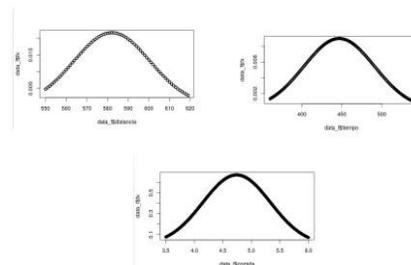


Figura 14. Distribución de probabilidad Distancia-Comida-Tiempo.



### 3.2 Visión.

Debido a la situación actual, no fue posible realizar pruebas físicas del sistema y no se encontraron secuencias de video que cumplieran los requisitos del proyecto (pecera con cámara en la parte superior de la pecera fija y peces en movimiento), por lo que se utilizó una grabación de una pecera con varios peces dentro. El video está en formato .mp4 y se reproduce a 30 FPS – dentro del requerimiento inicial. Cada cuadro tiene un alto de 480 pixeles por 854 pixeles de ancho. Las imágenes de entrada se redimensionan dentro del sistema para homologar el proceso independientemente de la resolución de entrada.

En la Figura 15 se muestra el objeto correctamente delimitado en un rectángulo verde, así como el contador de la distancia recorrida y una bandera de “movimiento” que devuelve si hay un objeto en movimiento en la secuencia de video para el fotograma actual. Cabe destacar que la distancia se determina en pixeles dado que, al implementarse de manera física, se debe calibrar la cámara con las dimensiones de la pecera y así mostrar el resultado de la distancia recorrida en el sistema de unidades que el usuario defina.

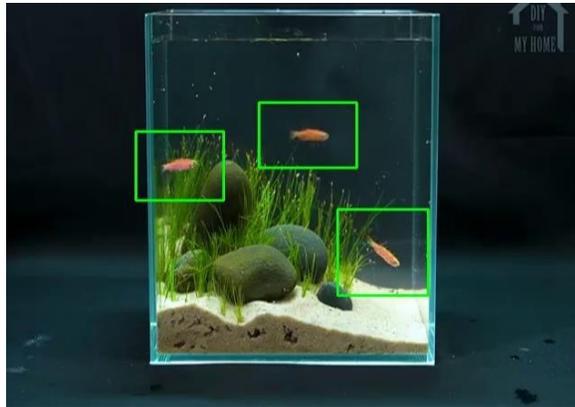


Figura 15. Detección de peces en movimiento.

### 3.3 Modelado 3D

Como primera fase del diseño se eligió el motor a utilizar. Se decidió utilizar un motor a pasos de 42 mm y un torque de  $0.186N \cdot m$  ya que se concluyó que estas características eran suficientes debido a que solo desplazaría la comida a través de la camisa del dispensador. Debido a que todas las piezas se fabricarían mediante impresión 3D, se tomó como base la forma del motor para empezar el diseño de los demás componentes, ya que esto ayudaría a la hora de ensamblar todas las piezas.

Lo primero que se diseñó fue la camisa del dispensador. En la parte superior tiene un depósito en el cual se coloca la comida del pez. Este compartimiento se encarga de llevar la comida en un tornillo sin fin, el cual la desplazará hasta ser depositada en la pecera. Para evitar la entrada de cualquier contaminante a la comida, se diseñó una tapa la cual cubre la comida una vez que se llenó el depósito.

El diseño del tornillo sin fin se realizó pensando en que la comida del pez no llegará a atorarse, por tal motivo el paso es de 1 cm y la longitud de la hélice es también de 1 cm. La longitud del tornillo es de 5 cm. Para evitar que se presentará algún desplazamiento entre la flecha del motor y el tornillo sin fin, se diseñó una pequeña chaveta. De esa manera quedaran fijos dichos componentes.

Para realizar el diseño del soporte principal, se realizó una investigación previa para determinar la mejor manera de fijar nuestros componentes con la pecera. Se determinó que la mejor forma era por medio de una sujeción tipo C, ya que esto nos daría el soporte necesario que se necesitaba. El tornillo

seleccionado es un M14x1.5. Continuando con el soporte principal, se diseñó una base donde se coloca el puente H basado en el L293 y una ranura para colocar el conector DS-01.

Para cubrir el compartimento donde van colocados los componentes eléctricos, se diseñó una tapa la cual se coloca en el soporte principal para ocultar dichos componentes.

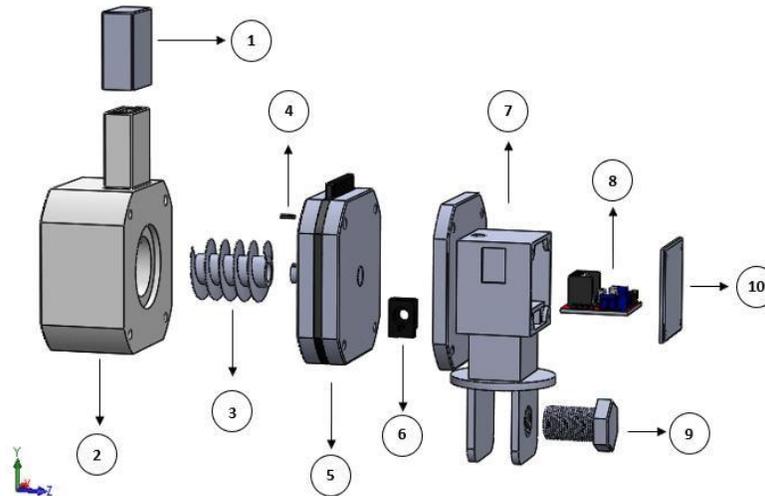


Figura 16. Despiece del ensamble.

Tabla 1. Lista de los componentes que conforman el dispensador.

ID	Componente	Descripción
1	Tapa (alimentador)	Componente que evita la entrada de polvo a la comida.
2	Camisa	Cilindro que soporta el desplazamiento de la comida.
3	Tornillo sin fin	Desplaza la comida dentro de la camisa para ser arrojada a la pecera.
4	Chaveta	Evita el deslizamiento entre la flecha del motor y el tornillo sin fin.
5	Motor a pasos	Actuador responsable de proveer el par necesario para mover el tornillo.
6	Conector FC-01	Conector 3.5 mm para alimentación CD.
7	Soporte principal	Base que fija y sostiene todos los componentes a la pecera.
8	Puente H	Controlador de potencia eléctrica basado en el L293.
9	Tornillo de fijación	Fija el soporte principal con la pecera.
10	Tapa del soporte	Cubre los componentes eléctricos colocados en el soporte principal.



Figura 17. Renderizado del dispensador fijado en la pecera.



### 3.3.1 Análisis estático

Considerando las dimensiones de la entrada del depósito de comida, es correcto suponer que el llenado del mismo debe llevarse a cabo mediante un dispositivo auxiliar, p. ej., un embudo. Este caso de uso en particular podría implicar la aplicación de una fuerza vertical (cortante) en el dispensador si el embudo se apoya en la entrada del depósito de comida. Nótese que el diseño propuesto del dispensador se asemeja a una viga en voladizo. En ese sentido, es posible estudiar la flexión producida en el componente cuando el extremo libre de este último es sometido a la fuerza previamente descrita.

Por lo anterior, se optó por realizar un *análisis de elementos finitos* con el propósito de determinar el punto crítico del dispensador considerando la situación planteada. Para dicho propósito, se optó por utilizar el paquete Solidworks 2018. En general, el *objetivo* es determinar si el esfuerzo de Von Mises en el punto crítico del componente supera el límite de fluencia del material seleccionado cuando el dispensador es sometido a la condición de carga descrita.

Debido al costo computacional que requeriría la simulación del ensamble, este último se aproximó mediante un sólido con la misma geometría y dimensiones. Las restricciones fueron colocadas en aquellas áreas destinadas a la sujeción del dispensador con la pecera (ver Fig. 18). Con el fin de estudiar el comportamiento del componente bajo un escenario relativamente pesimista, la fuerza vertical aplicada se estableció en un valor de 40 N, simulando que el usuario apoya el peso de su brazo mientras sostiene con firmeza el embudo durante el llenado del depósito de comida.

Con el fin de lograr bajos costos de fabricación, se consideró la manufactura aditiva (impresión 3D) como el enfoque idóneo para la fabricación de los subcomponentes del ensamble. Tomando en cuenta lo anterior, el material seleccionado para la simulación es el polímero ABS (Acrylonitrile Butadiene Styrene), cuyo límite de fluencia es de  $29.6 \times 10^6 Pa$ . Por su parte, se utilizó el mallado automático que provee el software con el fin de aprovechar el algoritmo integrado que estima un tamaño de malla que considera el volumen, área de superficie y otros detalles geométricos (ver Fig. 18).

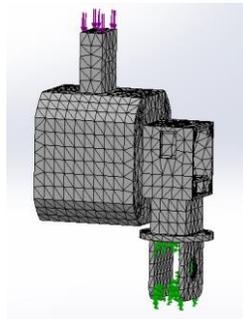


Figura 18. Mallado automático generado por el software Solidworks.

Con las consideraciones anteriormente descritas, se procedió a ejecutar la simulación. Los resultados se muestran en la Fig. 19.

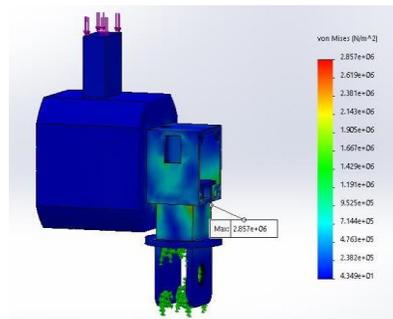
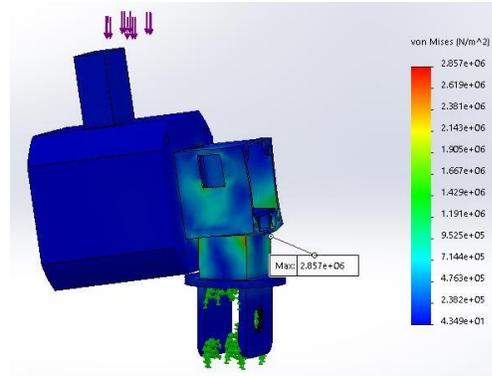


Figura 19. Punto crítico localizado en el soporte del dispensador.

De acuerdo a la Fig. 19, el esfuerzo máximo de Von Mises alcanza un valor de  $2.857 \times 10^6 \text{ MPa}$ . La Fig. 20. ilustra la deformación generada (sobrestimada) al aplicar la fuerza vertical de 40 N.



**Figura 20. Ilustración (sobrestimada) de la deformación elástica producida.**

El factor de seguridad se obtiene como sigue:

$$FS = \frac{\sigma_{fl}}{\sigma_e} = \frac{29.6 \times 10^6 \text{ Pa}}{2.857 \times 10^6 \text{ Pa}} = 10.36 \quad (17)$$

El resultado obtenido nos asegura que el caso de uso planteado no implica riesgo alguno para la integridad del componente. Dicho de otras palabras, ya que el esfuerzo máximo de Von Mises no excede el límite de fluencia del material, la deformación generada se denomina elástica, es decir, la deformación desaparece al retirar la fuerza.

Dependiendo de los requerimientos específicos del cliente, es posible aumentar el factor de seguridad del ensamblaje mediante dos enfoques. Por un lado, es posible seleccionar otro material con un límite de fluencia mayor. O bien, se puede redimensionar la sección del dispensador donde se encuentra el punto crítico. Se elegirá aquel enfoque que no genere conflicto con algún otro requerimiento. No obstante, dado que los posibles casos de uso del dispensador de comida son limitados, no se considera necesario aumentar la seguridad del ensamblaje.

### **3.4 Componentes adicionales**

Los demás componentes modelados no serán expuestos a trabajo mecánico constante por lo que se considera que no es necesario someterlos a pruebas de esfuerzos. Los componentes modelados serán listados a continuación junto a una breve explicación de su funcionalidad:

- Soporte de la cámara: Elemento empleado para asegurar la posición de la cámara en un punto por encima de la pecera. Se debe asegurar que la cámara no cambie de posición de manera repentina.
- Gabinete eléctrico: Elemento en el que algunos componentes electrónicos serán aislados de las condiciones de humedad. Aquí se almacenará el sistema controlados, la fuente de alimentación y un convertidor de DC-DC. El gabinete deberá contar con ventanas lo suficientemente grandes como para que los cables de alimentación a corriente alternan y unión de la cámara pasen, pero lo suficientemente pequeños como para que la humedad no se filtre a los componentes.

Los materiales elegidos para estos elementos son ABS para el gabinete y PLA para el soporte de la cámara. Estos materiales fueron elegidos por su bajo coste y buena resistencia en las condiciones en que se planean ser usados. Realizar el soporte de la cámara con PLA resultara en una reducción de costos muy significativa ya que en el mercado los soportes de cámaras de marcas chinas oscilan entre los \$1000 y \$2000 pesos y mediante manufactureras el precio puede llegar a \$520 pesos mexicanos. Las Figuras 21 y 22 muestran los componentes ensamblados por separado y la Figura 23 muestra el ensamble completo.



Figura 21. Soporte de la cámara para una cámara Logitech C525.

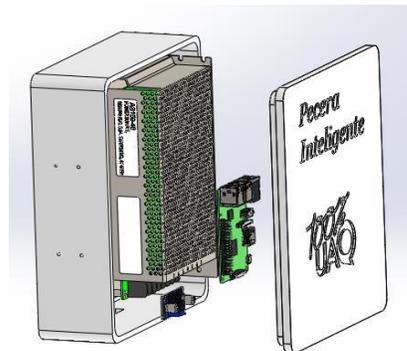


Figura 22. Ensamble del gabinete con los componentes que contendrá.

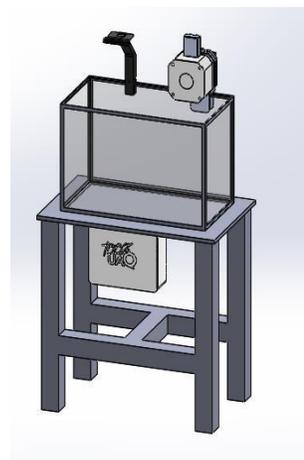


Figura 23. Ensamble completo del sistema montado en una pecera.



### 3.5 Aplicación móvil y base de datos.

Esta parte del proyecto constó de tres secciones: 1) Crear la base de datos, 2) Desarrollar la aplicación móvil y realizar la conexión con la base de datos y 3) Conexión entre la Raspberry y la base de datos.

#### 3.5.1 Creación de la base de datos.

Para la creación de la base de datos se hizo uso de la plataforma Firebase. Firebase es una plataforma gratuita para el desarrollo de aplicaciones web y aplicaciones móviles desarrollada por Google. En la Figura 24 se puede observar el diseño y contenido de la base de datos, a continuación, se describe cada uno de estos datos.

- H1 – Se utiliza para registrar la primera hora programada para alimentar al pez (H1:M1).
- H2 – Se utiliza para registrar la segunda hora programada para alimentar al pez (H2:M2).
- HoraComida – Establece el número de comidas al día (Puede ser 1 o 2).
- M1 – Se utiliza para registrar la primera hora programada para alimentar al pez (H1:M1).
- M2 – Se utiliza para registrar la segunda hora programada para alimentar al pez (H2:M2).
- NombrePez – Registra el nombre del pez.
- TPez – Registra el tamaño del pez.
- Comida – Manda señal para alimentar al pez por horario programado o de manera manual.
- Distancia – Registra la distancia recorrida por el pez. □ EstadoPez – Monitorea el estado del pez.

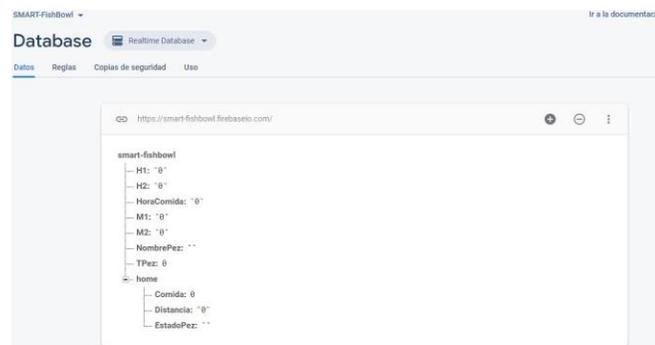


Figura 24. Vista general de la base de datos creada.

#### 3.5.2 Desarrollo de la aplicación móvil y conexión con la base de datos.

Para el desarrollo de la aplicación móvil se utilizó la plataforma Android Studio, una herramienta para desarrollar distintos tipos de aplicaciones de manera gratuita.

##### 3.5.2.1 MainActivity.

Se trata de la actividad principal, muestra el inicio de la aplicación y se presentan las opciones de las que dispone el usuario. En la Figura 25 se muestra la actividad principal, en ella el usuario puede ingresar a la parte de configuración, monitoreo o la operación manual para alimentar al pez.



Figura 25. MainActivity.

### 3.5.2.2 ConfigActivity.

En esta actividad, el usuario puede registrar el nombre de su pez, su tamaño y selecciona el número de comidas al día para su pez, al establecer estos parámetros, el usuario deberá fijar el horario de comida.

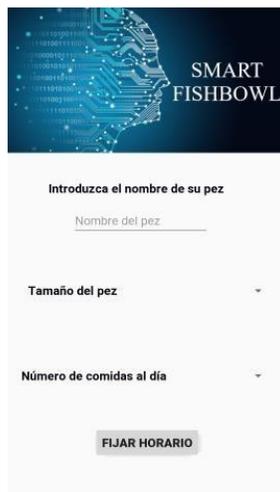


Figura 26. ConfigActivity.

### 3.5.2.3 HoraActivity.

En esta actividad el usuario programa el horario en que desea alimentar a su pez, si el usuario seleccionó solamente una comida al día, deberá establecer por lo menos el horario de la comida 1. Al pulsar el botón de finalizar, la aplicación retorna a la actividad principal indicando que la configuración fue exitosa.

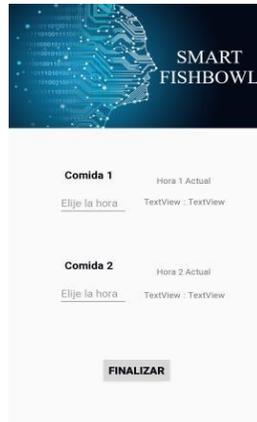


Figura 27. HoraActivity.

## 4. Conclusiones

La importancia de la modelación y simulación computacional se demuestra mediante el presente trabajo. Es de suma importancia construir un modelo matemático que contenga todos los parámetros clave del modelo físico, este simple hecho permitirá acelerar y hacer más económico el ciclo de diseño, manufactura y pruebas del producto que se esté fabricando. Hoy en día con los avances tecnológicos se tienen acceso a recursos electrónicos tales como software de modelación o paquetes de simulación para realizar un ensayo bastante preciso del fenómeno a analizar, por lo que se debe hacer uso de ellos dependiendo de la aplicación.

En particular en el presente proyecto, se observó como la modelación y simulación ha permitido tener una idea más aterrizada de lo que se quiere construir y cómo se comportará al momento de ponerlo en funcionamiento. Asimismo, durante el mismo proceso de modelación y simulación se tuvo que regresar en muchas ocasiones a un replanteamiento de ideas desde origen, ya que a través de este proceso es claro que ideas pueden llegar a ser factibles y cuáles no, ese mismo es un producto que nos deja la modelación y simulación antes de ir directamente a la fabricación.

Un proyecto como el actual, puede ayudar a expandir el concepto de automatización y control a distancia, así como la incorporación de funcionalidades inteligentes dentro de aparatos u objetos de uso común, en especial, aquellas que pueden ayudar a mascotas, niños, pacientes o incluso personas de edad avanzada. Esto no es más que algunos de los efectos del Internet de las Cosas penetrando lentamente en nuestra vida diaria.

## Referencias

- [1] Bonamente, Massilmiliano. Statistics and Analysis of Scientific Data. SPRINGER, 2018.