INSTITUTO POLITÉCNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA EN INGENIERÍA Y TECNOLOGÍAS AVANZADAS

Maquina de Estados

“Imágenes de clase televisada en conjunto con definición, uso y diferencia con Máquina de estados por eventos”

Mendoza Meza Manuel Everardo

Instrumentación Virtual

3MV5

upiita-ipn

FEBRERO 2022

Clase televisada

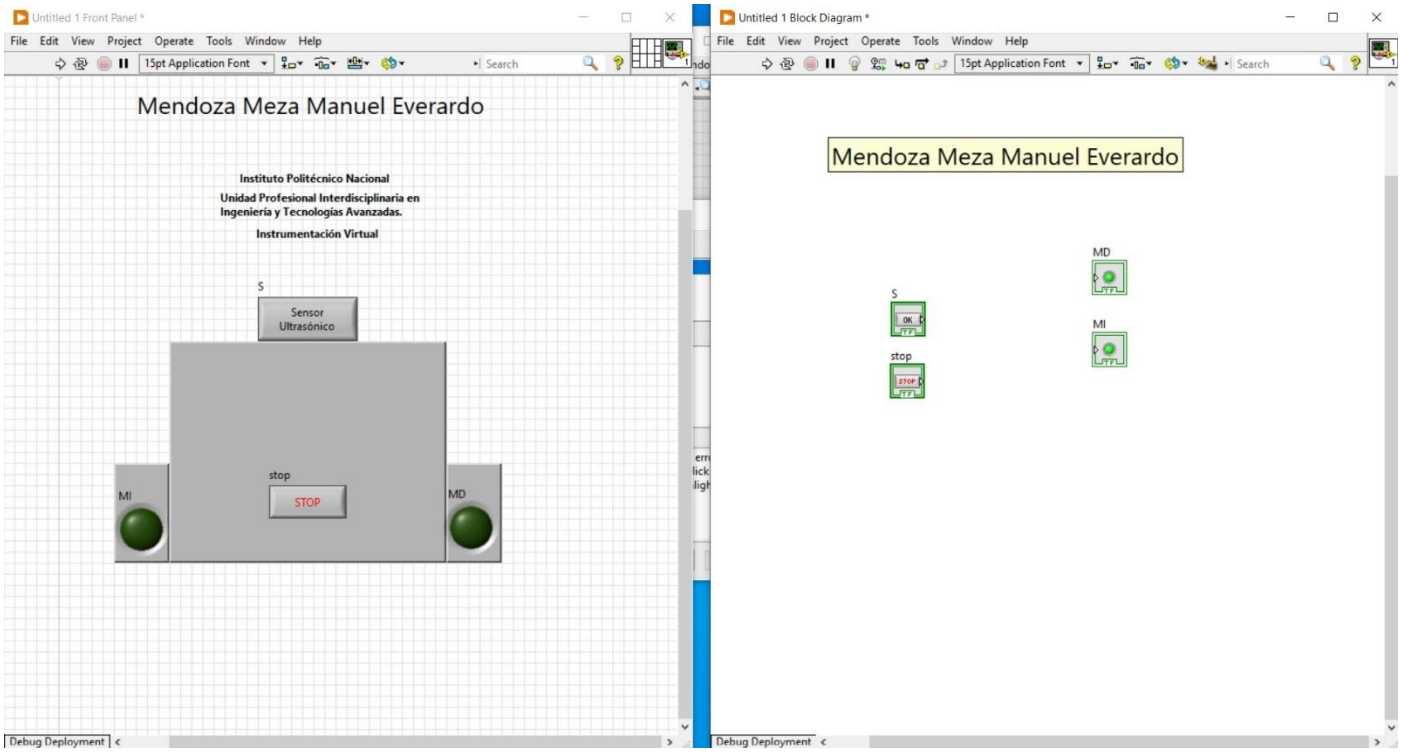
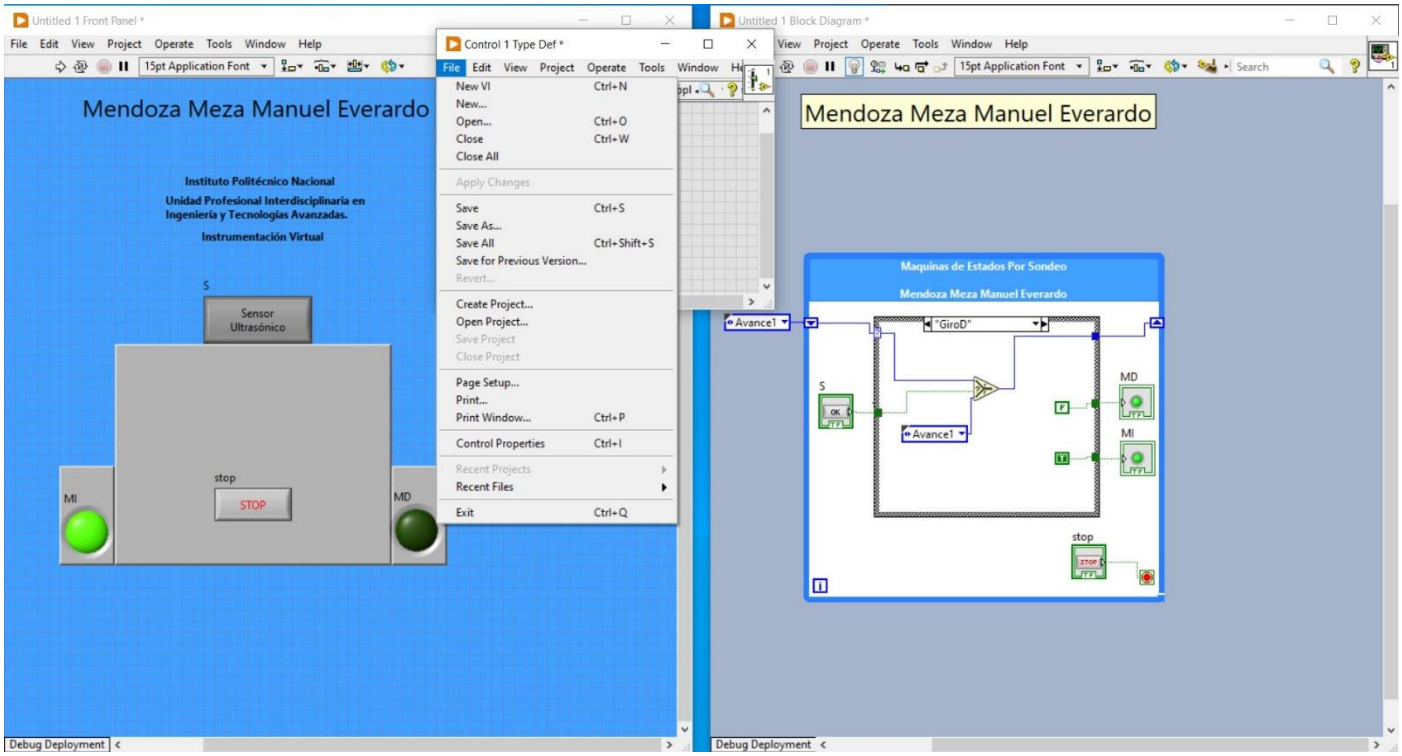


Imagen 1: Comienzo de maquina de estados por eventos



II

Imagen 2: Abriendo pestaña Control 1 Type Def.

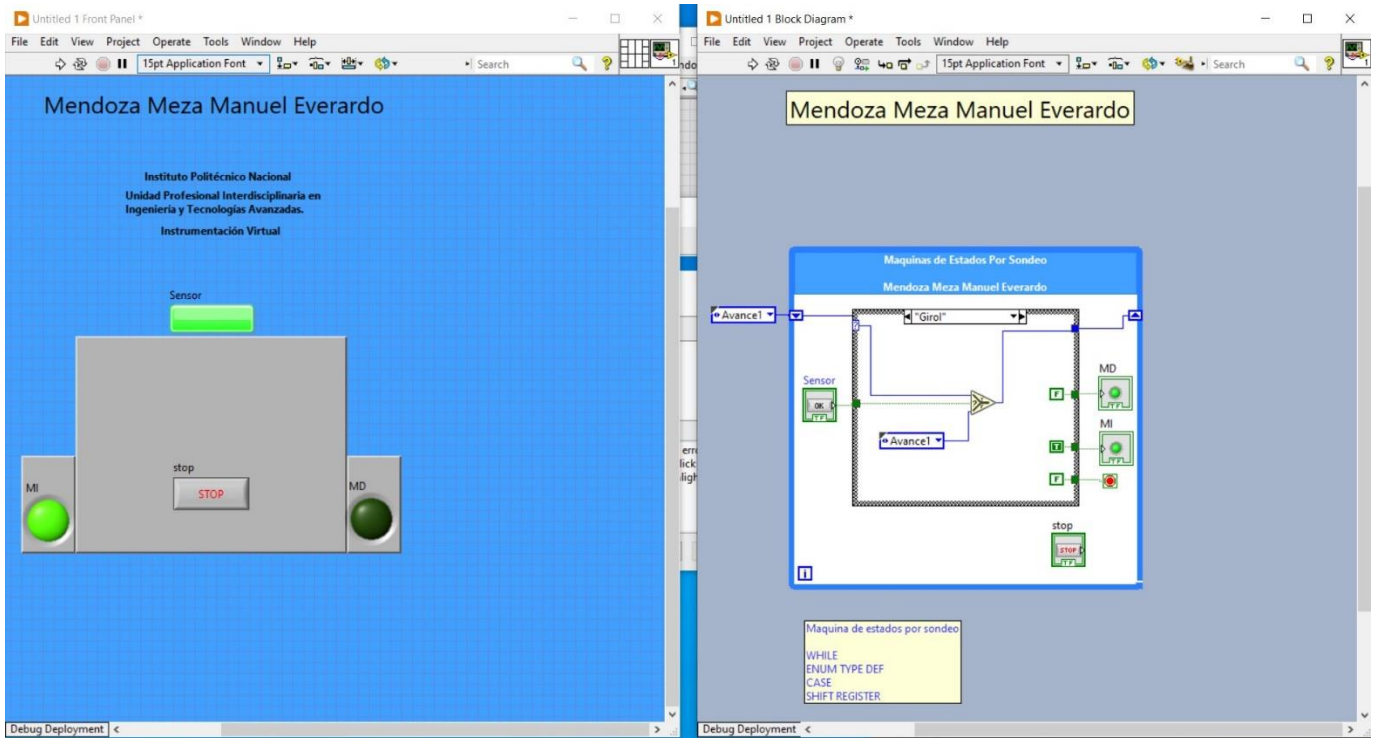


Imagen 3: Simulación de todos los casos

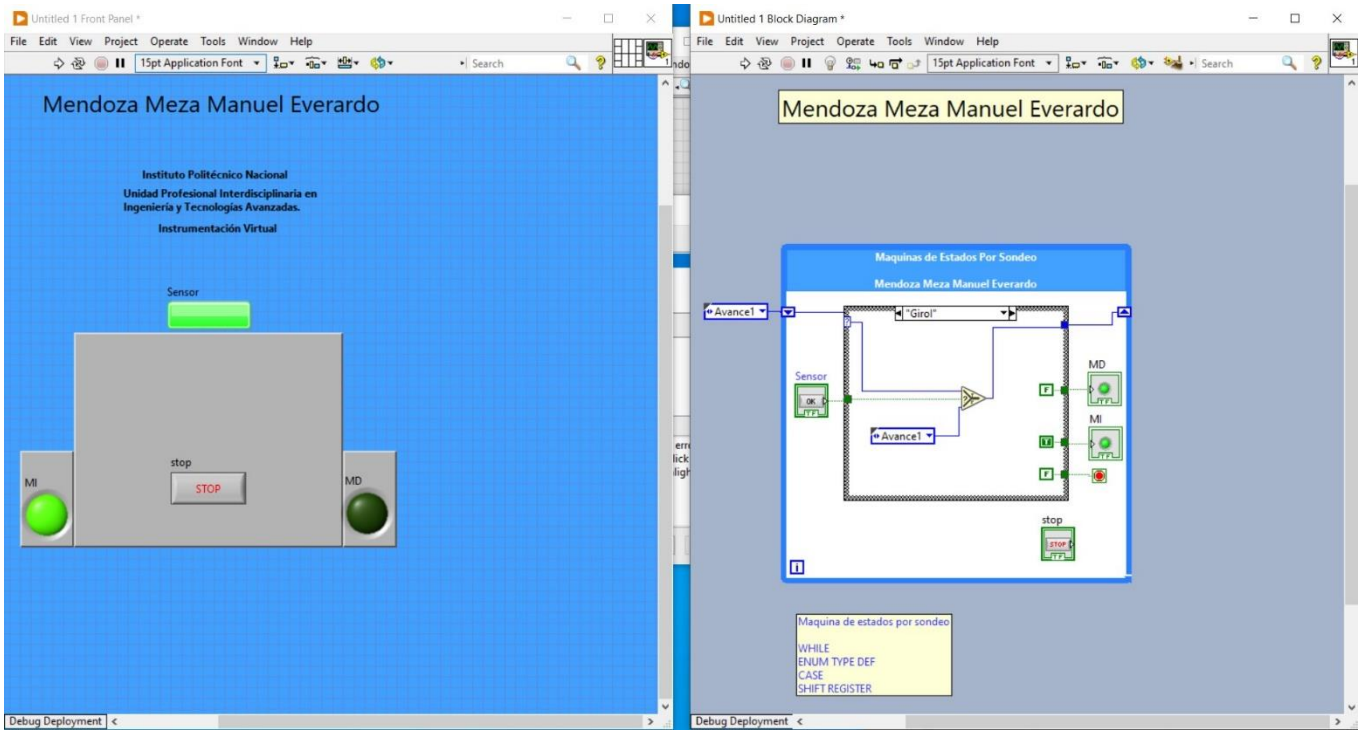


Imagen 4: Definiendo maquina de estado por sondeo e implementación de casos

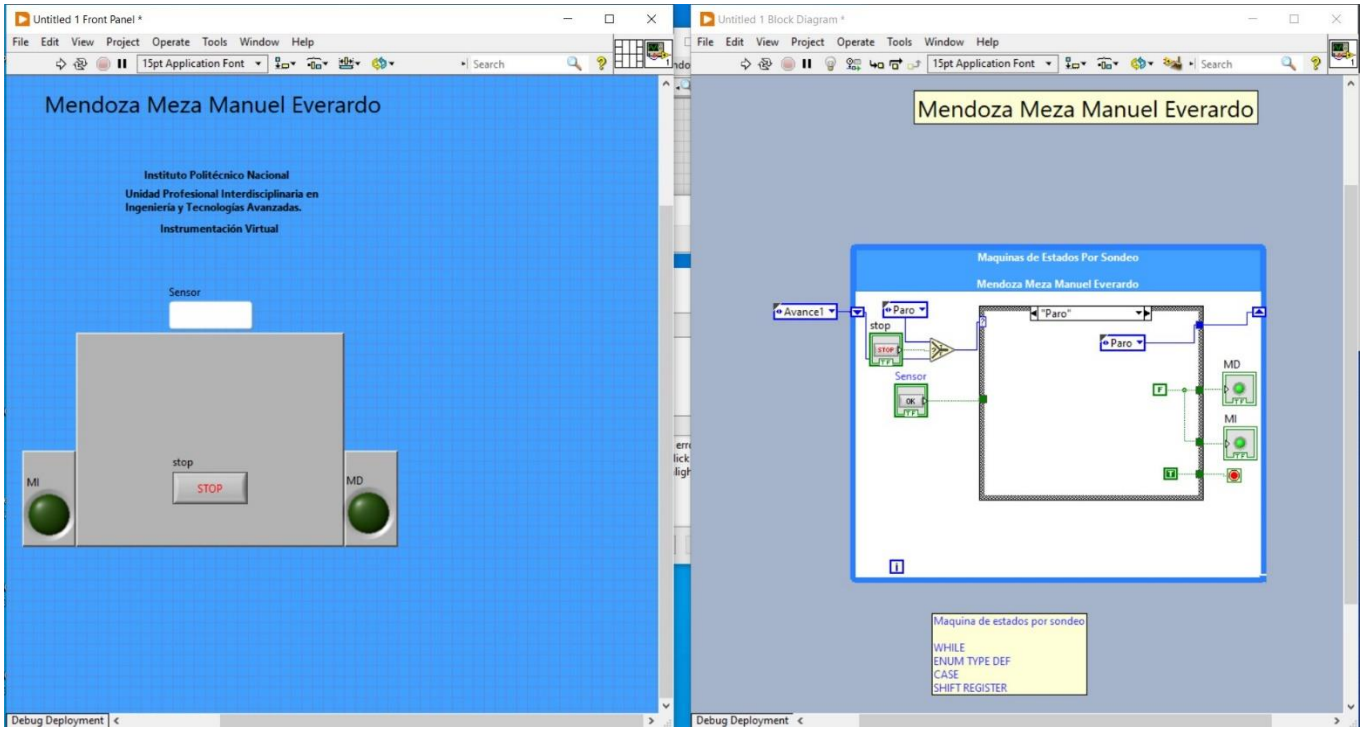


Imagen 5: Determinación de error y obtención de maquina de estados por sondeo

Investigación

Definición

La máquina de estados es una de las arquitecturas fundamentales que los desarrolladores de LabVIEW utilizan con frecuencia para crear aplicaciones rápidamente. La arquitectura de máquina de estado se puede utilizar para implementar algoritmos complejos de toma de decisiones que son representados por diagramas de estado o de flujo. Esta máquina de estados permite el flujo dinámico a los estados según valores de los estados anteriores o la información del usuario.

Esta arquitectura es ideal para aplicaciones que pueden describirse como una combinación de:

- Estados
- Lógica para toma de decisiones que determina cuando pasar de un estado en particular

Definimos a Estado como el comportamiento en el que se encuentra un programa al realizarse una tarea general. Estos estados pueden ser:

- Inicializar
- Esperar
- Ejecutar
- Ejecutar un cálculo
- Verificar el estado
- Etc.

Normalmente estos tipos de estados son determinados gracias a la declaración lógica, la cual determina cuándo pasar de un nuevo estado y a que estado debe pasar. Estos eventos se pueden utilizar para activar los movimientos del estado siguiente; o en su uso pueden ser programáticos u definidos por el usuario. Por ejemplo, presionar un botón.

Por otro lado, a cada estado en una máquina de estado hace algo único y llama a otros estados, esto requiere de comunicación, por lo que, esta comunicación depende específicamente de alguna condición o secuencia, de esta forma, la comunicación de estado necesita algún tipo de traducción en LabVIEW, la cual es la siguiente estructura:

1. Ciclo While – Ejecuta continuamente los distintos estados
2. Estructura Case – Cada caso contiene un código que se ejecutará para cada estado
3. Registro de desplazamiento – Contiene información de transición de estado
4. Código de transición – Determina el siguiente estado en la secuencia

Importantes consideraciones

Redundancia de código

Problema: La parte más difícil de crear una máquina de estado es diferenciar entre los posibles estados en el diagrama de estado. Por ejemplo, en el diagrama de estado de la máquina de Coca Cola (Fig. 4), podríamos haber tenido estados de 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 centavos en lugar de tener un estado "wait for response" que va de un estado a otro dependiendo del tipo de moneda que se inserte. Eso crearía 11 estados diferentes con exactamente el mismo diagrama de caso. El código redundante puede crear un gran problema en una aplicación más grande.

Solución : Si diferentes estados tienen el mismo diagrama de caso, intente combinarlos en un solo estado. Por ejemplo, el estado "wait for response" se crea para evitar la redundancia del código.

Uso de enum

Problema: Las enums se utilizan ampliamente como selectores de casos en las máquinas de estado. Si el usuario intenta agregar o eliminar un estado de esta enum, se romperán los cables conectados restantes de las copias de esta enum . Este es uno de los obstáculos más comunes al implementar máquinas de estado con enums.

Solución: Dos posibles soluciones a este problema son:

1. Si todas las enums se copian de la enum modificada, las interrupciones desaparecerán.
2. Crear un nuevo control con la enum y seleccionar "typedef" en el submenú. Al seleccionar typedef, todas las copias de enum se actualizarán automáticamente si el usuario agrega o elimina un estado.

Usos

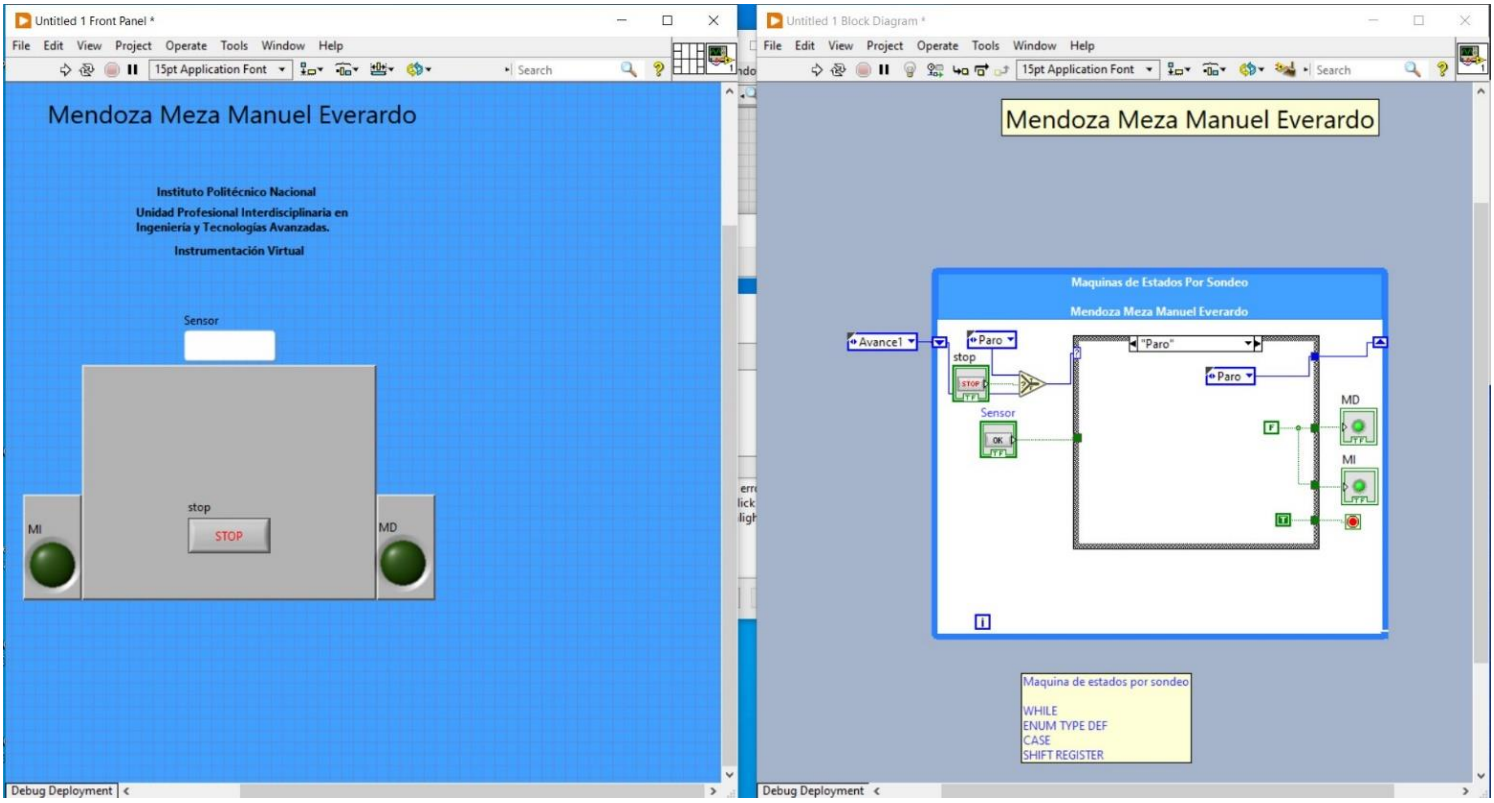
- Cuadros de diálogo de una sola página o con pestañas. Cada pestaña del cuadro de diálogo corresponde a un estado. Un usuario inicia las transiciones de estado haciendo clic en una pestaña en particular. Para cada pestaña, las acciones que el usuario puede realizar se incluyen en el estado.
- Un cajero automático (ATM). Los estados en esta aplicación podrían ser esperar la información del usuario, verificar la cantidad solicitada con el saldo de la cuenta, entregar el dinero, imprimir el recibo, etc.
- Una aplicación que realiza una medida, la registra en el disco y luego espera otra acción del usuario. Los estados en esta aplicación podrían ser esperar la información del usuario, realizar la medida, registrar los datos, mostrar los datos, etc.
- Las máquinas de estado se utilizan más comúnmente al programar interfaces de usuario. Al crear una interfaz de usuario, las diferentes acciones de usuario envían a la interfaz de usuario a diferentes segmentos de procesamiento. Cada uno de estos segmentos actuará como estado en la máquina de estado. Estos segmentos pueden conducir a otro segmento para su procesamiento posterior o esperar a otro evento del usuario. En este ejemplo, la máquina de estado monitorea constantemente al usuario para la siguiente acción a realizar.
- Cuando realizamos una red neuronal, que ayude con el aprendizaje autónomo de cierto dispositivo, suponiendo un robot seguidor, dron, un robot solucionador de laberinto etc.

Diferencias

Primero reconociendo principalmente a la maquina de estados por eventos como maquina de estados por sondeo, tenemos que observar características importantes del mismo, empezando por que una maquina de estados por sondeo es en sí una maquina de estados, sin embargo, la maquina de estados, tiene más configuraciones y no depende específicamente de ciertos arreglos o condiciones para hacer viable y funcional.

En caso contrario, la máquina de estados por sondeo consta principalmente de la ocupación de un ciclo While y un ciclo Case, en conjunto con un Enum para poder realizar de forma completa sus estados, así como, de la función Shift Register, para realizar su operación. Esto debido a que la maquina de estados por sondeo, recorre un dato que es ingresado por el

usuario, el cual puede ser booleano, esto con el objetivo de generar información nueva por cada ciclo con el que se esté ejecutando. Dado que hablamos de un ciclo Case, solo cuando



se tome como valor de entrada True en su implementación Stop produce que el programa y maquina de estado en cuestión termine su operación.

Imagen 6: Comportamiento de una máquina de estados por sondeo

Como se puede observar en la imagen, las características antes mencionadas que definen a este tipo de máquina producen un uso dirigido a la implementación de sistemas complejos como lo es el robot seguidor. Requerimos una secuencia que necesita constantemente de revisar el estado en el que se encuentra, una vez el estado detecta algo nuevo, algún cambio, (en este caso la detección de un obstáculo en frente del robot) este sistema toma ese valor y produce una nueva instrucción, la cual solo es conseguida una vez se determina el caso siguiente, para producir el nuevo estado, el cual se mantendrá, como se mencionó, en el mismo estado hasta un nuevo caso.

Para mostrar más el contraste analizaremos un caso con ejemplo de una máquina de estados:

Uno a varios usando arreglos: Si tiene varios estados a los que podría hacer la transición, podría usar un área booleana asociada con una constante enum para programar la transición. Se lleva a cabo algún código donde el resultado del código determina la transición, cuya salida es un arreglo de valores booleanos. El arreglo

booleano se correlaciona con una constante Enum que contiene una lista de posibles estados a los que se podría hacer la transición. Usando la función Index Array, se genera el índice del primer booleano “Verdadero” en el arreglo booleano. Luego, usando la función Array Subset, puede extraer lo apropiado de la constante Enum con la que se correlaciona.

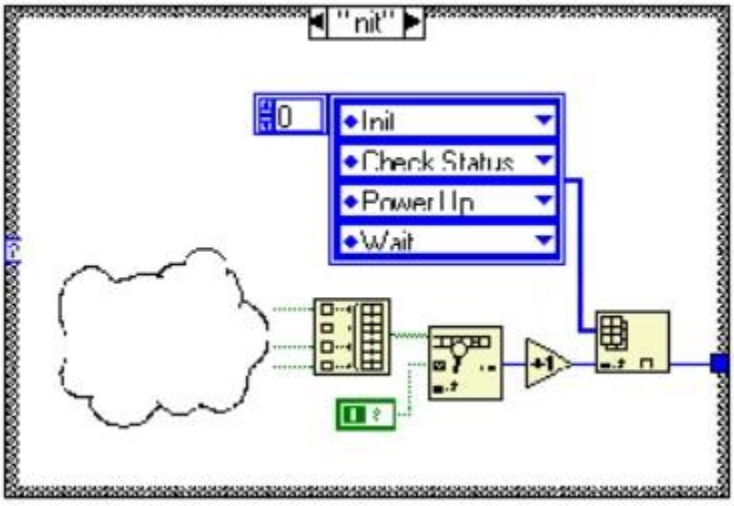


Imagen 7: Diagrama de bloques para representar el uso de la máquina de estado

En este caso, la maquina de estados no depende especialmente de un estado, o el sistema requiere que el estado en el que se encuentre y con el mantenerse en el mismo comportamiento, generando alguna repetición con su valor determinado, para este caso la máquina de estado solo muestra un valor de salida para el ciclo While, es más ni siquiera muestra algún tipo de paro programado.

En todo caso, el comportamiento es distinto, buscando que en este caso solo muestre valores en un Array determinados por las entradas que están dentro de la “nube” que se observa. En el caso de la función num su comportamiento es el normal, el cual determina que variable es la que se muestra en el mismo Array.

Ahora, como menciona el título del ejemplo, pueden hacerse más de un ciclo While, sin embargo, el comportamiento que diferencia principalmente entre la maquina de estados y la maquina de estados por sondeo es la utilización de la secuencia del ciclo While para mantener el propio estado dado el calor ingresado.