MAESTRÍA EN SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

Cámara IoT para detección facial con conectividad Wi-Fi

Autor:
Esp. Ing. Mauricio Barroso Benavides

Director:
Mg. Ing. Gonzalo Sanchez (FF.AA, FIUBA)

Jurados:
Mg. Ing. Edgardo Torrelli (FIUBA)
Mg. Lic. Leopoldo Zimperz (FIUBA)
Mg. Ing. Sebastián Guarino (FIUBA)

Este trabajo fue realizado en la ciudad de Tupiza,
entre junio de 2021 y diciembre de 2022.

Resumen

Esta memoria describe el proceso de desarrollo de un dispositivo electrónico compuesto principalmente por un módulo de procesamiento con conectividad Wi-Fi y una cámara, que puede capturar imágenes para procesarlas mediante algoritmos de Inteligencia Artificial y así detectar rostros humanos. Los datos generados por el dispositivos son transmitidos hacia servidores en la nube encargados de procesar, almacenar y facilitar su visulización para los usuarios finales. La principal aplicación de este trabajo es generar información sobre la presencia de personas para, por ejemplo, activar otros dispositivos como bocinas o mecanismos de cierre/apertura de puertas.

En la realización del presente trabajo se utilizaron conocimientos adquiridos a lo largo de la carrera como desarrollo de firmware, visión artificial, diseño de hardware, sistemas distribuidos, gestíon de proyectos y gestión de tecnología.

Agradecimientos

A Gonzalo Sanchez, director de este trabajo, por sus valiosos consejos y criterios que se ven reflejados a lo largo de este desarrollo.

A los profesores de la Maestria en Sistemas embebidos, por contribuir en mi formacion academica con sus conocimientos y experiencias.

Índice general

Índice de figuras

Índice de tablas

Este trabajo se lo dedico a mi familia, eternas gracias por sus apoyo incondicional en cada etapa de mi vida, Ustedes son la luz que guia mi camino

.

Capítulo 1

Introducción general

1.1. Deteccion facial

La vision artificial es un campo cientifico interdisciplinario que se encarga de como los sistemas computacionales pueden obtener un entendimiento de alto nivel de imagenes y videos digitales, para comprender y automatizar tareas como lo haria un sistema de vision humano. Las tareas que ejecuta un sistema de vision artificial son de adquisicion, procesamiento, analisis y entendimiento de imagenes. Un sistema de vision artificial esta compuesto de los siguientes elementos.

Uno de los campos de estudio mas importantes de la vision artificial es la deteccion facial. La deteccion facial puede ser considerada como un caso particular de la deteccion de objetos y tiene los objetivos de detectar y localizar todos los rostros humanos contenidos en una imagen digital. En la figura ...

Hoy en dia, muchos dispositivos comerciales y profesionales como smartphones, tablets y robots, utilizan la deteccion facial como primer paso para otro tipo de aplicaciones mas complejas, entre las que destacan: - reconocimiento facial, - computacion afectiva, - grabacion de video inteligente

1.2. Inteligencia Artificial, Aprendizaje Automatico y Aprendizaje Profundo

AI (*Artificial Intelligence*, Intelegencia Artificial), ML (*Machine Learning*, Aprendizaje Automatico) y DL (*Deep Learning*, Aprendizaje Profundo), son terminos muy utilizados hoy en dia en el mundo del desarrollo tecnologico [ref]. Aunque estos terminos son muy parecidos, entre ellos existen dependencias que pueden ser visulizadas con ayuda de la figura 1.1.

AI es un area de la computacion que permite a los sistemas computacionales imitar la inteligencia humana para entender su entorno y tomar acciones que maximicen sus posibilidades de lograr sus objetivos [ref]. Sus aplicaciones más importante se ecuentran en la áreas de comercio, educación, robótica, salud, agricultura, automotriz y finanzas[https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/artificial-intelligence-applications]. Todos los sistemas de inteligencia artificial reales e hipotéticos pueden ser clasificados en alguno de los siguientes tipos:

- ANI (*Artificial Narrow Intelligence, Inteligencia Artificial Estrecha*): también conocida como inteligencia artificial débil, su objetivo es llevar a cabo un solo tipo de tarea. Estos sistemas no poseen consciencia y no son manejados por

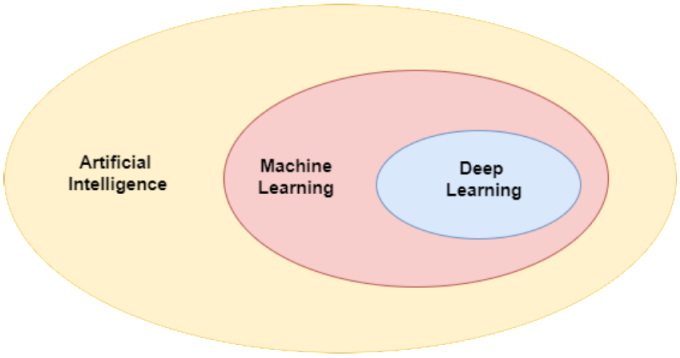


FIGURA 1.1. Diferencias entre AI, ML y DL.

sentimientos como lo haría un humano. Algunos ejemplos de ANI son los chat bots o los automóviles autónomos.

- AGI (*Artificial General Intelligence*, Inteligencia Artificial General): también conocida como inteligencia artificial fuerte, es un concepto en el que las máquinas exhiben inteligencia humana. Estos sistemas tendrían la capacidad de aprender, entender y actuar de tal manera que sería indistinguible a un humano. AIG actualmente no existe, pero es utilizado en industrias como el cine
- ASI (*Artificial Super Intelligence*, Super Inteligencia Artificial): ASI también forma parte de la inteligencia artificial fuerte. Se le considera muy poderosa por ser capaz de volverse consciente y autónoma. No solo replica el comportamiento humano, sino que lo supera. Puede pensar mejor y tener más habilidades. Sin embargo, esta tecnología aún está en desarrollo.

ML es un subconjunto de AI que utiliza algoritmos de aprendizaje estadísticos para contruir sistemas con la habilidad de aprender automáticamente y mejorar a partir de experiencias previas sin ser explícitamente programados para esto. Muchos de los servicios de recomendación utilizados por empresas con Netflix, YouTube o Spotify, utilizan ML para adapatarse a un usuario en particular y ofrecer una mejor experiencia más personalizada. Estos algoritmos pueden ser clasificados de la siguiente manera:

- Aprendizaje supervisado: se refiere al aprendizaje modelos a partir de un conjunto de datos, mejor conocidos como *dataset*, cuyas respuestas son conocidas con antelación y están asociadas a una etiqueta o *label*. Por ejemplo, el *dataset* pueden ser muchas fotografías de gatos y el *label* asociado el nombre de este animal. De esta manera el modelo es entrenado para generar predicciones de datos nuevos. En la figura 1.2 se representa gráficamente el aprendizaje supervisado.
- Aprendizaje no supervisado: es utilizado cuando los datos utilizados para el aprendizaje no tienen *labels*. Su objetivo principal es aprender acerca de los datos e inferir patrones sin ningún tipo de referencia sobre las respuestas esperadas. Es mayormente utilizado como parte del análisis exploratorio de datos [https://towardsdatascience.com/understanding-the-difference-between-ai-ml-and-dl-cceb63252a6c]. En la figura 1.3 se representa gráficamente el aprendizaje no supervisado.

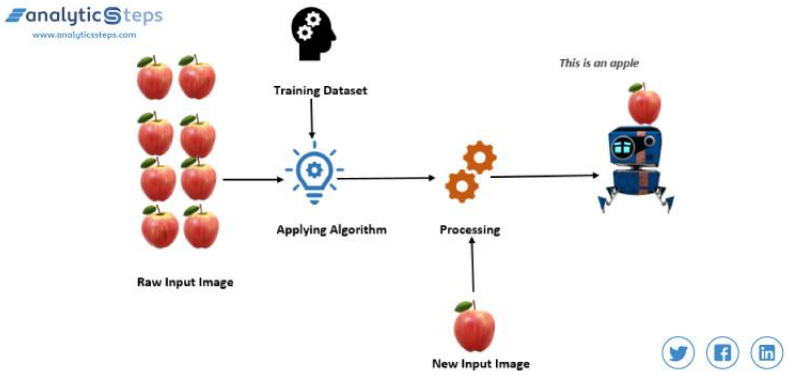


FIGURA 1.2. Aprendizaje supervisado.

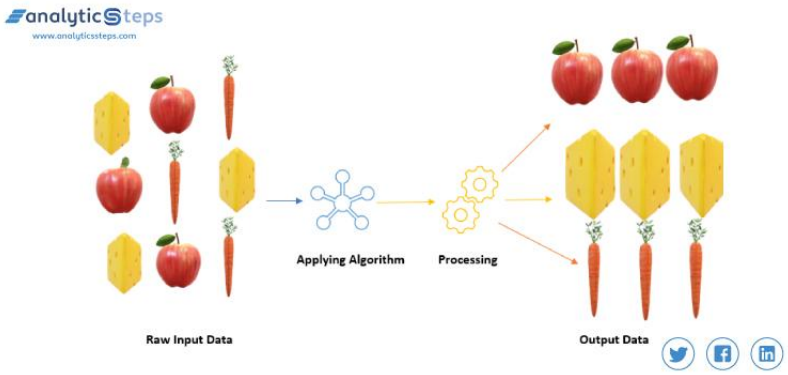


FIGURA 1.3. Aprendizaje no supervisado.

- Aprendizaje reforzado: es el aprendizaje mediante la interacción continua con el entorno con el método de prueba y error, y utiliza continuamente la retroalimentación de sus acciones y experiencias previas. Este tipo de aprendizaje utiliza recompensas si se realizan acciones correctas y penalizaciones si son incorrectas. En la figura 1.4 se representa gráficamente el aprendizaje reforzado.

FIGURA 1.4. Aprendizaje reforzado.

DL es una técnica de ML que está inspirada en la forma en la que el cerebro humano filtra información. Como DL procesa información de manera similar al cerebro humano sus aplicaciones son tareas que un humano generalmente realiza, como distinguir entre un peatón o un poste de luz en el caso de automóviles

autónomos. El componente principal de DL son las redes neuronales artificiales, que son capas de nodos interconectadas, donde existen una capa de entrada, una o varias capas ocultas y una capa de salida. En la figura 1.5 se puede observar la arquitectura de una red neuronal artificial.

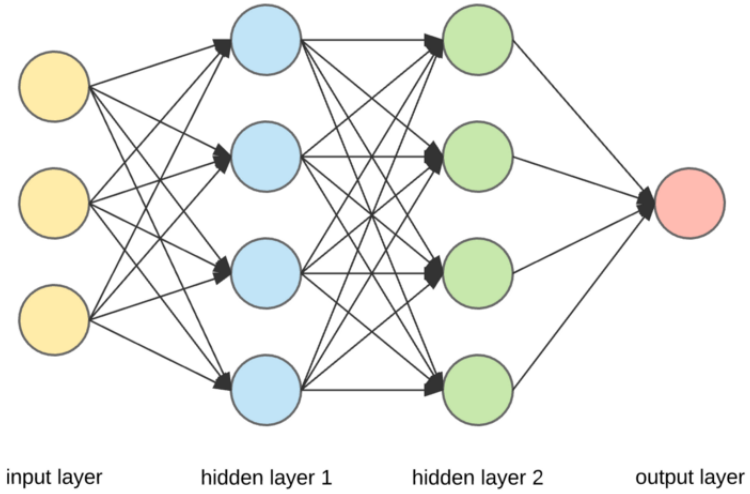


FIGURA 1.5. Arquitectura de una red neuronal artificial.

Cada uno de los nodos de las capas ocultas y de salida, tienen como entrada la salida de los nodos anteriores multiplicadas por unos términos denominados pesos o *weigths* y que sumados junto a otro término llamado sesgo o *bias* pasan por una función de activación no lineal para generar su salida. En la figura 1.6 se visualizar un nodo de las capas ocultas o de salida.

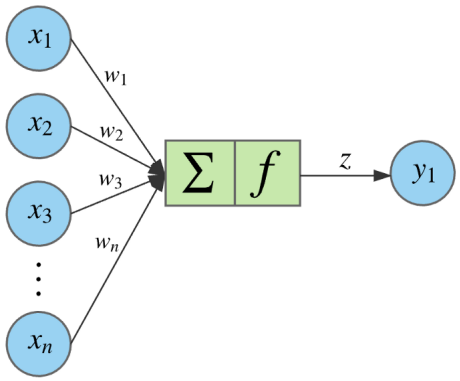


FIGURA 1.6. Nodo de una red neuronal artificial.

1.3. Redes neuronales convolucionales

También conocidas como CNN (*Convolutional Neural Networks*, Redes Neuronales Convolucionales) o ConvNet, son un algoritmo de DL que están orientadas a recibir como entrada una, asignarle *weights* y *biases* entrenables a varios aspectos/objetos en la imagen para poder diferenciarlas unas de otras. Su uso reduce el pre procesamiento de las imágenes de entrada con respecto a otros modelos de clasificación, ya que los filtros necesarios son incorporados en su arquitectura y tienen la habilidad de ser entrenados.

Computacionalmente una imagen puede ser muy dificil de procesar, esto depende del espacio de colores donde se encuentra [ref] y las dimensiones que posee. Por ejemplo una imagen RGB (*Red Green Blue*, Rojo Verde Azul) y de dimensiones 1920x1080 pixeles, tiene un tamaño de 6220800 bytes. El objetivo principal de las CNN es reducir la dimensionalidad de las imágenes de entrada, de tal forma que sean más fáciles de procesar y no pierdan sus características principales que son críticas para obtener una buena predicción.

La arquitectura de una CNN es independiente del tipo de aplicación, donde las capas que lo componen son elegidas en función de los objetivos que se persiguen. En la figura 1.7 se puede observar la arquitectura de una CNN para clasificar digitos escritos a mano.

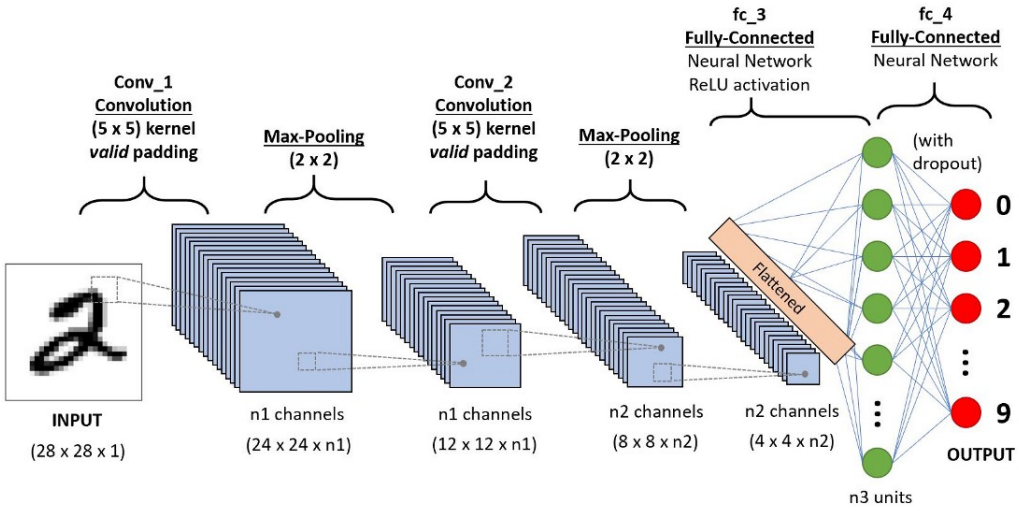


FIGURA 1.7. CNN para clasificar digitos escritos a mano.

En la arquitectura de la figura 1.7 se pueden observar tres capas principales para construir una CNN: capa de convoluciones, capa de *pooling* y capa *fully-connected*.

1.3.1. Capa de convoluciones

Esta capa esta encargada de aplicar la operación de convolución sobre las imágenes de entrada para encontrar patrones que más adelante permitirán clasificarla. La convolución de una imagen con un *kernel* no es más que la aplicación del operador punto entre ambos. Este tipo de capas se definen por:´

- El número de los *kernels* o filtros que se aplican a la imagen, que es el número de matrices por las que se van a convolucionar las imágenes de entrada.
- El tamaño de los *kernels*, donde casi siempre tienen dimensiones cuadradas e impares como 3x3 o 5x5.
- El *stride* o paso, se refiere a la forma en como el *kernel* recorre la imagen.

En la figura 1.8 se puede observar como un

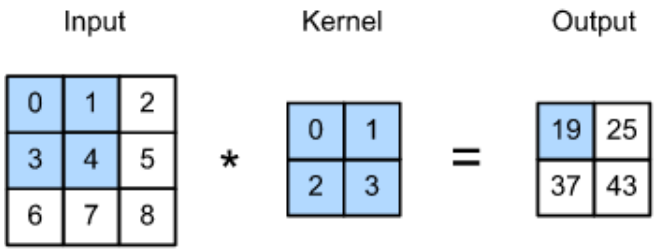


FIGURA 1.8. Convolución de una entrada de 3x3 con un *kernel* de 2x2 y stride de 1.

1.3.2. Capa de *pooling*

1.3.3. Capa *fully-connected*

1.4. Servicios en la nube

El termino servicios en la nube hace referencia a un amplio rango de servicios ofrecidos bajo demanda a companias y usuarios a traves de internet. Estos servicios estan disenados para proveer de una manera facil y asequible acceso a aplicaciones y recursos, sin la necesidad de una infrastructura o hardware propios.

Los servicios en la nube son administrados totalmente por proveedores de computacion en la nube [ref]. Estos se encuentran disponibles para los usuarios desde los servidores de los proveedores, por lo que no es necesario que una empresa aloje aplicaciones en sus propios servidores.

De manera general, existen tres tipos basicos de servicios en la nube [ref]:

- SaaS (*Software as a Service*, Software como un Servicio): en este servicio el proveedor solo proporciona el software o aplicaciones en la nube mediante internet. Los clientes tienen acceso a traves de APIs (*Application Programming Interface*, Interfaz de Programacion de Aplicaciones) o a traves de la web, que les permite interactuar de manera sencilla, sin la necesidad de gestionar, instalar ni actulizar el software.
- IaaS (*Infrastructure as a Service*, Infrastructura como un Servicio): este servicio implica la contatacion de una infraestructura de hardware a un tercero, donde varios cliente comparten los recursos de una maquina fisica. El proveedor proporciona a sus clientes el acceso a los recursos computacionales necesarios para almancenar o ejecutar tareas que pueden incluir servidores, redes, *backup*, *firewalls*, entre otros.
- PaaS (*Platform as a Service*, Plataforma como un Servicio): es un servicio que se encuentra conceptualmente entre SaaS e IaaS al eliminar la parte fisica de la infraestructura y ofrece una plataforma donde los cliente pueden crear, desarrollar, gestionar y distribuir sus aplicaciones. El proveedor es el encargado de la gestion y mantenimiento de la plataforma y permite que los clientes se dediquen exclusivamente en el desarrollo.

En la figura 1.9 se puede observar las diferencias entre estos servicios en funcion de los elementos que pueden gestionar.

[1]Imagen tomada de: https://expansionba.com.ar/2020/05/23/medidas-para-amortiguar-el-costo-energetico-en-pymes/

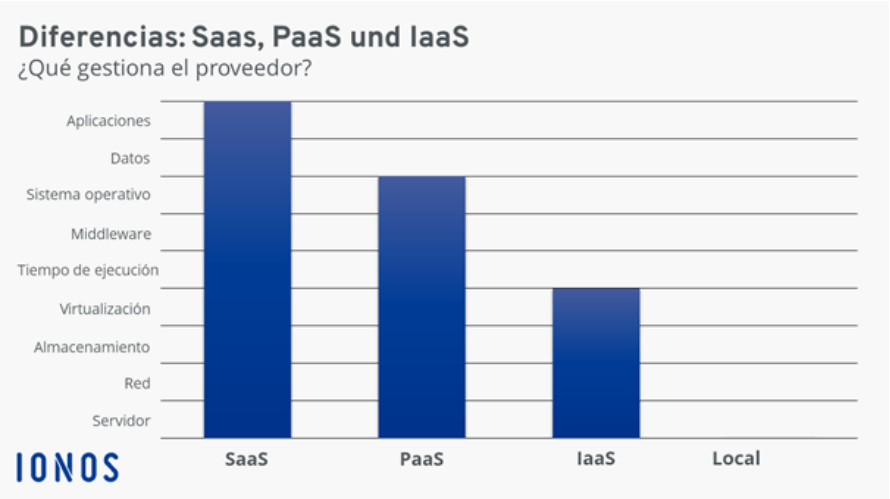


FIGURA 1.9. Capacidades de SaaS, IaaS y PaaS[1].

1.5. Motivacion

1.6. Estado del arte

1

1.7. Objetivos y alcance

El objetivo principal de este trabajo fue desarrollar un sistema embebido con la capacidad de ejecutar modelos de AI para detectar y localizar rostros humanos de imagenes digitales capturadas por su camara.

El alcance de este trabajo incluyo:

- Construccion de un prototipo de pruebas
- Desarrollo e implementacion de los modelos de AI necesarios
- Implementacion de los servicios en la nube necesarios

1.8. Requerimientos

Capítulo 2

Introducción específica

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

2.1. Estilo y convenciones

2.1.1. Uso de mayúscula inicial para los título de secciones

Si en el texto se hace alusión a diferentes partes del trabajo referirse a ellas como capítulo, sección o subsección según corresponda. Por ejemplo: "En el capítulo 1 se explica tal cosa", o "En la sección 2.1 se presenta lo que sea", o "En la subsección 2.1.2 se discute otra cosa".

Cuando se quiere poner una lista tabulada, se hace así:

- Este es el primer elemento de la lista.
- Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

Si se desea poner una lista numerada el formato es este:

1. Este es el primer elemento de la lista.
2. Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

2.1.2. Este es el título de una subsección

Se recomienda no utilizar **texto en negritas** en ningún párrafo, ni tampoco texto subrayado. En cambio sí se debe utilizar *texto en itálicas* para palabras en un idioma extranjero, al menos la primera vez que aparecen en el texto. En el caso de palabras que estamos inventando se deben utilizar "comillas", así como también para citas textuales. Por ejemplo, un *digital filter* es una especie de "selector" que permite separar ciertos componentes armónicos en particular.

La escritura debe ser impersonal. Por ejemplo, no utilizar "el diseño del firmware lo hice de acuerdo con tal principio", sino "el firmware fue diseñado utilizando tal principio".

El trabajo es algo que al momento de escribir la memoria se supone que ya está concluido, entonces todo lo que se refiera a hacer el trabajo se narra en tiempo pasado, porque es algo que ya ocurrió. Por ejemplo, "se diseñó el firmware empleando la técnica de test driven development".

En cambio, la memoria es algo que está vivo cada vez que el lector la lee. Por eso transcurre siempre en tiempo presente, como por ejemplo:

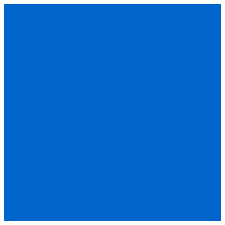
“En el presente capítulo se da una visión global sobre las distintas pruebas realizadas y los resultados obtenidos. Se explica el modo en que fueron llevados a cabo los test unitarios y las pruebas del sistema”.

Se recomienda no utilizar una sección de glosario sino colocar la descripción de las abreviaturas como parte del mismo cuerpo del texto. Por ejemplo, RTOS (*Real Time Operating System*, Sistema Operativo de Tiempo Real) o en caso de considerarlo apropiado mediante notas a pie de página.

Si se desea indicar alguna página web utilizar el siguiente formato de referencias bibliográficas, dónde las referencias se detallan en la sección de bibliografía de la memoria, utilizado el formato establecido por IEEE en [**IEEE:citation**]. Por ejemplo, “el presente trabajo se basa en la plataforma EDU-CIAA-NXP [**CIAA**], la cual...”.

2.1.3. Figuras

Al insertar figuras en la memoria se deben considerar determinadas pautas. Para empezar, usar siempre tipografía claramente legible. Luego, tener claro que **es incorrecto** escribir por ejemplo esto: “El diseño elegido es un cuadrado, como se ve en la siguiente figura:”



La forma correcta de utilizar una figura es con referencias cruzadas, por ejemplo: “Se eligió utilizar un cuadrado azul para el logo, como puede observarse en la figura 2.1”.

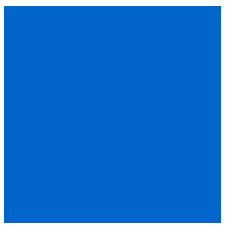


FIGURA 2.1. Ilustración del cuadrado azul que se eligió para el diseño del logo.

El texto de las figuras debe estar siempre en español, excepto que se decida reproducir una figura original tomada de alguna referencia. En ese caso la referencia

FIGURA 2.2. Imagen tomada de la página oficial del procesador[1].

de la cual se tomó la figura debe ser indicada en el epígrafe de la figura e incluida como una nota al pie, como se ilustra en la figura 2.2.

La figura y el epígrafe deben conformar una unidad cuyo significado principal pueda ser comprendido por el lector sin necesidad de leer el cuerpo central de la memoria. Para eso es necesario que el epígrafe sea todo lo detallado que corresponda y si en la figura se utilizan abreviaturas entonces aclarar su significado en el epígrafe o en la misma figura.

FIGURA 2.3. ¿Por qué de pronto aparece esta figura?

Nunca colocar una figura en el documento antes de hacer la primera referencia a ella, como se ilustra con la figura 2.3, porque sino el lector no comprenderá por qué de pronto aparece la figura en el documento, lo que distraerá su atención.

Otra posibilidad es utilizar el entorno *subfigure* para incluir más de una figura, como se puede ver en la figura 2.4. Notar que se pueden referenciar también las figuras internas individualmente de esta manera: 2.4a, 2.4b y 2.4c.

(A) Un caption. (B) Otro. (C) Y otro más.

FIGURA 2.4. Tres gráficos simples

El código para generar las imágenes se encuentra disponible para su reutilización en el archivo **`Chapter2.tex`**.

2.1.4. Tablas

Para las tablas utilizar el mismo formato que para las figuras, sólo que el epígrafe se debe colocar arriba de la tabla, como se ilustra en la tabla 2.1. Observar que

[1]Imagen tomada de https://goo.gl/images/i7C70w

sólo algunas filas van con líneas visibles y notar el uso de las negritas para los encabezados. La referencia se logra utilizando el comando `\ref{<label>}` donde label debe estar definida dentro del entorno de la tabla.

```
\begin{table}[h]
\centering
\caption[caption corto]{caption largo más descriptivo}
\begin{tabular}{l c c}
\toprule
\textbf{Especie}     & \textbf{Tamaño} & \textbf{Valor}\\
\midrule
Amphiprion Ocellaris & 10 cm           & \$ 6.000 \\
Hepatus Blue Tang    & 15 cm           & \$ 7.000 \\
Zebrasoma Xanthurus  & 12 cm           & \$ 6.800 \\
\bottomrule
\hline
\end{tabular}
\label{tab:peces}
\end{table}
```

TABLA 2.1. caption largo más descriptivo

Especie	**Tamaño**	**Valor**
Amphiprion Ocellaris	10 cm	$ 6.000
Hepatus Blue Tang	15 cm	$ 7.000
Zebrasoma Xanthurus	12 cm	$ 6.800

En cada capítulo se debe reiniciar el número de conteo de las figuras y las tablas, por ejemplo, figura 2.1 o tabla 2.1, pero no se debe reiniciar el conteo en cada sección. Por suerte la plantilla se encarga de esto por nosotros.

2.1.5. Ecuaciones

Al insertar ecuaciones en la memoria dentro de un entorno *equation*, éstas se numeran en forma automática y se pueden referir al igual que como se hace con las figuras y tablas, por ejemplo ver la ecuación 2.1.

$$ds^2 = c^2 dt^2 \left(\frac{d\sigma^2}{1 - k\sigma^2} + \sigma^2 \left[d\theta^2 + \sin^2 \theta d\phi^2 \right] \right) \tag{2.1}$$

Es importante tener presente que si bien las ecuaciones pueden ser referidas por su número, también es correcto utilizar los dos puntos, como por ejemplo "la expresión matemática que describe este comportamiento es la siguiente:"

$$\frac{\hbar^2}{2m} \nabla^2 \Psi + V(\mathbf{r}) \Psi = -i\hbar \frac{\partial \Psi}{\partial t} \tag{2.2}$$

Para generar la ecuación 2.1 se utilizó el siguiente código:

```
\begin{equation}
\label{eq:metric}
```

```
ds^2 = c^2 dt^2 \left( \frac{d\sigma^2}{1-k\sigma^2} +
\sigma^2\left[ d\theta^2 +
\sin^2\theta d\phi^2 \right] \right)
\end{equation}
```

Y para la ecuación 2.2:

```
\begin{equation}
\label{eq:schrodinger}
\frac{\hbar^2}{2m}\nabla^2\Psi + V(\mathbf{r})\Psi =
-i\hbar \frac{\partial\Psi}{\partial t}
\end{equation}
```

Capítulo 3

Diseño e implementación

3.1. Análisis del software

La idea de esta sección es resaltar los problemas encontrados, los criterios utilizados y la justificación de las decisiones que se hayan tomado.

Se puede agregar código o pseudocódigo dentro de un entorno lstlisting con el siguiente código:

```
\begin{lstlisting}[caption= "un epígrafe descriptivo"]
las líneas de código irían aquí...
\end{lstlisting}
```

A modo de ejemplo:

```
#define MAX_SENSOR_NUMBER 3
#define MAX_ALARM_NUMBER  6
#define MAX_ACTUATOR_NUMBER 6

uint32_t sensorValue[MAX_SENSOR_NUMBER];
FunctionalState alarmControl[MAX_ALARM_NUMBER]; //ENABLE or DISABLE
state_t alarmState[MAX_ALARM_NUMBER];           //ON or OFF
state_t actuatorState[MAX_ACTUATOR_NUMBER];     //ON or OFF

void vControl() {

  initGlobalVariables();

  period = 500 ms;

  while(1) {

    ticks = xTaskGetTickCount();

    updateSensors();

    updateAlarms();

    controlActuators();

    vTaskDelayUntil(&ticks, period);
  }
}
```

CÓDIGO 3.1. Pseudocódigo del lazo principal de control.

Capítulo 4

Ensayos y resultados

4.1. Pruebas funcionales del hardware

La idea de esta sección es explicar cómo se hicieron los ensayos, qué resultados se obtuvieron y analizarlos.

Capítulo 5

Conclusiones

5.1. Conclusiones generales

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se puedo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

5.2. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.