Proyecto PIS Unidad 2.

Rodríguez Israel, Jima Royel, Pardo Steeven, Calopino Juan, Granda Carlos.

Computación, Facultad de la Energía, las Industrias y los Recursos no Renovables, UNL.

I. Definición del problema.

Como se sabe la creciente problemática que se ha generado a lo largo de los años en torno al cambio climático ha incentivado al desarrollo de tecnologías entorno a la energía renovable, con el fin de concientizar y mejorar el entorno en el que se vive. Por lo cual se ha buscado cómo aprovechar un recurso renovable que se tiene al alcance todos los días, en este caso la energía solar, la cual es generada por el sol viajando a través de radiaciones hacia el planeta Tierra, siendo inagotable y muy abundante por lo que es sumamente aprovechada por diferentes tecnologías actuales, debido a que esta puede ser transformada en electricidad.

Para entender cómo se realiza la transformación de la energía solar a electricidad, es necesario saber que esta se genera mediante reacciones de fusión las cuales son producidas por el sol, llegando a viajar hacia la tierra por medio de ondas electromagnéticas y posteriormente aprovechadas para su uso e incluso almacenamiento, principalmente por medio de los paneles solares.

El problema que aborda el presente proyecto se centra en cómo aprovechar esta energía renovable a través de algoritmos y componentes eléctricos que permitan aprovechar la energía solar de la manera más eficiente a lo largo del día, con el fin de poder presentar ideas de cómo implementarlo en la vida cotidiana.

En el caso del algoritmo, el problema que surge entorno a este, se centra en la necesidad de calcular la posición del sol en el cielo en cualquier momento dado, teniendo en cuenta que este cálculo es dinámico y debe actualizarse continuamente debido al movimiento aparente del sol a lo largo del día y del año, permitiendo así al cargador solar tener una orientación óptima del panel, para maximizar la exposición a la energía solar y, por consiguiente, tener una mayor eficiencia del mismo.

II. Objetivos del presente proyecto.

El objetivo del presente proyecto trata de proporcionar una secuencia lógica y detallada de pasos o instrucciones que permitan calcular la posición del sol en una hora determinada y así realizar un sistema de seguimiento del sol a base de componentes o dispositivos que permitan implementarlo, en menor escala, para su uso en el día a día.

En el caso específico del algoritmo, como objetivo de optimización se busca tomar en consideración cómo las coordenadas influyen en el tiempo, e incluso en la fecha, a utilizar dentro de los cálculos a ejecutar. Entonces se debe considerar cómo la zona horaria es distinta de acuerdo a las coordenadas presentadas. Asimismo, como otro objetivo de optimización, se busca lograr compatibilizar los resultados que brinda el algoritmo con los resultados presentados por sensores de luz en el panel solar.

III. Método SPA (Solar Position Algorithm).

El método SPA es un método ampliamente utilizado para determinar la posición del sol en el cielo en función de la fecha, la hora y la ubicación geográfica específicas. La cuál se basa en un conjunto de ecuaciones trigonométricas y astronómicas que modelan la posición aparente del sol en el cielo en función de varios parámetros.

Las fórmulas utilizadas en el método SPA se derivan de modelos astronómicos complejos que tienen en cuenta la órbita de la Tierra alrededor del sol, la inclinación del eje de la Tierra, la rotación de la Tierra sobre su eje, y otros factores astronómicos. Algunas de las fórmulas comúnmente utilizadas en el método SPA incluyen:

Declinación Solar: Se trata del ángulo entre el plano del ecuador terrestre y la línea que conecta el centro de la Tierra con el Sol en un momento dado. El cual se calcula utilizando ecuaciones que tienen en cuenta la posición orbital de la Tierra y la posición del sol en la eclíptica.

$$\boldsymbol{\delta = -23.44^{\circ} \times \cos\left(\frac{360}{365} \times (n + 10)\right)}$$

$\boldsymbol{\delta}$ = Declinación solar

$\mathbf{n}$ = Días transcurridos desde el inicio del año.

Ángulo de Elevación Solar: Se trata del ángulo vertical entre la dirección del sol y el plano del horizonte en un lugar específico y en un momento dado, siendo calculado por medio de ecuaciones que utilizan la declinación solar, la ecuación del tiempo, la longitud y latitud del lugar, y el tiempo solar verdadero.

$$\mathbf{EoT = 9.87 \times \sin(2B) - 7.53 \times \cos(B) - 1.5 \times \sin(B)}$$

$\mathbf{EoT}$ = Ecuación del tiempo

$\mathbf{B} = \frac{360}{365} \times (n - 81)$

$\mathbf{B}$ = Ángulo horario en radianes

$\mathbf{n}$ = Días transcurridos desde que inicio el año

$$\mathbf{TSV = Hora\ local + \frac{4 \times (longitud\ local - longitud\ estandar) + EoT}{60}}$$

$\mathbf{TSV}$ = Tiempo Solar Verdadero

$\mathbf{EoT}$ = Ecuación del tiempo

$$\boldsymbol{\alpha = \sin^{-1}(\sin(\delta) \times \sin(\phi) + \cos(\delta) \times \cos(\phi) \times \cos(H))}$$

$\boldsymbol{\alpha}$ = Altura del sol
$\boldsymbol{\delta}$ = Declinación solar
$\boldsymbol{\phi}$ = Latitud en radianes
$\mathbf{H} = \mathbf{15}° \times (\mathbf{TSV} - \mathbf{12})$
$\mathbf{H}$ = Altura del sol sobre el horizonte
$\mathbf{TSV}$ = Tiempo Solar Verdadero

Ángulo Azimutal Solar: Se trata del ángulo horizontal entre la dirección norte y la proyección del sol en el plano horizontal. Siendo calculado por medio de ecuaciones trigonométricas que tienen en cuenta la declinación solar, la latitud del lugar y el ángulo de elevación solar.

$$\mathbf{Azimut} = \mathbf{cos}^{-1}\left(\frac{\mathbf{sin}(\boldsymbol{\delta}) - \mathbf{sin}(\boldsymbol{\alpha}) \times \mathbf{sin}(\boldsymbol{\phi})}{\mathbf{cos}(\boldsymbol{\alpha}) \times \mathbf{cos}(\boldsymbol{\phi})}\right)$$

$\boldsymbol{\delta}$ = Declinación solar
$\boldsymbol{\alpha}$ = Altura del sol
$\boldsymbol{\phi}$ = Latitud en radianes

IV. Variables a utilizar en la creación del algoritmo y como influyen en los cálculos.

Las variables que se van a utilizar para hacer que el proyectó, en este caso la parte del algoritmo, permita que los componentes a utilizar se muevan en torno a donde se encuentre el sol, dependen de diversas variables las cuales son:

Hora Actual: Es una variable muy indispensable para el proyecto ya que se necesitarán las horas y minutos, que son dos variables que influyen en casi todos los cálculos.

Fecha Actual: Esta variable siempre va a variar a medida que pasan los días, donde los valores como la declinación solar o el TSV o incluso la altura del sol no son constantes ya que estas fórmulas dependen del tiempo, es decir, su valor cambia a medida que pasan los días.

Días transcurridos desde que inició el año: Variable que sirve para poder realizar las ecuaciones principales del método SPA.

Días totales del año: Esta variable puede variar dependiendo si es año bisiesto; en este caso como es año bisiesto, la fórmula de declinación solar cambia el valor de 365 a 366, así mismo en la ecuación del tiempo.

Declinación Solar: Variable que tiene muchas constantes, sin embargo, para calcularla existe una variable a tomar en cuenta la cual es el número de días desde el inicio del año hasta el día actual.

Latitud y longitud: Variables que pueden variar dependiendo del lugar en el que se encuentren. En este caso, la latitud promedio en la ciudad de Loja esta entre un valor desde -4.03009 como mínimo y -3.9748 como máximo, en cambio la longitud de Loja está en promedio desde los -79.2184 hasta los -79.1867. Y es necesario conocer estas variables ya que mediante estas se pueden sacar los datos que permiten realizar las fórmulas correctamente de acuerdo al lugar.

Ecuación del Tiempo: Esta variable es importante ya que nos permite conocer el ángulo solar de acuerdo a la hora o día en el que se esté.

Zona Horaria: Variable relevante ya que depende mucho sobre la ubicación geográfica de un lugar dependiendo de la rotación de la Tierra y división de uso terrestre dependiendo de los horarios.

Longitud estándar: Variable que está relacionada con el meridiano de Greenwich como de la zona horaria, y es necesaria para algunos cálculos en el método SPA.

Altura del sol sobre el horizonte (H): Variable que permite entender en donde se encuentra el sol en cierta hora del día, y que influye dentro del cálculo para obtener el azimut.

Tiempo Solar Verdadero (TSV): Variable que varía a lo largo del día, ya que el valor de una de las variables no permanece constante debido a que se trata del tiempo actual.

Azimut: Variable que juega un papel muy importante porque indica la dirección horizontal de un punto en relación con el norte.

V. Estructuras de datos y funciones utilizadas en el código.

En el presente código se ha implementado distintas estructuras que guardan datos similares, pero con diferentes unidades. Entre las estructuras creadas se podrá observar que "Ubicacion" guarda los valores de las coordenadas, es decir, latitud y longitud; "TiempoActual" y "Tiempo" guardan los valores de fechas, horas y minutos; "ZonaHoraria" guarda el valor de la longitud estándar y de la zona horaria en la que se encuentra. Las demás estructuras guardan el valor obtenido de cada fórmula del SPA en grados y en radianes. Por último, está la estructura "PanelSolarSpa", la cual guarda los valores necesarios para hacer girar el panel solar; en este caso, guarda el azimut y la altura solar en grados, individualmente.

En cuanto a las funciones, se ha procedido a utilizar procedimientos y valores por referencia, ya que se ha observado una mayor eficiencia en la actualización de las variables después de cada cálculo. Para comenzar, se han creado procedimientos que permitan convertir los datos ingresados de una unidad a otra, es decir, de grados a radianes y viceversa. Luego, se ha continuado desarrollando procedimientos para tomar los datos del tiempo desde el sistema, determinar la ubicación respecto al plano geográfico, calcular los días transcurridos desde que inició el año y realizar todas las fórmulas correspondientes dentro del método SPA.

VI. Pseudocódigo y diagrama de flujo.

```
#include <stdio.h>
#include <time.h>
#include <math.h>
//ESTRUCTURAS
struct Ubicacion{
    float lat, lon;
};
struct TiempoActual{
    int aC,mC,dC,hC,minC;
};
struct Tiempo{
```

```
        int a,m,d,a1,m1,d1;
    };
    struct DeclinacionSolar{
        double dsG, dsR;
    };
    struct EcuacionTiempo{
        double etG, etR;
    };
    struct ZonaHoraria{
        int lonEstand, zH;
    };
    struct HoraSolarVerdadera{
        double TsD;
        int TsH, TsM;
    };
    struct AlturaSol{
        double HG, HR, asG, asR;
    };
    struct Azimut{
        double aziG, aziR;
    };
    struct PanelSolarSpa{
        double aziG, asG;
    };
    //FUNCIONES Y PROCEDIMIENTOS
    //CONVERSIONES
    void Pgar(double *variable){
        double Pi=3.1415926535;

*variable=((*variable*Pi)/180.0);
//Conversion de Grados a Radianes
    }
    void Prag(double *variable){
        double Pi=3.1415926535;

*variable=((*variable*180)/Pi);
//Conversion de Radianes a Grados
    }
    //TIEMPO DEL SISTEMA
    void PtiempoActual(int
*aC,int *mC, int *dC, int *hC, int
*minC){
        time_t t;
        time(&t);
        struct tm *tm_info =
localtime(&t);
        *aC = tm_info->tm_year +
1900; //Numero de aÑos desde 1900
        *mC = tm_info->tm_mon +
1;     //Numero de meses desde (0-
11)
        *dC = tm_info->tm_mday;
        *hC = tm_info->tm_hour;
        *minC = tm_info->tm_min;
    }
    //CALCULOS Y DATOS
NECESARIOS PARA LAS ECUACIONES
    void
PubicacionDireccion(float lat,
float lon){
        //Procedimiento para
determinar direcciones: N,S,E,O
        if (lat>0){
            printf("Valor de
latitud de su zona geografica\n%f
N\n", lat);
        } else{
            printf("Valor de
latitud de su zona geografica\n%f
S\n", (-1*lat));
        }
        if (lon>0){
            printf("Valor de
longitud de su zona geografica\n%f
E\n", lon);
        } else{
            printf("Valor de
longitud de su zona geografica\n%f
O\n", (-1*lon));
        }
    }
    void Ptiempo(int *a, int aC,
int *m, int mC, int *d, int dC,
int *a1, int *m1, int *d1){
        //Procedimiento para
determinar el tiempo transcurrido
durante este aÑo.
        *d=1; //Dia en el que se
empieza
        *m=1; //Mes en el que se
empieza
        *a1=aC-*a; //Para
contabilizar que el año siempre
sea 0
        *m1=mC-*m; //Para
contabilizar los meses completos
que han habido
        for(*m=1; *m<mC;
*m=*m+1){ //Bucle para que sume
los dias correspondientes de
acuerdo a cada mes del año
            if
(*m==1||*m==3||*m==5||*m==7||*m==8
```

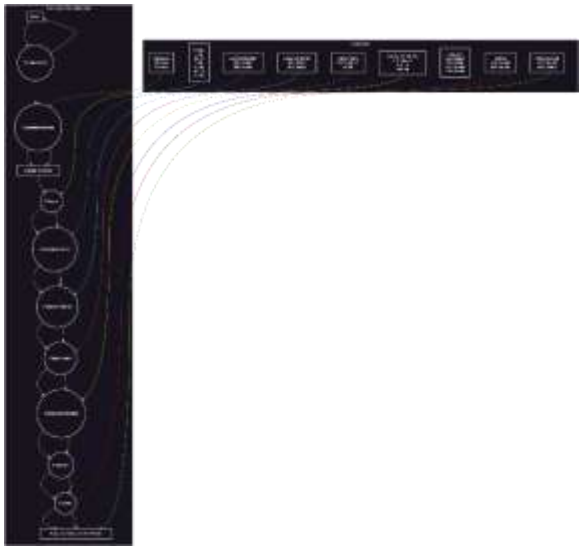
```
||*m==10||*m==12){ //Meses de 31 dias
                    dC+=31;
                } else if(*m==2){
//Mes de 28 o 29 dias de acuerdo a si el año es bisiesto
                    if(aC%4==0){
                        dC+=29; //El año si es bisiesto
                    }else{
                        dC+=28; //El año no es bisiesto
                    }
                } else if(*m==4||*m==6||*m==9||*m==11){
//Meses de 30 dias
                    dC+=30;
                }
            }
            *d1=dC; //Para guardar en una variable el total de dias desde que empezo el año
        }
        //1era FORMULA: DECLINACION SOLAR
        void PdeclinancionSolar(int a, int d1, double *decSolar, double *decSolarR){
            double ec; //Inicializo una variable para realizar calculos
            if(a%4==0){

ec=360.0/366.0*(d1+10); //Para el caso de que fuera año bisiesto
            } else{

ec=360.0/365.0*(d1+10); //Para el caso de que no fuera aÑo bisiesto
            }
            Pgar(&ec); //Convierto el resultado a Radianes
            *decSolar=(-23.44)*cos(ec); //Resultado en grados
            *decSolarR=*decSolar; //Asigno valor para ser convertido
            Pgar(&*decSolarR); //Convierto el resultado a Radianes
        }
        //2nda FORMULA: ECUACION DEL TIEMPO
        void PecuacionTiempo(int a, int d1, double *ecTiempo, double *ecTiempoG){
            double B; //Inicializo una variable para realizar calculos
            if(a%4==0){
                B=360.0/366.0*(d1-81); //Para el caso de que fuera año bisiesto
            } else{
                B=360.0/365.0*(d1-81); //Para el caso de que no fuera aÑo bisiesto
            }
            Pgar(&B); //Convierto el resultado a Radianes

*ecTiempo=9.87*(sin(2*B))-(7.53*cos(B))-(1.5*sin(B)); //Resultado en radianes
            *ecTiempoG=*ecTiempo; //Asigno valor para ser convertido
            Prag(&*ecTiempoG); //Convierto el resultado a Grados
        }
        //3era FORMULA: TIEMPO SOLAR VERDADERO
        void PzonaHoraria(int lonEntero, int *lonEs, int *zH){
            //Procedimiento para obtener la longitud Estandar y la Zona Horaria
            if (lonEntero%15==0){ //En el caso de que la parte entera de la coordenada si fuera divisible para 15
                *lonEs=lonEntero;
                *zH=lonEntero/15;
            }else{
                if (lonEntero>0){ //En el caso de que la parte entera de la coordenada no fuera divisible para 15 y sea un valor positivo

while(lonEntero%15!=0){ //Bucle para al valor restarle hasta que si sea divisible
                        lonEntero--;
```

```
                    }

*lonEs=lonEntero;

*zH=lonEntero/15;
                } else{  //En el
caso de que la parte entera de la
coordenada no fuera divisible para
15 y sea un valor negativo

while(lonEntero%15!=0){ //Bucle
para al valor sumarle hasta que si
sea divisible
                        lonEntero++;
                    }

*lonEs=lonEntero;

*zH=lonEntero/15;
                }
            }
        }
        void PhoraSolarVerdadera(int
hC, int minC, float lon, int
lonEs, double EcTiempo, double
*TsD, int *TsH, int *TsM){
            double
hdec=hC+(minC/60.0); //Inicializo
una variable para calcular la hora
en decimales
            *TsD=hdec+((4*(lon-
lonEs)+EcTiempo)/60.0);
//Resultado del Tiempo Solar en
decimales
            *TsH=*TsD; //Asigno el
valor y obtengo el Tiempo Solar en
horas
            *TsM= round((*TsD-
*TsH)*60); //Calculo y redondeo el
valor, obteniendo asi el valor del
Tiempo Solar en minutos
        }
        //4rta Formula: ALTURA DEL
SOL
        void PalturaSol(double TsD,
double *HG, double *HR, double lat
, double decSolar, double *asR,
double *asG){
            *HG=15*(TsD-12);
//Calculo la altura del sol sobre
el horizonte en Grados
            *HR=*HG; //Asigno valor
para ser convertido
            Pgar(&*HR); //Convierto
el resultado a Radianes
            Pgar(&lat); //Convierto
la latitud a Radianes

*asR=asin((sin(decSolar)*sin(lat))
+(cos(decSolar)*cos(lat)*cos(*HR))
); //Resultado en radianes
            *asG=*asR; //Asigno
valor para ser convertido
            Prag(&*asG); //Convierto
el resultado a Grados
        }
        //5nta FORMULA: AZIMUT
        void Pazimut(double
decSolar, double asR, double lat,
double HR ,double *aziR, double
*aziG){
            double Pi=3.1415926535;
//Inicializo el valor de Pi
            Pgar(&lat); //Convierto
la latitud a Radianes

*aziR=acos((sin(decSolar)-
sin(asR)*sin(lat))/(cos(asR)*cos(l
at))); //Resultado en grados
            if (HR>0){ //Condicion
para cuando la altura del sol
sobre el horizonte sea mayor a 0
                *aziR=(2*Pi)-*aziR;
                printf("El valor del
azimut cambio\n"); //Indicamos en
el caso de que se cumpla la
condicion
            }
            *aziG=*aziR; //Asigno
valor para ser convertido
            Prag(&*aziG);
//Convierto el resultado a Grados
        }
        //FUNCION PRINCIPAL
        int main(){
            //FECHA Y HORA
            struct TiempoActual
tmpa; //  <-------- ESTRUCTURA

PtiempoActual((&tmpa.aC),(&tmpa.mC
),(&tmpa.dC),(&tmpa.hC),(&tmpa.min
C)); //Llamo a la funcion
```

```
            printf("Fecha: %d/%d/%d
, Hora: %d:%d\n",
(tmpa.dC),(tmpa.mC),(tmpa.aC),(tmp
a.hC),(tmpa.minC)); //Escribo los
valores que me devuelve la funcion
            //COORDENADAS
            struct Ubicacion
coordenada; //  <--------
ESTRUCTURA
            printf("Bienvenido al
programa para calcular la
direccion y altura del sol\nSe
requieren de tus coordenadas para
realizar el calculo\n");
            while(1){ //Bucle para
detectar errores
                int datoingresado;
                printf("Ingresa como
dato flotante el valor de la
latitud de tu zona. Teniendo en
cuenta el rango [-90,90]\n");

datoingresado=scanf("%f",
&coordenada.lat); //Para ingresar
el valor de latitud
                if
(datoingresado==1){ //Si el dato
ingreado es correcto, procede a
salir del bucle e hacer los
calculos
                    break; //Para
salir del bucle
                } else{
                    printf("Error:
Entrada invalida. El dato
ingresado no es un valor
numerico\n"); //Indico que existe
un error
                    while
(getchar()!='\n'); //Para limpiar
el buffer de entrada y descartar
los caracteres hasta encontrar un
salto de linea
                }
            }
            while(1){ //Bucle para
detectar errores
                int datoingresado;
                printf("Ingresa como
dato flotante el valor de la
longitud de tu zona. Teniendo en
cuenta el rango [-180,180]\n");

datoingresado=scanf("%f",
&coordenada.lon); //Para ingresar
el valor de longitud
                if
(datoingresado==1){ //Si el dato
ingreado es correcto, procede a
salir del bucle e hacer los
calculos
                    break; //Para
salir del bucle
                } else{
                    printf("Error:
Entrada invalida. El dato
ingresado no es un valor
numerico\n"); //Indico que existe
un error
                    while
(getchar()!='\n'); //Para limpiar
el buffer de entrada y descartar
los caracteres hasta encontrar un
salto de linea
                }
            }
            if (coordenada.lat==0){
//Para ingresar automaticamente el
valor de prueba
                coordenada.lat=-
3.99313;
            }
            if (coordenada.lon==0){
//Para ingresar automaticamente el
valor de prueba
                coordenada.lon=-
79.20422;
            }

PubicacionDireccion((coordenada.la
t),(coordenada.lon)); //Llamo a la
funcion
            //DIAS TRANSCURRIDOS
            struct Tiempo tmp; //
<-------- Estructura
            tmp.a=tmpa.aC; //Asigno
el mismo valor para el año de
referencia

Ptiempo((&tmp.a),(tmpa.aC),(&tmp.m
),(tmpa.mC),(&tmp.d),(tmpa.dC),(&t
mp.a1),(&tmp.m1),(&tmp.d1));
//Llamo a la funcion
```

```
          printf("Fechas hasta
ahora: %d anios, %d meses, %d
dias\n",
(tmp.a1),(tmp.m1),(tmp.d1));
//Escribo los valores que me
devuelve la funcion
          //1era FORMULA:
DECLINACION SOLAR
          //Variables utilizadas:
Año, Dias que han transcurrido
desde que inicio el año
          struct DeclinacionSolar
decSolar; //  <-----------
Estructura

PdeclinancionSolar((tmp.a),(tmp.d1
),(&decSolar.dsG),(&decSolar.dsR))
; //Llamo a la funcion
          printf("Valor de la
declinacion solar %f en grados\n",
decSolar.dsG); //Escribo los
valores que me devuelve la funcion
          //2nda FORMULA: ECUACION
DEL TIEMPO
        //Variables utilizadas:
Año, Dias que han transcurrido
desde que inicio el año
          struct EcuacionTiempo
ecTiempo; //  <-------------
Estructura

PecuacionTiempo((tmp.a),(tmp.d1),(
&ecTiempo.etR),(&ecTiempo.etG));
//Llamo a la funcion
          printf("Valor de la
ecuacion del tiempo %f en
radianes\n", ecTiempo.etR);
//Escribo los valores que me
devuelve la funcion
          //LONGITUD ESTANDAR Y
ZONA HORARIA
          struct ZonaHoraria
zhoraria; //  <----------------
Estructura

PzonaHoraria((coordenada.lon),(&zh
oraria.lonEstand),(&zhoraria.zH));
//Llamo a la funcion
          printf("Valor de la
longitud estandar: %d\nTu zona
horaria es: %d\n",
zhoraria.lonEstand, zhoraria.zH);
//Escribo los valores que me
devuelve la funcion
          //3era FORMULA: TIEMPO
SOLAR VERDADERO
          //Variables utilizadas:
Hora y Minuto actual, Longitud,
Longitud Estandar, Ecuacion del
Tiempo
          struct
HoraSolarVerdadera TiempoSolar; //
<------ Estructura

PhoraSolarVerdadera((tmpa.hC),(tmp
a.minC),(coordenada.lon),(zhoraria
.lonEstand),(ecTiempo.etR),(&Tiemp
oSolar.TsD),(&TiempoSolar.TsH),(&T
iempoSolar.TsM)); //Llamo a la
funcion
          printf("Valor de la Hora
solar:\nEn decimales: %f\nEn
horas: %d\nEn minutos: %d\n",
(TiempoSolar.TsD),(TiempoSolar.TsH
),(TiempoSolar.TsM)); //Escribo
los valores que me devuelve la
funcion
          //4rta FORMULA: ALTURA
DEL SOL
        //Variables utilizadas:
Tiempo Solar, ALtura del Sol sobre
el Horizonte (H), latitud,
Declinacion Solar
          struct AlturaSol altSol;
//  <--------------------
Estructura

PalturaSol((TiempoSolar.TsD),(&alt
Sol.HG),(&altSol.HR),(coordenada.l
at),(decSolar.dsR),(&altSol.asR),(
&altSol.asG)); //Llamo a la
funcion
          printf("Valor de H: %f
en radianes\nValor de la altura
solar:\nEn radianes: %f\nEn
grados: %f\n",
(altSol.HR),(altSol.asR),(altSol.a
sG)); //Escribo los valores que me
devuelve la funcion
          //5nta FORMULA: AZIMUT
        //Variables utilizadas:
Declinacion SoLar, Altura del Sol,
latitud, ALtura del Sol sobre el
Horizonte (H)
```

```
            struct Azimut azi; //
<--------------------------
Estructura

Pazimut((decSolar.dsR),(altSol.asR
),(coordenada.lat),(altSol.HR),(&a
zi.aziR),(&azi.aziG)); //Llamo a
la funcion
            printf("El valor del
Azimut: %f en radianes\nEl valor
del Azimut: %f en grados\n",
(azi.aziR),(azi.aziG)); //Escribo
los valores que me devuelve la
funcion
            struct PanelSolarSpa
psSpa; //  <-----------------
Estructura
            psSpa.asG=altSol.asG;
//Guardo los valores que son
necesarios en Y para el panel
solar
            psSpa.aziG=azi.aziG;
//Guardo los valores que son
necesarios en X para el panel
solar
            printf("Los valores que
van a ir hacia el panel son:\nPara
X: %f grados <-- Azimut\nPara Y:
%f grados <-- Altura Solar\n",
(psSpa.aziG),(psSpa.asG));
//Escribo los valores guardados en
la estructura
            return 0;
      }
```



VII. Discusión sobre como el diseño puede implementarse en un sistema real de paneles solares.

Para implementar un código en un sistema de paneles solares, es esencial integrar tanto el software como el hardware. El software se basa en generar órdenes que el hardware ejecuta para cumplir objetivos como el seguimiento solar. Esta integración debe ser optimizada para asegurar eficiencia y durabilidad del proyecto. La mejora continua del algoritmo también es fundamental, ya que influye directamente en la eficiencia y capacidad del sistema para aprovechar la energía solar de manera efectiva.

VIII. Referencias:

[1] “Posición del Sol | PVEducation”. PVEducation. Accedido el 16 de junio de 2024. [En línea].

[2] “Cálculo de la posición del sol en el cielo para cada lugar en cualquier momento”. Home ☼ SunEarthTools.com solar tools for consumers and designers. Accedido el 15 de junio de 2024. [En línea].

[3] “Cómo Calcular El Azimut De Una Placa Solar | Placas Solares”. Placas Solares. Accedido el 16 de junio de 2024. [En línea].

[4] “Calculadora en línea: Acimut y ángulo de elevación solar”. Calculadoras en línea. Accedido el 16 de junio de 2024. [En línea].