UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Polytechnical University of Catalonia

Aerospace Engineering

ANNEX: Design of an azimuth/elevation mount for Sun-tracking of a solar panel

Iker Blanco Bravo

Supervised by

Prof. Beatriz AMANTE
Projects and Construction Engineering Department

Prof. David DE LA TORRE
Physics Department

June 2020

Contents

A Radiation for different Sun-tracking PV

This appendix presents the code used to perform the radiation study, which is discussed in section 4.1.

```
% Radiation study script

% Control parameters
phi = 12.3; % Latitude [deg]
lambda = -1.53388 ; %Longitude [deg]
load('Database_irradiance\Horizontal_direct_radiation_2.mat')
load('Database_irradiance\Horizontal_difuse_radiation_2.mat')
load('Database_irradiance\k_factor_2.mat')

%Official time
UTC = 0:0.0625:23 ;

%Day of the year
day = 15:30:345 ;

% Declination of Sun [deg]
deltas = 23.45 * sind((360 * (day + 284)) / 365) ;

% Solar time parameters
M = (2*pi*day)/365.24 ;
EoT = 229.18 * (-0.0334*sin(M) + 0.04184*sin(2*M + 3.5884)) ;
GMT = 0 ;
AO = 0 ;
var_lambda = lambda - GMT*15 ;

% Preallocate omegas
omegas = NaN(length(day),length(UTC)) ;

%Solar time [deg]

```

```
for k = 1:length(day)
    for t = 1:length(UTC)
        omegas (k,t) = 15 * (UTC(t) - AO - 12) + var_lambda + (EoT(k)/4) ;
    end
end

% Preallocate vars
theta_zss = NaN(length(day),length(omegas));
psi_ss = NaN(length(day),length(omegas));
theta_s = NaN (4,length(day),length(omegas));

for t = 1:4

    for j = 1:length(day)

        for i = 1:length(omegas)
            %Current hour angle and declination
            omega = omegas(j,i) ;
            delta = deltas(j) ;

            % Pre-compute some parameters for performance
            cosd_phi = cosd(phi);
            sind_phi = sind(phi);
            cosd_delta = cosd(delta);
            sind_delta = sind(delta);
            cosd_omega = cosd(omega);
            sind_omega = sind(omega);

            % Compute Zenith angle
            cosd_theta_zs = (cosd_delta * cosd_omega * cosd_phi) + (sind_delta * sind_phi);
            sind_theta_zs = sind(acosd(cosd_theta_zs));
            theta_zs = acosd(cosd_theta_zs);

            % Compute azimuth angle
```

```
            sind_psi_s = (cosd_delta * sind_omega) / sind_theta_zs;
            cosd_psi_s = (cosd_delta * cosd_omega * sind_phi - cosd_phi * sind_delta) / sind_theta_zs;

            % Get correct value of angle psi_s
            psi_s = asind(sind_psi_s);

            if cosd_psi_s < 0
                if sind_psi_s > 0, psi_s = 180 - psi_s;
                elseif sind_psi_s < 0, psi_s = -180 - psi_s;
                else, psi_s = psi_s + 180;
                end
            end

        % Save angle values
        theta_zss(j,i) = theta_zs; % Zenith angle
        psi_ss(j,i) = psi_s; % Azimuth angle

        if t == 1 %CASE 1: static panel
            beta = 3.7 + 0.69 * phi ; %optimal tilt angle
            alfa = 0 ; %pointing to N-S axis

        elseif t == 2 %CASE 2: azimuth 1-axis panel

            beta = phi ; %optimal tilt angle
            alfa = psi_s ;

        elseif t == 3 %CASE 3: horizontal 1-axis panel
            beta = theta_zs ;

            if day(j) > 114 && day(j) < 232
                alfa = 180 ;
            else
                alfa = 0 ;
```

```
                end

            else  %CASE 4: 2-axis panel
                beta = theta_zs ;
                alfa = psi_s ;
            end

            %Angle between Sun and PV panel
            theta_s = ( sign(phi) * ( sind(beta)*cosd(alfa)*cosd(delta)*cosd(omega)*sind(phi) ...
                           - sind(beta)*cosd(alfa)*cosd(phi)*sind(delta)) ...
                          + (sind(beta)*sind(alfa)*cosd(delta)*sind(omega)) ...
                           + (cosd(beta)*cosd(delta)*cosd(omega)*cosd(phi)) ...
                           + (cosd(beta)*sind(delta)*sind(phi)) ) ;

            G_b(j,i,t) = (B(j,i)*max(0,theta_s))/(cosd_theta_zs) ;
            G_d(j,i,t) = ( D(j,i)*(1-k_factor(j,i))*((1+cosd(beta))/(2)) ) + ( (D(j,i)*k_factor(j,i)*max(0,theta_s))/(cosd_theta_zs) ) ;
            G(j,i,t) = G_b(j,i,t) + G_d(j,i,t) ;
            end

        end

end

S = sum(G)/12 ;
dry_season = ( G(1,:,:) + G(2,:,:) + G(3,:,:) + G(4,:,:) + G(5,:,:) + G(10,:,:) + G(11,:,:) + G(12,:,:) ) / 8 ;
rainy_season = ( G(6,:,:) + G(7,:,:) + G(8,:,:) + G(9,:,:) ) / 4 ;
```

```
jan = G(1,:,:) ;
feb = G(2,:,:) ;
mar = G(3,:,:) ;
apr = G(4,:,:) ;
may = G(5,:,:) ;
jun = G(6,:,:) ;
jul = G(7,:,:) ;
aug = G(8,:,:) ;
sep = G(9,:,:) ;
oct = G(10,:,:) ;
nov = G(11,:,:) ;
dec = G(12,:,:) ;

for j = 1:4

    for i = 1 : (length(omegas)-1)

        dA_dry(i,j) = ( ( dry_season(1,i+1,j) +
          dry_season(1,i,j) ) * (UTC(i+1)-UTC(i) ) ) /
           2 ;
        dA_rainy(i,j) = ( ( rainy_season(1,i+1,j) +
          rainy_season(1,i,j) ) * (UTC(i+1)-UTC(i) ))
          / 2 ;

        dA_jan(i,j) = ( ( jan(1,i+1,j) + jan(1,i,j) ) *
            (UTC(i+1)-UTC(i) ) ) / 2 ;
        dA_feb(i,j) = ( ( feb(1,i+1,j) + feb(1,i,j) ) *
            (UTC(i+1)-UTC(i) ) ) / 2 ;
        dA_mar(i,j) = ( ( mar(1,i+1,j) + mar(1,i,j) ) *
            (UTC(i+1)-UTC(i) ) ) / 2 ;
        dA_apr(i,j) = ( ( apr(1,i+1,j) + apr(1,i,j) ) *
            (UTC(i+1)-UTC(i) ) ) / 2 ;
        dA_may(i,j) = ( ( may(1,i+1,j) + may(1,i,j) ) *
            (UTC(i+1)-UTC(i) ) ) / 2 ;
        dA_jun(i,j) = ( ( jun(1,i+1,j) + jun(1,i,j) ) *
            (UTC(i+1)-UTC(i) ) ) / 2 ;
        dA_jul(i,j) = ( ( jul(1,i+1,j) + jul(1,i,j) ) *
            (UTC(i+1)-UTC(i) ) ) / 2 ;
```

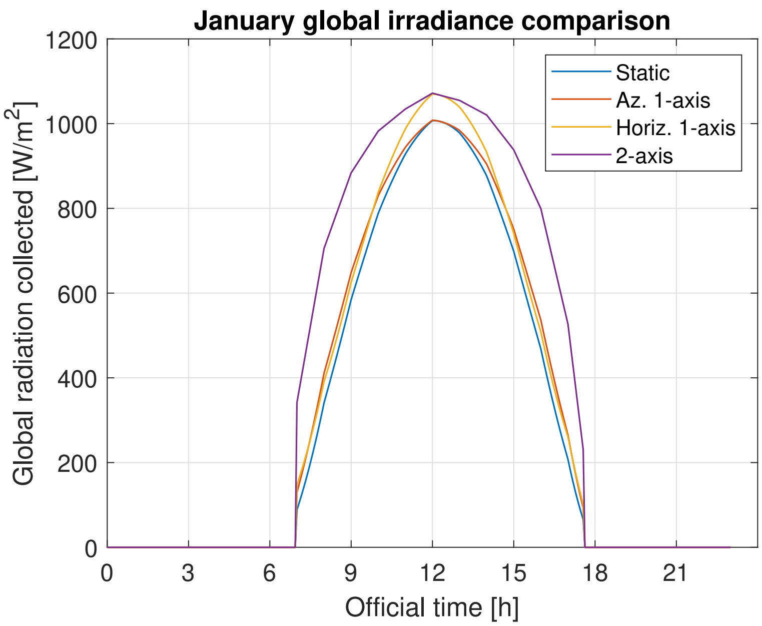
```
            dA_aug(i,j) = ( ( aug(1,i+1,j) + aug(1,i,j) ) *
                (UTC(i+1)-UTC(i) ) ) / 2 ;
            dA_sep(i,j) = ( ( sep(1,i+1,j) + sep(1,i,j) ) *
                (UTC(i+1)-UTC(i) ) ) / 2 ;
            dA_oct(i,j) = ( ( oct(1,i+1,j) + oct(1,i,j) ) *
                (UTC(i+1)-UTC(i) ) ) / 2 ;
            dA_nov(i,j) = ( ( nov(1,i+1,j) + nov(1,i,j) ) *
                (UTC(i+1)-UTC(i) ) ) / 2 ;
            dA_dec(i,j) = ( ( dec(1,i+1,j) + dec(1,i,j) ) *
                (UTC(i+1)-UTC(i) ) ) / 2 ;

        end

end

A_dry = sum (dA_dry,1) ;
A_rainy = sum (dA_rainy,1) ;

A_jan = sum (dA_jan,1) ;
A_feb = sum (dA_feb,1) ;
A_mar = sum (dA_mar,1) ;
A_apr = sum (dA_apr,1) ;
A_may = sum (dA_may,1) ;
A_jun = sum (dA_jun,1) ;
A_jul = sum (dA_jul,1) ;
A_aug = sum (dA_aug,1) ;
A_sep = sum (dA_sep,1) ;
A_oct = sum (dA_oct,1) ;
A_nov = sum (dA_nov,1) ;
A_dec = sum (dA_dec,1) ;

kWh_meses = [A_jan;A_feb;A_mar;A_apr;A_may;A_jun;A_jul;
    A_aug;A_sep;A_oct;A_nov;A_dec] ;
min_rad = min (kWh_meses) ;

%INSTALLATION
%Consumo equipo
C = 1000:2000:11000 ;
```

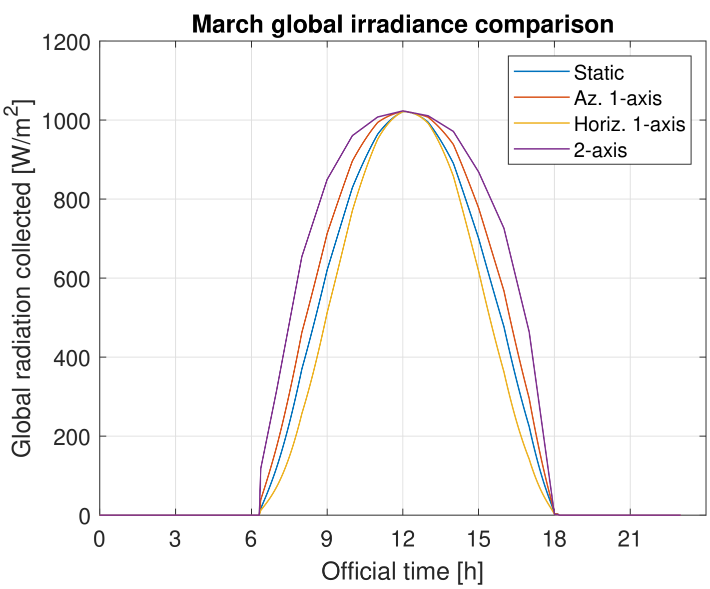
```matlab

%Rendimiento instalacion
eta = 0.75 ;
E = C ./ eta ;

%Hora solar pico
x = 4 ; %Tipo de tracker: 1)Static,2)Azimuth,3)Horiz,4)2-axis
HSP = min_rad(x) / 1000 ;

%Numero de modulos
p = 0.8 ; %perdidas ensuciamiento
W = 50:1:450 ;

for i=1:length(E)
    for j=1:length(W)
        N(i,j) = ( E(i) ) / (HSP * p * W(j) ) ;
    end
end

E_ob = (6 * HSP * p * 340)*eta ;

%Capacidad acumuladores
dias_autonomia = 1:1:4 ;
V = 24 ;
descarga=0.5:0.01:0.8 ;

for j=1:length(E)

    for i=1:length(dias_autonomia)

        for k=1:length(descarga)

        A(j,k,i) = (E(j) * dias_autonomia(i) )/ (V*descarga(k)) ;

        end
    end
```

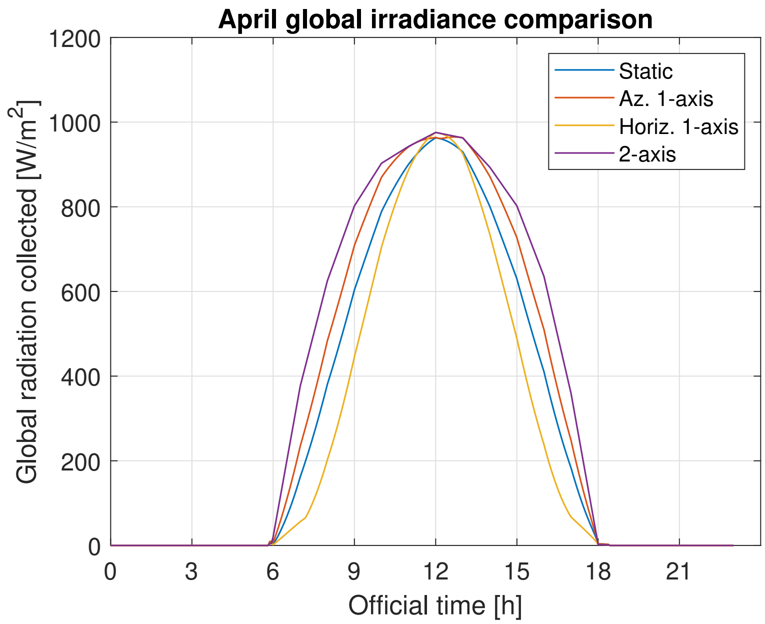
```
end
```

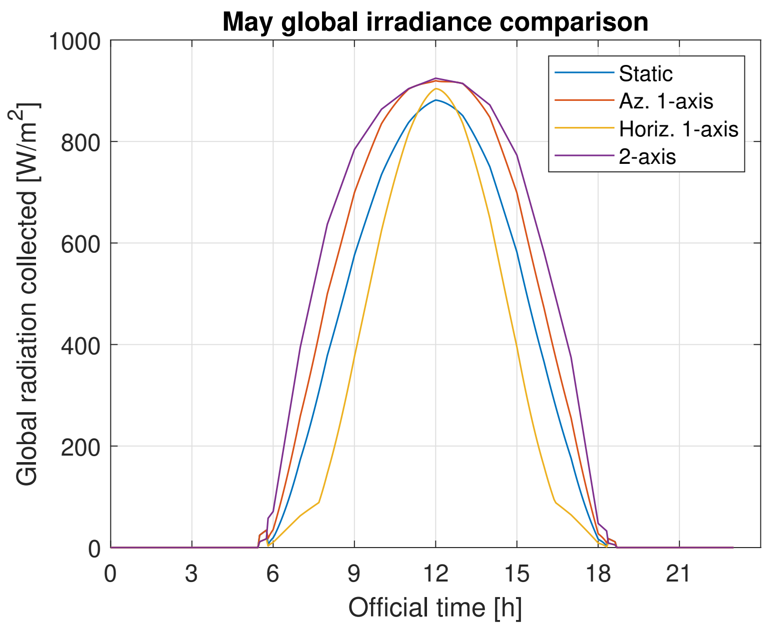
B Monthly global irradiance comparison

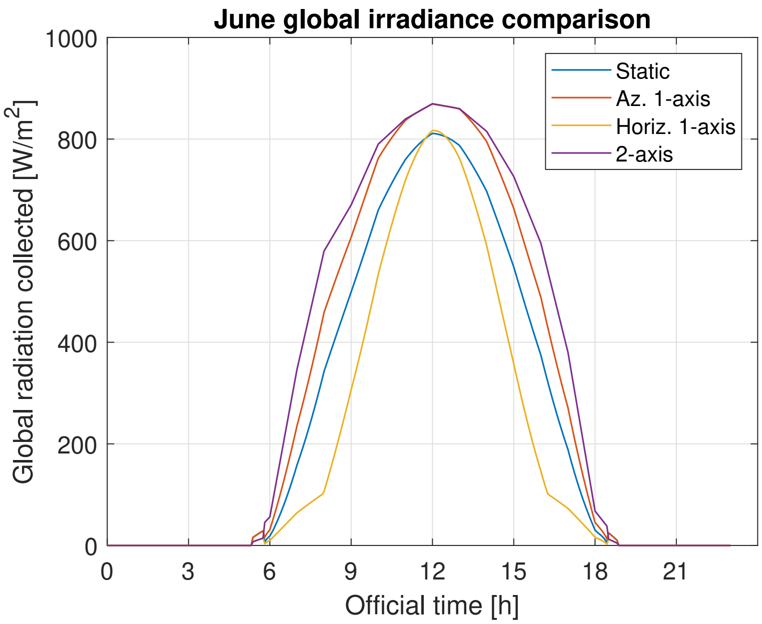
This appendix presents the monthly comparison between different tracker systems, which are discussed in section 4.1.

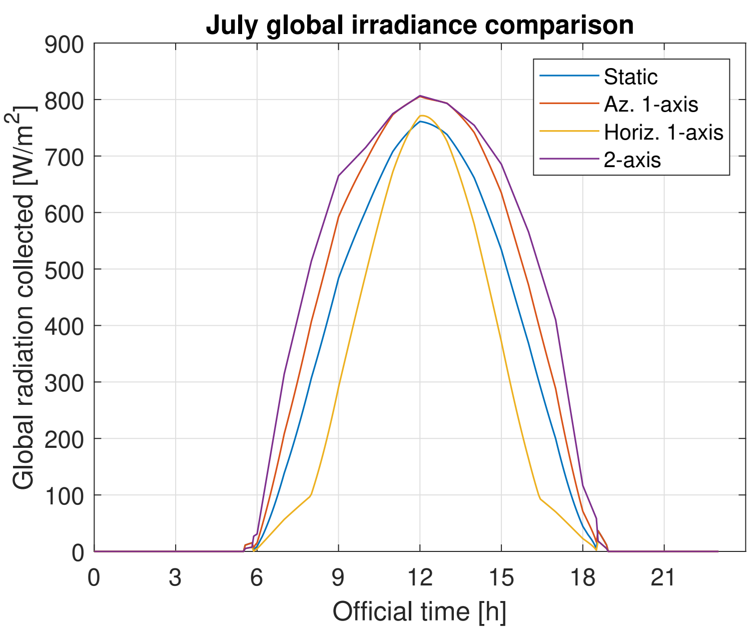


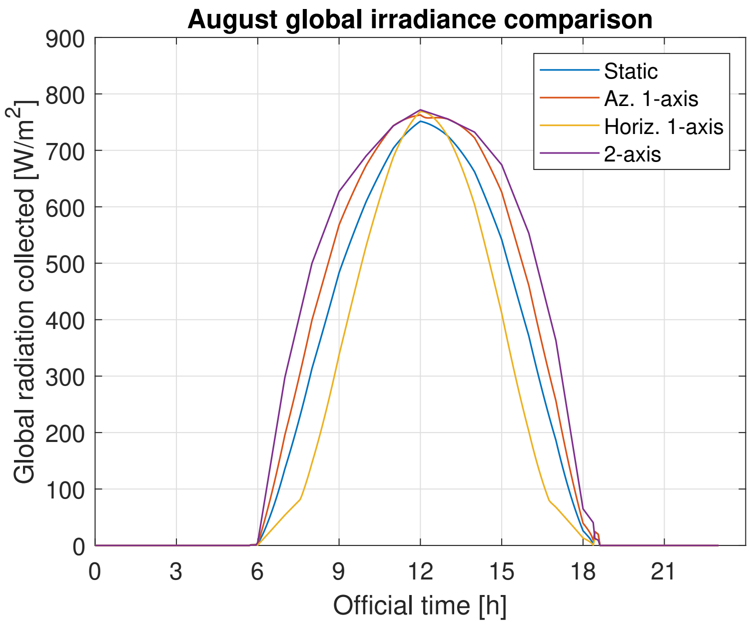


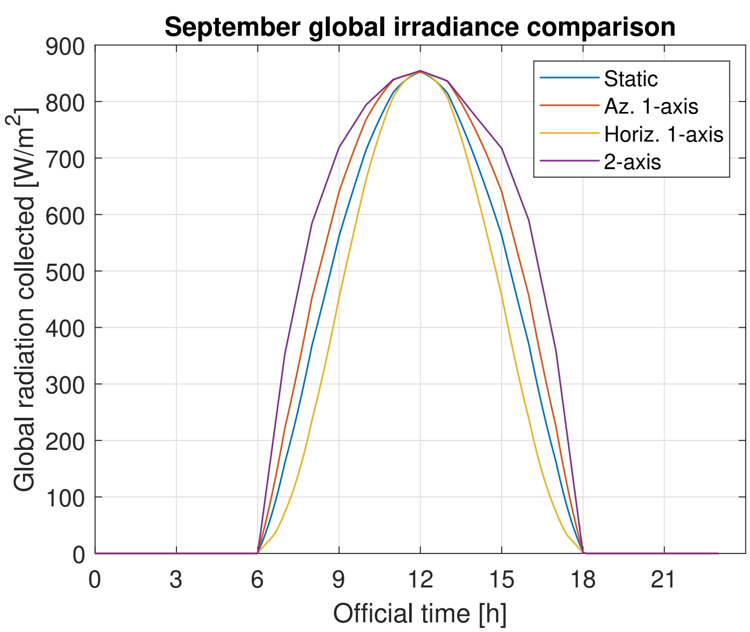

April global irradiance comparison
Static
Az. 1-axis
Horiz. 1-axis
2-axis
Global radiation collected [W/m2]
Official time [h]
0
3
6
9
12
15
18
21
0
200
400
600
800
1000
1200

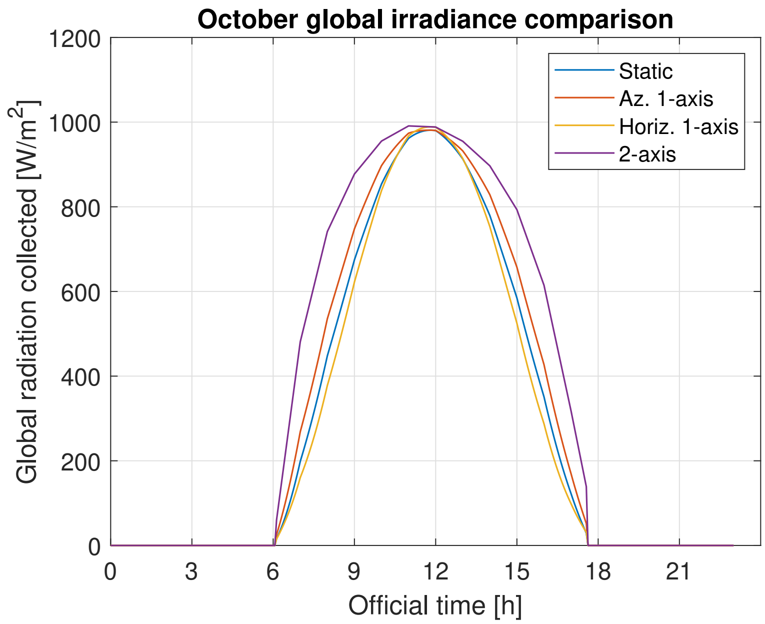

May global irradiance comparison
Static
Az. 1-axis
Horiz. 1-axis
2-axis
Global radiation collected [W/m2]
Official time [h]
0
3
6
9
12
15
18
21
0
200
400
600
800
1000

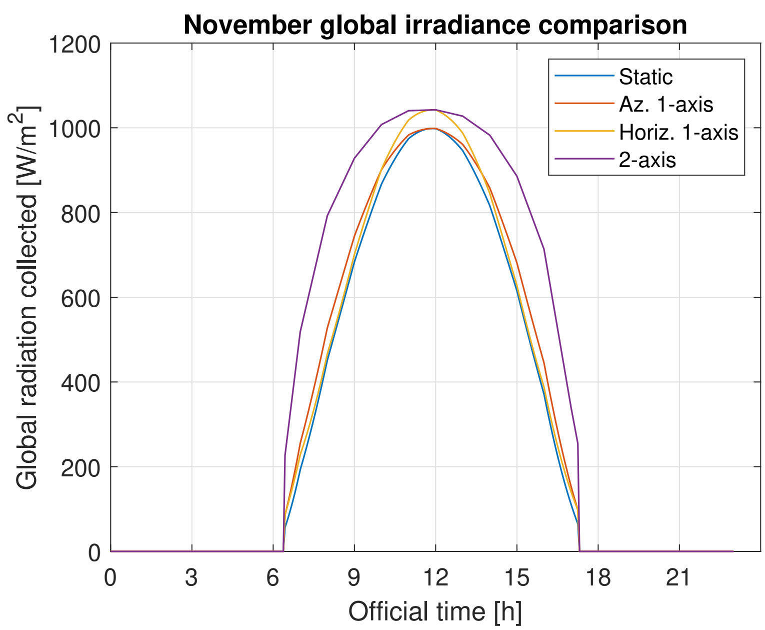

June global irradiance comparison
Global radiation collected [W/m^2]
Official time [h]
Static
Az. 1-axis
Horiz. 1-axis
2-axis
0
200
400
600
800
1000
0
3
6
9
12
15
18
21


July global irradiance comparison
Global radiation collected [W/m^2]
Official time [h]
Static
Az. 1-axis
Horiz. 1-axis
2-axis
0
100
200
300
400
500
600
700
800
900
0
3
6
9
12
15
18
21


August global irradiance comparison
Global radiation collected [W/m2]
Official time [h]
Static
Az. 1-axis
Horiz. 1-axis
2-axis
0
100
200
300
400
500
600
700
800
900
0
3
6
9
12
15
18
21


September global irradiance comparison
Global radiation collected [W/m2]
Official time [h]
Static
Az. 1-axis
Horiz. 1-axis
2-axis
0
100
200
300
400
500
600
700
800
900
0
3
6
9
12
15
18
21


October global irradiance comparison
Global radiation collected [W/m2]
Official time [h]
Static
Az. 1-axis
Horiz. 1-axis
2-axis
0
200
400
600
800
1000
1200
0
3
6
9
12
15
18
21


November global irradiance comparison
Global radiation collected [W/m2]
Official time [h]
Static
Az. 1-axis
Horiz. 1-axis
2-axis
0
200
400
600
800
1000
1200
0
3
6
9
12
15
18
21

C Solar geometry in Burkina Faso

This appendix presents the azimuth and zenith angle values for a year in Burkina Faso. The results are discussed in section 4.3.2.

```
% Solar geometry

% Control parameters
phi = ; % Latitud [deg]
omegas = -180:1:180; % Hora solar [deg] [-180,180]
days = 1:1:365; % Dia any [1,365]

% Preallocate arrays
zenmd = NaN(length(days),1); % Zenith at midday
azmd = NaN(length(days),1); % Azimuth at midday
deltas = NaN(length(days),1); % Sun declination

% Loop days
for k = 1:length(days)

    % Day
    day = days(k);

    % Declination of Sun [deg]
    delta = 23.45 * sind((360 * (day + 284)) / 365);
    deltas(k) = delta;

    % Preallocate vars
    theta_zss = NaN(length(omegas),1);
    psi_ss = NaN(length(omegas),1);

    % Loop solar hour angle
    for i=1:length(omegas)

        % Current hour angle
        omega = omegas(i);

        % Pre-compute some parameters for performance
```

```
        cosd_phi = cosd(phi);
        sind_phi = sind(phi);
        cosd_delta = cosd(delta);
        sind_delta = sind(delta);
        cosd_omega = cosd(omega);
        sind_omega = sind(omega);

        % Compute Zenith angle
        cosd_theta_zs = (cosd_delta * cosd_omega *
           cosd_phi) +  (sind_delta * sind_phi);
        sind_theta_zs = sind(acosd(cosd_theta_zs));
        theta_zs = acosd(cosd_theta_zs);

        % Compute azimuth angle
        sind_psi_s = (cosd_delta * sind_omega) /
           sind_theta_zs;
        cosd_psi_s = (cosd_delta * cosd_omega *
           sind_phi - cosd_phi * sind_delta) /
           sind_theta_zs;

        % Get correct value of angle psi_s
        psi_s = asind(sind_psi_s);
        if cosd_psi_s < 0
            if sind_psi_s > 0, psi_s = 180 - psi_s;
            elseif sind_psi_s < 0, psi_s = -180 - psi_s
               ;
            else, psi_s = psi_s + 180;
            end
        end

        % Save angle values
        theta_zss(i) = theta_zs; % Zenith angle
        psi_ss(i) = psi_s; % Azimuth angle

    end

    % Get minimum zenith
```

```
    zenmd(k) = min(theta_zss); % Minimum Zenith (sun is highest on sky)
    idxmd = find(theta_zss==zenmd(k)); % Index at min zenith
    azmd(k) = psi_ss(idxmd); % Azimuth at min zenith

end

% Plot min zenith + azimuth vs day of year
figure(); hold on; grid on; box on;
xlabel('Day of the year');
xlim(days([1,end]));
xticks([1,30:30:365]);
yyaxis left;
plot(days,zenmd,'.');
ylabel('Minimum Zenith angle [deg] (0 is up, 180 is down)');
yyaxis right;
plot(days,azmd,'x');
ylabel('Azimuth angle at min zenith [deg] (0 is south, 180 is north)');

% Plot sun declination vs day of year
figure(); hold on; grid on; box on;
xlabel('Day of the year');
xlim(days([1,end]));
xticks([1,30:30:365]);
plot(days,deltas,'.'); % Sun declination
plot(days,0*days+phi,'--'); % Latitud geografica
ylabel('Sun declination angle [deg]');
legend('Sun Declination','Geographic Latitude');
```

D Bolt geometry tables

This appendix presents bolt geometry tables, which are used in section 5.5. Both tables have been adapted from *Uniones*, Universidade da Coruña.

	M 10	M 12	M 16	M 20	M 22	M 24	M 27	M 30	M 33	M 36
longitud nominal l (mm)	longitud de la caña l_g (mm)									
30	10	8								
35	15	13	9							
40	20	18	14	10	8					
45	25	23	19	15	13	11				
50	30	28	24	20	18	16				
55	35	33	29	25	23	21				
60	40	38	34	30	28	26	23			
65	45	43	39	35	33	31	28			
70	50	48	44	40	38	36	33			
75	55	53	49	45	43	41	38			
80		58	54	50	48	46	43	40		
85		63	59	55	53	51	48	45		
90		68	64	60	58	56	53	50		
(95)		73	69	65	63	61	58	55		
100		78	74	70	68	66	63	60	57	54
(105)		83	79	75	73	71	68	65	62	59
110		88	84	80	78	76	73	70	67	64
(115)		93	89	85	83	81	78	75	72	69
120		98	94	90	88	86	83	80	77	74
(125)			99	95	93	91	88	85	82	79
130			104	100	98	96	93	90	87	84
140			114	110	108	106	103	100	97	94
150			124	120	118	116	113	110	107	104
160				130	128	126	123	120	117	114
170				140	138	136	133	130	127	124
180					148	146	143	140	137	134
190					158	156	153	150	147	144
200					168	166	163	160	157	154

Se evitarán en lo posible los valores entre paréntesis

TIPO	vástago	cabeza			área resistente
	d (mm)	k (mm)	s (mm)	e (mm)	A_s (cm^2)
M 10	10	7	17	19,6	0,580
M 12	12	8	19	21,9	0,843
M 16	16	10	24	27,7	1,570
M 20	20	13	30	34,6	2,450
(M 22)	22	14	32	36,9	3,030
M 24	24	15	36	41,6	3,530
(M 27)	27	17	41	47,3	4,560
M 30	30	19	46	53,1	5,610
(M 33)	33	21	50	57,7	6,940
M 36	36	23	55	63,5	8,170

Se recomienda no utilizar los tornillos cuyo tipo figura entre paréntesis