German International University

Engineering Faculty

Implementation of a Dual-Axis Tracking System for Parabolic Dish Collectors

Bachelor Thesis

Author: Ahmed El Araby

Supervisor: Dr. Yehia Eissa

Submission Date: 09 February, 2022

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree

(ii) due acknowledgement has been made in the text to all other material used

---

Ahmed El Araby
09 February, 2022

Acknowledgments

First and foremost I am thankful to ALLAH for giving me the strength, knowledge and opportunity to undertake this study and complete it satisfactorily.

It is with immense gratitude that I acknowledge the assistance of my Supervisor Dr. Yehia Eissa for his guidance along this journey and for always supporting me during the making of this thesis. I would also like to mention that I'm thankful to all my tutors being Professors and Teaching Assistants in the past 3 years of my undergraduate career who taught me how to be an engineer and professional in all standards.

Furthermore, I would like to thank my father for his help with the manufacturing of the body of the mechanism at Arabo filters' factory, which otherwise achieving it would not have been possible.
Moreover, I can't express my gratitude enough to my colleagues and friends who helped me throughout the years and for their feedback and moral support.

Last but not least, I have to appreciate my family's support and motivation, without them all of this could never have been achieved.

Abstract

Solar energy is abundant in our environment, and making use of it is a must as it is clean and renewable, and solar generation has seen growth over the last decades. Thus, this experimental study examines the implementation challenges faced in the development process of a dual-axis solar tracker for a parabolic dish collector (PDC) in the Egyptian market. Firstly, this system uses chronological solar tracking technology which is time-based and automatic, and the movement degrees of this system is 2D since based on various previous research, is more energy efficient. Secondly, This system uses a simple economical electro-mechanical system that consists of motors, drivers, and a microcontroller to execute the program performing the solar tracking. Finally, the hardware was machined and assembled in a factory using lathe machines and other machining tools to make sure the system is as precise as needed. The system was tested and successfully achieved accurate solar tracking of the sun throughout the day. Thus, we conclude that a relatively large-scale solar tracker for PDCs is possible to create with a manageable budget.

Nomenclature

Symbols [1]

α	Elevation angle
β	Module tilt angle
δ	Declination angle
γ	Azimuth angle
λ	Longitude
ϕ	Latitude
ψ	Module azimuth angle
θ_z	Zenith angle
I_0^n	Extraterrestrial solar radiation (solar constant)

Acronyms

CSP	Concentrating Solar Power
DHI	Direct Horizontal Irradiation
DNI	Direct Normal Irradiation
EoT	Equation of Time
JD	Julian Date
LST	Local Solar Time
LSTM	Local Standard Time Meridian
MCS	MicroChip Studio

MCU Microcontroller Unit

PDC Parabolic Dish Collector

SC Solar Constant

TC Time Correction Factor

Contents

Chapter 1

Introduction

1.1 Background

There's no denying the importance of solar energy. The return on investment of going solar is highly valuable, and not just financially, but for the sake of public health and environmental sustainability[2]. One of the solar energy systems used for the generation of electricity or heat are solar collectors which are used in this paper to demonstrate our implementation of a solar tracking system and algorithm.

The Sun rays reaches the sky from the time it rises until the time it sets.The sun path across the day is an essential variable as the position of the sun must be precisely normal to the solar collector's focal point, hence, increasing the collector's overall heat gain. This can be achieved by the use of a solar tracking system that can ensure that maximum energy is constantly and persistently harvested by accurately orienting the solar collector towards the Sun.

Sunlight is composed of two components; direct beam and diffuse radiation. . Non-concentrating solar collectors accept both direct and diffuse light. However, concentrating solar collectors only accept the direct beam portion since the energy produced is directly proportional to the reflected direct beam radiation that successfully hits its focal line/point. This makes the solar tracking system more essential in concentrating solar plants because unless directly pointed at the Sun they produce a negligible amount of energy.

There are multiple types of concentrating solar collectors but the most mainly used are:

1. **Parabolic trough collector** are made by simply bending a sheet of reflective or highly polished material into a parabolic shape called a parabola. Since solar light waves essentially travel parallel to each other, this type of solar collector can be pointed directly into the sun and still achieve a total focal output from all parts of the trough-shaped reflector as shown in figure 1.1 [3].

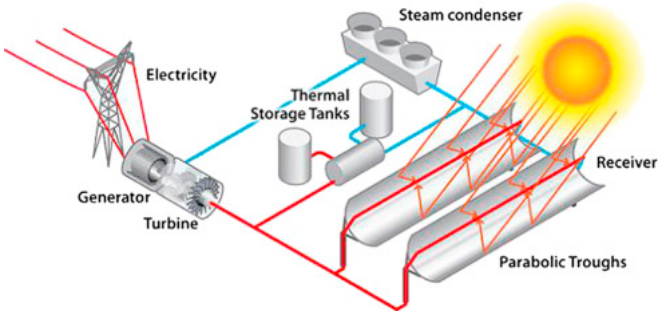

Figure 1.1: Figure showing the parabolic trough receiver system[3].

2. **Power tower receiver** In power tower concentrating solar power systems, a large number of flat, sun-tracking mirrors, known as heliostats, focus sunlight onto a receiver at the top of a tall tower as shown in figure 1.2 [4].

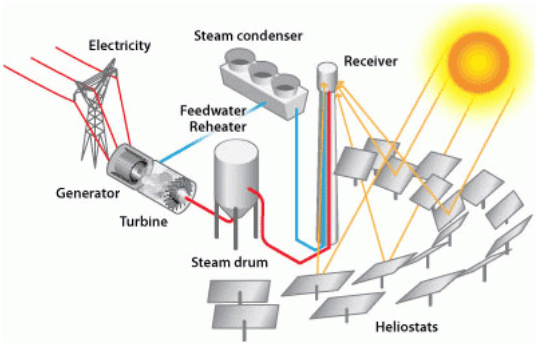

Figure 1.2: Figure showing the Power tower receiver system.[4]

3. **Parabolic dish collector** can be described as a concentrating solar collector that comes in a shape and appearance similar to that of a satellite dish as shown in figure 1.3. A parabolic dish does have reflectors like mirrors or any other reflective material and has an absorber at its focal point [5].

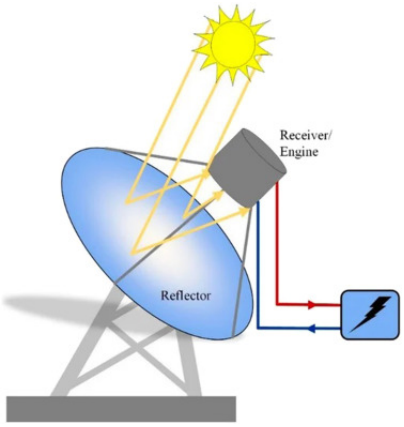

Figure 1.3: Principle of operation of PDC [6].

4. **Fresnel lens collector** The Fresnel Collector is a linear concentrating solar thermal collector optimized for industrial applications. It can provide heat up to 400°C and operate with pressures up to 120 bars. Besides being operated with pressurized hot water and thermal oil, it can also generate solar steam directly, shown in figure 1.4 [**?**]

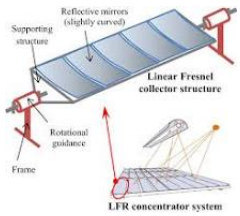

Figure 1.4: Principle of operation of PDC [7].

1.2 Related Work

1.2.1 Concentrating collector

There's a wide variety of concentrating collectors in the current market for heat generation. However, there are mainly three reasons for the use of PDC over the other concentrating collectors, the main reason is that the PDC's body which is the dish part of the whole machine can be replaced with flat plate collectors, photo-voltaics or even Solar panels, thus giving more meaning and use to the solar tracker. The second reason is that the

PDC has proved to be the most favourable concentrating collector (linear Fresnel lens with a secondary compound parabolic collector) in terms of technical, economic and environmental criteria [8]. The third and final reason is that other concentrating methods are large scale and economically challenging for a student. To summarize, the PDC was selected for:

1. PDC mechanical body can be replaced with other Solar generation methods
2. Superior economical, technical, and environmental capabilities compared to other solar collection methods
3. economically achievable

1.2.2 Solar tracking

In the global solar energy production there is continuously a push for more efficient energy production methods and technologies, which is measured by the LCOE (levelised cost of electricity) [9]. For the majority of uses, solar energy already has the lowest LCOE in the world, but future developments in solar power plants will allow them to produce power even more effectively.

In this subsection I'll give a brief explanation on the importance and differences in solar trackers used nowadays in the solar power generation industry.

What is a solar tracker

Solar tracking for Concentrated solar power (CSP) is required to be more precise and sensitive, unlike PV which isn't needed to be as exacting as CSP [10]. To maximize solar irradiance on the panel, solar trackers position the panels to get the most direct sunlight possible using a variety of electrical components [10]. Actuators, motors, sensors, and more components are included in solar trackers. By positioning the panels or reflectors to face the sun at the most direct angle, these tools allow the solar plant to receive the maximum amount of sunlight, improving its performance and efficiency. [10]

Solar tracking technologies

There are multiple technologies used for tracking the sun, apart from the degrees of movement, there are various principles of operations with different usabilities and applications:

1. Active Solar Tracking
2. Passive Solar Tracking
3. Chronological Solar Tracking

Active Solar Tracking The words active and passive occur commonly in engineering fields. Those devices which must be provided with a source of energy for their full performance are classed as active[11]. Those which do not need any external source of energy are called passive. Thus, active trackers must be provided with energy for their actuators to move, as illustrated in figure 1.5. That energy may even be part of the energy harvested by the PV or CSP systems they are driving. The actuator systems may consist of motors and other elaborate mechanical devices. Therefore the existence of sensors that read data is a necessity for this technology [11].

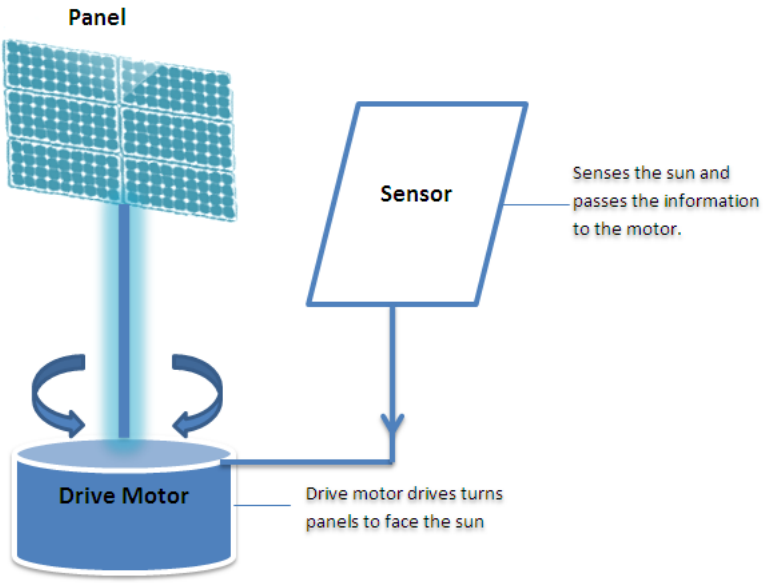

Figure 1.5: Active solar tracker using sensors[11].

Passive Solar Tracking Passive trackers will be using a low boiling point compressed gas fluid that is driven to one side or the other (by solar heat creating gas pressure) to cause the tracker to move in response to that of an imbalance. For such concentrated solar collector types, this non-precision orientation is undesirable, but it is suitable for typical PV panel kinds. [12], passive trackers could also be operated manually by hand (tilt angle).

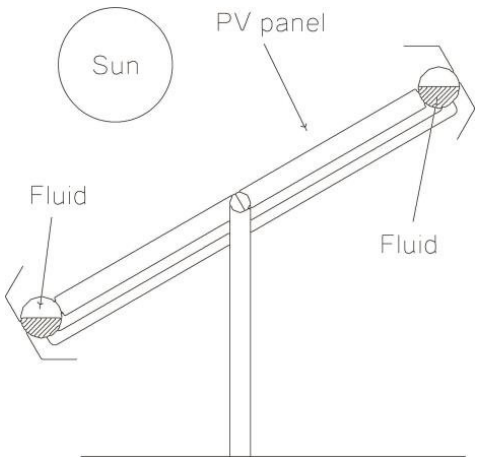

Figure 1.6: Passive solar tracker using cylindrical tubes filled with fluid.[13].

Chronological Solar Tracking The chronological tracking is time-based, whereby the structure is moved at a fixed rate throughout the day. The theory behind this is that the Sun moves across the sky at a fixed rate. Thus the drive motors are programmed to continuously rotate at a "slow average rate of one revolution per day (15 degrees per hour)". This method of Sun-tracking is of high accurate. However, the continuous rotation of the motor or actuator means more power consumption, and tracking the sun on a very cloudy day is unnecessary [14].

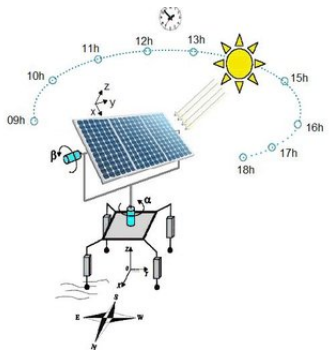

Figure 1.7: Example of chronological technology applied to Dual-Axis tracker following sun path[14]

1.3 Aims

what are the challenges faced when implementing dual-axes tracking for parabolic dish collectors? The aim of this study is to construct an open-loop dual-axis solar tracking system that follows chronological technology and examine its challenges, which utilizes the use of time-based algorithms for locating the position of the sun. This system uses a Dual axis tracking system which uses the calculated Altitude-Azimuth angles to follow the sun's path, as this is a PDC, which needs direct beams to be normal to its focal point at all times.

1.4 Outline

A brief introduction to the basics of solar energy, concentrating collectors, and solar trackers were explained, for the rest of the thesis, chapter 2 is a thorough literature review on Solar Radiation, Solar Geometry, Solar Trackers, and solar tracking algorithm. Chapter 3 elaborates on the methodology, the approach used to make this system operate, Chapter 4 reveals the experimental results of the study, moreover the general discussion.

Chapter 2

Literature Review

2.1 Solar Radiation

The design of solar systems requires knowledge of the useful solar radiation received on the surface of the installation, it is one of the essential variables to take into account for the production of solar energy.

A solar collector (PDC) is required to absorb solar radiation and use it depending on its application, in assessing the performance of said application, it is, therefore, crucial not only to determine its ability to absorb solar radiation but also to characterize its heat losses [15].

Solar radiation consists of direct beam, diffuse, and reflected radiations from the ground and adjacent surfaces. The amount of diffuse radiation falling on the solar collector depends on the view factor of the collector to the sky. The reflected radiation striking the collector's surface depends on the reflectivity of the surface, as well as on view factors and the amount of solar radiation reaching the reflecting surfaces. The amount of reflected radiation coming from the ground can be of an appreciable amount and can be amplified using special reflecting surfaces [16].

Solar radiation consists of three parts Direct Normal Irradiation (DNI), Direct Horizontal Irradiation (DHI), and Global Horizontal Irradiation (GHI): extraterrestrial solar energy which is above the atmosphere, and global solar energy which is under the atmosphere. The measured solar energy values can be used for developing solar energy models which describe the mathematical relations between solar energy and meteorological variables such as ambient temperature, humidity, and sunshine ratio[16]. These models can be later used to predict the direct and diffuse solar energy using historical meteorological data at sites where there is no solar

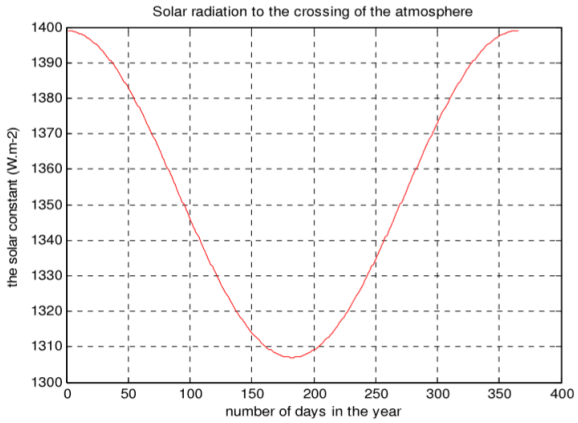

Figure 2.1: Variation of the extraterrestrial solar irradiance[20].

energy measuring device installed. Solar energy of daylight utilization for any site is dependent upon the quality of the available flux [17].

2.1.1 Solar radiation reaching earth's atmosphere

The earth's atmosphere intercepts some of the power radiated by the sun; it radiates an approximately constant power. The extraterrestrial irradiance on a surface normal to the solar radiation is given by [18] [2] [19] (see figure 2.1):

$$I_0 = SC(1 + 0.034 \cos \frac{360n}{365})$$

where: I_0^n is the Extraterrestrial solar radiation (solar constant). SC is the solar constant n is the day of the year

2.1.2 Solar energy at earth's surface

In the previous section, we posed a number of definitions allowing us to formulate the concept of solar radiation outside the atmosphere. This quantity is readily measurable because it is considered that it depends only on the distance between the measuring points of the Sun. We will see in this paragraph, the passage through the atmosphere will tend to complicate the understanding of the phenomenon, and the multiple interactions that may occur, in what follows, we consider the model of inclination of

global radiation in the case of a clear sky. Direct-beam radiation: It is used to describe solar radiation traveling in a straight line from the sun down to the surface of the earth[21].

The total amount of radiation received by a solar collector, also referred to as global horizontal irradiance, G, is composed of direct (beam), B, and diffuse, D, components.

$G = B + D$

Direct irradiation

This irradiation is received directly from the sun traveling through the atmosphere without modification. The capture area of the solar collector is placed in the following conditions:

- The sun's position is defined by its elevation (α) and azimuth (γ);
- The module tilt (β) and orientation (γ_s);
- The collector's system location in terms of Latitude (ϕ) and Longitude (λ).

However, for Dual-Axis tracking systems, which is our case, the solar collector is already adjusting itself to face the sun in the same fashion as sunflowers do. Thus, it is unnecessary to do the previous calculations, it is only mentioned for knowledge and for fixed solar applications [22].

Diffuse irradiation

Diffuse irradiation describes the sunlight falling on the surface of the Earth, which is scattered by air molecules or atmospheric particles such as aerosols or clouds [23][24].

The percentage of Direct and Diffuse irradiation is highly dependent on the insolation, taking that into account, The percentage of the sky's radiation that is diffuse is much greater in higher latitude, cloudier places than in lower latitude, sunnier places [24].

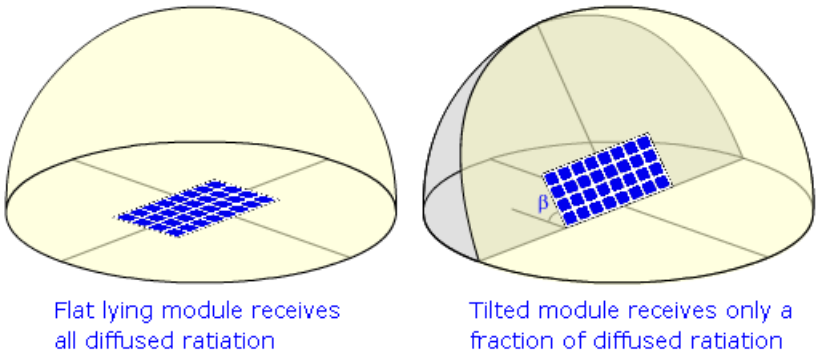

Figure 2.2: Although the simple model works pretty well for the desert locations it will introduce errors for the places with significant cloud coverage. For those parts of the Earth, special models of the sky should be used to achieve higher accuracy.[25][26]

Despite its uncertainty, its calculation is simpler, From a simple model, assuming isotropic radiation from the whole sky dome, it follows that the collector tilted by the angle β will receive just the proportional part of Diffuse Horizontal Irradiation, DHI.

2.2 Parabolic Solar Dish principle of operation

The parabolic Solar Dish collector as mentioned in 1.2.1 is one of the most favorable types of solar concentrators to work with[8]. Thus, in this section, we will discuss its principle of operation and conduct a literature review on it.

The PDC or sometimes referred to as Solar Dish Reflector is a concave mirror that optically reflects and focuses the sun's incident solar energy (DNI) onto a small receiving area using mirrors or lenses that focuses sunlight into a receiving focal point. By concentrating the sunlight to a single spot called the focal point, the intensity of the receiving solar energy is magnified many times over with each mirror or lens acting as a single sun shining directly at the same focal point on the solar dish meaning that more overall power/square meter of dish is achieved [27]. Mirrors or lenses can be facetted-segmented surfaces or a single parabolically shaped surface made in some forming process [28].

PDCs depend on 2-D axis tracking systems which is the purpose of this study, as it requires to be normal to the sun at all times. The concentration factor of the PDC is higher than all other concentrating collectors, where the concentration factor is the ratio between the area of sunlight collected and the area of the solar receiver onto which it is focused. The concentration factor, which is also known as the "number of suns" of a PDC can reach up to 1000 suns reaching temperatures at the "target" (focal point)[27] is one of the critical parameters affecting the efficiency of a Concentrated Solar Power system[29]. PDCs could reach temperatures ranging from $500°C$ to more than $1500°C$, depending on the size of the dish and the collector's location.

PDC from its name forms a paraboloidal-like shape by stamping them out from thin aluminum coated Mylar or thin sheet metal, with varying sizes and results, thoroughly explained in this study [30], concluding that depending on the dish diameter and degree of inclination control the location of the foci (target), and the lower the convexity, the lower the concentration of the reflected radiation, less the radiation intensity and less the temperature.

The PDC could also be covered with small mirror reflectors to help concentrate the thermal energy into a single focal point where the heat absorber is located producing more overall thermal energy per square meter of the dish. The same study previously mentioned studies the effect of using mirror lenses as reflectors [30], reaching that it has usability but gets easily damaged and is difficult to install. Therefore nickel sheet metal is preferred for use.

PDC also requires a collection method or better called "thermal receiver" which collects all this solar energy to convert it into heat. The solar receiver can be as simple as a small evacuated tube or a more complex solar heat engine, such as a Stirling engine generator [31].

2.3 Solar Geometry

The Astronomical Almanac is an algorithm that calculates the Elevation and Azimuth angles based on three parameters, Latitude, Longitude, and

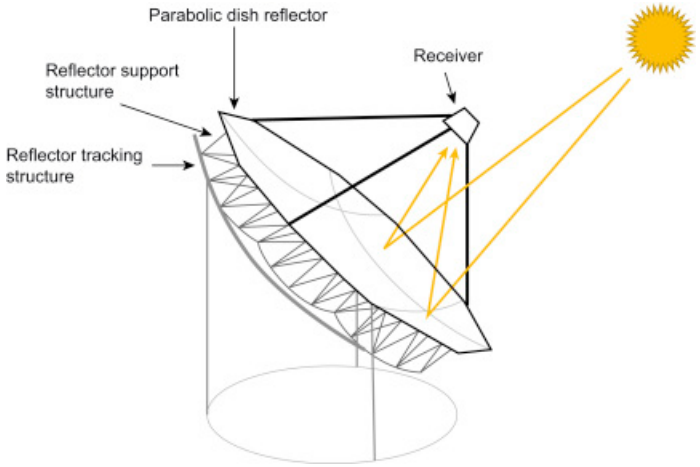

Figure 2.3: Reciever is fixed to the sun using 2-D solar tracking. The size of the receiver needs to be optimized to minimize the shadow it, and its support structure might create on the reflector. Also, a Stirling engine was placed at the receiver. Alternatively, the receiver might have a heat transfer fluid to drive an independent process or heat engine [32]

Time [33]. The algorithm has an accuracy of 0.01 deg until the year 2050. A brief and expansive explanation of the motion of the sun, Solar time, and the solar tracking angles will be elaborated for a better understanding of how the system works.

2.3.1 Motion of the Sun

The apparent motion of the sun, caused by the rotation of the Earth about its axis, changes the angle at which the direct component of light will strike the Earth. From a fixed location on Earth, the sun appears to move throughout the sky. The position of the sun depends on the location of a point on Earth, the time of day, and the time of year.[34]

This apparent motion of the sun has a major impact on the amount of power received by a solar collector. When the sun's rays are perpendicular to the absorbing surface, the power density on the surface is equal to the incident power density. However, as the angle between the sun and the absorbing surface changes, the intensity on the surface is reduced. When the module is parallel to the sun's rays (and the angle to the module normal = 90°) the intensity of light essentially falls to zero.[34] For intermediate

angles, the relative power density is $\cos(\theta)$ where θ is the angle between the sun's rays and the module normal.[34]

The angle between the sun and a fixed location on Earth depends on the particular location (the longitude of the location), the time of year and the time of day. [34]In addition, the time at which the sun rises and sets depends on the longitude of the location. Therefore, complete modeling of the sun's angle to a fixed position on Earth requires the latitude, longitude, day of the year, and time of day [34].

2.3.2 Solar Time

Until the late 19^{th} century most people used local solar time so that noon was when the sun was directly overhead during the day, and each town had its own definition. Transport was slow, so it did not matter that the time in a town miles away varied by a few minutes. The advent of railways necessitated more accurate timekeeping and time zones were introduced to keep an entire region on the same time [35]. Time zones follow political boundaries so that local time may be up to 2 hours different from solar time[34].

Twelve-noon local solar time (LST) is defined as when the sun is highest in the sky. Local time (LT) usually varies from LST because of the eccentricity of the Earth's orbit, and because of human adjustments such as time zones and daylight saving.

Local Standard Time Meridian (LSTM)

The Local Standard Time Meridian (LSTM) is a reference meridian used for a particular time zone and is similar to the Prime Meridian, which is used for Greenwich Mean Time. The LSTM is calculated according to the equation:

$$LSTM = 15^{\circ}\Delta T_{UTC}$$

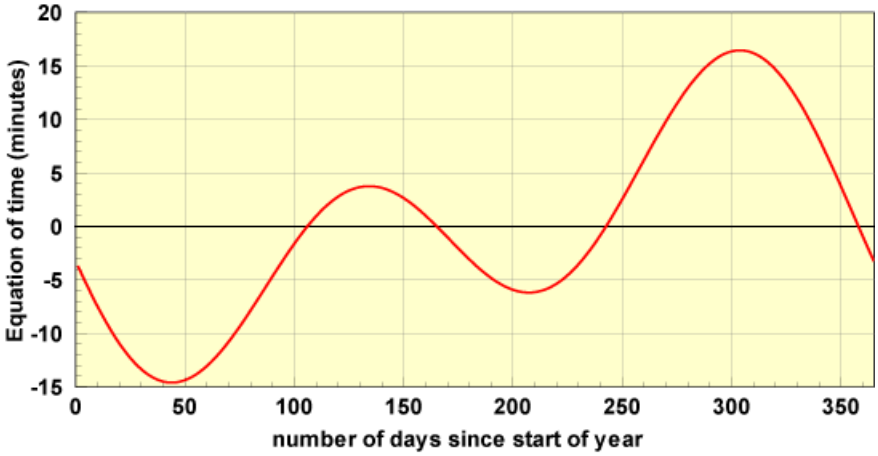

Figure 2.4: The LSTM, measured in degrees, which runs through the center of each time zone. It can be calculated by multiplying the differences in hours from Greenwich Mean Time by 15 degrees per hour [37].

where T_{UTC} is the difference of the Local Time (LT) from Universal Coordinated Time (UTC) in hours. ΔT_{UTC} is also equal to the time zone. 15°= 360°/24 hours. For instance, Sydney Australia is UTC +10 so the Local Standard Time Meridian is 150 °E. Phoenix, USA is UTC -7 so the LSTM is 105 °W.

Equation of Time (EoT)

The equation of time (EoT) (in minutes) is an empirical equation that corrects for the eccentricity of the Earth's orbit and the Earth's axial tilt. An approximation [36] accurate to within $\frac{1}{2}$ minute is:

$$EoT = 9.87 \sin 2B - 7.53 \cos B - 1.5 \sin B$$

where,

$$B = \frac{360}{365}(d - 81)$$

in degrees and d is the number of days since the start of the year [1,365]. The time correction EoT is plotted in the figure below.

Time Correction Factor (TC)

The net Time Correction Factor (in minutes) accounts for the variation of the Local Solar Time (LST) within a given time zone due to the longitude variations within the time zone and also incorporates the EoT above.

$$TC = 4(Longitude - LSTM) + EoT$$

The factor of 4 minutes comes from the fact that the Earth rotates 1° every 4 minutes.

Local Solar Time (LST)

The Local Solar Time (LST) can be found by using the previous two corrections to adjust the local time (LT).

$$LST = LT + \frac{TC}{60}$$

Hour Angle (HRA)

The Hour Angle converts the local solar time (LST) into the number of degrees which the sun moves across the sky. By definition, the Hour Angle is 0° at solar noon. Since the Earth rotates 15° per hour, each hour away from solar noon corresponds to an angular motion of the sun in the sky of 15°. In the morning the hour angle is negative, in the afternoon the hour angle is positive.

$$HRA = 15(LST - 12)$$

2.3.3 Declination Angle

The declination angle, denoted by δ, varies seasonally due to the tilt of the Earth on its axis of rotation and the rotation of the Earth around the

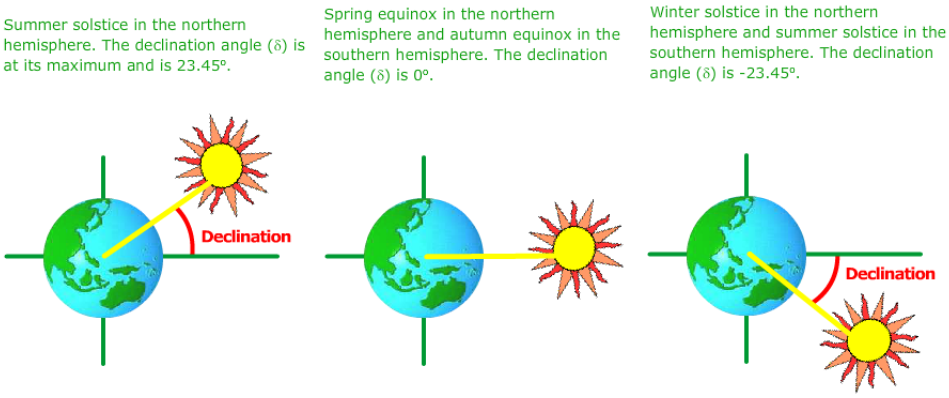

Figure 2.5: Despite the fact that the Earth revolves around the sun, it is simpler to think of the sun revolving around a stationary Earth. This requires a coordinate transformation. Under this alternative coordinate system, the sun moves around the Earth[38].

sun. If the Earth were not tilted on its axis of rotation, the declination would always be 0°. However, the Earth is tilted by 23.45° and the declination angle varies plus or minus this amount. Only at the spring and fall equinoxes is the declination angle equal to 0°. The declination of the sun is the angle between the equator and a line drawn from the center of the Earth to the center of the sun.

The declination angle can be calculated by the equation[39]:

$$\delta = -23.45^\circ \cos \frac{360}{365}(d + 10)$$

where d is the day of the year with Jan 1 as d = 1

The declination is zero at the equinoxes (March 22 and September 22), positive during the northern hemisphere summer, and negative during the northern hemisphere winter. The declination reaches a maximum of 23.45° on June 22 (summer solstice in the northern hemisphere) and a minimum of -23.45° on December 21-22 (winter solstice in the northern hemisphere). In the equation above, the +10 comes from the fact that the winter solstice occurs before the start of the year. The equation also assumes that the sun's orbit is a perfect circle and the factor of 360/365 converts the day number to a position in orbit.

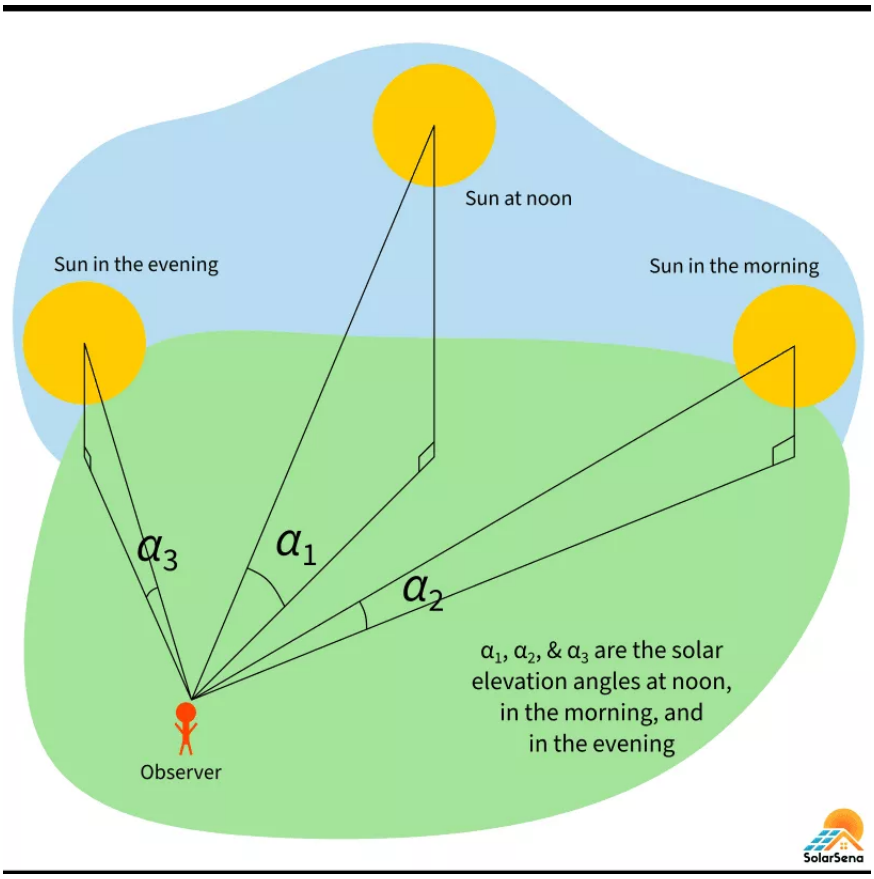

Figure 2.6: An important parameter in the design of photovoltaic systems is the maximum elevation angle, that is, the maximum height of the sun in the sky at a particular time of year. This maximum elevation angle occurs at solar noon and depends on the latitude and declination angle as shown in the figure above [40].

2.3.4 Elevation Angle

The elevation angle (used interchangeably with altitude angle) is the angular height of the sun in the sky measured from the horizontal. Confusingly, both altitude and elevation are also used to describe the height in meters above sea level. The elevation is 0° at sunrise and 90° when the sun is directly overhead (which occurs for example at the equator on the spring and fall equinoxes).

The elevation angle varies throughout the day. It also depends on the latitude of a particular location and the day of the year, as illustrated in figure 2.6.

At the Tropic of Cancer on the summer solstice, the sun is directly overhead

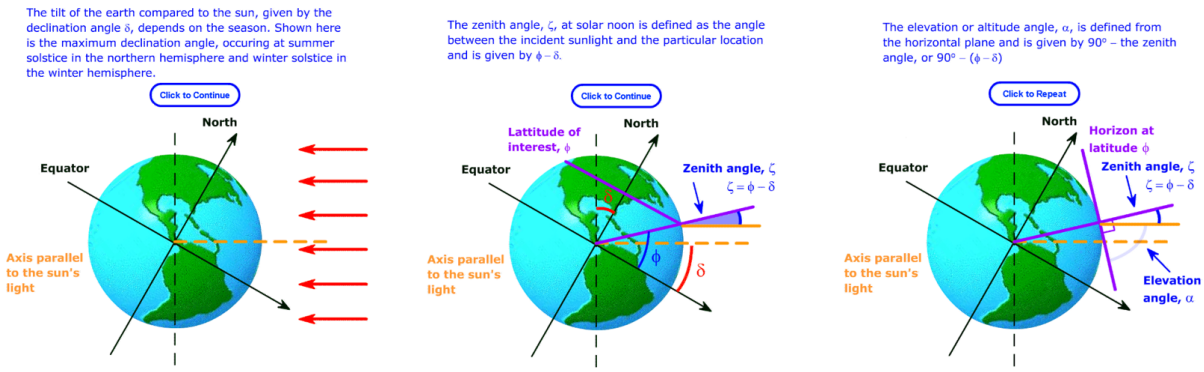

Figure 2.7: The maximum elevation angle at solar noon (α) is a function of latitude and the declination angle (δ). [40]

and the elevation angle is 90°. In summer at latitudes between the equator and the Tropic of Cancer, the elevation angle at solar noon is greater than 90°, implying that the sunlight is coming from the north rather than from the south as in most of the northern hemisphere. Similarly, at latitudes between the equator and the Tropic of Capricorn, during some periods of the year, sunlight is incident from the south, rather than from the north.

While the maximum elevation angle is used even in very simple solar application design, more accurate solar tracking system simulation requires the knowledge of how the elevation angle varies throughout the day.

Elevation angle α is calculated using the following equation

$$\alpha = \arcsin[\sin\delta \sin\varphi + \cos\delta \cos\varphi \cos HRA]$$

where: HRA is the hour angle

φ is the latitude of the location of interest (+ve for the northern hemisphere and -ve for the southern hemisphere).

δ is the declination angle, which depends on the day of the year.

Zenith angle

The zenith angle is the angle between the sun and the vertical. The zenith

angle is similar to the elevation angle but it is measured from the vertical rather than from the horizontal, thus making the zenith angle = 90° - elevation.

$\theta_z = 90° - \alpha$

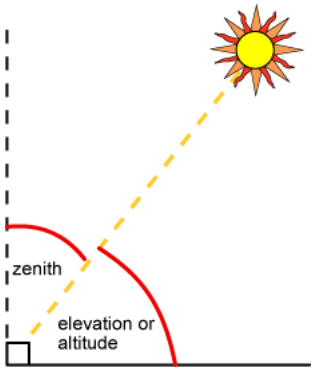

Figure 2.8: The solar zenith angle is the zenith angle of the sun, i.e., the angle between the sun's rays and the vertical direction.[40]

Sunrise and Sunset

To calculate the sunrise and sunset time the elevation is set to zero and the elevation equation above is rearranged to give:

$Sunrise = 12 - \frac{1}{15°} \arccos\left(-\tan\varphi \tan\delta\right) - \frac{TC}{60}$

$Sunrise = 12 + \frac{1}{15°} \arccos\left(-\tan\varphi \tan\delta\right) - \frac{TC}{60}$

where TC is the time correction.

2.3.5 Azimuth Angle

The azimuth angle is the compass direction from which the sunlight is coming. At solar noon, the sun is always directly south in the northern hemisphere, and the same for the southern hemisphere. The azimuth angle varies throughout the day as shown in the animation below. At the

equinoxes, the sun rises directly east and sets directly west, thus making the azimuth angles 90° at sunrise and 270° at sunset. In general, however, the azimuth angle varies with the latitude and time of year, and the full equations to calculate the sun's position throughout the day are given on the following page.

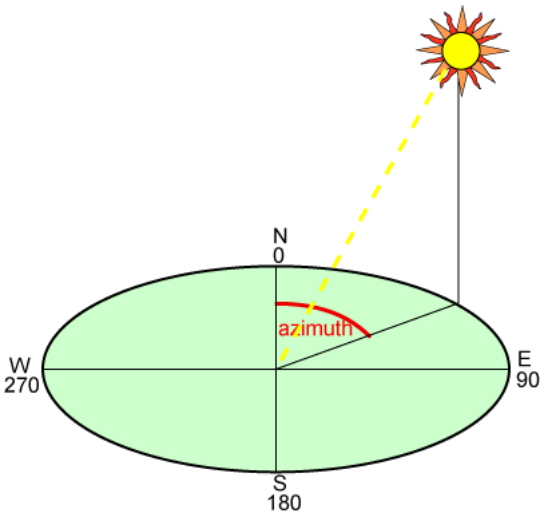

Figure 2.9: The azimuth angle is like a compass direction with North = 0° and South = 180°. Other authors use a variety of slightly different definitions (i.e., angles of ± 180° and South = 0°).[41]

The azimuth is calculated from the above parameters:

$$\gamma = \arccos\left[\frac{\sin\delta\cos\varphi - \cos\delta\sin\varphi\cos HRA}{\cos\alpha}\right]$$

where α is the elevation, φ is the latitude, and δ is the declination.

The above equation only gives the correct azimuth in the solar morning so that:

$\gamma = A_{zi}$, for LST < 12 or HRA < 0

$\gamma = 360° - A_{zi}$, for LST > 12 or HRA > 0

2.4 Solar Tracking

The Sun travels across the sky from the time it rises until the time it sets. The position of the Sun is a key factor in the performance of solar energy

systems as it can immensely reduce the collector's energy gain by different ratios, depending on the type of application.

Solar tracking is a mechanism that ensures that the solar collector is precisely normal to the direction of the sun's direct beams by calculating the following parameters, solar irradiance, solar azimuth angle, elevation angle, inclination angle, declination angle, and zenith angle are the important parameters that identify the best position of solar tracking system [28], since it's a dual-axis control system. More on how solar tracking works and the algorithm works is mentioned in 2.3.1

The below section is a literature review on the various technologies used for solar tracking and the different movement degrees of freedom used.

Technologies of solar tracking

Solar tracking technologies can be classified to three main types that control the movement according to this study [42]. Namely, active tracking, passive tracking and chronological tracking [43].

Active solar tracking includes the use sensors during the day to determine the position of the sun in the sky during the day [12]. Light sensors will trigger the actuator or motor to move the drive system towards the sun. If the sunlight is not perpendicular to the tracker, then there will be a difference in the illumination on one light sensor compared to the other. This difference can be used to determine the direction in which the tracker has to be directed in order to be perpendicular to the sun. Active trackers are the most commonly used technology [42], it has up to a percentage of 76.42% usage the solar tracker drive systems types in the recent studies [44]. Moreover, it proved to have high accuracy and efficiency in tracking the position of the sun, on the down side however, it requires extra power consumption and is not very accurate on cloud days.

Passive solar tracking are different from the latter as it doesn't need mechanical drives to orient itself towards the sun, what it does is that it

depends on the thermal expansion in materials or an imbalance in pressure between two points at both ends of the tracker, which usually consists of a fluid (expansible gas or liquid). The passive solar tracking system relies on a low boiling point compressed gas fluid, which cause the structure of the tracker to move to an imbalance [44]. Passive solar tracker's merits are that it does not require any motors or actuators, easy to install and has low maintenance cost. However, its demerits are that it has strong dependence on weather conditions and its inferior accuracy-wise, this technique has a usage of 6.6% in recent studies [44].

Chronological Solar tracking is time-based, all external conditions are considered irrelevant to the functionality of this system, as this system is set to follow an algorithm that moves this structure at a fixed rate and angle throughout the day, all day, for weeks, months or even years without any human or other weather conditions. According to the user this system can control the motor or actuator to move in the desired time-steps. Taking this into account, the results of the chronological tracker are that it has no energy losses and negligible tracking errors. Despite that, continuous rotation requires more energy due to unnecessary work in cloudy days [45], this technique has a usage of 7.55% in recent studies [44].

For the sake of knowledge and a complete comprehensive research on this topic, there are two other less frequently used yet valuable, are the manual and semi-passive tracking. Semi-passive tracking system is a technique where the solar tracking concentrator can track the sun and keep the sun's rays perpendicular to the absorber's cross-sectional area with a minimal mechanical effort and reduced movement for sun tracking, this technique has a usage of 3.77% in recent studies [44]. Finally, Manual solar tracker is a method where the system can track the sun angle from season to season with manual tilt angle changing per seasons using a manual gear for ease of the system construction and maintenance. The significance of this method is that its cheap compared to the alternatives, this technique has a usage of 2.83% in recent studies [44].

Needless to say that all of these techniques have their own merits and demerits, especially depending on it's scale and location. Some of

these technologies can be integrated to work together, like Active and Chronological.

Movement degrees of solar tracking

Solar trackers existed as a target for a better overall output energy generated, as an upgrade for just still solar panels or collectors or whatever solar application.

Mainly, there are two types of solar tracking system invented based on movement degree of freedoms. One is a single axis solar tracker and another is Dual axis solar tracker [46]. There is a plethora of researches and papers published that prove that the use of solar trackers, despite their initial cost, are better efficiently in terms of power output and cost [47][48][49].

A fixed axis solar collector is one that doesn't have any moving parts or a solar tracker, in this study we take this as our reference point when comparing it to our two solar tracking methods. Fixed tilt solar arrays, are solar panels or collectors which are installed in an optimal tilt angle that is facing the sun all day, this basic construction, immobile as it is, is simple, easy to design, and has low maintenance. However it will produce less energy.

Single axis trackers have one degree of freedom that acts as an axis of rotation. According to the direction of the rotation axis, single-axis tracking is further classified as follows, the single axis tracking tracks daily the sun from east to west and can be divided into horizontal single axis tracker and vertical single axis tracker in order to perform solar tracking centered on the horizontal and vertical axis of the PV panel or whatever desired solar application [50].

Dual axis trackers have two degrees of freedom that act as axes of rotation. These axes are typically normal to one another. The axis that is fixed with respect to the ground can be considered a primary axis. The axis that is referenced to the primary axis can be considered a secondary axis. There are several common implementations of dual axis trackers. They

are classified by the orientation of their primary axes with respect to the ground. Two common implementations are tip-tilt dual axis trackers and azimuth-altitude dual axis trackers. The orientation of the module with respect to the tracker axis is important when modeling performance. Dual axis trackers typically have modules oriented parallel to the secondary axis of rotation. No matter where the Sun is in the sky, dual axis trackers are able to angle themselves to be in direct contact with the Sun.

By comparing three different systems in different types of criteria, we have found that dual-axis sun tracking solar panel system is more efficient in terms of output power and generating total energy [51].

By considering the movement degree of freedom in solar tracking system, the single-axis solar tracker is the simplicity when compared to the dual axis rotation. In addition, the single-axis solar tracker used lower cost and energy consumption since this system only use one motor for tracking the sun horizontally or vertically. However, this system can only track the sun during the daily movement and not the yearly movement. In addition, the efficiency of the tracking system is reduced by a large amount due to the rotation around only single-axis caused by the cloudy days. Although single-axis solar tracker is less complexity construction, but their efficiency is lower than that of dual-axis solar tracker. The main advantage of using dual-axis tracking in comparison with single one is tracking the sun movement not only during the day, but also for the yearly movement. Although dual-axis solar tracker is more expensive construction, they deliver efficient performance when compared to the single axis solar tracker.

Chapter 3

Methodology

As mentioned before, the research aim of this thesis is to implement a global dual-axis solar tracking system based on open-loop astronomical equations, thus the name chronological technology. As to maximize the amount of sunlight collected at the focal point while being low maintenance and having an efficient output.

Taking it a step further, an attempt to integrate active and chronological technologies will be implemented for higher precision.

3.1 Software design

The two main angles we are trying to calculate are the Elevation and Azimuth angles, both angles are necessary for the control of the position of the concentrating collector's focal point.

To be able to calculate these angles an 2.3.1 had to be followed and translated into 3.1.2.

3.1.1 Microchip studio IDE

The coding environment I chose is Microchip Studio (MCS), previously known as Atmel studio, which is an Integrated Development Environment (IDE) for developing and debugging AVR and SAM microcontroller applications. It is a well-supported development tool that gives a seamless and easy-to-use environment for writing, building, and debugging applications written in C/C++ or assembly code, which in my case Embedded C is used.

The reason I chose MCS over its popular counterpart Arduino, is basically because it is an upgrade in terms of almost everything, sometimes an Arduino board is not enough. They often hide much of the functionality of the MicroController Unit (MCU) and the Arduino IDE does not provide ready access to the underlying modules. This isn't very convenient, as some projects can't have the extra overhead given by a standard function, such as digitalWrite. This is where Atmel Studio comes into play. It is an IDE used by professionals that lets you write programs in C, C++, and even assembly for nearly all of Atmel's MCUs.

Not to mention, Arduino's libraries tend to limit the rotation to strictly 180 degrees in stepper motors, on the other hand, MCUs allow for all 360 degrees of rotation.

3.1.2 Sun position calculator code

The calculations of the 2.3.1 were initially constructed in C using Visual Studio IDE, then implemented on MCS which uses the same programming language but embedded as this is the code that will be boot loaded to the microcontroller.

The code has the time and location predefined to the German International University's geographical locations, these variables could also be changed to any location all over the world and any time of the year for the next 27 years [33][52-55].

The code will be referred to as "project" and not code since the project has multiple files of two types, C and header files where both are codes, only difference is that the latter exists to provide interfaces that allow a file to access functions, global variables, and macros from other files.

This project follows the AutoSAR Layered Architecture, which is one of the most common architectural styles. The idea behind Layered Architecture is that modules or components with similar functionalities are organized into horizontal layers. As a result, each layer performs a specific role within the application. [52]

This project is split into 4 system layers:

- Microcontroller Abstraction Layer
- Electronic Unit Abstraction Layer
- Application
- Utilities

The MicroController Abstraction Layer (MCAL) is a software module that enables direct access to on-chip MCU peripheral modules and makes the upper software layer independent of the MCU. In my case, it includes the stepper motor and external timer modules.

The Electronic Unit Abstraction Layer (ECUAL) is the module responsible for the declarations and initializations of all used pins in the program, by writing, reading or toggling any of the corresponding registers on the MCU. In this project, it's the Digital/Input Output driver.

The code itself is made to take a step every 10 minutes which is modifiable. However, for peak efficiency, 10-15 minutes is advised to minimize energy losses[44][53].

Since this is an open-loop system and no feedback is given since the lack of sensors, thus the code had to be a little bit more extensive, the current flowchart shows how the elevation and azimuth angles are reached.

Since the hour angle changes around noon as explained in **??**, the values of the azimuth and elevation also differ, all of this is illustrated in the flowchart and the code is available to read in the appendix.

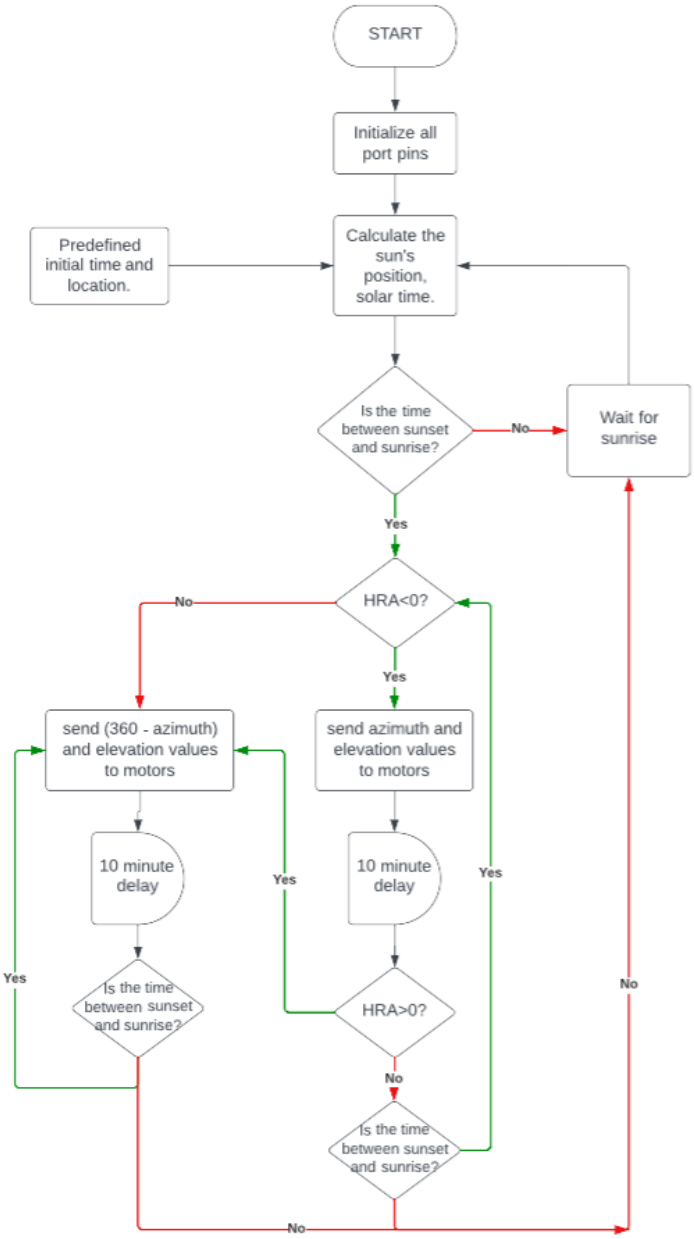

Figure 3.1: Flowchart for the system

3.2 Hardware design

The hardware setup was all designed and simulated on SolidWorks before the actual implementation phase started. The design is to scale, which made the implementation phase well-ordered, fast and modifiable.

Here's an electronic schematic of the system showing the connections between the components of the system.

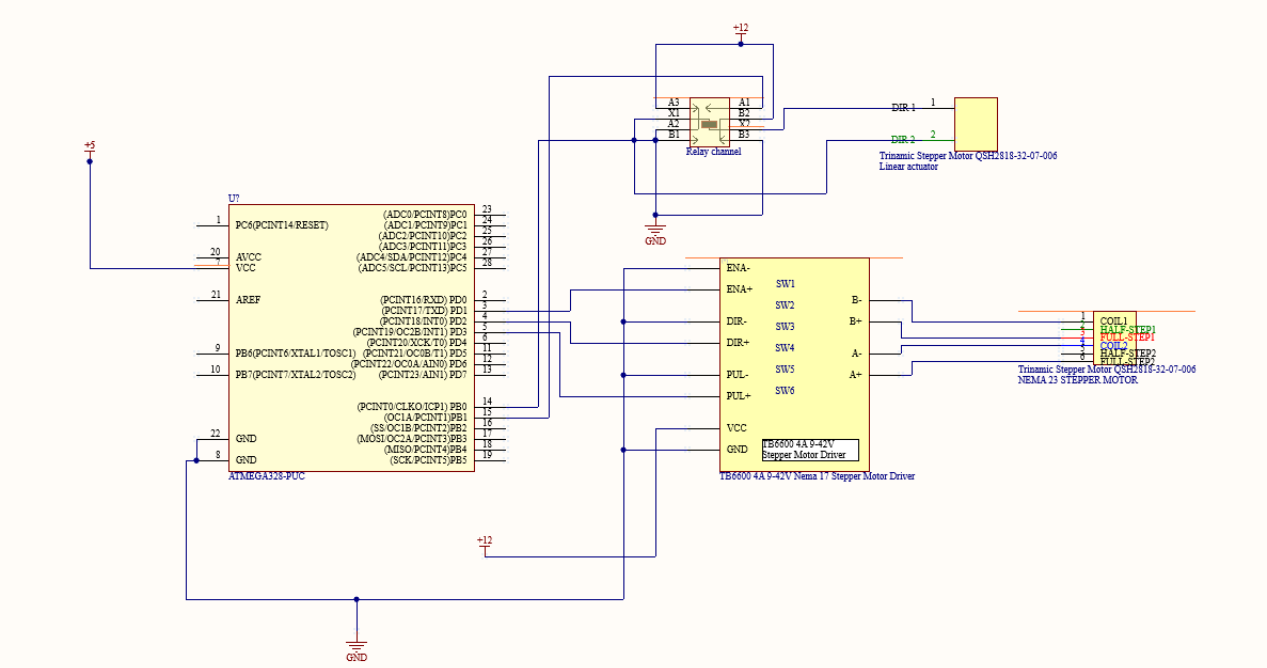

Figure 3.2: Electronic schematic diagram of the system

3.2.1 Solar Dish

The 70cm prifix satellite dish was chosen as the reflecting parabola, and it was wrapped with a self-adhesive mirror sticker roll.

The focal length of the satellite dish calculated using the following equation:

$f = (D * D)/(16 * c)$

where, f is focal length. D is diameter. c is depth.

Since the diameter is 70cm and the depth is 730mm, the focal length is equal to 420mm.

3.2.2 supporting beam

The beam that holds the solar dish, it has a length of 60cm and diameter of 50mm and a thickness of 5mm, and Material Galvanized steel.

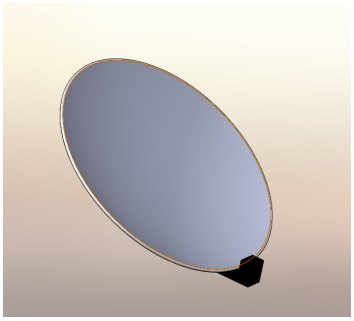

Figure 3.3: PDC in solidworks

Figure 3.4: Actual satellite dish used as reflector[54]

Figure 3.5: Supporting column on Solidworks

At the top of the beam two joint parts are at the top to hold the solar dish.

3.2.3 NEMA 23 stepper motor

Principle of operation

What is the NEMA 23 stepper motor? The NEMA 23 is mostly a hybrid bipolar stepper motor with high torque. Additionally, the motor is a 2-

phase stepping motor used to generate power. Notably, it stands out because its step angle is 1.8 degrees. With this angle, there are 200 steps for every single turn. Therefore, each revolution of this dual-shaft motor covers 1.8 degrees. [55]

Figure 3.6: Image of a disassembled stepper motor set [55]

Also, the motor's design focuses on delivering maximum torque at operating speeds. As a result, the design reduces vibration and noise. Thus it is preferred in precision motion controllers.

Pin configuration Notably, the NEMA23 motor wirings have four wires that are different colors. These are; (Black, Green, Red, and Blue). However, these wires are uncovered.

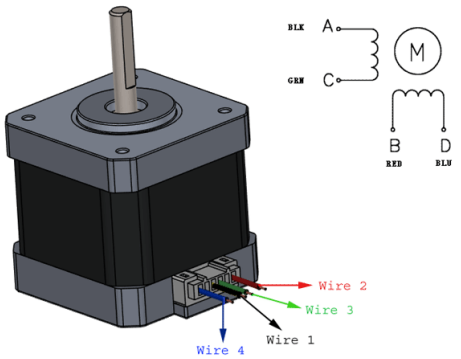

Figure 3.7: Diagrammatic representation of the NEMA 23 pin configuration[55]

Simply, one coil is wired with black and green wires. However, the remaining coil attaches to the red and blue wires. Additionally, using two H-bridges can control this motor. But, keep in mind that you should always use a motor drive.

Specifications of the NEMA23 Stepper Motor [56]

- Model: 23HS7628
- Shaft Diameter: 8mm
- Step Angle: 1.8 degrees
- Motor Length: 76mm
- Rated Current: 2.8A
- Phase Resistance: 1.13 ohms
- Phase Inductance: 3.6mH
- Holding Torque: 18.9Kg.cm

3.2.4 TB6600 stepper motor driver

Stepper motor drivers are small pin-compatible modules designed particularly to drive stepper motors capable of continuously rotating with precise position control without the need for a feedback system. The stepper motor drivers have built-in converters that offer multi-stage resolution and variable current control that allows control of stepper motors with simple step and direction inputs.[57]

The TB6600 stepper motor driver IC is a very easy, effective, and professional device that can drive 2-phase stepper motors. Compatible with any type of microcontroller and Arduino to generate 5V digital output pulse signals. The TB6600 stepper motor driver has a wide input power range, (9-42 VDC) of power supply, and generates 4 Amps peak current,

supporting direction control and speed that is sufficient for most stepper motors.[57]

Figure 3.8: TB6600 Stepper Motor Driver Module

TB6600 Specifications

- Operating voltage: 9 – 42 V
- Max output current: 4.5 A per phase, 5.0 A peak1
- Microstep resolution: full, 1/2, 1/4, 1/8, 1/16, and 1/32
- Protection: Low-voltage shutdown, overheating, and over-current protection
- Dimensions: 96 x 72 x 28/36 mm
- Hole spacing: 88, diameter 5 mm

A stepper motor driver is necessary for the operation of a stepper motor since it has high current demand and the control of the steps since microstepping is unachievable without drivers.

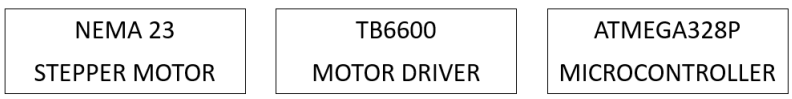

Figure 3.9: Block diagram of the driver, motor, and controller

3.2.5 Linear actuator

The linear actuator is responsible for the elevation of the solar dish, it s the one used in the old satellite dishes, 36V power using a power supply and is controlled using a relay and MCU, it is responsible for the elevation angle of the mechanism.

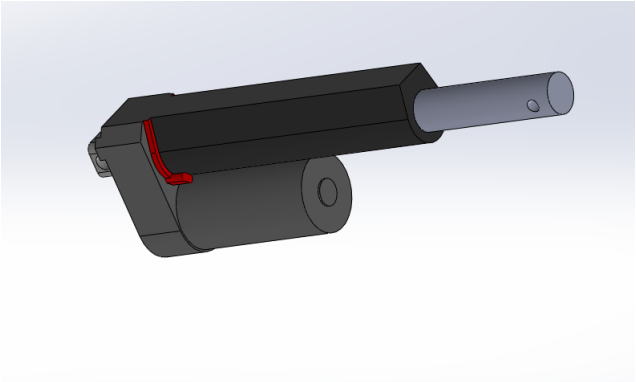

Figure 3.10: Linear actuator in solidworks

Using the **position coordinate equation** to find the elevation angle: The angular motion of the solar dish can be obtained using the coordinate α, whereas the extension along the linear actuator is defined using coordinate s, which measures its length from the point the linear actuator is fixed to the endpoint attached to the dish, where L1 and L2 are constant variables equal to 216mm and 171mm respectively. These points can be related to each other using the law of cosines, namely,

$$S^2 = L1^2 + L2^2 - 2(L1)(L2)(cos(\alpha))$$

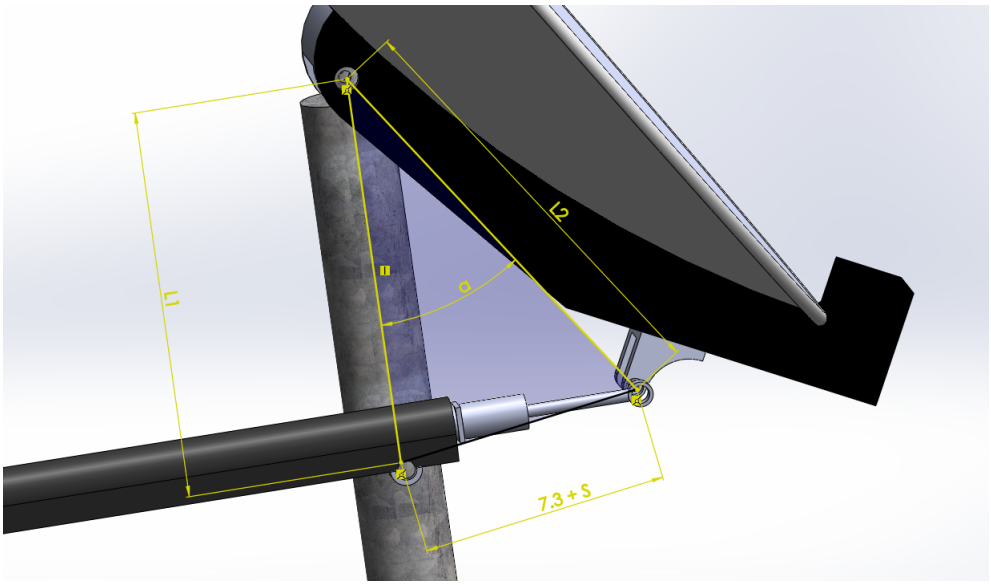

Figure 3.11: Figure showing the position coordinates to determine the angle of elevation

3.2.6 Gears

Since Solar trackers rely heavily on precision, the stepper motor's step angle of 1.8 degrees will not suffice, thus we have to use gears and manipulate the gears so that we have the smallest step angle possible for optimum accuracy while also maintaining enough torque to move to rotate the body. A total gear ratio of 1:32 is the overall gear ratio needed since using only two gears would result in one being too large. Hence, we will use a compound gear train of 2 stages, being 13:52, 13:52 meaning (13x13:52x52) which equals 1:16, and then adding another gear to reach 1:18. The stepper motor is also capable of microstepping, which goes as far as 1/32, but as not to lose much torque only half stepping will be used (1/2), therefore reaching a total ratio of 1:32, meaning every step will result in 0.05 degrees rotation, convenient enough considering the available hardware in the market.

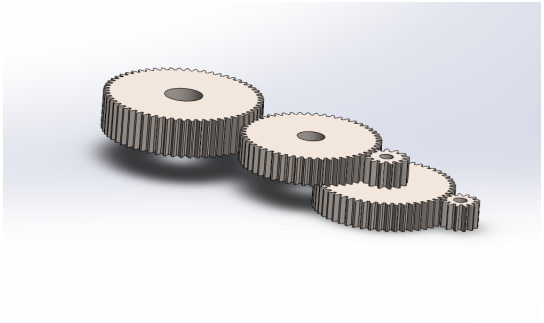

Figure 3.12: Gear train in SolidWorks

Of course, the gears had to be supported by a shaft since they can't be hanging in the air between the motor's shaft and the solar tracker's column. Thus, shafts steel shafts were machined on a center lathe machine to support the gears to prevent any axial force to be exerted on the stepper motor. The shafts were mounted on thrust bearings to allow rotation, specifically thrust bearings since they are designed specifically to support axial loads [58]

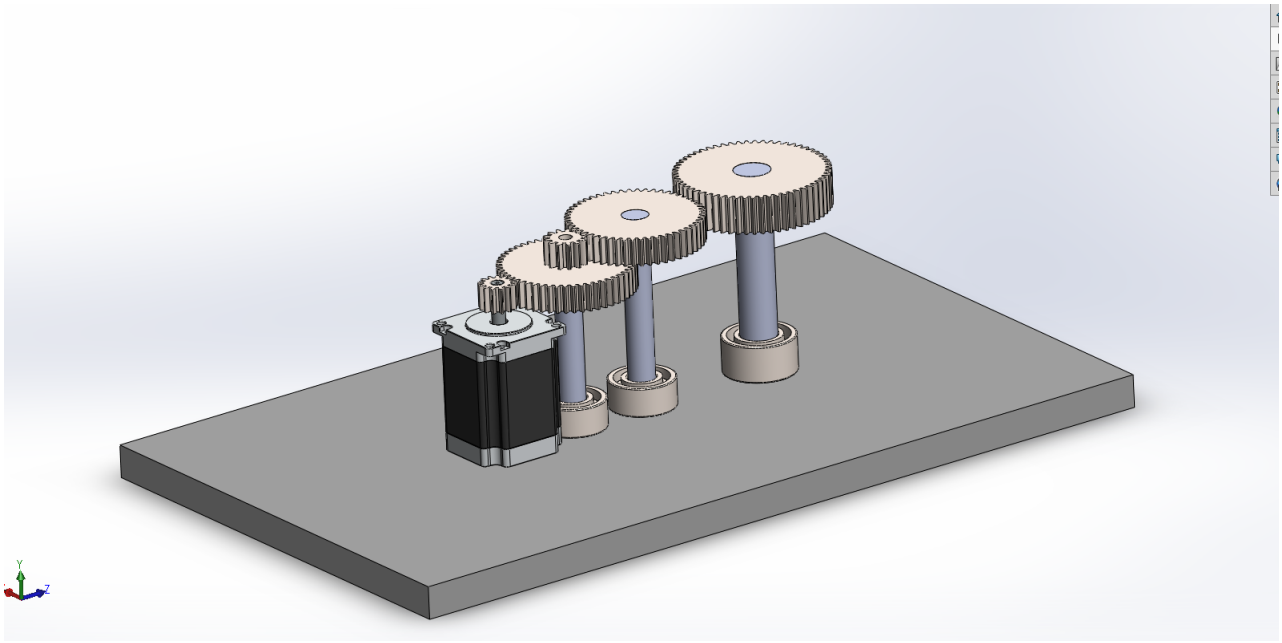

Figure 3.13: Gear train assembly with supporting shafts and thrust bearing illustrated in solidworks

The whole assembly is as simulated on SolidWorks as in the figure below.

Figure 3.14: The whole assembly excluding the linear actuator

Chapter 4

Results

4.1 Setup and implementation

After the design phase the hardware implementation started by testing the code of the algorithm on the atmega328p MCU, using an Arduino UNO as an In-Circuit Programmer (ISP) to upload and burn the code from the generated hex file from MCS to the MCU.

After that two 13-tooth gears and two 52-tooth gears and one 59-tooth gear were bought and gas cleaned to remove rust and clean residue in between the teeth and for a better overall look.

The whole project will be mounted on a 2cm thick base with a 5mm metal sheet layer to be able to fix the motor to the base and weld the bearings and shafts to the base. Wheels can also be added to the bottom of the base for easier project mobility.

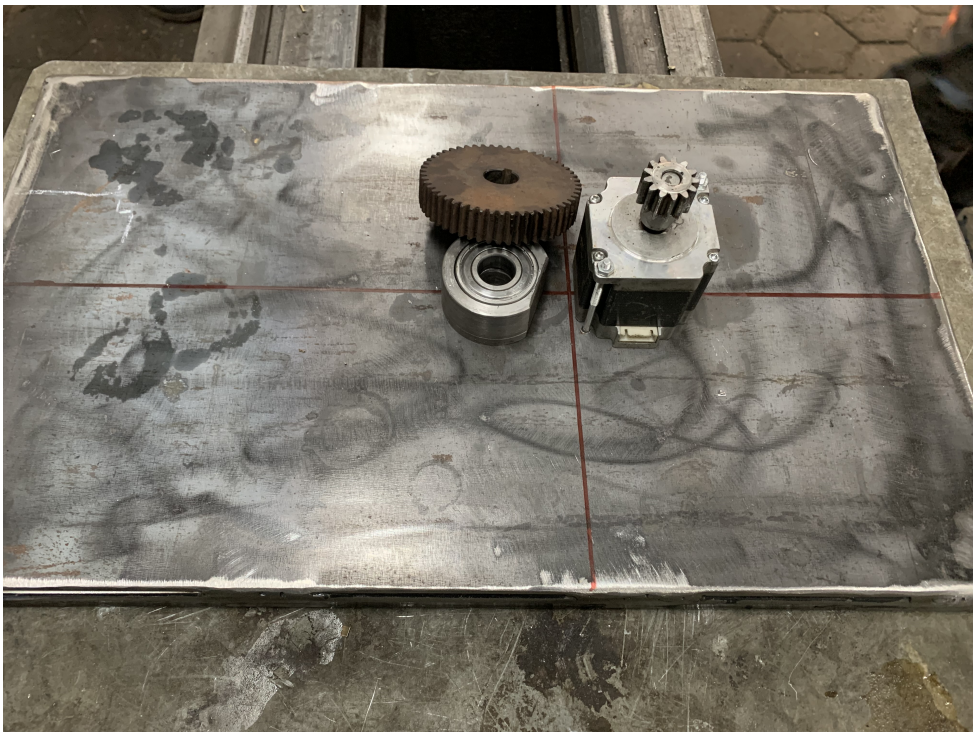

Figure 4.1: The metal base for the project with the motor mounted

The first 13-tooth gear was mounted and fixed to the motor shaft, this is the driver gear for the assembly. Then it was meshed with the first 52-tooth gear, the shaft for this gear, and the rest of the gears were machined and cut on a center lathe machine.

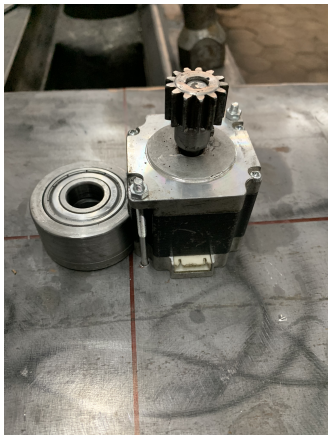

Figure 4.2: The motor with the 13-tooth gear

After, calculating the central distance using the following equation[59]: $a = (d1 + d2)/2$

where: a = distance between the centers of the gears d1 = pitch diameter of gear 1 d2 = pitch diameter of gear 2

Figure 4.3: Thrust bearings welded to the base

The final gear train after welding the thrust bearings holding the shafts and gears is in the figure below, thus reaching the desired 1/18 gear ratio and a step angle of 0.1 degrees, and using the TB6600 driver and halfstepping we can reach 0.05 degrees accuracy.

Figure 4.4: Full gear train

Now the main column is to be welded on top of the final driven gear, however, it will be welded on the final gear while rotating and locked in

the lathe machine, to ensure that it will be straight and uniform with no slag, cracking, holes, and the welding meets the desired strength.

Figure 4.5: The column being welded on the 59 teeth gear

Then, the linear actuator was welded to the main column to provide for the elevation angle of the dish.

Figure 4.6: Linear motor welded to the main rotating column

A small metal sheet was cut and welded on top of the rotating column to provide as a base for the dish.

Figure 4.7: Base of the dish welded to the top of the column

The end of the linear actuator was fixed to the end of the dish for the maximum possible elevation range, which ranges from 17 degrees to 81 degrees, which makes up for more than 92 percent of the elevation in Egypt. [60]

Figure 4.8: Revolute joint between end-effector of linear motor link and the dish link

Finally, the solar dish was coated with a mirror-like reflective material to test later the solar tracker for whether the reflectiveness is successful or not.

Figure 4.9: reflective tape over the area of the dish

This is the final assembly with the gears, linear actuator, and stepper motor all assembled and fully functional. Since this is an open-loop system, the solar dish must be facing north before operating, this can be known from the compass welded on the base. Also, if this is the first operating it, ensure that the linear motor is fully retracted (included in the code), otherwise, the linear motor automatically retracts depending on the time of the day according to the algorithm and code.

Figure 4.10: Picture of the whole assembly

There are two power sources for this project, 5V, and 12V power supplies, to be connected and activated simultaneously.

Figure 4.11: Focal point visible on a piece of carton

4.2 Conclusion

Implementation of the solar tracker was successfully achieved, with the azimuth rotation of 0.05 degrees per step which considering the scale and budget is impressive. Also, the elevation angle was in the range between 17 and 81 degrees, which is 92 percent of the day's elevation throughout the year [61], which is not to be underestimated considering a linear actuator for satellite dishes was used.

Thus in conclusion the solar tracking system worked as intended, the DNI was efficiently reflected off the PDC throughout the day over several days with a step of 10 minutes from sunrise till sunset, then it rotates in the opposite direction to its initial position (pointing north) waiting for the next sunrise to start operating again.

4.3 Discussion

Implementing a PDC in the current market has been a hassle considering the limited options for the available motors, hence one must be creative with working with what's in hand, which is the job of every engineer, problem solving.

This thesis proves that it is possible to implement a relatively large-scale PDC with high precision in dual axis with affordable finances any student can handle.

4.4 Future work

There are various factors that could be reconsidered if this is ever to be recreated.

Firstly, a larger dish is recommended, since the larger is the surface area of the dish, the more DNI is reflected of the dish. Also, the weight of the dishes is not that much compared to the rest of the assembly.

Secondly, integrate light dependent resistors to the system to achieve active solar tracking along side chronological solar tracking, which much increases the accuracy of the system.

Last but not least, using a longer column supporting the dish, as the higher the dish is in altitude, the better is the collection of the sun and also avoids the shadows of any possible obstructing surrounding.

Appendix

main.c file

```

#include <avr/io.h>
#include "ECUAL/Stepper/stepper.h"
#define F_CPU 16000000UL

int main(void)
{

  APP_init();

  //Main loop
  while(1){

    APP_start();
  }

}
```

In the Utilities file:

types.h file

```
#ifndef TYPES_H_
#define TYPES_H_


typedef unsigned char uint8_t;

typedef unsigned short uint16;

typedef unsigned int uint32;


#endif
```

registers.h file

```
#ifndef REGISTERS_H_
#define REGISTERS_H_
#include "types.h"
#include "bit_manipulation.h"
#include "interrupts.h"
#include <math.h>
#define PI 3.14159265359

//DIO Registers


//volatile as it may change from external signals

//Timer0 Registers
```

```
15 #define TCCR0B (*(volatile uint8_t*)(0x45))
16 #define TCCR1A (*(volatile uint8_t*)(0x80))
17 #define TCCR1B (*(volatile uint8_t*)(0x81))
18 #define TCCR1C (*(volatile uint8_t*)(0x82))
19 #define TCNT1H (*(volatile uint8_t*)(0x85))
20 #define TCNT1L (*(volatile uint8_t*)(0x84))
21 #define TIFR1 (*(volatile uint8_t*)(0x36))
22 #define TIMSK1 (*(volatile uint8_t*)(0x6F))
23 #define TCNT1 (*(volatile uint8_t*)(0x1FF))
24
25
26
27 // PORTA registers
28 #define PORTA *((volatile uint8_t*)0x3B)
29 #define DDRA *((volatile uint8_t*)0x3A)
30 #define PINA *((volatile uint8_t*)0x39)
31
32
33 // PORTB registers
34 #define PORTB *((volatile uint8_t*)0x25)
35 #define DDRB *((volatile uint8_t*)0x24)
36 #define PINB *((volatile uint8_t*)0x23)
37
38 // PORTC registers
39 #define PORTC *((volatile uint8_t*)0x28)
40 #define DDRC *((volatile uint8_t*)0x27)
41 #define PINC *((volatile uint8_t*)0x26)
42
43 // PORTD registers
44 #define PORTD *((volatile uint8_t*)0x2B)
45 #define DDRD *((volatile uint8_t*)0x2A)
46 #define PIND *((volatile uint8_t*)0x29)
47
48 //PINS
49 #define PIN0 0
50 #define PIN1 1
51 #define PIN2 2
52 #define PIN3 3
53 #define PIN4 4
54 #define PIN5 5
55 #define PIN6 6
56 #define PIN7 7
57
58 //External Interrupts registers
59 #define MCUCR (*(volatile uint8_t*)(0x55))
60 //#define MCUCSR (*(volatile uint8_t*)(0x54))
61 #define SREG (*(volatile uint8_t*)(0x5F))
62 //#define GICR (*(volatile uint8_t*)(0x5B))
63 //#define GIFR (*(volatile uint8_t*)(0x5A))
64
65
66
67 #endif /* REGISTERS_H_ */
```

interrupts.h file

```
#ifndef INTERRUPTS_H_
#define INTERRUPTS_H_
#include "registers.h"

//External interrupt request 1
#define EXT_INT_0 __vector_1

//External interrupt request 2
#define EXT_INT_1 __vector_2

//Enable interrupts

//Set Global Interrupts, Set the I-bit in status register to 1
#define sei()  __asm__ __volatile__ ("sei" ::: "memory")

//Clear Global Interrupts, Set the I-bit in status register to 0
#define cli()  __asm__ __volatile__ ("cli" ::: "memory")

//Rising edge ISC01 ISC00 are set to 1
#define RISING_EDGE_SETUP() MCUCR|= (1<<1) | (1<<0)

//Enable external interrupt INT0
#define SETUP_INT0() GICR|=(1<<6)

//ISR definition
#define ISR(INT_VECT)void INT_VECT(void) __attribute__ ((signal,used)); \
void INT_VECT(void)

#endif /* INTERRUPTS_H_ */
```

bitManipulation.h file

```
#ifndef BIT_MANIPULATION_H_
#define BIT_MANIPULATION_H_




#define CLEAR_BIT(REG,NUM) REG&=~(1<<NUM)
#define SET_BIT(REG,NUM) REG|=(1<<NUM)
#define TOGGLE_BIT(REG,NUM) REG^=(1<<NUM)
#define READ_BIT(REG,NUM) ((REG&(1<<NUM))>>NUM)

#endif /* BIT_MANIPULATION_H_ */
```

In the MCAL file:

dio.h file

```
#ifndef DIO_H_
#define DIO_H_
#include "../../Utilities/registers.h"
```

```
#include "../../Utilities/bit_manipulation.h"

#define PORT_A 'A'
#define PORT_B 'B'
#define PORT_C 'C'
#define PORT_D 'D'

//Direction defines
#define IN 0
#define OUT 1

//Values defines
#define HIGH 1
#define LOW 0


void DIO_init(uint8_t portNumber, uint8_t pinNumber, uint8_t direction);//
    Initialize dio direction
void DIO_write(uint8_t portNumber, uint8_t pinNumber, uint8_t value); //
    write data to dio
void DIO_toggle(uint8_t portNumber, uint8_t pinNumber); //toggle dio
void DIO_read(uint8_t portNumber, uint8_t pinNumber, uint8_t* value); //
    read dio



#endif /* DIO_H_ */
```

dio.c file

```
#include "dio.h"

void DIO_init(uint8_t portNumber, uint8_t pinNumber, uint8_t direction){
  switch(portNumber){


    case PORT_A:
    if(direction == IN){
      DDRA &= ~(1<<pinNumber); //input
    }
    else if(direction == OUT)
    {
      DDRA |= (1<<pinNumber); //output
    }
    else{
      //error
    }
    break;



    case PORT_B:
    if(direction == IN){
```

```
24        DDRB &= ~(1<<pinNumber); //input
25      }
26      else if(direction == OUT)
27      {
28        DDRB |= (1<<pinNumber); //output
29      }
30      else{
31        //error
32      }
33      break;
34
35
36
37      case PORT_C:
38      if(direction == IN){
39        DDRC &= ~(1<<pinNumber); //input
40      }
41      else if(direction == OUT)
42      {
43        DDRC |= (1<<pinNumber); //output
44      }
45      else{
46        //error
47      }
48      break;
49
50
51
52      case PORT_D:
53      if(direction == IN){
54        DDRD &= ~(1<<pinNumber); //input
55      }
56      else if(direction == OUT)
57      {
58        //
59        DDRD |= (1<<pinNumber); //output
60      }
61      else{
62        //error
63      }
64      break;
65
66    }
67  }
68  void DIO_write(uint8_t portNumber, uint8_t pinNumber, uint8_t value){
69    switch(portNumber){
70      case PORT_A :
71      if(value == LOW){
72        PORTA &= ~(1<<pinNumber); //input
73      }
74      else if(value == HIGH)
75      {
76        PORTA |= (1<<pinNumber); //output
```

```
77      }
78      else{
79        //error
80      }
81      break;
82      case PORT_B :
83      if(value == LOW){
84        PORTB &= ~(1<<pinNumber); //input
85      }
86      else if(value == HIGH)
87      {
88        PORTB |= (1<<pinNumber); //output
89      }
90      else{
91        //error
92      }
93      break;
94      case PORT_C :
95      if(value == LOW){
96        PORTC &= ~(1<<pinNumber); //input
97      }
98      else if(value == HIGH)
99      {
100       PORTC |= (1<<pinNumber); //output
101     }
102     else{
103       //error
104     }
105     break;
106
107     case PORT_D :
108     if(value == LOW){
109       PORTD &= ~(1<<pinNumber); //input
110     }
111     else if(value == HIGH)
112     {
113       PORTD |= (1<<pinNumber); //output
114     }
115     else{
116       //error
117     }
118     break;
119
120   }
121 }
122 void DIO_toggle(uint8_t portNumber, uint8_t pinNumber){ // toggle dio
123   switch(portNumber){
124     case PORT_A:
125     PORTA ^= (1<<7); //This line serves no function but it was added as the
        code for some reason skipped the first line, so I added anything here
      as it is already skipped
126     PORTA ^= (1<<pinNumber);
127     break;
```

```
128
129     case PORT_B:
130     PORTB ^= (1<<pinNumber);
131     break;
132
133     case PORT_C:
134     PORTC ^= (1<<pinNumber);
135     break;
136
137     case PORT_D:
138     PORTD ^= (1<<pinNumber);
139     break;
140   }
141 }
142 void DIO_read(uint8_t portNumber, uint8_t pinNumber, uint8_t* value); //
    read dio (this function was not used)
```

In the ECUAL file:

linear.h file

```
1  #include <inttypes.h>
2  #include <stdint.h>
3  #include <stdlib.h>
4  #include <string.h>
5  #include <stdio.h>
6  #include <inttypes.h>
7  #include <math.h>
8  #include <time.h>
9
10 #define LINEAR_PORT PORT_C
11 #define FORWARD_PIN PIN5
12 #define BACKWARD_PIN PIN4
13
14 //Motor time stroke = 2 mins 9s
15 //length 18cm
16
17 #define LOW   0
18 #define HIGH  1
19
20 void LINEAR_init(uint8_t linearPort, uint8_t linearPin);
21
22 void LINEAR_on(uint8_t linearPort,uint8_t linearPin);
23
24 void LINEAR_off(uint8_t linearPort,uint8_t linearPin);
25
26 void step_forward(float time);
27
28 void step_backward(float time);
29
30 void step_lin(float s);
31
32 #endif
```

linear.c file

```
#include "linear.h"

void LINEAR_init(uint8_t linearPort,uint8_t linearPin){
  DIO_init(linearPort,linearPin,OUT);
}

void LINEAR_on(uint8_t linearPort,uint8_t linearPin){
  DIO_write(linearPort,linearPin,HIGH);
}

void LINEAR_off(uint8_t linearPort,uint8_t linearPin){
  DIO_write(linearPort,linearPin,LOW);
}

void step_forward(float time){


  //Forward linear
  LINEAR_off(LINEAR_PORT,FORWARD_PIN);
  LINEAR_on(LINEAR_PORT,BACKWARD_PIN);
  TIMER_delay_lin(time);

   //Stop linear
  LINEAR_on(LINEAR_PORT,FORWARD_PIN);
  LINEAR_on(LINEAR_PORT,BACKWARD_PIN);
}


void step_backward(float time){


  //backward linear
  LINEAR_on(LINEAR_PORT,FORWARD_PIN);
  LINEAR_off(LINEAR_PORT,BACKWARD_PIN);
  TIMER_delay_lin(time);

  //Stop linear
  LINEAR_on(LINEAR_PORT,FORWARD_PIN);
  LINEAR_on(LINEAR_PORT,BACKWARD_PIN);
}


void step_lin(float s){

  if(s<0){
    float stroke = floor((float) (fabs(s)*627.78));
    step_backward(stroke);
}
  else{
    float stroke = floor((float) (fabs(s)*627.78));
    step_forward(stroke);
```

```
  }
}
```

stepper.h file

```
#ifndef SERVER
#define SERVER

#include "../../MCAL/DIO driver/dio.h"

#include <inttypes.h>
#include <stdint.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <inttypes.h>
#include <math.h>
#include <time.h>

//#define step 60*10
#define tDays 365.25
#define pressure 1015.2 //average barometric pressure in mbar Cairo, Egypt
//#define T 25
#define timeZone 2 //GMT +2
#define PI 3.14159265
#define conv 0.01745329252
#define w 360/365
#define conv2 57.29577951


#define STEPPER_PORT PORT_D
#define ENA_PIN PIN1
#define DIR_PIN PIN2
#define PUL_PIN PIN3

#define LOW   0
#define HIGH  1

void STEPPER_init(uint8_t stepPort, uint8_t stepPin);

void STEPPER_on(uint8_t stepPort,uint8_t stepPin);

void STEPPER_off(uint8_t stepPort,uint8_t stepPin);

void step(double angle);

void convert(double hourD);
#endif
```

stepper.c file

```
#include "stepper.h"

```

```
3  void convert(double hourD) {
4
5    double fractPart, hour,m,sec,fractM,fractS,min;
6
7    fractM = modf(hourD, &hour);
8
9   m = fractM * 60;
10
11   fractS = modf(m, &min);
12
13   sec = (60 * fractS);
14 }
15
16 void STEPPER_init(uint8_t stepPort,uint8_t stepPin){
17   DIO_init(stepPort,stepPin,OUT);
18 }
19
20 void STEPPER_on(uint8_t stepPort,uint8_t stepPin){
21   DIO_write(stepPort,stepPin,HIGH);
22 }
23
24 void STEPPER_off(uint8_t stepPort,uint8_t stepPin){
25   DIO_write(stepPort,stepPin,LOW);
26 }
27
28 void step(double angle){
29   long i = round(angle/0.0247875);
30   long j = i;
31   long pulse10 = 12;
32   int pulse = 12;
33
34
35
36   //Azimuth
37
38   while(i>0){
39
40     STEPPER_on(STEPPER_PORT,PUL_PIN);
41     PORTB ^= (1<<0);
42     TIMER_delay(pulse);
43     STEPPER_off(STEPPER_PORT,PUL_PIN);
44     PORTB ^= (1<<0);
45     TIMER_delay(pulse);
46     i--;
47
48 }
49
50   PORTB ^= (1<<0);
51   long afterStep = 600000 - (j * pulse * 2);
52   TIMER_delay_10(afterStep);
53   PORTB ^= (1<<0);
54
55
```

```
56
57 }
```

timer.h file

```
1 #ifndef TIMER_H_
2 #define TIMER_H_
3
4 #include "../../Utilities/registers.h"
5 #include <math.h>
6
7
8
9
10
11 //set timer initial value
12 void Timer_init();
13
14 void TIMER_delay(int millisec); //delay of specific amount default uses 256
     prescalar
15
16 void TIMER_delay_lin(float millisec); //delay for linear actuator timer
17
18 void TIMER_delay_10(long millisec); //delay of specific amount default uses
     256 prescalar
19
20 #endif /* TIMER_H_ */
```

timer.c file

```
1 #include "timer.h"
2
3 void TIMER_init(){
4   TCCR0B = 0x00;
5   TCCR1A = 0x00;
6   TCCR1B = 0x00;
7   TCCR1C = 0x00;
8
9
10 }
11
12 int overflows;
13 long overflows_10;
14 int overflowCounter;
15 long overflowCounter_10;
16 float Tmax;
17
18 void TIMER_delay(int millisec){
19   TCCR1B |= (1<<0); //no prescalar (1)
20     //Ttick = 0.0000625; //in ms
21   //Tmax = Ttick * pow(2,16);
22   Tmax = 4.096; //seconds
23   overflows = ceil(millisec/Tmax);
```

```
24   overflowCounter = 0;
25
26   /*the and condition breaks the timer if the button is pressed, thus you
27    don't have to wait for the delay to finish so that the ISR is executed
      */
28   while(overflowCounter < overflows){
29     while( (TIFR1 & (1<<0)) == 0 );
30     //clear bit
31     TIFR1 |= (1<<0);
32     //Timer Stop
33     overflowCounter++;
34   }
35   overflowCounter = 0;
36
37   TCCR1B = 0x00;
38 }
39
40 void TIMER_delay_lin(float millisec){
41   TCCR1B |= (1<<0); //no prescalar (1)
42     //Ttick = 0.0000625; //in ms
43   //Tmax = Ttick * pow(2,16);
44   Tmax = 4.096; //seconds
45   overflows = ceil(millisec/Tmax);
46   overflowCounter = 0;
47
48   /*the and condition breaks the timer if the button is pressed, thus you
49    don't have to wait for the delay to finish so that the ISR is executed
      */
50   while(overflowCounter < overflows){
51     while( (TIFR1 & (1<<0)) == 0 );
52     //clear bit
53     TIFR1 |= (1<<0);
54     //Timer Stop
55     overflowCounter++;
56   }
57   overflowCounter = 0;
58
59   TCCR1B = 0x00;
60 }
61
62 void TIMER_delay_10(long millisec){
63   TCCR1B |= (1<<0); //no prescalar (1)
64     //Ttick = 0.0000625; //in ms
65   //Tmax = Ttick * pow(2,16);
66   Tmax = 4.096; //seconds
67   overflows_10 = round((long)(millisec)/(Tmax));
68   overflowCounter_10 = 0;
69
70   /*the and condition breaks the timer if the button is pressed, thus you
71    don't have to wait for the delay to finish so that the ISR is executed
      */
72   while( overflowCounter_10 < overflows_10){
73     while( (TIFR1 & (1<<0)) == 0 );
```

```
74      //clear bit
75      TIFR1 |= (1<<0);
76      //Timer Stop
77      overflowCounter_10 = overflowCounter_10 +1;
78    }
79    overflowCounter_10 = 0;
80
81    TCCR1B = 0x00;
82  }
```

In the Application file:

app.h file

```
1  #ifndef APPLICATION_H_
2  #define APPLICATION_H_
3
4
5  #include "../ECUAL/Stepper/stepper.h"
6  #include "../ECUAL/Timer/timer.h"
7  #include "../Utilities/interrupts.h"
8  #include "../ECUAL/Linear/linear.h"
9
10
11 double hourD = 16.5;
12 int days = 12;
13 #define lat  30.00409436499994
14 #define lon 31.70039206325549
15 #define GMT 2
16
17 void APP_init(void); // initializes all lights, timers, and interrupt
       registers needed
18 void APP_start(void); // main loop
19
20
21
22 #endif /* APPLICATION_H_ */
```

app.c file

```
1  #include "app.h"
2  #include "../ECUAL/Stepper/stepper.h"
3  #include "../ECUAL/Linear/linear.h"
4  #include <math.h>
5
6  void APP_init(void){
7
8    //Timer init
9    TIMER_init();
10
11   //Stepper motor
12   STEPPER_init(STEPPER_PORT,ENA_PIN); //Holding torque
13   STEPPER_init(STEPPER_PORT,DIR_PIN); //Direction
```

```
14    STEPPER_init(STEPPER_PORT,PUL_PIN); //Pulse
15    DDRB = 0xFF;
16
17    //Linear actuator
18    LINEAR_init(LINEAR_PORT,FORWARD_PIN);
19    LINEAR_init(LINEAR_PORT,BACKWARD_PIN);
20
21
22      //Interrupt init
23    sei(); //enable global interrupt
24
25
26
27  }
28
29
30
31  void APP_start(void){
32
33    while(1){
34
35
36      //step_backward(20000);   //for setting up for the first time linear
        actuator must be fully retracted
37
38      STEPPER_on(STEPPER_PORT,DIR_PIN); //HIGH CCW
39
40       //first step which is used as a reference
41      double LSTM = 15 * GMT;double B = ((double)(360) / (double)(365)) * (
        days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(B * conv)
        - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));double LST =
        hourD + ((double)(TC) / (double)(60));double HRA = (15 * (LST - 12));
        double declination = (23.45 * sin(B * conv));   double sunrise = 12 - ((
        double)1 / (double)15) * (acos(-tan(lat * conv) * tan(declination * conv
        )) * conv2) - (TC / 60);double sunset = 12 + (((double)1 / (double)15) *
         acos(-tan(lat * conv) * tan(declination * conv)) * conv2) - (TC / 60);
        double elevation = asin(((sin(declination * conv) * sin(lat * conv)) + (
        cos(declination * conv) * cos(lat * conv) * cos(HRA * conv)))) * conv2;
42      float A = 21.6;float Beta = 17.1;float phi = elevation * PI/180;float
        alpha = pow(A,2) + pow(Beta,2);float beta = (float)2*A*Beta;
43
44
45
46
47      if((HRA<0) && (hourD > sunrise && hourD < sunset)){
48
49        float S = 10* (sqrt(alpha - (beta*cos(phi))) - 7.3);
50        if(elevation>13 && elevation<81){
51          step_lin(S);
52        }
53        double RefAzimuth = (acos(((sin(declination * conv) * cos(lat * conv)
        ) - (cos(declination * conv) * sin(lat * conv) * cos(HRA * conv))) / (
        cos(elevation * conv))) * conv2);
```

```
54
55        step(RefAzimuth);
56        hourD += 0.166667;
57        double sunrise2 = sunrise;
58        double sunset2 = sunset;
59        double HRA2 = HRA;
60
61        while((HRA2<0) && (hourD > sunrise2 && hourD < sunset2)){
62          double LSTM = 15 * GMT;double B = ((double)(360) / (double)(365)) *
     (days - 81);
63          double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(B * conv) - 1.5
    * sin(B * conv);
64          double TC = EoT + (4 * (lon - LSTM));double LST = hourD + ((double)
    (TC) / (double)(60));
65          double HRA = (15 * (LST - 12));
66          double declination = (23.45 * sin(B * conv));
67          double sunrise = 12 - ((double)1 / (double)15) * (acos(-tan(lat *
    conv) * tan(declination * conv)) * conv2) - (TC / 60);
68          double sunset = 12 + (((double)1 / (double)15) * acos(-tan(lat *
    conv) * tan(declination * conv)) * conv2) - (TC / 60);
69          double elevation = asin(((sin(declination * conv) * sin(lat * conv)
    ) + (cos(declination * conv) * cos(lat * conv) * cos(HRA * conv)))) *
    conv2;float phi = elevation * PI/180;float alpha = pow(A,2) + pow(Beta
    ,2);float beta = (float)2*A*Beta;
70          float newS = 10* (sqrt(alpha - (beta*cos(phi))) - 7.3);
71          float diffS = newS - S;
72          S = newS;
73          if(elevation>13 && elevation<81){
74            step_lin(diffS);
75
76          }
77
78          double NewAzimuth = acos(((sin(declination * conv) * cos(lat * conv
    )) - (cos(declination * conv) * sin(lat * conv) * cos(HRA * conv))) / (
    cos(elevation * conv))) * conv2;
79
80          double diffAzi = NewAzimuth - RefAzimuth;
81          RefAzimuth = NewAzimuth;
82          step(diffAzi);
83          hourD += 0.166667;
84          double LSTM2 = 15 * GMT;
85          double B2 = ((double)(360) / (double)(365)) * (days - 81);
86          double EoT2 = 9.87 * sin(2 * B2 * conv) - 7.53 * cos(B * conv) -
    1.5 * sin(B * conv);
87          double TC2 = EoT + (4 * (lon - LSTM2));
88          double LST2 = hourD + ((double)(TC2) / (double)(60));
89          double HRA2 = (15 * (LST2 - 12));
90          double declination2 = (23.45 * sin(B2 * conv));
91          double sunrise2 = 12 - ((double)1 / (double)15) * (acos(-tan(lat *
    conv) * tan(declination2 * conv)) * conv2) - (TC2 / 60);
92          double sunset2 = 12 + (((double)1 / (double)15) * acos(-tan(lat *
    conv) * tan(declination2 * conv)) * conv2) - (TC2 / 60);
93          if(HRA2>0){
```

```
94          while(HRA2>0){
95            double LSTM = 15 * GMT;double B = ((double)(360) / (double)
    (365)) * (days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(
    B * conv) - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));
    double LST = hourD + ((double)(TC) / (double)(60));double HRA = (15 * (
    LST - 12));double declination = (23.45 * sin(B * conv));double sunrise =
     12 - ((double)1 / (double)15) * (acos(-tan(lat * conv) * tan(
    declination * conv)) * conv2) - (TC / 60);double sunset = 12 + (((double
    )1 / (double)15) * acos(-tan(lat * conv) * tan(declination * conv)) *
    conv2) - (TC / 60);double elevation = asin(((sin(declination * conv) *
    sin(lat * conv)) + (cos(declination * conv) * cos(lat * conv) * cos(HRA
    * conv)))) * conv2;
96            float phi =  -1*elevation * PI/180;
97            float alpha = pow(A,2) + pow(Beta,2); //7
98            float beta = (float)2*A*Beta;
99            float newS = fabs(10* (sqrt(alpha - (beta*cos(phi))) - 7.3));
100           float diffS = newS - S;
101           S = newS;
102           if(elevation>13 && elevation<81){
103             step_lin(diffS);
104           }
105
106           double NewAzimuth2 = 360 - (acos(((sin(declination * conv) *
    cos(lat * conv)) - (cos(declination * conv) * sin(lat * conv) * cos(HRA
    * conv))) / (cos(elevation * conv))) * conv2);
107
108           double diffAzi1 = NewAzimuth2 - NewAzimuth;
109           NewAzimuth = NewAzimuth2;
110           step(diffAzi1);
111           hourD += 0.166667;
112           if(hourD > sunset){
113             STEPPER_off(STEPPER_PORT,DIR_PIN); //CW
114             double LSTM = 15 * GMT;double B = ((double)(360) / (double)
    (365)) * (days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(
    B * conv) - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));
    double LST = hourD + ((double)(TC) / (double)(60));double HRA = (15 * (
    LST - 12));double declination = (23.45 * sin(B * conv));double elevation
     = asin(((sin(declination * conv) * sin(lat * conv)) + (cos(declination
    * conv) * cos(lat * conv) * cos(HRA * conv)))) * conv2;double sunrise =
    12 - ((double)1 / (double)15) * (acos(-tan(lat * conv) * tan(declination
     * conv)) * conv2) - (TC / 60);double sunset = 12 + (((double)1 / (
    double)15) * acos(-tan(lat * conv) * tan(declination * conv)) * conv2) -
     (TC / 60);
115             double finAzimuth = (360 - (acos(((sin(declination * conv) *
    cos(lat * conv)) - (cos(declination * conv) * sin(lat * conv) * cos(HRA
    * conv))) / (cos(elevation * conv))) * conv2));
116             step(finAzimuth);
117             break;
118           }
119         }
120         break;
121       }
122     }
```

```
123
124
125
126
127
128
129
130
131
132
133
134         while(HRA>0){
135           double LSTM = 15 * GMT;double B = ((double)(360) / (double)(365)) *
          (days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(B * conv
          ) - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));double LST
          = hourD + ((double)(TC) / (double)(60));double HRA = (15 * (LST - 12));
          double declination = (23.45 * sin(B * conv));double sunrise = 12 - ((
          double)1 / (double)15) * (acos(-tan(lat * conv) * tan(declination * conv
          )) * conv2) - (TC / 60);double sunset = 12 + (((double)1 / (double)15) *
           acos(-tan(lat * conv) * tan(declination * conv)) * conv2) - (TC / 60);
          double elevation = asin(((sin(declination * conv) * sin(lat * conv)) + (
          cos(declination * conv) * cos(lat * conv) * cos(HRA * conv)))) * conv2;
136           float phi =  -1*elevation * PI/180;
137           float alpha = pow(A,2) + pow(Beta,2); //7
138           float beta = (float)2*A*Beta;
139           float newS = fabs(10* (sqrt(alpha - (beta*cos(phi))) - 7.3));
140           float diffS = newS - S;
141           S = newS;
142           if(elevation>13 && elevation<81){
143             step_lin(diffS);
144           }
145           double NewAzimuth = 360 - (acos(((sin(declination * conv) * cos(lat
          * conv)) - (cos(declination * conv) * sin(lat * conv) * cos(HRA * conv)
          )) / (cos(elevation * conv))) * conv2);
146           double diffAzi = NewAzimuth - RefAzimuth;
147           RefAzimuth = NewAzimuth;
148           step(diffAzi);
149           hourD += 0.166667;
150           if(hourD > sunset){
151             STEPPER_off(STEPPER_PORT,DIR_PIN); //CW
152             double LSTM = 15 * GMT;double B = ((double)(360) / (double)(365))
          * (days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(B *
          conv) - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));double
          LST = hourD + ((double)(TC) / (double)(60));double HRA = (15 * (LST -
          12));double declination = (23.45 * sin(B * conv));double elevation =
          asin(((sin(declination * conv) * sin(lat * conv)) + (cos(declination *
          conv) * cos(lat * conv) * cos(HRA * conv)))) * conv2;double sunrise = 12
          - ((double)1 / (double)15) * (acos(-tan(lat * conv) * tan(declination *
          conv)) * conv2) - (TC / 60);double sunset = 12 + (((double)1 / (double)
          15) * acos(-tan(lat * conv) * tan(declination * conv)) * conv2) - (TC /
          60);
153           double finAzimuth = (360 - (acos(((sin(declination * conv) * cos(
          lat * conv)) - (cos(declination * conv) * sin(lat * conv) * cos(HRA *
```

```
conv))) / (cos(elevation * conv))) * conv2));
154           step(finAzimuth);
155           break;
156         }
157       }
158     }
159
160
161     //after 12:00 in solar time
162     else if((HRA>0) && (hourD > sunrise && hourD < sunset)){
163       float S = 10* (sqrt(alpha - (beta*cos(phi))) - 7.3);
164       if(elevation>13 && elevation<81){
165         step_lin(S);
166       }
167       double RefAzimuth = 360 - (acos(((sin(declination * conv) * cos(lat *
      conv)) - (cos(declination * conv) * sin(lat * conv) * cos(HRA * conv)))
      / (cos(elevation * conv))) * conv2);
168       step(RefAzimuth);
169       hourD += 0.166667;
170
171       while((HRA<0) && (hourD > sunrise && hourD < sunset)){
172         double LSTM = 15 * GMT;double B = ((double)(360) / (double)(365)) *
      (days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(B * conv
      ) - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));double LST
      = hourD + ((double)(TC) / (double)(60));double HRA = (15 * (LST - 12));
      double declination = (23.45 * sin(B * conv));      double sunrise = 12 -
      ((double)1 / (double)15) * (acos(-tan(lat * conv) * tan(declination *
      conv)) * conv2) - (TC / 60);double sunset = 12 + (((double)1 / (double)
      15) * acos(-tan(lat * conv) * tan(declination * conv)) * conv2) - (TC /
      60);double elevation = asin(((sin(declination * conv) * sin(lat * conv))
      + (cos(declination * conv) * cos(lat * conv) * cos(HRA * conv)))) *
      conv2;float phi = elevation * PI/180;float alpha = pow(A,2) + pow(Beta
      ,2);float beta = (float)2*A*Beta;
173         float newS = 10* (sqrt(alpha - (beta*cos(phi))) - 7.3);
174         float diffS = newS - S;
175         S = newS;
176         if(elevation>13 && elevation<81){
177           step_lin(diffS);
178         }
179         double NewAzimuth = acos(((sin(declination * conv) * cos(lat * conv
      )) - (cos(declination * conv) * sin(lat * conv) * cos(HRA * conv))) / (
      cos(elevation * conv))) * conv2;
180         double diffAzi = NewAzimuth - RefAzimuth;
181         RefAzimuth = NewAzimuth;
182         step(diffAzi);
183         hourD += 0.166667;
184       }
185
186       while(HRA>0){
187         double LSTM = 15 * GMT;double B = ((double)(360) / (double)(365)) *
      (days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(B * conv
      ) - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));double LST
      = hourD + ((double)(TC) / (double)(60));double HRA = (15 * (LST - 12));
```

```
double declination = (23.45 * sin(B * conv));double sunrise = 12 - ((
double)1 / (double)15) * (acos(-tan(lat * conv) * tan(declination * conv
)) * conv2) - (TC / 60);double sunset = 12 + (((double)1 / (double)15) *
 acos(-tan(lat * conv) * tan(declination * conv)) * conv2) - (TC / 60);
double elevation = asin(((sin(declination * conv) * sin(lat * conv)) + (
cos(declination * conv) * cos(lat * conv) * cos(HRA * conv)))) * conv2;
188       float phi =  -1*elevation * PI/180;
189       float alpha = pow(A,2) + pow(Beta,2); //7
190       float beta = (float)2*A*Beta;
191       float newS = fabs(10* (sqrt(alpha - (beta*cos(phi))) - 7.3));
192       float diffS = newS - S;
193       S = newS;
194       if(elevation>13 && elevation<81){
195         step_lin(diffS);
196       }
197       double NewAzimuth = 360 - (acos(((sin(declination * conv) * cos(lat
 * conv)) - (cos(declination * conv) * sin(lat * conv) * cos(HRA * conv)
)) / (cos(elevation * conv))) * conv2);
198       double diffAzi = NewAzimuth - RefAzimuth;
199       RefAzimuth = NewAzimuth;
200       step(diffAzi);
201       hourD += 0.166667;
202       if(hourD > sunset){
203         STEPPER_off(STEPPER_PORT,DIR_PIN); //CW
204         double LSTM = 15 * GMT;double B = ((double)(360) / (double)(365))
 * (days - 81);double EoT = 9.87 * sin(2 * B * conv) - 7.53 * cos(B *
conv) - 1.5 * sin(B * conv);double TC = EoT + (4 * (lon - LSTM));double
LST = hourD + ((double)(TC) / (double)(60));double HRA = (15 * (LST -
12));double declination = (23.45 * sin(B * conv));double elevation =
asin(((sin(declination * conv) * sin(lat * conv)) + (cos(declination *
conv) * cos(lat * conv) * cos(HRA * conv)))) * conv2;double sunrise = 12
 - ((double)1 / (double)15) * (acos(-tan(lat * conv) * tan(declination *
 conv)) * conv2) - (TC / 60);double sunset = 12 + (((double)1 / (double)
15) * acos(-tan(lat * conv) * tan(declination * conv)) * conv2) - (TC /
60);
205         double finAzimuth = (360 - (acos(((sin(declination * conv) * cos(
lat * conv)) - (cos(declination * conv) * sin(lat * conv) * cos(HRA *
conv))) / (cos(elevation * conv))) * conv2));
206         step(finAzimuth);
207         break;
208       }
209     }
210   }
211
212
213
214   while(hourD < sunrise || hourD > sunset){
215     if(floor(hourD) ==24){
216       days+=1;
217       hourD = 0;
218       TIMER_delay_10(600000); //10 minutes delay
219
220     }
```

```
221         else{
222           TIMER_delay_10(600000); //10 minutes delay
223           hourD += 0.166667;
224         }
225       }
226
227
228
229
230 }
231 }
```

List of Figures

Bibliography

[1] BeamTypesWeb. https://www.e-education.psu.edu/eme810/node/575.

[2] Ahmed Ennaoui. Photovoltaic solar energy conversion (pvsec) helmholtz-zentrum berlin für materialien und energie-free. *University of Berlin-Allemand*, 2014.

[3] Trough. https://www.alternative-energy-tutorials.com/solar-hot-water/parabolic-trough-reflector.html.

[4] PowerTower. https://www.energy.gov/eere/solar/power-tower-system-concentrating-solar-thermal-power-basics.

[5] pdc. https://renewablepedia.com/parabolic-dish-solar-collectors/.

[6] pdcImage. https://renewablepedia.com/parabolic-dish-solar-collectors/.

[7] frensel. https://www.industrial-solar.de/en/technologies/fresnel-collector/.

[8] JD Nixon, PK Dey, and PA Davies. Which is the best solar thermal collection technology for electricity generation in north-west india? evaluation of options using the analytical hierarchy process. *Energy*, 35(12):5230–5240, 2010.

[9] Solar trackingintro. https://www.linkedin.com/pulse/introduction-solar-trackers-nick-lusson.

[10] C.B.Honsberg and S.G.Bowden, "Absorption Coefficient" sun position calculator. https://en.wikipedia.org/wiki/Concentrated_solar_power.

[11] Active Solar trackingfig. https://www.solarmango.com/scp/solar-tracker-tracking-the-sun-for-maximum-power/.

[12] Ahmad Razif bin Abdul Hamid, Ahmad Khusairy Hakiim bin Abdul Azim, and Mohd Hafizuddin bin Abu Bakar. A review on solar tracking system. In *e-Proceedings iCompEx17 Academic Paper*, 2017.

[13] KAREN BARBOSA DE MELO, BRUNO HENRIQUE KIKUMOTO DE PAULA, MICHELLE KITAYAMA DA SILVA, DANTE INGA NARVÁEZ, HUGO SOEIRO MOREIRA, MARCELO GRADELLA VILLALVA, and THAIS GAMA DE SIQUEIRA. A study on the influence of locality in the viability of solar tracker systems. In *Congresso Brasileiro de Automática-CBA*, volume 1, 2019.

[14] Miloš Jovanović, Željko Despotović, and ore Urukalo. The chronological system of solar tracking implemented on mobile solar generator-imp mseg. *Zbornik Meunarodne konferencije o obnovljivim izvorima električne energije–MKOIEE*, 5(1):107–113, 2017.

[15] Koray Ulgen. Optimum tilt angle for solar collectors. *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, 28(13):1171–1180, 2006.

[16] Nassar Yasser Fathi and Alsadi Samer. View factors of flat solar collectors array in flat, inclined, and step-like solar fields. *Journal of Solar Energy Engineering*, 138(6), 2016.

[17] Samy A Khalil and AM Shaffie. A comparative study of total, direct and diffuse solar irradiance by using different models on horizontal and inclined surfaces for cairo, egypt. *Renewable and Sustainable Energy Reviews*, 27:853–863, 2013.

[18] Chung-Yuen Won, Duk-Heon Kim, Sei-Chan Kim, Won-Sam Kim, and Hack-Sung Kim. A new maximum power point tracker of photovoltaic arrays using fuzzy controller. In *Proceedings of 1994 Power Electronics Specialist Conference-PESC'94*, volume 1, pages 396–403. IEEE, 1994.

[19] Alain Ricaud. Gisement solaire et transferts énergétiques. *Master Energies Renouvelables, Université de Cergy-Pontoise*, 2011.

[20] Mohamed Nfaoui and Khalil El-Hami. Extracting the maximum energy from solar panels. *Energy Reports*, 4:536–545, 2018.

[21] BeamTypesWeb. https://www.ftexploring.com/solar-energy/direct-and-diffuse-radiation.htm.

[22] TMYmaking use of tmy data. https://www.pveducation.org/pvcdrom/properties-of-sunlight/making-use-of-tmy-data.

[23] Diffuse irradiationmaking use of tmy data. https://inspire.ec.europa.eu/codelist/SolarPotentialValue/diffuseIrradiation.

[24] Diffuse irradiationmaking use of tmy data. https://www.firstgreen.co/difference-between-direct-and-diffused-radiation/.

[25] Diffuse irradiationmaking use of tmy data. https://www.pveducation.org/pvcdrom/properties-of-sunlight/making-use-of-tmy-data.

[26] Richard Perez, Pierre Ineichen, Robert Seals, Joseph Michalsky, and Ronald Stewart. Modeling daylight availability and irradiance components from direct and global irradiance. *Solar energy*, 44(5):271–289, 1990.

[27] Solar dish principlemaking use of tmy data. https://www.alternative-energy-tutorials.com/solar-hot-water/solar-dish-collector.html.

[28] Saban Yilmaz, Hasan Riza Ozcalik, Osman Dogmus, Furkan Dincer, Oguzhan Akgol, and Muharrem Karaaslan. Design of two axes sun tracking controller with analytically solar radiation calculations. *Renewable and Sustainable Energy Reviews*, 43:997–1005, 2015.

[29] Solar dish principledefinition. https://helioscsp.com/the-significance-of-the-concentration-factor-in-concentrated-solar-

[30] Sattar Aljabair, Laith Jaafer Habeeb, et al. Study the effect of diameter and depth of parabolic dish collector on the concentration ratio

and temperature amount of solar tower receiver. *Journal of University of Babylon for Engineering Sciences*, 27(1):142–156, 2019.

[31] Solar dish collectionsterling engine. https://www.alternative-energy-tutorials.com/solar-hot-water/stirling-engine-generator.html.

[32] MJ Blanco and S Miller. Introduction to concentrating solar thermal (cst) technologies. In *Advances in concentrating solar thermal research and technology*, pages 3–25. Elsevier, 2017.

[33] Joseph J Michalsky. The astronomical almanac's algorithm for approximate solar position (1950–2050). *Solar energy*, 40(3):227–235, 1988.

[34] PVeducation motion of the sun. https://www.pveducation.org/pvcdrom/properties-of-sunlight/motion-of-the-sun.

[35] Eviatar Zerubavel. The standardization of time: A sociohistorical perspective. *American journal of sociology*, 88(1):1–23, 1982.

[36] WJ Greenstreet. Index to the mathematical gazette no. 144, jan. 1920—no. 155, december 1921. *The Mathematical Gazette*, 10(I1):iii–xii, 1921.

[37] LSTM. https://www.pveducation.org/pvcdrom/properties-of-sunlight/the-suns-position.

[38] Declination2. https://www.pveducation.org/pvcdrom/properties-of-sunlight/declination-angle.

[39] PI Cooper. The absorption of radiation in solar stills. *Solar energy*, 12(3):333–346, 1969.

[40] Elevationsolar sena. https://solarsena.com/solar-elevation-angle-altitude/.

[41] Azimuth Angle. https://www.pveducation.org/pvcdrom/properties-of-sunlight/azimuth-angle.

[42] AR Amelia, YM Irwan, I Safwati, WZ Leow, MH Mat, and Mohd Shukor Abdul Rahim. Technologies of solar tracking systems: A review. In *IOP Conference Series: Materials Science and Engineering*, volume 767, page 012052. IOP Publishing, 2020.

[43] Rajesh Singh, Suresh Kumar, Anita Gehlot, and Rupendra Pachauri. An imperative role of sun trackers in photovoltaic technology: A review. *Renewable and Sustainable Energy Reviews*, 82:3263–3278, 2018.

[44] AZ Hafez, AM Yousef, and NM Harag. Solar tracking systems: Technologies and trackers drive types–a review. *Renewable and Sustainable Energy Reviews*, 91:754–782, 2018.

[45] Athanasios Aris Panagopoulos, Georgios Chalkiadakis, and Nicholas Jennings. Towards optimal solar tracking: a dynamic programming approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[46] DF Fam, SP Koh, SK Tiong, and KH Chong. Qualitative analysis of stochastic operations in dual axis solar tracking environment. *Research Journal of Recent Sciences* _____ *ISSN*, 2277:2502, 2012.

[47] George C Bakos. Design and construction of a two-axis sun tracking system for parabolic trough collector (ptc) efficiency improvement. *Renewable energy*, 31(15):2411–2421, 2006.

[48] Shashwati Ray and Abhishek Kumar Tripathi. Design and development of tilted single axis and azimuth-altitude dual axis solar tracking systems. In *2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, pages 1–6. IEEE, 2016.

[49] Mohammed Saifuddin Munna, Mohammad Ariful Islam Bhuyan, Kazi Mustafizur Rahman, and Md Ashiqul Hoque. Design, implementation and performance analysis of a dual-axis autonomous solar tracker. In *2015 3rd International Conference on Green Energy and Technology (ICGET)*, pages 1–5. IEEE, 2015.

[50] Yongqiang Zhu, Jiahao Liu, and Xiaohua Yang. Design and performance analysis of a solar tracking system with a novel single-axis tracking structure to maximize energy collection. *Applied Energy*, 264:114647, 2020.

[51] Kamrul Islam Chowdhury, Md Alam, Promit Shams Bakshi, et al. *Performance comparison between fixed panel, single-axis and dual-axis sun tracking solar panel system*. PhD thesis, BARC University, 2017.

[52] Focal Point formulafp. https://cs.uwaterloo.ca/~m2nagapp/courses/CS446/1195/Arch_Design_Activity/Layered.pdf.

[53] Athanasios Aris Panagopoulos, Georgios Chalkiadakis, and Nicholas Jennings. Towards optimal solar tracking: a dynamic programming approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[54] Prifix satellite dishspecs. https://eg.labeb.com/en/pc/prifix-satellite-dish-100-cm-size-wall-and-ground-fixtaion-95809.

[55] NEMA23. https://www.wellpcb.com/nema-23-stepper-motor.html.

[56] NEMA23. https://www.ampere-electronics.com/p/23hs7628-nema23-stepper-motor/.

[57] TB6600. https://www.watelectronics.com/tb6600-stepper-motor-driver-module/.

[58] thrustBearing. https://monroeengineering.com/blog/what-are-thrust-bearings-and-how-do-they-work/.

[59] central distance. https://khkgears.net/new/gear_knowledge/gear-nomenclature/center-distance.html.

[60] SunCalc sun position calculator. https://www.suncalc.org/#/30.0019,31.7022,13/2022.10.12/10:00/1/3.

[61] SunEarthTools sun position calculator. https://www.sunearthtools.com/dp/tools/pos_sun.php?lang=en.