MATLAB Tool for Analyzing Sun's Path

[1]Pulkit Singh Rana, [2]Prateek Gupta
[1]Student, [2]Student,
[1]Electrical and Electronics Engineering,
[1]Maharaja Agrasen Institute of Technology, Delhi, India

Abstract: The paper is focused upon finding an alternative to traditional solar tracker for the solar panels. In the paper, we developed a method which can track the sun without using any tracking device by calculating sun's path for given latitude. This tool will help us in determining the exact sun's path for any day of the year.

Index Terms - P.V- Photovoltaic, Solstices, Equinox

I. Introduction

The cities are becoming smart and technological advancements are scaling new heights everyday which further helps in making these smart cities even more environment friendly. Sustaining a content life and fulfilling basic needs are to be taken care with immediate effect from today onwards so that we can prepare for the future well in advance.

Solar energy has become inexpensive enough to manufacture and advanced enough to replace most or all of the energy that you get from the regular power grid. While you can set up a solar system without a tracking device, there are some pretty compelling reasons to include one. Stationary mounts which hold these panels in a fixed position have a decreased productivity when the sun passes to a less-than-optimal angle. It is calculated that 30.79% more PV electricity is obtained using the sun tracking system when compared to the latitude tilt fixed system [1]. Solar trackers generate more electricity in roughly the same amount of space needed for fixed tilt systems, making them ideal optimizing land usage. But these solar trackers are very expensive and are suitable for smaller jobsites.

Implementing the conventional tracking system might not be feasible for personal usage. This paper is aimed at developing an open loop system which can track the sun based on the latitude of the given location.

II. Sun's Path

Tracking down the sun's path is one of the crucial aspects of our project as the reflection system of our project depends upon the sun's pathway during a particular season. Sun's path will also determine the efficiency of the reflection system

To obtain the Sun's path first we need to define the North-South and East-West direction in X-Y plane. The configuration which we used is: negative Y direction equals North and negative X direction equals East. Also, we can approximate the sun's path of a day by cutting a hollow sphere completely with a plane whose normal is parallel to the "North pole of the sky" (the radius of the hollow sphere doesn't matter). If we can find the maximum Altitude in whole (let's say it Alpha – "α") than we can find equation of the plane and we can use this equation to find sun's path on 21st June (Summer Solstice). We also noted that maximum Altitude difference between Summer Solstice and Winter Solstice (21st Dec) is 46.9 degrees hence using "α" we can also calculate the path of the sun on 21st December, rest of the days sun's path lies in between these two paths like maximum Altitude difference between Summer Solstice and 23rd September (Equinox) is 23.45 degrees and sun's path on 23rd September and 20th March is same.

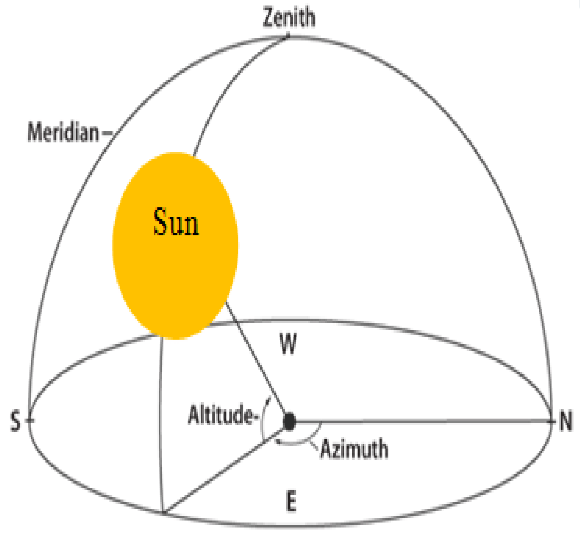

Fig.3.1

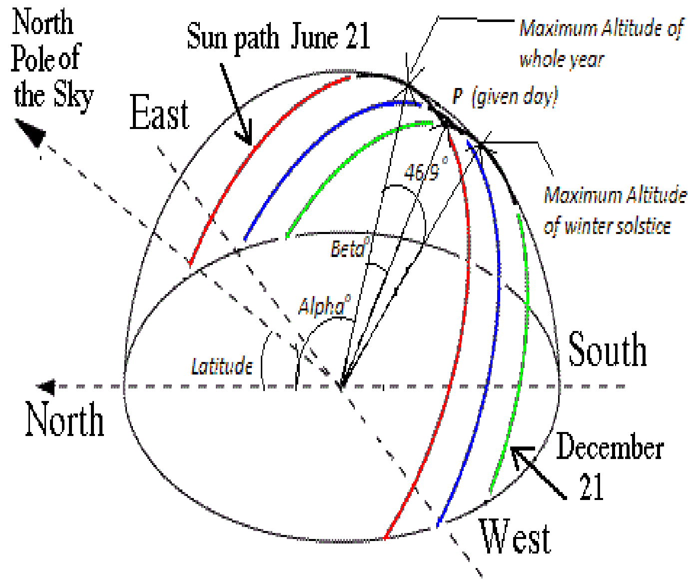

Fig 3.2

The variables which we required to calculate these paths are:

- the Altitude of "North pole of the sky" and
- Alpha (α)

The altitude of "North pole of the sky" of given location is equal to the Latitude of that location and alpha (α), if we measure it from north axis is "Latitude + 90 - 23.45" degrees of that location. For our project purpose, we found the Latitude and alpha (α) of Delhi is around 28.7 and 95.25 degrees respectively.

Now let say we want to calculate the sun's path of any day of the year, we know that maximum altitude difference between that day and summer soloists is going to be in between 0 and 46.9 degrees let's say it is Beta (β) for given day. For simplicity,

- let vector "North pole of the sky" = N
- The highest altitude points of that day's sun's path = P
- The radius of the sphere be = R

So here,

$$N = [0, -\cos(\text{Latitude}), \sin(\text{Latitude})] \quad (1)$$

$$P = R*[0, -\text{cosd}(\alpha+\beta), \sin(\alpha+\beta)] \quad (2)$$

Now if [ax + by + cz + d = 0] is the equation of the plane then,

$$a = N(1) \quad x_o = P(1) \quad (3)$$

$$b = N(2) \quad y_o = P(2) \quad (4)$$

$$c = N(3) \quad z_o = P(3) \quad (5)$$

and

$$d = -(a*x_o + b*y_o + c*z_o) \quad (6)$$

and the equation of the sphere is

$$X^2+Y^2+Z^2=R^2 \quad (7)$$

Now if [x, y, z] are the coordinates of sun's path and we chose 'z' as our controlling vector which varies from 0 to 'z_o' than on solving plane and sphere intersection equation we would get,

$$x = -\text{sqrt}(R^2-z^2-((d+c*z)/b^2) \quad (8)$$

$$y = -(d+c*z) / b \quad (9)$$

Above equation gives half of the sun's path other half is the mirror image where Y-Z plane act as a mirror and image form on positive side of X axis. Hence by changing 'z' from 0 to 'z_o' we can get coordinates of sun's path of given day. This sun path has a sharp curve around X-axis, so we need to find more points on this curve path so to have a smoother curve. Now theoretically if 'z' remains in between 0 and 'z_o', 'x' should always result in a real number but because computer uses numerical computations, but this might not be the case every single time. For example, in some cases as 'z' is approaching 'z_o' and 'x' is approaching zero and x comes out to be,

$$x = 0.0001 + j*0.00000001 \quad (10)$$

So, we have to use absolute value of "(R^2-z^2-((d+c*z)/b^2)" to compute 'x', that is why we used equation (8).

Also, we can see in figure as 'z' approaches 'z_o' slope of sun's path curve approaches zero hence we have to use more points near z = z_o, so that we can capture every important point of that sharp turn near z = z_o.

This can be done using "sine" function because the below figure it follows similar trajectory between 0 and π/2.

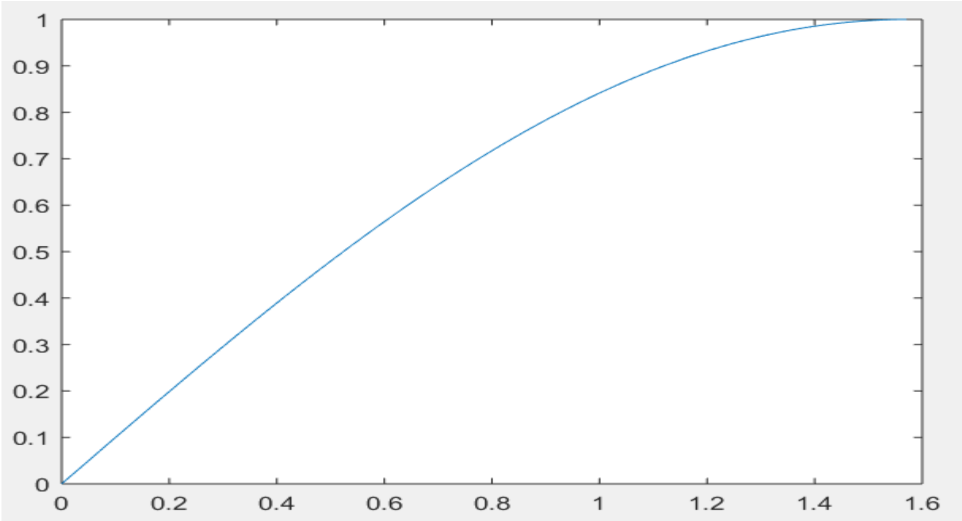

Fig 3.3

This way we can increase number of points of 'z' near 'z_o' hence,

$$z= z_o*\sin (0 \text{ to } \pi/2) \quad (11)$$

Now the thing we need to focus is controlling the minimum sun's altitude because photosynthesis does not start right after sun appears above horizon because initially intensity of sun light is not enough to carry out the process of photosynthesis or may be some buildings are on the sight of light which may prevent it to reach heliostats or even the horizon of the heliostats is different from ground horizon as they are being placed on the top of the building.

Let's call that minimum altitude "Gama" (γ).

We can use 'γ' to control minimum value of 'z' hence,

$$z = R*\sin(\gamma)+(z0-R*\sin(\gamma)) *\sin (0 \text{ to } \pi/2) \quad (12)$$

Now the variable we need to know to get a given day's sun's path:

1. Latitude (Depends on Geographic Location)
2. *β* (Maximum Altitude difference between Summer Solstice and Given day) (Depends on Given day)
3. γ (Minimum Altitude Limit) (Depends on user Requirement)
4. R (Distance of Apparent sun from Origin) (Depends on user Requirement)
5. Detail (Number of points on curve) (Depends on user Requirement)

Now all we need to do is to find a way of controlling the value of 'β' according to a given date. For this we have a function called "days365".

This function takes two input day1 and day2 and gives back the number of days between day2 and day1 for example "days365" ('1-jan', '21-june') = 171". It should be clear by now that sun repeats same path twice a year from 21-June (Summer Solstice) to 21-Dec (Winter Solstice) then back to 21-June hence value of 'β' also repeats itself from 0 to 46.9 and then back to 0 degrees. Now since we know exact value of 'β' of four days like 21-June, 23-Sep, 21-Dec and 20-March, we can get approximate value of 'β' between them using above mentioned function.

Let,

num1 = days365('21-june','23-sep') (13)

num2= days365('23-sep','21-dec') (14)

num3= days365('21-dec-2018','20-march-2019') (15)

num4= days365('20-march','21-june') (16)

Now we need to get num1 amount of equally spaced angles between 0 and 23.45 degrees let's call these vectors 'vec1' and we need to do this for all days, so function "linspace" is very useful here,

vec1= linspace (0,23.45, num1) (17)

vec2= linspace (23.45,46.9, num2) (18)

vec3= linspace (46.9,23.45, num3) (19)

vec4= linspace (23.45, 0, num4) (20)

Let's join these angle vector into one 365 day angle vector and call it 'β vector', here first value of 'β vector' is of 21-June and last value of 'β vector' is of 20-June now since "days365" ('21-june-2018', '1-jan-2019') =194" it means 195th value of 'β vector' is of 1-Jan hence if we consider 'β vector' like a circular vector and shift every element 195 position backward or (365-194=171) 171position forward we would end up with a vector whose first 'β' value is of 1-Jan and last 'β' value is of 31-Dec.

Basically, we are rearranging Sun's path from "21st June to 20th June" to "1st January to 31st December" as it will allow us to get the path of sun at any desired day of the year using this function easily

Let's test this function, let

1) Latitude = 28.7
2) Gama (γ) = 23.66
3) R = 100
4) Detail = 50
5) Day = Whole Year

This what result we got,

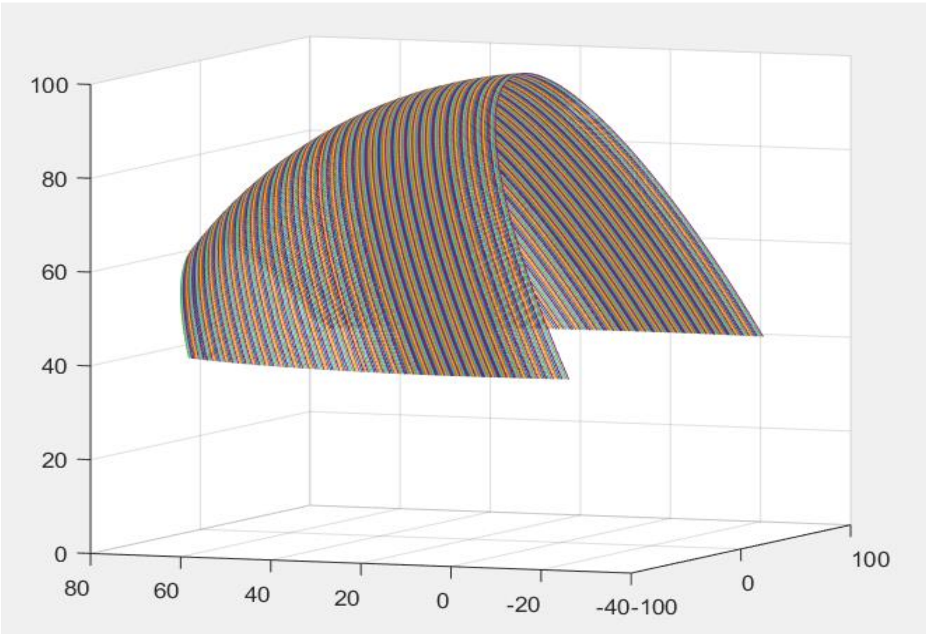

Fig 3.4

Here every single curve represents every single day.

III. Conclusion

The computer model which is developed is capable of tracking the sun using latitude of the given location which can be provided by the user. The detail of final output of the model can be controlled by user and also by defining the value of "Gamma" (γ), minimum altitude from which the actual tracking of the sun starts, can be controlled.

The calculated sun's path might differ from the actual path from 2-3 days. This is because the earth rotates around the sun in an elliptical orbit and due to this elliptical orbit, the velocity of rotation of earth around sun is different for different time-periods and as we move closer to 'solstices' and 'equinoxes' the accuracy of this model increases and vice-versa.

References

[1] Eke, Rustu & Sentürk, Ali. (2012). Performance comparison of a double-axis sun tracking versus fixed PV system. Solar Energy. 86. 2665-2672. 10.1016/j.solener.2012.06.006

[2] Adrian W.Y.W., Durairajah V., Gobee S. (2014) Autonomous Dual Axis Solar Tracking System Using Optical Sensor and Sun Trajectory. In: Mat Sakim H., Mustaffa M. (eds) The 8th International Conference on Robotic, Vision, Signal Processing & Power Applications. Lecture Notes in Electrical Engineering, vol 291. Springer, Singapore

[3] F. I. Mustafa, A. S. Al-Ammri and F. F. Ahmad, "Direct and indirect sensing two-axis solar tracking system," *2017 8th International Renewable Energy Congress (IREC)*, Amman, 2017, pp. 1-4

[4] Chang T.P, "The Sun's apparent position and the optimal tilt angle of a solar collector in the northern hemisphere", Vol.83, Issue 8, August 2009, Pages 1274-1284

[5] Kalogirou S.A, "Design and construction of a one-axis sun-tracking system", Volume 57, Issue 6, December 1996, Pages 465-469

[6] Jenkins A, "The Sun's Position in Sky", European Journal of Physics, Volume 34, Number 3

[7] MATLAB and Financial Toolbox Release R2006a, The MathWork, Inc., Natick, Massachusetts, United States