INFORME DEL PROYECTO INTEGRADO DE SABERES

Luis Blacio – Steven Arévalo - Ethan Soto - Johan Chumbi

Facultad de la Energía, las Industrias y los Recursos Naturales no Renovables, Universidad Nacional de Loja, Ecuador

luis.blacio@unl.edu.ec

frank.arevalo@unl.edu.ec

johan.chumbi@unl.edu.ec

ethan.soto@unl.edu.ec

I. INTRODUCCIÓN

Según un informe realizado por la Agencia Internacional de Energía (IEA) nos menciona que las energías renovables están creciendo rápidamente alrededor del mundo e incluso nos menciona que habar un aumentaran significativamente en el uso de energías limpias para el 2030. Además, la influencia de energías limpias esta empezando a cambiar la perspectiva de estos. Uno de ellos es China una de las grandes potencias que era responsable de casi dos tercios del aumento del uso mundial del petróleo empezó a disminuir su constante demanda. [1]

Por lo tanto, el satisfacer las necesidades de desarrollo de manera sostenible es esencial para poder avanzar. Este sería el fin que justifica nuestro proyecto el mejor y cambiar la mentalidad que tenemos para así evitar que ocurra más situación que ya antes se han vivió dentro del país como por ejemplo en la ciudad de Loja con los frecuentes apagones de la luz que afectan a las familias, los trabajadores y empresas. Incluso se podría decir que la finalización de nuestro proyecto sería el primer paso para tomar con mayor seriedad el uso de energía limpia y no conformarnos con lo que ya poseemos. [3]

Nuestro objetivo es estudiar, diseñar y probar un cargador solar completamente automatizado en el que pueda girar 180 grados y pueda seguir la luz solar, más adelante se les procederá a explicar cómo funciona y los componentes que posee. Se desarrollo un algoritmo en el programa de Visual Studio Code en el lenguaje de C que calcule y determine la ubicación del sol permitiendo que paneles solares puedan moverse y seguir al sol como si fuera un girasol. De esta manera podremos comprobar la efectividad del algoritmo mediante simulaciones que se realizaran dentro de la Universidad Nacional de Loja. Algunas variables que nos piden son la hora del día y datos meteorológicos básicos entre otros.

II. PROBLEMÁTICA

Hay algunos tipos de objetivos para el reconocimiento de la programación legítima para recoger, almacenar y tratar los datos recogidos a través de fuentes de alimentación verde de una placa Arduino. El objetivo principal de este trabajo es desarrollar un algoritmo para ampliar la eficiencia energética solo con la energía procedente del sol, ya que este problema surge en la mayor parte específica de los establecimientos de cargadores basados en la luz solar, que no aprovechan al máximo la energía del sol, ya que los rayos solares estarán en constante cambio por el hecho de que la tierra sigue rotando durante el día, causando un cierto nivel de inclinación a el sol y por lo tanto un cambio en el módulo y dirección de los rayos solares, aparte los rayos solares ascenderán desde el este y se podrá hacia el oeste. Por lo tanto, los paneles solares fijos no tendrán la misma intensidad de radiación solar durante el día. [1]

III. PROPUESTA

Nuestra solución conlleva en desarrollo de un prototipo que integrará paneles solares junto con un sistema de posicionamiento y control con la función algorítmica del Arduino. Se realizarán los cálculos para determinar la posición óptima de los paneles solares según la ubicación geográfica, aparte teniendo en cuenta la hora del día y la estación del año.

Son los cálculos matemáticos que utilizaron en el código fue la geometría esférica, esta se enfoca en las curvas, líneas y profundidad en una esfera o en nuestro caso sería el exterior debido a que la ubicación por donde sale el sol depende mucho de la ubicación, por lo que serán la base para el desarrollo de un programa que procesará los datos de la posición del sol y enviarán las instrucciones necesarias para establecer la orientación y ángulo de inclinación con respecto al norte de los paneles. [2]

De acuerdo a ya lo antes mencionado, los servomotores podrán seguir el movimiento del sol durante las horas convenientes, permitiendo que pueda maximizar la captación de energía solar y superar las limitaciones que comúnmente se ve en sistemas fijos convencionales, lo que será una nueva manera de aprovechar la energía solar, en paneles, aumentando así la eficiencia en la captura de energía renovable.

IV. EVIDENCIA DEL ALGORTIMO

El código que se realizo fue subido al sitio web de Github y ejecutado en Visual Studios Code.

- Link sobre el código:

https://github.com/159785/Desarrollo-del-Algoritmo-sobre-el-PROYECTO-INTEGRADO-DE-SABERES-PIS-/blob/b7dcac03b4f65a5a04141a4bf2ce01f201d22559/%E2%80%A2%09Algoritmo%20para%20el%20Panel%20Solar%20Automatizado

V. EXPLICACIÓN DE LAS VARIABLES DEL CODIGO

- **PI:** Estable dirigiéndose al valor de π (pi), en torno a 3,14159265359.
- **año:** Número que indica el año.
- **mes:** Número que indica el tramo largo del año (1-12).
- **día**: Número que indica el día del mes (1-31).
- **días_por_mes**: Muestra que contiene la cantidad de días en cada tramo largo del año (para un año sin saltos).
- **N**: Número que indica la cantidad de días transcurridos desde el 1 de enero hasta la fecha actual.
- **Term_1:** Valor fijo - 23,44, utilizado en la estimación del punto de declinación solar.
- **Term_2:** Estima determinada que es el resultado de una actividad con la cantidad de días transcurridos, utilizada en la estimación del punto de declinación basado en la luz solar.
- **DS_rad**: Valor en radianes obtenido de Term_2, utilizado en el cálculo del punto de declinación basado en la luz solar.
- **COSX:** Valor en coseno de DS_rad, utilizado en la estimación del punto de declinación orientado al sol.
- **Decl:** Estima del punto de declinación orientado al sol en radianes.
- **B:** Estima determinada utilizada en la situación de tiempo, adquirida a partir de la cantidad de días transcurridos.
- **EoT**: Valor de la situación de tiempo.
- hr: Número que indica la hora en curso (0-23).
- min: Número entero que indica los minutos en curso (0-59).
- **hr_local**: Estimación de la hora cercana en organización decimal.
- **zona_hor**: Número que indica la zona horaria cercana (establecida físicamente, para esta situación - 5).
- **long_stand**: Valor de la longitud estándar para la región horaria cercana.
- **TSV**: Estimación horaria genuina basada en la luz solar en configuración decimal.
- **H:** Estimación de la hora GMT en radianes.
- **Lat:** Estima de alcance en radianes (en lugar de grados).
- **Long:** Estima de la longitud en grados.
- **Elev**: estima del punto de salida orientado al sol en radianes.
- **Azim**: Estima del azimut orientado al sol en radianes.
- **now**: Valor de tipo time_t que indica la hora actual.
- **tm**: Puntero a una estructura tm que contiene la fecha y la hora separadas.

VI. EXPLICACIÓN DE LAS FUNCIONES DEL CODIGO

1. **es_bisiesto (int ano):**

Razón: Decide si un año es un año de salto.
Límite: "ano" (año).
Devuelve: 1 en caso de que el año sea de salto, 0 en caso contrario.
Justificación:

- Un año es de salto si es separable por 4.
- No obstante, si es distinto de 100, no es un año de salto, excepto si también es separable de 400.

2. **calcular_número_días(int mes, int día, int año):**

Razón: Calcula la cantidad de días transcurridos desde el 1 de enero hasta una fecha predeterminada.
Límites: mes (mes), día (día), año (año).
Retorno: número de días transcurridos.
Justificación:

- Suma los tiempos del mes en curso.
- Suma los tiempos de los meses anteriores.
- Añade un día si se trata de un año bisiesto y la fecha es posterior a febrero.

3. **calcular_ángulo_declinación(int N):**

Razón: Calcula la declinación del sol para un día del año.
Límite: N (número de días transcurridos desde el 1 de enero).
Devuelve: punto de declinación orientado al sol en radianes.
Justificación:

- Utiliza una receta en vista de la declinación por días y convierte el resultado a radianes.

4. **calcular_tiempo_ecuación(int N):**

Razón: Calcula la condición del tiempo para un día concreto del año.
Límite: N (número de días transcurridos desde el 1 de enero).
Devolución: Valor de la situación del tiempo en minutos.
Justificación:

- Utiliza una receta con términos sinusoidales en función del deslizamiento por días.

5. **calculate_TSV(double hr, double min, double Lengthy, double EoT):**

Razón: Calcula la auténtica hora orientada al sol (TSV).
Límites: hr (hora), min (minuto), Long (longitud), EoT (condición del tiempo).
Retorno: Hora auténtica orientada al sol en horas decimales.
Justificación:

- Establece la hora cercana, la longitud y la condición de tiempo para obtener el TSV.

6. **calcular_H(doble TSV):**

Razón: Calcula la hora GMT en radianes.
Límite: TSV (auténtica hora solar).
Devuelve: Hora GMT en radianes.
Justificación:

- Determina la diferencia en grados de la hora solar de la tarde y la convierte en radianes.

7. **calculate_elevation(double Decl, double Lat, double H):**

Razón: Determina el punto de altura en función del sol.
Límites: Decl (punto de declinación basado en el sol), Lat (alcance), H (hora GMT en radianes).
Retorno: punto de altura basado en el sol en radianes.
Justificación:

- Utiliza la receta de subida basada en el sol en función de la declinación basada en el sol, el alcance y la hora GMT.

8. **calculate_Azimuth(double Decl, twofold Elev, twofold Lat, twofold H):**

Razón: Calcula el azimut orientado al sol.
Límites: Decl (punto de declinación orientado por el sol), Elev (punto de elevación orientado por el sol), Lat (alcance), H (hora GMT en radianes).
Devuelve: azimut orientado al sol en radianes.
Justificación:

- Utiliza la receta del azimut orientado al sol y cambia el valor en caso de que la hora GMT sea positiva (tardía).

9. **main():**

Razón: Capacidad fundamental que ejecuta el programa.
Lógica:

- Establece el altitud y la longitud.
- Obtiene la fecha y hora en curso.
- Ajusta la oportunidad al tramo de 5 minutos más cercano.
- Calcula la cantidad de deslizamiento por días.
- Calcula el punto de declinación orientado hacia el sol, condición del tiempo, el tiempo genuino impulsado por el sol, el tiempo GMT, la salida basada en el sol y el azimut basado en el sol.
- Convierte los resultados a grados y los imprime.
- Espera 5 minutos antes de repetir el ciclo.

VII. PSEUDOCÓDIGO Y DIAGRAMA DE FLUJO

En cuanto a lo que representa el pseudocódigo se encuentra dentro del repositorio de Github y se puede acceder mediante el siguiente link

https://github.com/159785/Desarrollo-del-Algoritmo-sobre-el-PROYECTO-INTEGRADO-DE-SABERES-PIS-/blob/256d16711de5f7dcae5c2f9efaeca6f07535f829/PSEUDOCO%CC%81DIGO%20PIS

Ahora por otra parte el diagrama de flujo se lo puede encontrar, visualizando las figuras 1, 2, 3 y 4 referenciadas en la parte de anexos

VIII. LA APLICACIÓN DEL ALGORITMO EN LA VIDA REAL

Particularmente es importante recalcar, que este tipo de algoritmo si puede funcionar en la vida real, en tal caso ayudaría en la instalación de paneles solares, sobre todo, como una nueva visión dentro del campo laboral y una nueva forma de utilizar las energías renovables y llevarlas a su máxima optimización, ya que con este algoritmo solo se necesita insertar las coordenadas de la zona geográfica y configurar la hora en cuanto al meridiano de Greenwich y el algoritmo resolverá y devolverá un resultado preciso, que mediante ayuda de la parte Hardware y una correcta calibración se pueden obtener resultados favorables.

IX. REREFERENCIAS

I. REFERENCIAS

[1] IEA, «World Energy Outlook 2023,» IEA, Octubre 2023. [En línea]. Available: https://www.iea.org/reports/world-energy-outlook-2023/executive-summary?language=es. [Último acceso: 15 6 2024].

[2] H. Maltby, . J. Thomas y S. Dash, «Brilliant | Learn interactively,» [En línea]. Available: https://brilliant.org/wiki/spherical-geometry/. [Último acceso: 16 6 2024].

[3] J. Z. A. y. M. A. Ramirez, «Análisis de rendimiento y estrategias de mejora para una planta fotovoltaica de 90 KW,» Guayaquil, 2024.

X. ANEXOS

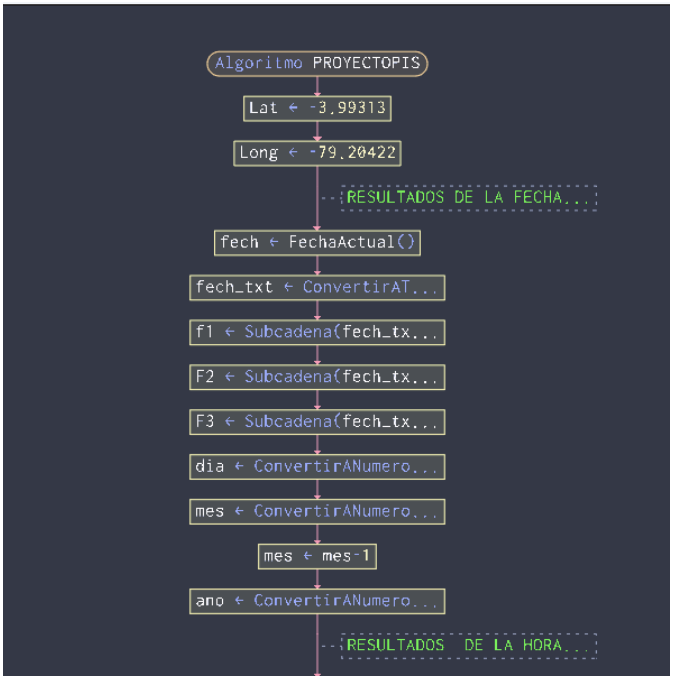

FIGURA No.1 Inicio del algoritmo en el diagrama de flujo

FIGURA No.2 Desarrollo del algoritmo en el diagrama de flujo

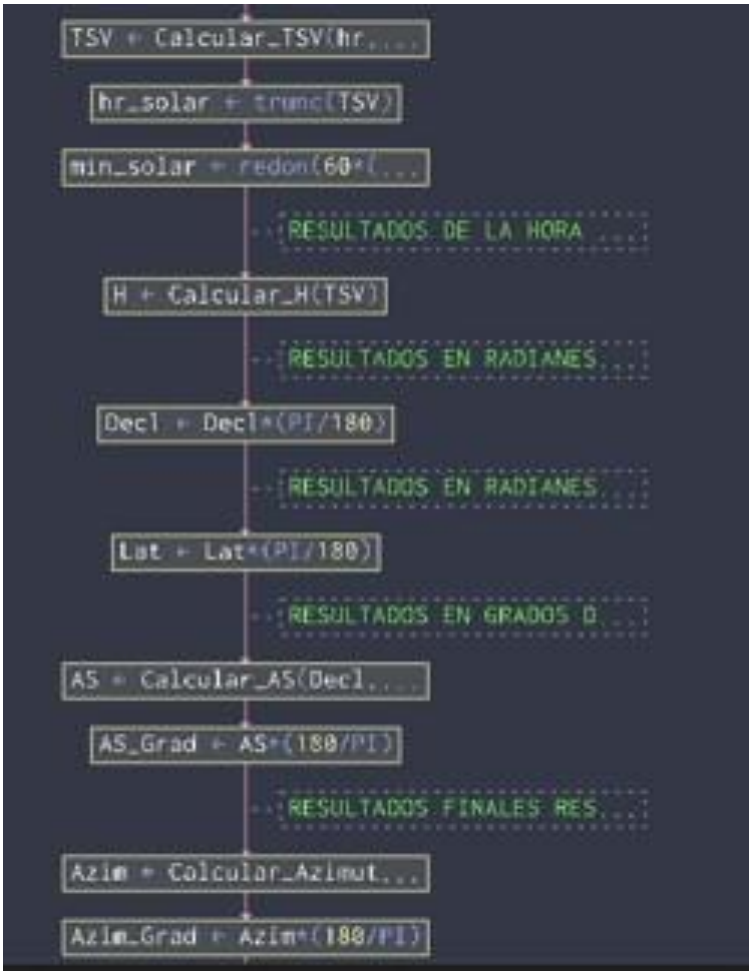

FIGURA No.3 Continuación del desarrollo del algoritmo en el diagrama de flujo

FIGURA No.4 Finalización del algoritmo en el diagrama de flujo