Photovoltaic applications in demanding situations : estimation and optimisation of solar ressources for autonomous power supplies

Kha Bao Khanh Cao

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue le *27/10/2023* **par :**

Kha Bao Khanh CAO

Photovoltaic applications in demanding situations: estimation and optimisation of solar ressources for autonomous power supplies.

JURY

Corinne ALONSO	Professeur d'Université	Membre du Jury
Sonia BEN DHIA	Professeur d'Université	Membre du Jury
Vincent BOITIER	Maître de conférences HdR	Membre du Jury
Vincent DEBUSSCHERE	Maître de conférences HdR	Rapporteur
Anne MIGAN DUBOIS	Professeur d'Université	Rapporteur
Thierry TALBERT	Maître de conférences HdR	Membre du Jury

École doctorale et spécialité :

GEET : Génie Electrique

Unité de Recherche :

Laboratoire d'analyse et d'architecture des systèmes

Directeur de Thèse :

Vincent BOITIER

Acknowledgments

The first person I would like to thank is my thesis director, Mr. Boitier, for having guided me throughout this three-year project. It was intensive but fun, where I learnt so much and developed a large set of skills. I would also like to thank Mme. Alonso for the various tips she provided throughout my three years working at LAAS and put me in contact with various other colleagues. I must also thank Mr. Seguier for the oscillating solar platform; it was quite regrettable that I did not make the most of it for this work. I want to mention Yanandlall Gopee, a close friend in the lab where we spent so much time discussing life and procrastinating together. Finally, on the technical side, I also would like to thank my gym buddy Nguyen Ngoc Bao, who helped me with the evaluation of neural network-based maximum power point tracking algorithms. I asked for some general input and he shipped me a thesis worthy review of each individual paper with razor-sharp analysis from the point of view of a data scientist.

However, this thesis could not have been possible without my family. My parents, Chu Thi Hai Yen and Cao Thi, being doctors and teachers in their respective fields, were a major inspiration for the path I chose. My brother, Cao Kha Bao Khang, always motivated me to do things and try to do them better every day. During this period, my girlfriend Nguyen Thai Mai Chi provided important psychological support so that I could overcome the challenges during the last three years. Furthermore, I also would like to thank a close friend, Quoc An, who really motivated me to believe that folding should not be an option and that persistence is a virtue that leads to favourable results. I believe that without their help, I could not have accomplished everything with such heart and commitment.

Contents

Glossary **v**

Introduction **1**

1 Shading forecast for autonomous PV systems **7**
1.1 Literature review 10
1.1.1 Computer Assisted Design Software 10
1.1.2 Digital Surface Models 13
1.1.3 Sky imaging 14
1.1.4 Solar estimation under forest canopies 15
1.2 Solar position relative to our camera 17
1.2.1 Clarification on geometry terms 17
1.2.2 Solar coordinates in our camera's frame of reference 18
1.3 How the camera maps 3D points onto a 2D image 20
1.3.1 The principle of camera calibration 20
1.3.2 Choosing software toolbox for fisheye calibration 22
1.4 Process shading data and generate solar estimate 25
1.4.1 Selecting the software environment and irradiance data . . . 25
1.4.2 Calculating the direct shading factor 26
1.4.3 Calculating the diffuse shading factor 26
1.5 Preliminary solar estimation evaluation 29
1.5.1 Validating the solar position on an image 29
1.5.2 Experimental measurement using a solar panel 29
1.6 Summary of user procedure 33
1.6.1 Taking the sky photo 34
1.6.2 Taking the calibration images 35
1.7 Python script for system energy forecast 37
1.8 Conclusion and perspective of Chapter 1 40

2 PV simulation for studying GMPPT **43**
2.1 Modelling a PV array under PSCs 49
2.1.1 PV cell modelling 49
2.1.2 PV module modelling 50
2.1.3 PV block modelling 53
2.1.4 Modelling a string of PV blocks 55
2.1.5 Determining model parameters 56
2.2 Graph the distribution of GMPPs 62
2.2.1 Selecting the set of irradiance for the G-T sweep 62
2.2.2 Selecting the set of temperature for the G-T sweep 63
2.2.3 Studying the GMPP distribution from the G-T sweeps 64

2.3 Compute optimisations 67
2.3.1 Two basic G-T sweep scripts 67
2.3.2 Improvement one: Reducing program instructions 70
2.3.3 Improvement two: Vectorisation 72
2.3.4 Improvement three: Look-up table (LUT) 72
2.3.5 Improvement four: Parallel computing 72
2.3.6 Evaluating the execution time impact of the optimisations . . 73
2.4 Electrical simulation of solar harvesting system 74
2.4.1 The basics of software GMPPT algorithm 74
2.4.2 Electrical model of the PV string 76
2.4.3 Electrical model of the converter board 77
2.4.4 Simple battery and load model 81
2.4.5 Evaluating the Simulink model 82
2.5 Conclusion and perspective of Chapter 2 84

3 GMPPT algorithms 87
3.1 Literature review 92
3.1.1 Overview of software MPPT 92
3.1.2 In-depth analysis of MPPT algorithms 93
3.1.3 In-depth analysis of GMPPT algorithms 101
3.1.4 State of the literature 107
3.2 Probabilistic GMPPT algorithm 114
3.2.1 Algorithm description 114
3.2.2 Theoretical evaluation of the search mechanism 116
3.3 Validation of our algorithm proposal 118
3.3.1 Test profiles 118
3.3.2 Test setups 120
3.3.3 Results for a sequence of irradiance steps 121
3.3.4 Results for under varying irradiance profiles 123
3.4 Conclusion and perspective of Chapter 3 128

Conclusion 131

Bibliography 135

Glossary

$(1, \theta_s, \varphi_s)$ Spherical coordinates of the sun in ground reference frame

(h_{shade}, w_{shade}) Height and width the shadow in the shading profile creator

$(O\vec{x'}\vec{y'}\vec{z'})$ Spherical coordinates of the sun in ground reference frame

$(O\vec{x_a}\vec{y_a}\vec{z_a})$ Reference frame used by AstroPy

$(O\vec{x_c}\vec{y_c}\vec{z_c})$ Spherical coordinates of the sun in ground reference frame

$(O\vec{x_w}\vec{y_w}\vec{z_w})$ Arbitrary reference frame of an object in real world

$(O\vec{x}\vec{y}\vec{z})$ Reference frame connected to ground

(x'_s, y'_s, z'_s) Spherical coordinates of the sun in ground reference frame

(x_s, y_s, z_s) Spherical coordinates of the sun in ground reference frame

(x_{shade}, y_{shade}) x and y coordinates of the shadow in the shading profile creator

$(\theta_{a_s}, \varphi_{a_s})$ Azimuth and zenith of the sun in the AstroPy coordinate

$(x_{c_s}, y_{c_s}, z_{c_s})$ Spherical coordinates of the sun in ground reference frame

η_{battin} Charge efficiency of the battery

$\eta_{battout}$ Discharge efficiency of the battery

$\eta_{conversion}$ Efficiency of converter

η_{solar} Efficiency of solar panel

ω Inclination of the smartphone used to take the sky photo

ψ Orientation of the smartphone used to take the sky photo

θ Azimuth of an arbitrary object

θ_{shade} Movement direction of the shadow in the shading profile creator

φ Zenith of an arbitrary object

A Equivalent diode ideality factor of a PV module

A_{byp} Ideality factor of the bypass diode used in Shockley equation

A_{cell} Diode ideality factor of a PV cell

C Arbitrary constant used to evaluate whether a projection is equal area

C_{batt} Capacity of baterry

$cter$ Counter of iterations in HC phase in fast GMPPT

$cter_{limit}$ Maximum limit of $cter$

D^k Duty cycle sent at iteration k

d_x^i Duty cycle of candidate solution i at search iteration x

d_{est}^i Duty cycle estimate to get to target voltage i in fast GMPPT

G Irradiance received by a PV cell or module

G_{global} Global irradiance in the shading profile creator

G_{ref} Reference irradiance at $1000Wm^{-2}$

$I_{block} = f(V_{pv})$ Shortened representation of equation 2.5 where the input is voltage and output is current of the PV block

I_{block} Current of a PV block

I_{byp} Reverse saturation current of the bypass diode used in Shockley equation

I_{db} Current through the bypass diode

$I_{pv} = f_m(V_{pv})$ Shortened representation of equation 2.4 where the input is voltage and output is current of the PV module

I_{sc} Short circuit current of a PV cell or module

inv Number of duty cycle inversions

inv_{limit} Maximum limit of inv

k_1 Constant to transform the zenith of an object to a vertical position on an image

k_2 Constant to transform the azimuth of an object to a horizontal position on an image

k_B Boltzman constant

l Unit based on the length of a square solar panel in the shading profile creator

N_s Number of PV cells in series in a PV module

$ninv$ The length of the streak of duty cycle updates without inversions

$ninv_{limit}$ Maximum limit of $ninv$

p_x^i Power measured by candidate solution i at search iteration x

P_{gmpp} Power at global maximum power point of a PV string with bypass diodes

P_{mpp} Power at maximum power point of a PV module

P_{pv}^k PV string power measured at iteration k

q Elemental charge

r Distance from a point to origin of reference frame

R_{don} On resistance of a bypass diode used in piecewise modelling

R^k_{solar} Solar irradiation at timestamp k

R_s Equivalent series resistance in the PV module model

S Size of solar panel

SOC^k State of charge of battery at timestamp k

SOC_{max} Maximum state of charge of the battery

SOC_{min} Minimum state of charge of the battery

T Temperature of a PV cell or module

T_s Sampling time

T_{global} Global temperature in the shading profile creator

T_{ref} Reference temperature at $25°C$

u Horizontal position of a pixel on an image

v Vertical position of a pixel on an image

V_f Forward voltage of a bypass diode used in piecewise modelling

$V_{battnom}$ Nominal voltage of battery

V_{gmpp} Voltage at global maximum power point of a PV string with bypass diodes

v^i_{low} Lower target voltage bound i in fast GMPPT

V_{mpp} Voltage at maximum power point of a PV module

V_{oc} Open circuit voltage of a PV cell or module

$V_{pv} = g(I_{block})$ Shortened representation of equation 2.5 where the input is current and output is voltage of the PV block

$V_{pv} = g_m(I_{pv})$ Shortened representation of equation 2.4 where the input is current and output is voltage of the PV module

V^k_{pv} PV string voltage measured at iteration k

v_{shade} Velocity of the shadow in the shading profile creator

V_{string} Voltage of a PV string with bypass diodes

v^i_{target} Target voltage i in fast GMPPT

v^i_{up} Upper target voltage bound i in fast GMPPT

$E_{batt}{}^k$ Energy flow to the battery at iteration k

$E_{conv}{}^{k}$ Energy generated by converter at timestamp k

$E_{load}{}^{k}$ Energy consumed by the load at timestamp k

I_0 Equivalent reverse saturation current of the diode in a PV module

I_{0cell} Reverse saturation current of the diode in a PV cell

I_d Equivalent current through the diode of a PV module

I_{dcell} Current through the diode of a PV cell

I_L Equivalent photocurrent of a PV module

I_{Lcell} Photocurrent of a PV cell

I_{pv} Current of a PV module

$I_{pv_{cell}}$ Current of a PV cell

I_{scn} Nominal short circuit current of a PV module

$I_{scncell}$ Nominal short circuit current of a PV cell

K_i Current temperature coefficient of a PV module

K_{icell} Current temperature coefficient of a PV cell

K_v Voltage temperature coefficient of a PV module

K_{vcell} Voltage temperature coefficient of a PV cell

R_p Equivalent parallel resistance in the PV module model

$R_{p_{cell}}$ Parallel resistance in the PV cell model

R_{scell} Series resistance in the PV cell model

V_{ocn} Nominal open circuit of a PV module

$V_{ocncell}$ Nominal open circuit of a PV cell

V_{pv} Number of PV string in parallel in a PV module

V_{pv} Voltage of a PV module

$V_{pv_{cell}}$ Voltage of a PV cell

AC Alternating Current

ADC Analogue-to-Digital Converter

CAD Computer-Assisted Design

CB Characteristics-based

D Duty cycle

DC Direct Current

DSM Digital Surface Model

ECONECT Ecosystème connecté Sentinelles de l'Environnement or Connected Environmental Sentinels Ecosystem

FOV Field-of-view

fVoc Fractional open circuit voltage

G-T Irradiance and temperature

GMPP Global Maximum Power Point

GMPPT Global Maximum Power Point Tracking

HC Hill Climbing

I-V Current-Voltage

InC Incremental Conductance

IntelC Intelligent Control

ISAE Institut Supérieur de l'Aéronautique et de l'Espace

LiDAR Light Detection And Ranging

LUT Look-up Table

MPC Multiple Peak Capable

MPP Maximum Power Point

MPPT Maximum Power Point Tracking

MT/s Megatransfers per second

NVME SSD Non-Volatile Memory Express Solid State Drive

nVoc GMPPT schema based on performing MPPT on different zones of the voltages.

OptA Optimisation Algorithms

P-V Power-Voltage

P&O Perturb and Observe

PD Parameter-Dependant

PSC Partial Shading Conditions

PV Photovoltaic

RAM Random Access Memory

ROM Read Only Memory

s-iteration Search iteration of an optimisation algorithm

SP Series Parallel

UTC Coordinated Universal Time

Introduction

In this thesis, we discuss **solar harvesting and optimisation for autonomous low to medium power supplies between 10W and 100W**. In situations where the load does not consume a significant amount of energy, a battery could suffice. However, there are many use cases where this is not possible or simply not practical for the entire life cycle of the system. Therefore, there is a need to harvest energy from the environment. Most of the time, the only practical energy source would be the sun. It is everywhere and relatively predictable, and solar panels are becoming more affordable by day thanks to the renewable energy boom.

Figure 1: Illustration of the ECONECT project as shown on its website econect.cnrs.fr

There are several examples of autonomous power supplies to better understand the challenges they present. Note that none of the application examples provided is the centre point of this thesis work because we aim to address the general problem of solar autonomous power supplies operating under non-optimal situations as a whole. The examples only serve to address how the findings presented in this manuscript could be used and where we gathered our inspirations to conduct the work. One such use case is the project ECONECT (Ecosystème connecté Sentinelles de l'Environnement) whose objective is to study and survey the ecosystem's response to climate change with remote sensors placed in the wild [1]. An overview of the project can be found in Figure 1. Although very low-power sensors can get by with a sufficiently large battery capacity, a video camera and wireless transfers of substantial data do not last long on battery power. The frequent and tedious battery change trips to keep the ECONECT infrastructure running highlight this fact very well, where Maxime Cauchoix, a researcher involved in the project, had to make a two-hour trip every two days to change the battery.

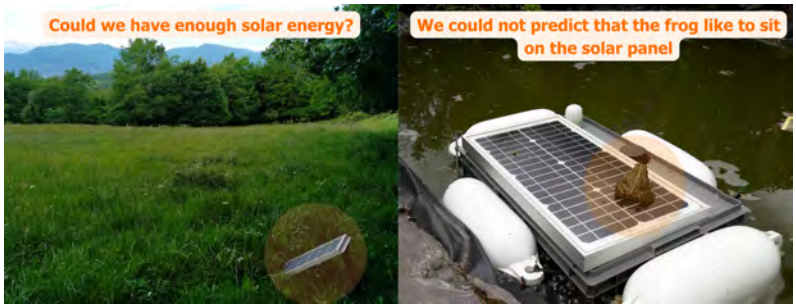


Figure 2: The two problems that we face: how do we estimate the solar energy generated by the solar panels and what could be done to mitigate the effect of unexpected shadows.

This project presents us with two problems. Before deploying the measurement system, we need a **good solar energy estimate** so that a proper sizing of solar panels and batteries could be made. If solar panels are placed in open, clear spaces or we could control the environment optimally similar to large industrial photovoltaic plants, there is not much of a challenge. However, it is likely that they will be deployed in the forest among trees and wildlife activities (i.e. prairie, hedges, forest edge, middle of the forest, in a pond, etc.), which can reduce actual solar availability to the solar panel and complicates the estimation process significantly (Figure 2).

Figure 3: Some example of unexpected shadows. A is mostly a design oversight, while B is unexpected build-ups.

After the system is installed, we must consider the unexpected shadows. This is not limited to low-power solar harvesting and could range from design oversight to static shadows cast by dust and leaves, as shown in Figure 3. We have also shown a funny example of a frog sitting on the solar panel in Figure 2. These shadows are very significant because **even a small shadow on a solar panel could cause a significant power loss**, which we will explore later. However, these are just static shadows, which is not the only problem we have.

There are some more contexts for low-power solar harvesting in which moving shadows could be present. Figure 4A highlights an autonomous solar boat from the Iboat project by Baptiste Genet in collaboration with ISAE [2] where we have a solar powered sailing boat and the sails cast shadows on the solar panels. This

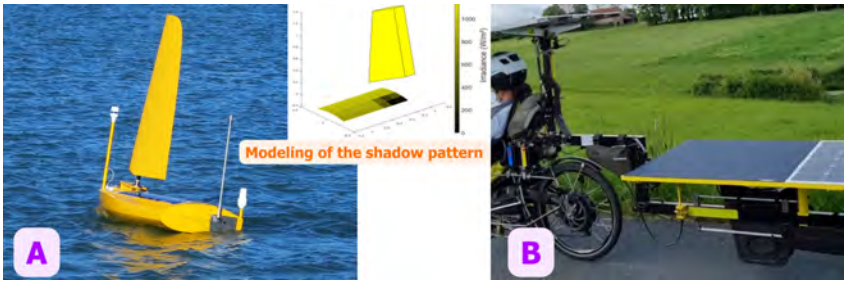


Figure 4: Illustration of some other low-power solar harvesting applications that face moving shadows: a solar powered sailing boat (A) and solar-aided bicycles (B)

alone was already troublesome; they have seen that only two covered solar cells could shutdown the production of the entire array of 32 solar cells. Even if that was rectified, the fact is that the boat is operating in the sea, rocking back and forth, side-to-side, we are still left with the problem of shadows moving in hard-to-predict patterns. In Figure 4B, solar-aided bicycles are shown where we could imagine that mobile shadows from trees and buildings constantly cast shadows, hindering its performance. These examples are intended to highlight the context in which we situate our work. This is important because these partial shading conditions have received significant attention in the literature, but most of the work seems to focus on larger installations where they do not occur constantly nor change continuously.

Most autonomous power supplies would be excessively dimensioned to continuously power the load, but it is impractical to actually deploy an excessive amount of solar harvesting surface and battery capacity. Therefore, designing them just enough could be a major advantage in terms of logistics and financial gain. However, unexpected shadows could then cause power drops leading to intermittent blackouts. Seeing these problems, there is a need to implement a **shading management** technique to ensure that the system is still harvesting the maximum amount of energy possible even under these unexpected shadowing events.

To summarise several problems studied in this thesis, we composed it into three questions as follows:

- How could we estimate the solar availability of the PV system under these challenging situations with relative accuracy?
- What is the impact of the partial shading conditions on the available power of the PV system?
- How could we extract the maximum energy from the system under these unstable irradiance conditions caused by unexpected shadowing events?

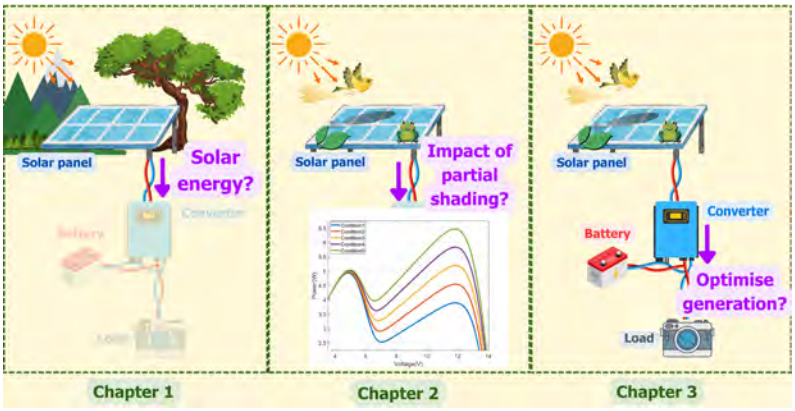


Figure 5: Illustrating the themes of the three chapters of this thesis.

In these frameworks, we propose three chapters dedicated to different problems, as shown in Figure 5. Because this work is intended to solve some problems with the ECONECT project, there will be an overarching theme of trying to keep the end solution as simple and inexpensive as possible for the user, and it is important to keep this lens when considering the various decisions made.

Chapter 1 will be dedicated to answering how we could obtain a **good solar energy estimate** and use this information to preliminary size the solar panels and batteries for a given consumption profile. We start with an evaluation of the existing tools at our disposal and propose a solution that is inexpensive, easy to use, and highly maintainable. We then apply the tool to assess the evolution of the state of charge of a battery in an autonomous power supply, which is a good indicator of whether there is intermittent blackout during operation.

Chapter 2 will be dedicated to analysing the effect of partial shading conditions, where we mathematically model and simulate a **photovoltaic (PV)** system under **partial shading conditions (PSC)**. The study was carried out on a string of four small PV modules with bypass diodes, and each module consists of six solar cells in series. From this we gain important information about the **maximum power point (MPP)** and **global maximum power point (GMPP)** of the system such as how these GMPPs are distributed, the various different operating regimes of the PV array, how the shading patterns impact the power characteristics of the PV array, etc.

Chapter 3 discusses **maximum power point tracking (MPPT)** and **global maximum power point tracking (GMPPT)** based on the strong foundation of Chapter 2 where we eventually optimise photovoltaic generation under fluidly changing partial shadows. This chapter involves a deep dive into the literature of MPPT and GMPPT, and many critiques of the existing techniques, testing methodology, result presentation, etc. We then propose our own very simple and lightweight GMPPT algorithm based on the knowledge gained from Chapter 2, and test our method against three others that are equally light and easy to implement on low-cost

microcontrollers. We also discuss a novel testing methodology to better match the research context, which are unexpected shadows with no clear movement patterns.

Finally, we will provide a general conclusion to the various contributions of this work in relation to the contexts we have identified in this introductory chapter. Furthermore, we discuss how the various results are not simply limited to autonomous low- to medium-power supplies, but also to larger systems.

Chapter 1

Shading forecast for autonomous photovoltaic systems

In renewable energy applications such as solar energy, an omnipresent problem is the discrepancy between when the energy is generated and when it is consumed. For example, our system has to operate 24/7 while the sun is clearly not high in the sky 24/7. It manifests itself in all situations from the electrical reliability of self-consumption solar households [3] where most people work during daylight hours and go home at night to use electricity to solar buses [4]. Without reliable solar energy estimation, we could always put an excessive solar harvesting surface, which is not ecologically friendly [5], or an excessive energy storage capacity, which is also a disaster to the environment [6]. Furthermore, this also comes with logistical and maintenance inconveniences, not to mention a higher cost. All of this justifies the need for a precise and timely estimate of solar energy, and specifically for our application, we need a good **daily solar energy estimation**.

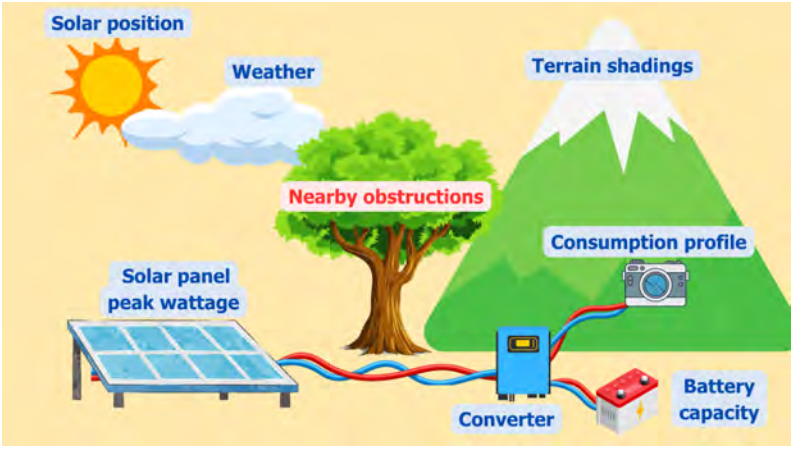


Figure 1.1: Elements impacting solar energy available for autonomous PV systems

Let us take a look at what a solar engineer would do to perform a solar estimation. The different elements to consider are summarised in Figure 1.1. First up on the list is the **solar position** which we can reliably predict in relation to any location on Earth thanks to the countless measurements and observations made throughout

history. We then have the **weather** which is constantly monitored by numerous weather stations and remains relatively stable throughout different years as long as the window of observation is not as limited as hourly. Descending lower to the ground, we have **terrain shadings** such as mountains and hills, information that is well documented in geographical databases. Zooming into our deployment location, the impact of **nearby obstructions** like trees and buildings becomes apparent, which changes from place to place and is relatively difficult to assess. We now arrive at design choices to satisfy the system's needs, like determining the **solar panel's peak wattage** to satisfy the system's need, **battery capacity** to cover periods of low solar availability, choosing the right **converter** for the job, and optimising the **consumption profile** to prioritise critical tasks. These elements are valid for all types of solar installations, from large photovoltaic fields to residential self-consumption to our remote autonomous sensor system.

Among the elements shown in Figure 1.1, **nearby obstructions** will be the focus of this chapter. Specifically, we provide a low-cost, simple, and highly maintainable method for acquiring and integrating the shading data with other information, such as solar position, weather data, etc., to generate a good daily solar energy estimate. We start out with a literature review to see the different tools at our disposal for the job, their advantages and inconveniences, and from there select the toolboxes that could help us achieve our goal. We then describe our proposal for easy shading forecast, how we interpret the data, and how we generate a solar estimate with the shadings considered. We proceed with a presentation of preliminary test results to validate our method. An application case was presented in which we determined the minimum state of charge of the battery using the irradiance compensated with shading and highlighted the importance of accurately estimating the hourly solar irradiation. Finally, we conclude with an overview of how our proposed shading estimation could be used, its merits, and its limitations.

Contents

1.1 Literature review 10
1.1.1 Computer Assisted Design Software 10
1.1.1.1 Complex shading forecast in PVSyst 10
1.1.1.2 Complex shading forecast in PVGIS 11
1.1.2 Digital Surface Models 13
1.1.3 Sky imaging 14
1.1.4 Solar estimation under forest canopies 15
1.2 Solar position relative to our camera 17
1.2.1 Clarification on geometry terms 17
1.2.2 Solar coordinates in our camera's frame of reference 18
1.3 How the camera maps 3D points onto a 2D image 20
1.3.1 The principle of camera calibration 20
1.3.2 Choosing software toolbox for fisheye calibration 22
1.4 Process shading data and generate solar estimate 25
1.4.1 Selecting the software environment and irradiance data . . . 25
1.4.2 Calculating the direct shading factor 26
1.4.3 Calculating the diffuse shading factor 26
1.5 Preliminary solar estimation evaluation 29
1.5.1 Validating the solar position on an image 29
1.5.2 Experimental measurement using a solar panel 29
1.6 Summary of user procedure 33
1.6.1 Taking the sky photo 34
1.6.2 Taking the calibration images 35
1.7 Python script for system energy forecast 37
1.8 Conclusion and perspective of Chapter 1 40

1.1 Literature review

As discussed in the preceding section, local obstruction estimation is key to ensuring the proper function of our autonomous application. We first discuss the three main techniques that are referenced regularly in the literature for solar estimation: 3D Computer-Assisted Design (CAD) programs, Digital Surface Models (DSM) based, and sky imaging techniques. We then proceed with a short discussion on why we should and how we could build our own solar estimation toolbox.

1.1.1 Computer Assisted Design Software

3D CAD techniques have been developed since the mid-1990s, with software like SOMBRERO [7] paving the way for the sophisticated tools that we have today like PVGIS [8], PVSyst [9], PVWatts [10], SolarGIS [11], RETScreen [12], BlueSol [13], HelioScope [14], PVSOL [15] and its free alternative [16], Solarius PV [17], Solar Pro [18], PV F-chart [19], PolySun [20], SAM [21], Homer Grid [22], Archelios [23], and AutoCalSol [24]. They are the backbone of solar design companies, providing a suite of utilities that greatly improve the quality of life for solar engineers to evaluate all of the elements found in Figure 1.1. Nevertheless, while these CAD software allow for high-precision solar engineering, the incurred complexity presents itself as an important inconvenience for those who simply seek to perform a quick estimate. Furthermore, all of them require licencing cost. For a rough estimate, the free online tool provided by the EU Science Hub PVGIS could be a viable option. Milosavljevic et al. [25] published in 2022 an excellent overview and evaluation of 14 solar design solutions including PVGIS even though the latter is not technically a 3D CAD software. Overall, they concluded that most of them are reasonably accurate in estimating the total annual solar energy.

Now, we test two of these tools, PVGIS and PVSyst, to see how nearby shading estimation could be done using them.

1.1.1.1 Complex shading forecast in PVSyst

This software suite is very complete with solar position calculator, weather database integration, commercial solar panels, energy storage solution selector, etc. But we are interested in the complex shading evaluation tool that it provides. There is a simple option to import these data if they are available. If not, the user can build a 3D scene of the area and choose a location to put the solar panels (Figure 1.2). The program then calculates the solar position relative to the solar panel throughout the year, considers the shadows that might be cast, and generates an energy estimate. While the result is accurate, the procedure is complex and does not align well with our goal of a simple shading forecast method.

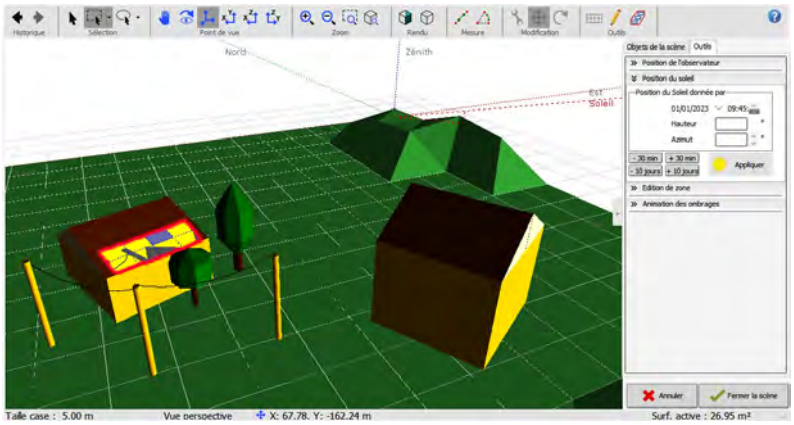


Figure 1.2: PVSyst scene builder utility. The eight solar panels are located on top of the building to the left and the solar position in this example is at 09h45 on 01 January 2023. The blue solar panel is not shaded while those in yellow are partially shaded with the grey area being the shadows.

1.1.1.2 Complex shading forecast in PVGIS

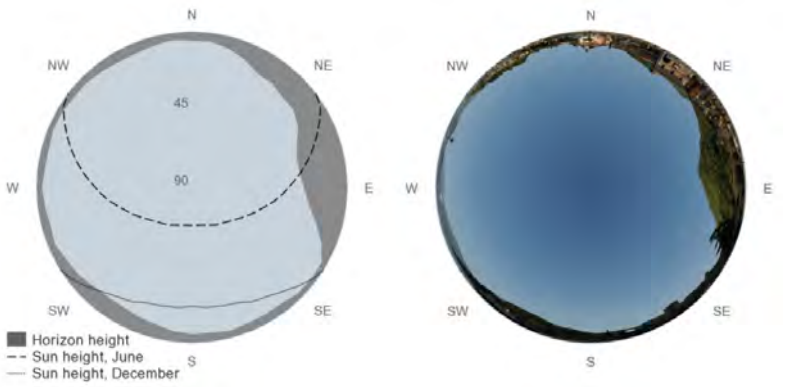


Figure 1.3: Illustration to understand the horizon diagram in PVGIS (Figure from PVGIS documentation [8])

Before going into a complex shading situation, it is important to first understand how PVGIS handles terrain obstruction in general. An illustration can be found in Figure 1.3. Let us first start with the right photo, which is a 180° **field of view (FOV)** of the surrounding area above our deployment location. For example, we could see a small mountain to the east (right side of the photo) and a small town on elevated terrain toward the north and north-east (top and top right of the photo, respectively). We then have the diagram on the left that represents these terrain obstructions in dark grey and the unobstructed sky in light grey. Overlaid on that diagram is also the portion of the sky that the sun traverses throughout the year, delimited by the June solar solstice in dashed lines and December solar solstice in

dotted lines. In the northern hemisphere, the solar trajectory will always be toward the south side of the sky.

This horizon diagram could be modified with custom horizon data to take into account nearby shading. To describe a particular terrain obstacle, PVGIS needs two information: in which direction this object is in (its azimuth) and how high this object is (its elevation). The user could simply upload a list of elevation points and PVGIS will assume equal azimuth difference between these points and that they are given in the order north, east, south, and west (clockwise on the horizon diagram). The elevation will be capped at 90°. To better understand its mechanism, we provide here a simple example in Figure 1.4. In it, we provide eight elevation points to PVGIS. This means that PVGIS assumes that the azimuth difference between them is 45° (because $360°/8 = 45°$). The first elevation point is 0° so we see in the horizon diagram that the elevation of the terrain towards the north is 0°. The next elevation point is 15° at the 45° azimuth clockwise, resulting in the horizon diagram with an elevation of 15° terrain toward the northeast. Continuing to the third point, we have 30° elevation at the 90° azimuth going clockwise, and this is what the horizon diagram shows.

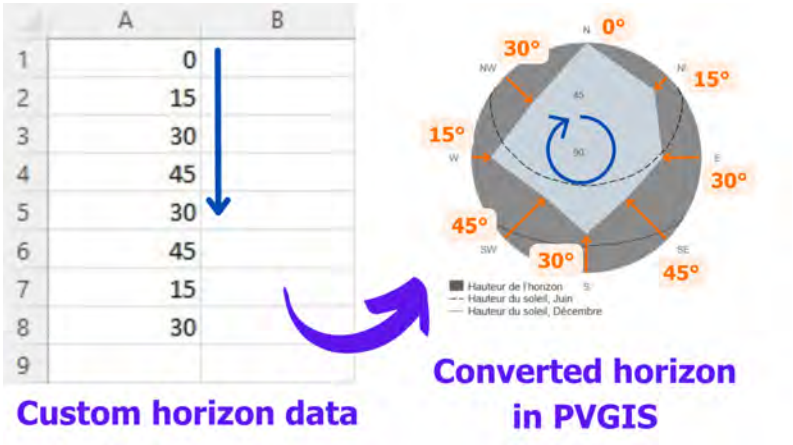


Figure 1.4: Illustration of how PVGIS interprets custom horizon data.

The above description tells us that the custom data will struggle with a situation like that shown in Figure 1.5. Given that only one single elevation point from the horizon is allowed per azimuth direction, the horizon diagram could not represent this tree. Therefore, this custom-horizon tool is not particularly useful to us. However, PVGIS does provide a free and accurate solar irradiance estimation that is very useful.

Figure 1.5: A simple example of a horizon that is impossible to describe using PVGIS horizon tool.

1.1.2 Digital Surface Models

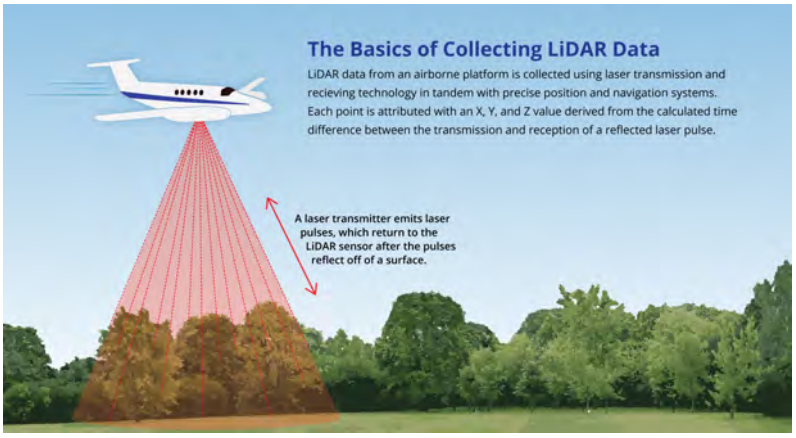


Figure 1.6: Illustration of LiDar technology [26]

If one wishes to avoid scouting the physical location altogether, it is possible to turn to DSM databases. They include several layers with crucial geographic information such as terrain, elevation, vegetation, infrastructure, buildings, etc. [27]. This is done using LiDAR technology (Figure 1.6) which, in a nutshell, means using laser that bounces off the surface to get the terrain details [26]. The authors of [28] used DSM data for remote horizon measurement along with an on-site camera to model their urban environment PV system, resulting in an impressive annual yield precision. Another research work that evaluated the use of DSM in urban environments has found that with available high resolution data, the estimation can be comparable to other methods such as sky imaging, while the result with lower resolution data may be hit or miss [29]. Therefore, the DSM-based strategy should work well in urban environments where high population concentration drives the demand for LiDAR data, not only for photovoltaic installations but also for urban planning. For remote deployments, we are simply out of luck most of the time.

Furthermore, DSM does not yet include the necessary data to evaluate complex shadings such as those cast by vegetation.

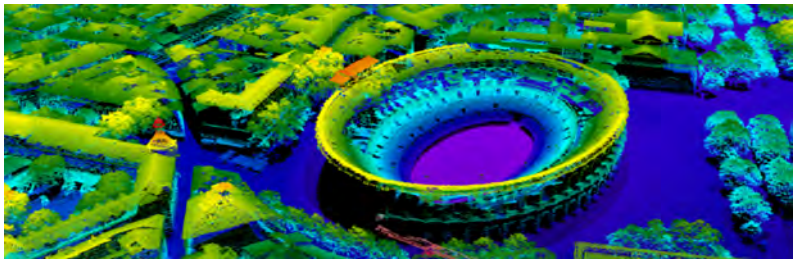


Figure 1.7: Example of high resolution LiDARHD data including complex objects like vegetation and architectural details [30]

Although the evaluations of DSM-based strategies are accurate as of the writing of this thesis, there are advances on the horizon that may drastically change the perspective above. For example, LiDARHD (Figure 1.7) is an open source database that GéoService is updating and should include all complex terrain details in France [30]. If open-access high-resolution LiDAR data would become the norm, it could be the ultimate simple and low-cost shading forecast strategy. With a program that incorporates these terrain data and an irradiance database, a user would only need to click several buttons from the comfort of his office and immediately get a good solar energy estimate.

1.1.3 Sky imaging

This method involves capturing the entire opening above the deployment location using a fisheye lens or a reflective sphere (Figure 1.8) and seeing where there are obstructions. There are several commercial solutions based on this technique, as well as a lot of discussion in the literature, thanks to its ability to determine a complex horizon with relative ease.

Figure 1.8: Reflective dome technique (1) [31] and fisheye camera imaging (2) [32]

Let us first consider some enterprise solutions for shading estimation which can be separated into two main categories: those that have integration with solar design software or weather database and those that stop at a sun path overlay on the

sky image. For those seeking the former category, we have tools like Panorama Master from The Solar Design Company to be used with their proprietary software HorizON [33], Horicatcher to be used with Meteonorm database [34], the all-in-one portable Solmetric SunEye 210 capable of capturing sky images with sun path overlay as well as generating immediate solar energy estimation output [35], and the Solar PathFinder Kit to be used with their proprietary PV Studio 2 [36]. If sun path estimation on sky image suffices, one can consider tools like Pacific Gas and Electric Company's Sunpath Sofware, which works with some specific fisheye camera models [37], or smartphone-based applications like Sun Seeker [38] and SunnyTrack [39]. A 2021 report by Duluk et al. [40] evaluated several of these commercial solutions and provided a fair comparison between them, albeit through an architectural lens rather than a photovoltaic lens. Based on this report and our own research on these solutions, there does not seem to be a solution that is both inexpensive and simple.

Therefore, we turn to academia for potential solutions. Orioli et al. [41] presented a method to accurately plot the sun's position onto an image, but they use a regular camera lens without a wide FOV, leading to the end user needing to take multiple sky photos to capture the entire surroundings. Sanchez-Segura et al. [31] proposed a low-cost sky imaging solution with promising results, but a large reflective metallic dome with a camera pointing toward it is not the most user-friendly tool. The use of inexpensive, hobby-grade fisheye lenses on a smartphone was explored by Oliveira Panao et al. [42] with success, but the steps to determine the camera characteristics are still on the more tedious side. Also using a smartphone, an Android application called Solar Survey proposed by Ranalli et al. [43] was closest to being the best portable shading evaluation tool that integrates imaging with weather data for complete solar estimation, but now it has a lot of bugs and is practically unusable on modern Android devices. In addition to capturing local obstructions, sky imaging also proved its usefulness in real-time irradiance forecast, being used to predict cloud movements overhead as shown in [44], [45], [46] and [47].

1.1.4 Solar estimation under forest canopies

Before proceeding, we should discuss some previous works on solar availability under forest canopies. In general, it is intuitive that estimation is hard due to complex shadings, and this was proven by Hardy et al. [48]. The good news is that even under such complex situations, the solar spectrum is still suitable for Si-based solar cells as shown by Wang et al. [49], and Gunasagaran et al. [50] carried out a detailed measurement of solar availability under forest canopies with promising results. Therefore, our goal to have a good hourly solar estimate for systems under complex shadings such as those found in ECONECT is, in theory, possible.

Another constraint inspired by the ecological project is the user-friendliness of the shading forecast procedure, both in terms of the work to obtain the shading information and in terms of how the program generates the solar estimate, since not

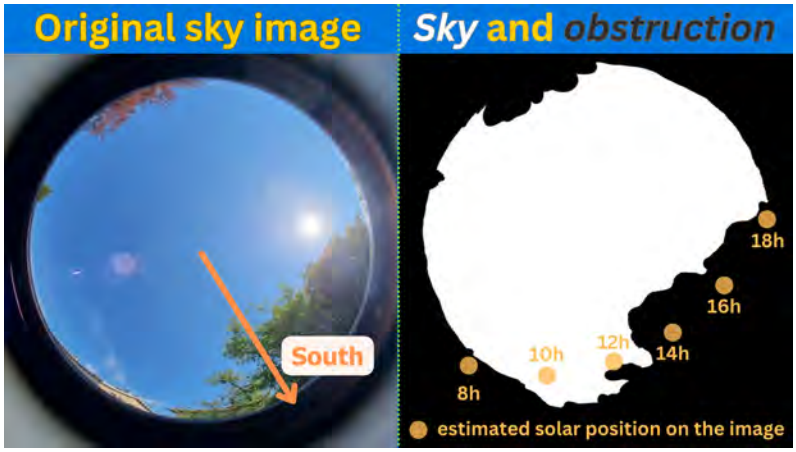


Figure 1.9: Illustration of using the sky image to determine sunlight obstruction. The example shows solar positions typical of March.

everyone working in the project is familiar with the ins and outs of solar engineering. It sums up to three criteria: **low budget**, **low complexity**, and **ease of maintenance**. From the above discussion, there is currently none that ticks all these boxes. Among the three estimation paradigms to choose from, sky imaging is the clear choice for a non-expert audience since no professional-grade tool is required. This is furthermore justified by the fact that inexpensive clip-on fisheye lenses for smartphones have been proven to be a viable option to take a photo in which we can reliably map the solar position onto the image [42]. From these photos, we are going to **map the sun onto the image at every moment in time** and **determine whether the sun is obstructed or not**. This gives us the **shading factor** the solar panel at every moment in time. It tells us how much irradiance is lost because of shadows. This information could then be **incorporated into irradiance data** requested from various databases. For example, if the shading factor at a particular moment in time is 80%, the effective irradiance left is only 20% of what the raw irradiance data would suggest. This should result in a solar energy estimation closer to reality compared to when no shadow was considered.

Two critical pieces of information are required: **the solar position relative to our camera** and **how the camera maps 3D points onto a 2D image**. The next sections discusses these two problems, which are eventually implemented using Python and open-source libraries to avoid licencing costs and to ensure long-term maintainability. Afterwards, we present our experimental setup and results to validate the work. To provide an overview of the process, we then describe the complete procedure that an end user needs to follow for their own shading estimation. Finally, we conclude with the merits, drawbacks, and perspective of the work.

1.2 Solar position relative to our camera

1.2.1 Clarification on geometry terms

To avoid confusion, we start with a summary of the useful geometric concepts for later discussions. A **3D coordinate reference** is a set of three orientated coordinate lines **(x-axis, y-axis, z-axis)** and an **origin O** where the positive direction is represented by an arrow (Figure 1.10). The 2D representation of vectors is also shown in the figure. This reference formed by three coordinate lines $\vec{Ox}$ (x-axis), $\vec{Oy}$ (y-axis), and $\vec{Oz}$ (z-axis) is described as $(O\vec{x}\vec{y}\vec{z})$. To distinguish different references, we add subscripts or superscripts to x, y, and z (e.g., $(O\vec{x_1}\vec{y_1}\vec{z_1})$ is not the same reference as $(O\vec{x_2}\vec{y_2}\vec{z_2})$). Since we only perform rotations and no translation, the origin O will remain unchanged throughout the discussion. A coordinate system is said to be orthonormal when the positive direction of $\vec{Ox}$, $\vec{Oy}$, and $\vec{Oz}$ respects the right-hand rule as shown in Figure 1.10.

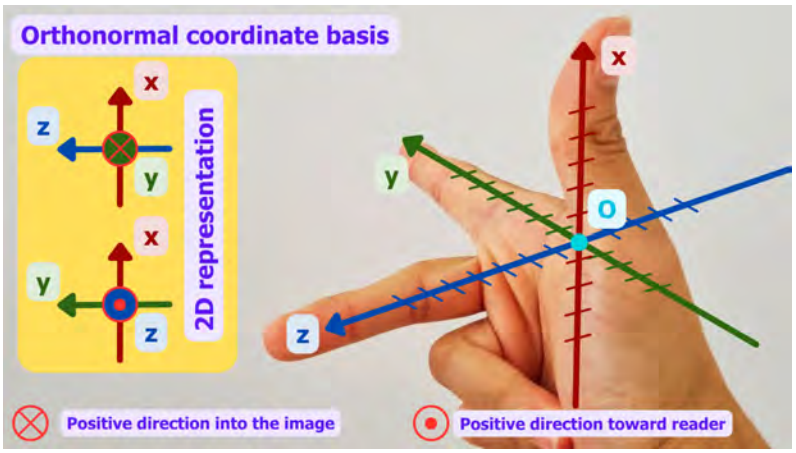


Figure 1.10: Illustration of a $(Oxyz)$ coordinate reference in 3D as well as how it could be represented in 2D.

A point in a 3D coordinate reference could be described using the **3D Cartesian coordinate system**. In this representation, a tuple of three numbers represents its distance from the origin relative to $\vec{Ox}$, $\vec{Oy}$, and $\vec{Oz}$ respectively. The generic representation of the coordinates of point A relative to a reference $(O\vec{x}\vec{y}\vec{z})$ would be (x_A, y_A, z_A) (Figure 1.11). When dealing with extraterrestrial objects, the **spherical coordinate system** is used more frequently. The coordinates of a point in the reference $(O\vec{x}\vec{y}\vec{z})$ are still a tuple of three numbers, but this time they represent its distance to the origin (r), its azimuth (θ), and its zenith (φ) respectively. The generic representation of the spherical coordinates of point A in reference $(O\vec{x}\vec{y}\vec{z})$ would be $(r_A, \theta_A, \varphi_A)$ (Figure 1.11). The distance to the origin is self-explaining; it is the length of the OA segment. The azimuth of point A in the $(O\vec{x}\vec{y}\vec{z})$ reference is the angle formed between $\vec{Ox}$ and the projection of the line OA in the $(O\vec{x}\vec{y})$ plane. The zenith of point A in $(O\vec{x}\vec{y}\vec{z})$ is the angle formed between the OA line

and $\vec{Oz}$. The positive direction of the azimuth and zenith angles can be found in Figure 1.11.

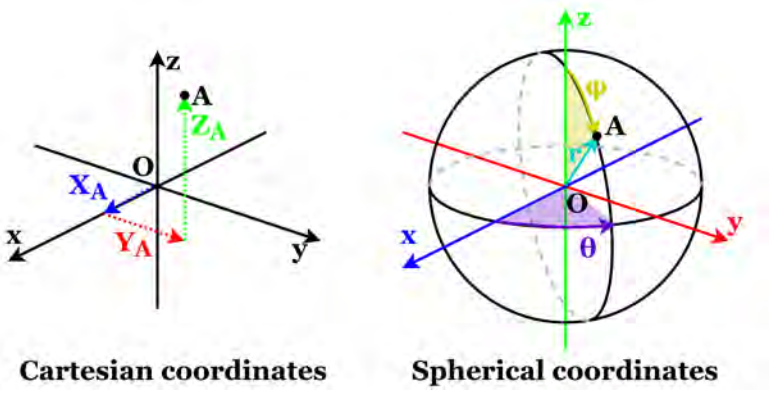


Figure 1.11: Illustration of the 3D Cartesian and spherical coordinates of point A in the reference frame $(Oxyz)$.

1.2.2 Solar coordinates in our camera's frame of reference

The Sun's distance from Earth makes it essentially a point in the sky, which we denote as point S. The objective is to find the coordinates of point S (x_{cs}, y_{cs}, z_{cs}) in the reference $(O\vec{x_c}\vec{y_c}\vec{z_c})$ connected to our camera sensor, as shown in Figure 1.12. This orientation was chosen because it is the de-facto norm used by the majority of camera calibration toolboxes as we will see in the next section. Furthermore, the choice of 3D Cartesian coordinates instead of spherical coordinates is dictated by a software library used in the later section on camera calibration.

Figure 1.12: Visualization of how the camera coordinate reference is oriented relative to the phone's camera.

We used a software library that generates a set of solar azimuths and zeniths $(\theta_{as}, \varphi_{as})$ in reference to $(O\vec{x_a}\vec{y_a}\vec{z_a})$ at each provided timestamp (Figure 1.13 section AstroPy coordinate). Note that the r component is not present because the Sun is so far away that its distance is not relevant for practical purposes. An interesting property arises due to our camera's sensor small size: any point having the same azimuth (θ component) and zenith (φ component) will map to the same pixel on the image. Therefore, an arbitrary r component for S could be chosen

without affecting the final result and it was defaulted to unity. This is the principle behind **projective coordinates** or **homogeneous coordinates**, but we proceed in 3D coordinates to keep the mathematical formulas familiar. Therefore, the solar coordinates in $(O\vec{x_a}\vec{y_a}\vec{z_a})$ are represented as $(1, \theta_{as}, \varphi_{as})$.

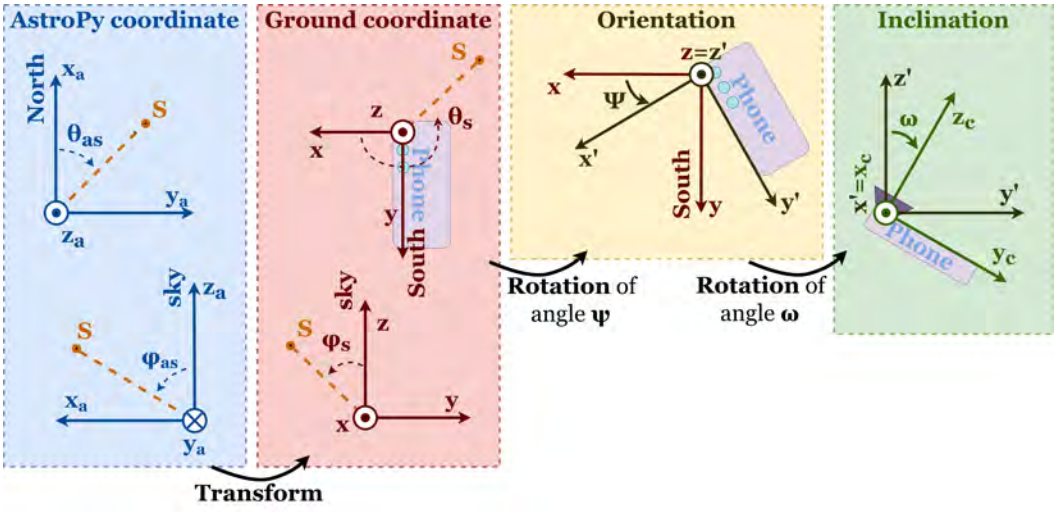


Figure 1.13: The transformations needed to get the solar position in camera's frame of reference.

The first step is to consider a basic photo taking setup with our phone flat to the ground and the South toward the bottom of the image. We call this the ground coordinate reference $(O\vec{x}\vec{y}\vec{z})$ with the y-axis pointing south (Figure 1.13 section Ground coordinate). The solar spherical coordinates in this reference are $(1, \theta_s, \varphi_s)$ and its 3D Cartesian coordinates in this reference are (x_s, y_s, z_s). Between $(1, \theta_{as}, \varphi_{as})$ and $(1, \theta_s, \varphi_s)$ is a simple coordinate transform shown in equation 1.1. We then convert the spherical coordinates to Cartesian coordinates in equation 1.2.

$$\begin{aligned} \theta_s &= \frac{3\pi}{2} - \theta_{as} \\ \varphi_s &= \varphi_{as} \end{aligned} \tag{1.1}$$

$$\begin{aligned} x_s &= sin(\varphi_s)cos(\theta_s) \\ y_s &= sin(\varphi_s)sin(\theta_s) \\ z_s &= cos(\varphi_s) \end{aligned} \tag{1.2}$$

Next, we account for the phone's orientation of an angle ψ where the positive angular direction is from South to East. This is essentially a rotation around the z-axis by ψ and the new reference is denoted $(O\vec{x'}\vec{y'}\vec{z'})$ with the new solar 3D coordinates being (x'_s, y'_s, z'_s) (Figure 1.13 section Orientation). The rotation from (x_s, y_s, z_s) in $(O\vec{x}\vec{y}\vec{z})$ to (x'_s, y'_s, z'_s) in $(O\vec{x'}\vec{y'}\vec{z'})$ is found in equation 1.3.

$$\begin{aligned} x'_s &= cos(\psi)x_s + sin(\psi)y_s \\ y'_s &= -sin(\psi)x_s + cos(\psi)y_s \\ z'_s &= z_s \end{aligned} \tag{1.3}$$

The final rotation transforming $(O\vec{x'}\vec{y'}\vec{z'})$ to $(O\vec{x_c}\vec{y_c}\vec{z_c})$ represents the phone's inclination of angle ω and we reach the desired solar coordinates (x_{cs}, y_{cs}, z_{cs}) in $(O\vec{x_c}\vec{y_c}\vec{z_c})$ (Figure 1.13 section Inclination).

$$\begin{aligned} x_{cs} &= x'_s \\ y_{cs} &= cos(\omega)y'_s - sin(\omega)z'_s \\ z_{cs} &= sin(\omega)y'_s + cos(\omega)z'_s \end{aligned} \tag{1.4}$$

As we shall see in the following section, getting the solar spherical coordinates in $(O\vec{x_c}\vec{y_c}\vec{z_c})$ would seem more intuitive for the task at hand, but we used Cartesian because this is what the software toolbox we chose requires.

1.3 How the camera maps 3D points onto a 2D image

1.3.1 The principle of camera calibration

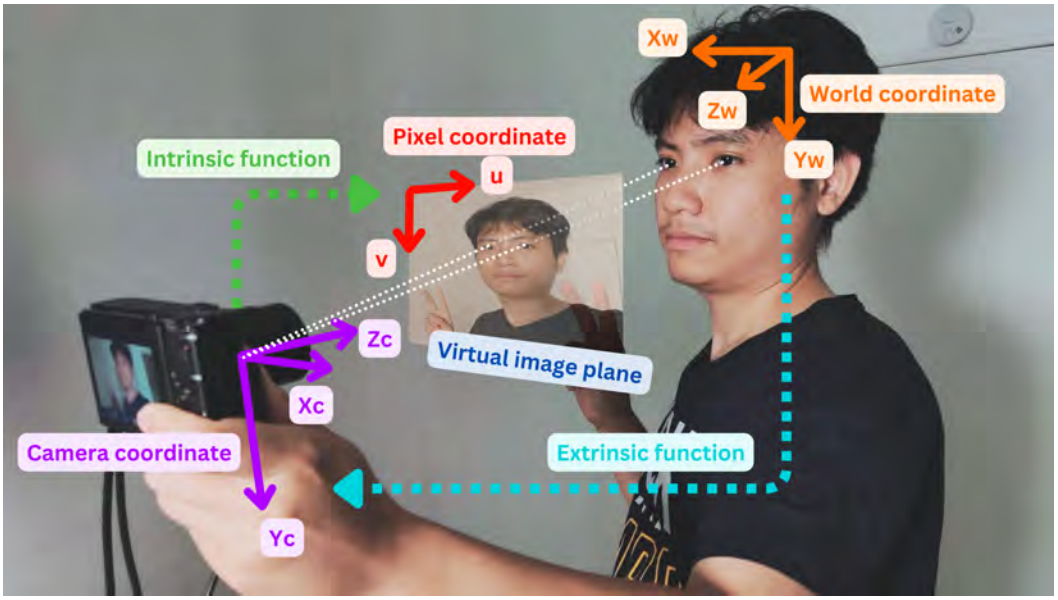


Figure 1.14: Illustration of extrinsic and intrinsic function of a camera model.

Mathematically speaking, a camera is a function that transforms a point in our 3D world into a 2D pixel in our image. We can distinguish two separate steps: an **extrinsic function** $\mathbb{R}^3 \to \mathbb{R}^3$ that maps the coordinates of a 3D world point in the world coordinate reference $(O\vec{x_w}\vec{y_w}\vec{z_w})$ into a set of coordinates in the camera

coordinate reference $(O\vec{x_c}\vec{y_c}\vec{z_c})$ and an **intrinsic function** $\mathbb{R}^3 \to \mathbb{Z}^2$ that maps the latter coordinates into a set of 2D pixel coordinates on an image plane (Figure 1.14). Note that the output of the intrinsic function is in $\mathbb{Z}^2$ because the pixel coordinates must be integers. For us specifically, we need the **intrinsic function** since we knew the solar coordinates in the camera's frame of reference (the extrinsic function for solar position). Obtaining the intrinsic function is called **camera calibration**.

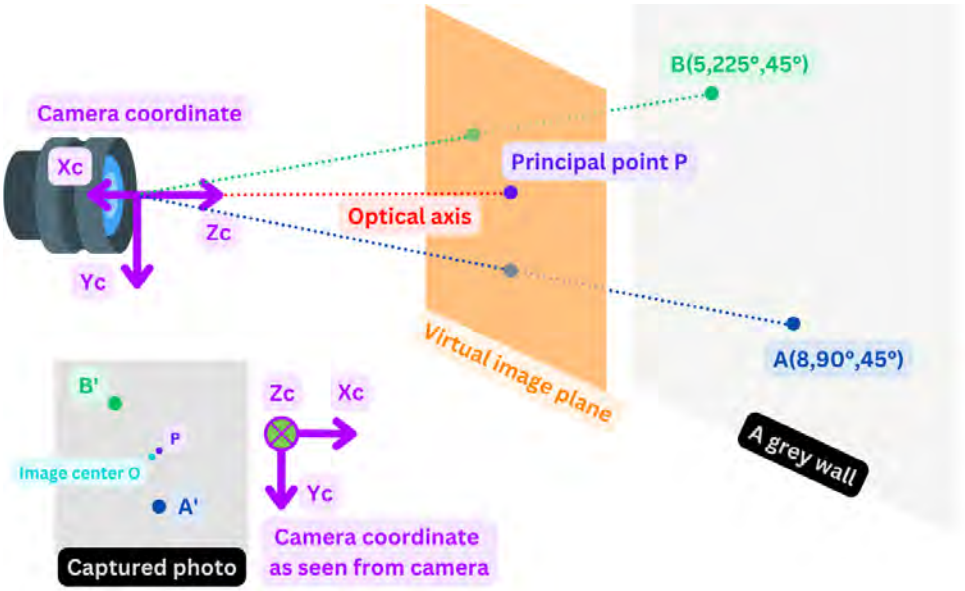


Figure 1.15: Example to show how the intrinsic function works.

We take an example to better understand the mechanism of the intrinsic function. Imagine taking a photo of two points, A and B, on a grey wall as shown in Figure 1.15. The intrinsic function is dependant on the **virtual principal point of the image** which is where the optical axis lands on the image plane. We call it point P in our example. It is important to distinguish the **image center O** (intersection of the photo's diagonals) and this principal point because they do not always overlap. The spherical coordinates of A and B in the camera reference $(O\vec{x_c}\vec{y_c}\vec{z_c})$ are $(5, 90°, 45°)$ and $(8, 225°, 45°)$ respectively and their image on our photo are A' and B' respectively. As discussed previously, the r component does not affect the image of the object so the intrinsic function is independent from $r_A = 5$ and $r_B = 8$. Next, the azimuth of a point dictates which **direction** around the principal point P its image is situated. In our example, point A has an azimuth of $\theta_A = 90°$ and based on how the camera coordinate reference is seen from the camera, we can expect A' to land somewhere directly below the principal point P in the photo which is the case here. Finally, the zenith of a point dictates the **distance** of its image from the principal point P. In our imaginary context, the point A and B both have the same $\varphi_A = \varphi_B = 45°$ zenith, meaning that their distance from point P is the same $PA' = PB'$. Most camera models consider that the distortion around the principal point is negligible (i.e. no significant distortion to the azimuth of

the object in the photo), therefore the bulk of the calibration is about determining the distortion from the principal point (i.e. distortion to the zenith angle). This also shows the importance of getting the highest precision possible on the principal point, otherwise the correction on zenith distortion would not work properly.

The authors of [42] took a photo of an object with known length at a controlled distance and counted the pixels in the image to determine the focal length of the camera. However, this process is tedious and hard to get consistently correct. Therefore, we explored **autocalibration** techniques that are frequently used in machine vision applications. The idea is similar to curve-fitting data where we have a known function with unknown parameters that maps a set of input data to a set of output data. Figure 1.16 illustrates how to obtain the list of 3D world input points and 2D image output points. The input world points are the evenly spaced vertices of a checkerboard and the world coordinates reference is linked to the pattern's surface itself, and the output is their pixel coordinate in the image. To ensure an optimal calibration, it is necessary to take multiple photos to have our set of output image points cover every part of the image's FOV. But this also means that the extrinsic function's parameters will change from photo to photo. The solution used by calibration toolboxes is to iterate between guessing the intrinsic parameters and optimising the extrinsic parameters until the desired remapping precision is reached.

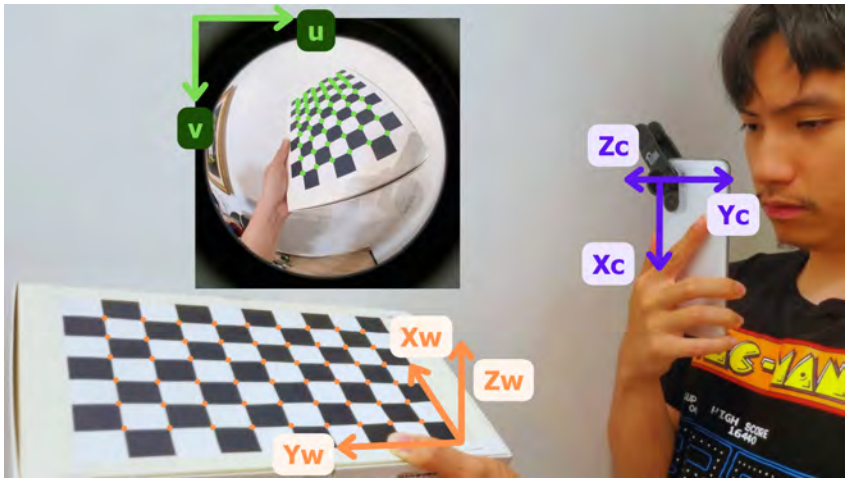


Figure 1.16: Illustration of the input 3D world points (orange) and the output 2D world points (green), with the relevant coordinate systems.

1.3.2 Choosing software toolbox for fisheye calibration

We evaluate three different calibration toolboxes to pick out the best for the job: OpenCV fisheye module [51], OpenCV Contrib Omnidir module [52], and a Python port of a MATLAB toolbox by Scaramuzza [53]. OpenCV is the most widely used library for imaging applications, which guarantees a long-term future. As for the Python port of the MATLAB toolbox, we had started the project out in MATLAB with promising results and wanted to see if there is an equivalent library in Python.

Although these are based on different camera models, their core mechanism is essentially the same. The difference in their results consequently lies in whether the intrinsic function is compatible with the camera's specification used, which in this case is a fisheye camera with an approximate 170° FOV. To compare their calibration accuracy, we first need a photo containing the 2D image of 3D reference points with known zenith angles. We then input this same set of zenith angles into our camera model and "ask" it to draw where it "thinks" the 2D image of these reference points should land in the photo.

The 3D reference points here are the clear angle marks from 0° to 90° at 5° steps of an acrylic protractor (Figure 1.17). The test starts by first taking a good calibration image set and letting the program determine the camera's intrinsic function. We then setup the smartphone vertically to take photos of the protractor. Horizontal alignment is done by levelling the base of the protractor with the fisheye lens. For vertical alignment, we first start with an estimation and then refine the phone's position until the principal point matches the 0° point via subsequent photos. To increase the visibility of the protractor in the evaluation photo, we added red dots to highlight the angle markings.

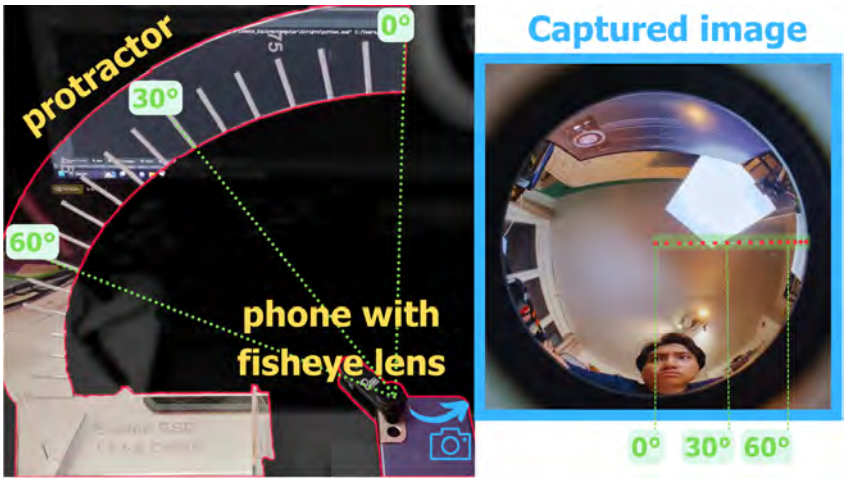


Figure 1.17: Remapping evaluation setup with an acrylic protractor with clear angle markings and the photo of the protractor as taken from the phone. The red dots here are excessively highlighted for higher visibility.

We gave the camera model generated by each module, OpenCV fisheye module, OpenCV Contrib omnidir module, and Scaramuzza's toolbox Python port, our set of known zenith angles which are the set of zenith angles from 0° to 90° at 5° steps. We then drew the camera model's estimation of where 3D objects having the same zeniths in the camera's reference should map onto the image, resulting in equi-zenith circles. The photos of the protractor with the angle estimate overlaid from the three toolboxes can be found in Figure 1.18. We resume the remapping error of each calibration method in Figure 1.19.

Using the fisheye module in OpenCV, the result is relatively precise up to around 45° but the deviation increases to an unusable level beyond that. This is consistent with a lot of discussion in online forums about the limited FOV where it is reliable. As for the OpenCV Contrib Omnidir, the result is wildly out of mark, and we conclude that this is most likely the wrong camera model to use with a fisheye lens. Finally, the Python port of the Scaramuzza OcamlCalib toolbox shows excellent results with a perfectly accurate remapping up to 75° which is the highest zenith visible in the photo. This toolbox will be chosen for our post-processing.

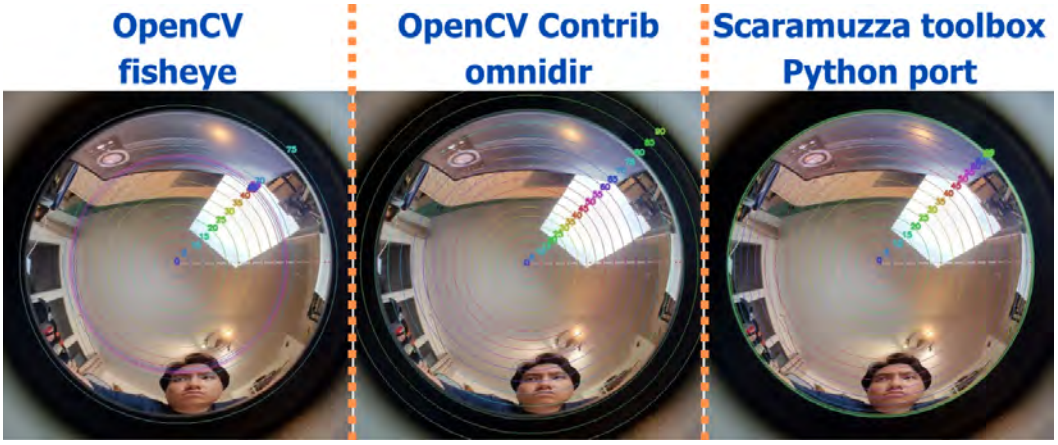


Figure 1.18: Protractor's photo with remapped angles overlaid to evaluate the precision of three calibration libraries tested.

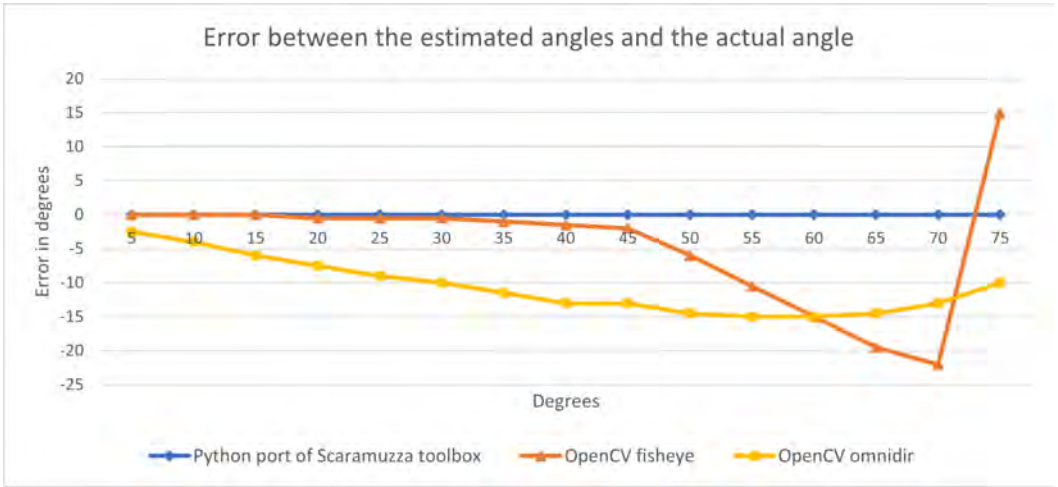


Figure 1.19: Remapping error of the three calibration toolboxes. Notice that the actual zenith angles only goes up to 75° because that is the highest reference zenith angle we could see in the photo.

1.4 Process shading data and generate solar estimate

1.4.1 Selecting the software environment and irradiance data

The language of choice will be Python because it is open-source with a strong community and having a multitude of open-source libraries will ensure the free access to our program, the ease of extension to adapt to different projects, and the long-term maintenance. Furthermore, it is well-known and relatively simple with online courses that could get a user up to speed in less than 20 hours.

Although being free and providing high-quality data as shown by Milosavljevic et al. in [25] was the reason we used PVGIS, this choice was arbitrary because other open-access and high-quality irradiance databases, such as the NASA Power Project [54], are available. Although PVGIS allows one to customise a horizon for solar estimation, it has some limitations, as shown in Section 1.1.1.2. For this reason, we opted to calculate the effective irradiance after shading ourselves. There are three components of irradiance provided by PVGIS: **direct**, **diffuse**, and **reflected** (Figure 1.20). Since most of the time the reflected component's contribution is inconsequential, our Python script only requests the other two. We used the hourly irradiance where each data point represents the average solar irradiance of each hour of the day. Now, we discuss concretely how to obtain the **direct shading factor** and **diffuse shading factor** and use them to compensate the raw direct and diffuse irradiance data, respectively. Note that we assume that the sky photo that the user input has already been transformed to black and white, where white represents the unobstructed sky and black represents the nearby obstructions. Nevertheless, for clarity, we are proceeding with examples using the original sky photo.

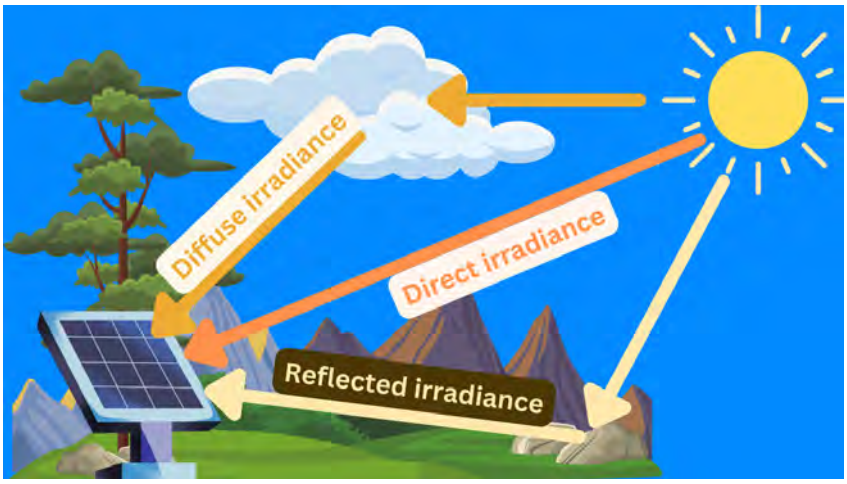


Figure 1.20: Illustration of the solar direct, diffuse, and reflected components.

1.4.2 Calculating the direct shading factor

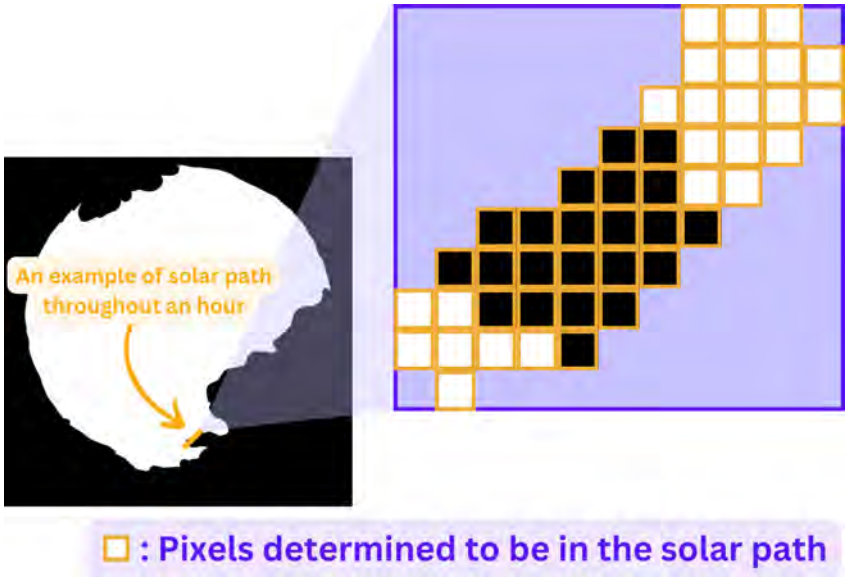


Figure 1.21: Zooming into the solar path traced onto an image to better understand the direct shading factor calculation. In this simplified example, among the 46 pixels that are determined to be representing the solar path throughout a one hour period, we counted 22 black pixels, which means that the average direct shading throughout that period is around $\frac{22}{46} = 47.8\%$

Iterating through the hourly time-stamped irradiance data, we determine the solar position on the image and adjust the estimation with any detected shading. This is done by tracing the solar path in the corresponding hour timeframe onto our image and **count the total pixels in the said solar path**, as well as the **number of black pixels (obstruction)** (illustrated in Figure 1.21). Dividing the number of black pixels by the total number of pixels, we get the **direct shading factor** for that hourly irradiance value. A shading factor of zero means no obstruction to direct sunlight and one means complete obstruction. Then it is used to proportionally compensate the irradiance. For example, for an estimated irradiance of $1000 W m^{-2}$ and a shading factor of 0.3, we get an effective direct irradiance of $700 W m^{-2}$.

1.4.3 Calculating the diffuse shading factor

Regarding the diffuse component, we use the isotropic sky diffuse model by Hottel and Whillier [55] which assumes that the diffuse irradiance comes from the whole sky. This means that **the diffuse shading factor** could be obtained by **dividing the obstruction area over the entire area of the sky**. However, this area is not on a flat surface but on the **hemisphere of the sky**, as illustrated in Figure 1.22.

Figure 1.22: Illustration of how the diffuse shading factor should be calculated properly. We need to consider the area of shading over the total area of the sky on the surface of a sphere, not on a flat image. This image was taken by determining the equivalent azimuth and zenith of each pixel and it is then mapped onto a unit sphere.

As the original photo has optical distortions, the **shaded area in the fisheye photo over the sky area in the fisheye photo** may not be the same as the **shaded area on the sky hemisphere over the sky hemisphere's area**, so proceeding directly with the fisheye image is not possible. But we do know how to map any point in the fisheye onto the sky hemisphere given that the extrinsic and intrinsic functions of the camera are known. So what we now need is the **equal area projection** (also called the **conformal projection**) of the hemisphere of the sky onto a flat image so that **the shaded area of the projected image over the total area of the projected image** is the same as the **shaded area on the hemisphere of the sky over the area of the hemisphere of the sky** (Figure 1.23).

Figure 1.23: Illustration of the transformation from fisheye photo to an equal-area projection of the demi-sphere.

Using this flat image makes determining the diffuse shading factor easier to understand in the software because a matrix form to represent the hemisphere would be bloated. Based on this document by the United States Geological Survey (USGS) on map projections [27], the criterion in equation 1.5 could be used to verify whether a projection of a sphere onto a flat surface is equal area, where u is the vertical position of the object in the projected image, v is the horizontal position of the object in the projected image, φ is the zenith of the projected object on the sphere, θ is the azimuth of the projected object on the sphere, and C is an arbitrary constant. Note that the top left corner of our flat rectangular in Figure 1.23 corresponds to $u = 0$ and $v = 0$ with the positive direction represented by the arrows.

$$\frac{\partial v}{\partial \theta}\frac{\partial u}{\partial \varphi} - \frac{\partial v}{\partial \varphi}\frac{\partial u}{\partial \theta} = C sin(\varphi) \tag{1.5}$$

In Figure 1.23, we used equation 1.6 to project any pair of zenith and azimuth on the sky hemisphere onto our equal area image where k_1 and k_2 are constants.

$$\begin{aligned} v &= k_1(1 - cos(\varphi)) \\ u &= k_2\theta \end{aligned} \tag{1.6}$$

To verify, we calculated all the partial derivatives of equation 1.6 as found in equation 1.5, resulting in the set of equations 1.7.

$$\begin{aligned} \frac{\partial u}{\partial \theta} &= k_2 \\ \frac{\partial v}{\partial \varphi} &= 0 + k_1 sin(\varphi) \\ \frac{\partial u}{\partial \varphi} &= 0 \\ \frac{\partial v}{\partial \theta} &= 0 \end{aligned} \tag{1.7}$$

Finally, we apply the terms from the set of equations 1.7 to equation 1.5 resulting in equation 1.8 which confirms that our projection is indeed equal area.

$$\frac{\partial u}{\partial \theta}\frac{\partial v}{\partial \varphi} - \frac{\partial u}{\partial \varphi}\frac{\partial v}{\partial \theta} = k_2 k_1 sin(\varphi) - 0 = k_2 k_1 sin(\varphi) \tag{1.8}$$

With the help of this projection, by dividing the total number of black pixels in the projected image (the obstructed sky area) over the total number of pixels in the projected image (the total sky area), we obtain the diffuse shading factor. Since the isotropic sky diffuse model does not take into account the solar position, this factor is applied to all diffuse irradiance values regardless of their timestamp.

1.5 Preliminary solar estimation evaluation

1.5.1 Validating the solar position on an image

We present two photos of the sun above our experimental setup to compare with our estimation as shown in Figure 1.24. Throughout the process of taking sky images at different sites, we noticed that it is a bit tricky to get the exact solar position in our image due to measurement error on the phone's orientation. Most azimuth measurements have to be slightly corrected, up to $\pm 3°$, to have a perfect overlap. To account for this in our shading forecast, we increased the size of the circle estimating the solar position in the photo when performing the irradiance compensation.

Figure 1.24: Example of two solar position evaluation photos. The yellow circle marks where our program estimates the sun to be in our image.

1.5.2 Experimental measurement using a solar panel

We chose a location with moderate shading toward the southern section of the sky (original sky image in Figure 1.9) and measured the short-circuit current of a solar panel, which is directly proportional to the irradiance it receives. Our 20W monocrystalline silicon ET-M53620 solar module [56] is slightly inclined at 8° and orientated 138° towards the South-East. It is worth remarking on the trees toward the southern part of the sky. The geographical coordinates of the test are $43°34'47"N, 1°27'47"E$. The test window was from 24 February 2023 to 19 April 2023 and the sampling rate is 10s. We then reconvert this short-circuit current back to equivalent irradiation received using the PV model shown in equation 2.4 which will be properly explained in the next chapter. As a reminder, **irradiance is the solar power per unit area** in Wm^{-2} and **irradiation is the solar energy per unit area** in Whm^{-2}.

The reason why we chose a solar panel over a pyranometer reading is because the solar panel is a large harvesting surface and a point measurement is not necessarily representative of the effective irradiance the panel receives when shadows could be present. **This is also true for the difference between the small sensor of**

a smartphone's camera and the solar panel, which should be taken into account. We provide a result snippet to highlight this problem in Figure 1.25. On the left, we provide a top-down simplified view of the system where we have two pyranometer readings at the two corners of the solar panel and on the right is the reading data from all three measurements. We see that the irradiance received by the solar panel is not exactly consistent with the pyranometer readings.

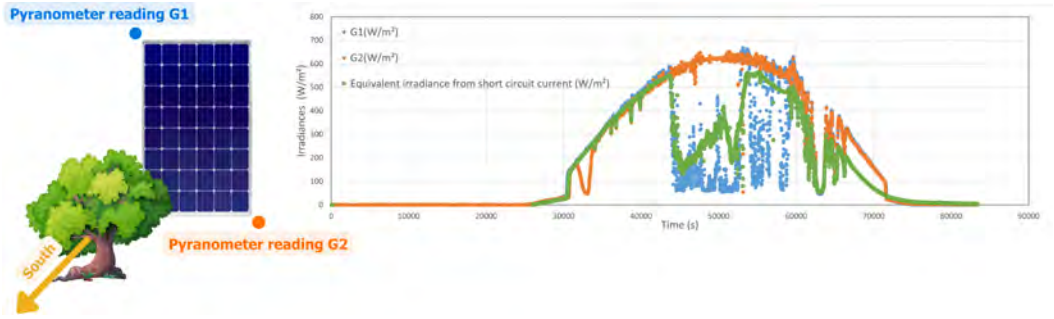


Figure 1.25: Result snippet to highlight the difference between the effective received by a solar panel and pyranometer reading due to the presence of shadows.

Data calibration was performed because the current sense slightly deviates over time. This caused us to sometimes have irradiation values below $0Whm^{-2}$ which does not make sense. For each measurement day, we take the average irradiation between 20h UTC and 04h UTC, which should be $0Whm^{-2}$, and then calibrate the data for that day with this average value. All irradiations below $0Whm^2$ are forced to $0Whm^2$. We provide in Figure 1.26 a snippet of the hourly data from our measurement, raw data from SoDa and data from SoDa after shading compensation.

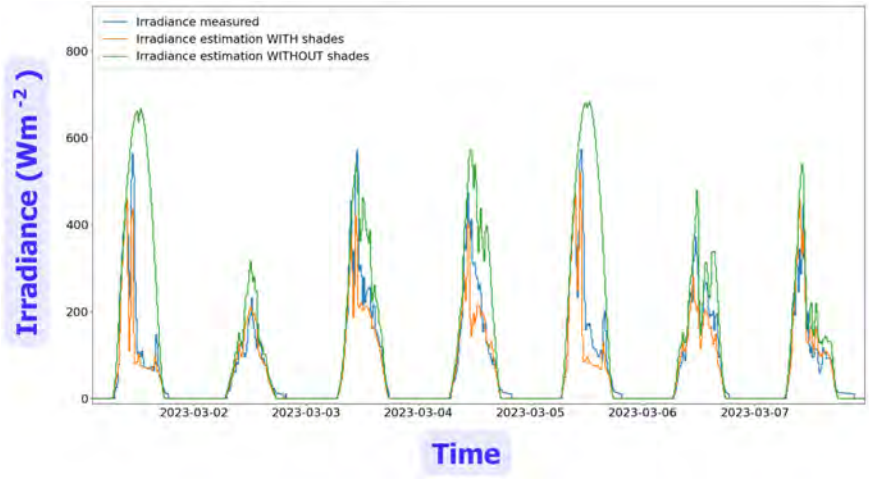


Figure 1.26: Snippet of hourly irradiance data from our measurement, raw data from SoDa and data from SoDa after shading compensation for the period from 01 March 2023 to 07 March 2023. The result is given in irradiance but hourly irradiance is the same as hourly irradiation.

To evaluate the accuracy improvement to solar estimate brought about by the shading estimate, we propose two separate comparisons. First, we need to know whether **our solar measurement is consistent with the irradiation satellite data in the same period** to confirm that our test setup accurately measured the irradiation received. For this, we contacted SoDa [57] and acquired their satellite data for the period from 24 February 2023 to 19 April 2023, which were sampled every 15 minutes. Originally, we wanted to use NASA open access irradiation database that usually has a very short delay on data availability, but this year's data were heavily delayed by more than six months due to technical issues. However, given that any reliable source of irradiation for the same period in 2023 should satisfy our needs, we gladly accepted SoDa's free-of-charge offer. Second, we must evaluate **how an end-user might benefit from our shading compensation tool**, so we compared our measurement with the hourly PVGIS irradiation data taken from the same period from 24 February to 19 April but from 2008 to 2016. PVGIS data was selected because it it was the irradiation database enquired by the current version of the programme.

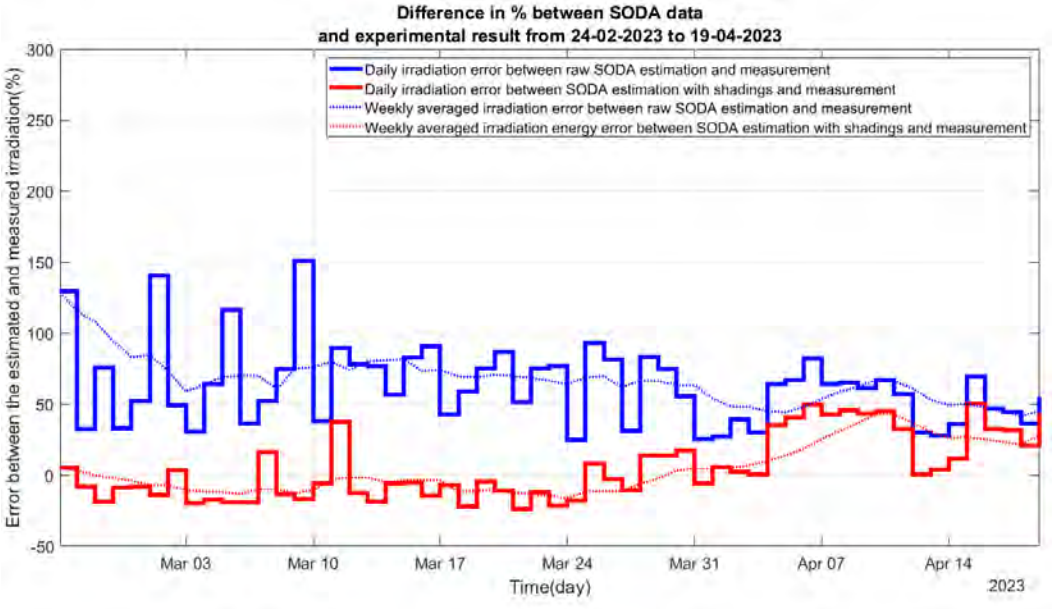


Figure 1.27: Difference between SoDa daily averaged irradiance data, raw and compensated, and our daily averaged measured irradiance.

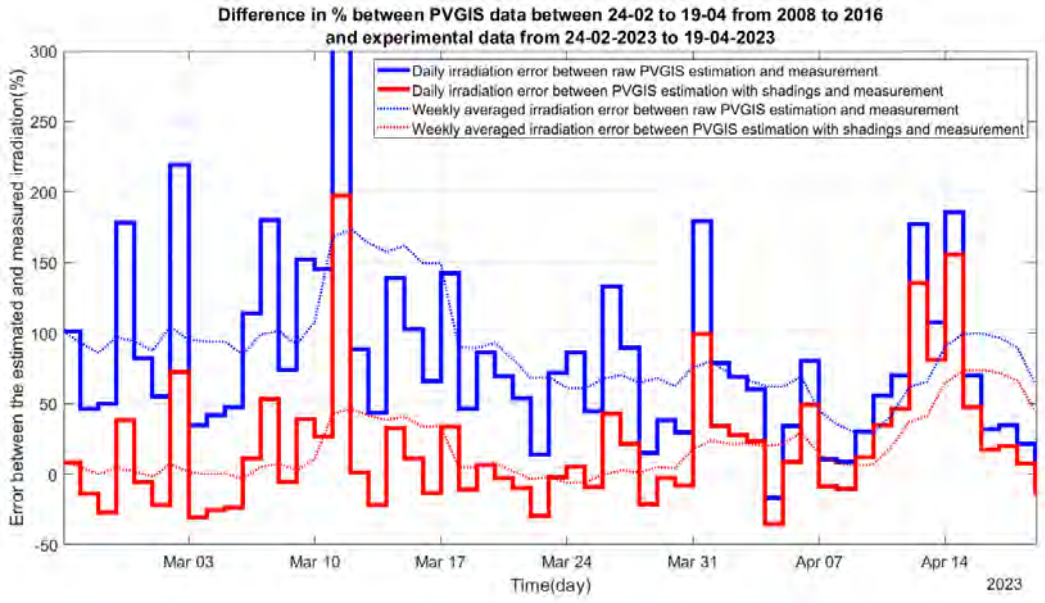


Figure 1.28: Difference between PVGIS daily averaged irradiance data, raw and compensated, and our daily averaged measured irradiance.

Figure 1.27 shows the difference between the SoDa daily irradiation, raw and compensated, and our measurement. We also plotted a rolling weekly average difference to observe the evolution of these two quantities over a longer period of time. Overall, the compensated data seems to better match our measurement, and this is further confirmed by the boxplots the daily deviation. Without compensation (SODA/Mes(NoShade) in Figure 1.29), the daily irradiation from SoDa is on average 62% higher than the daily averaged measured irradiation with a maximum of 151% and a minimum of 24%. After shading was considered (SODA/Mes(+Shade) in Figure 1.29), the median dropped down to -2% with a maximum of 50% and a minimum of -23%. This proves that our shading forecast is relatively precise. The rolling weekly average tells us that the tree mostly covers our solar panels between February and March and the sun path starts to move out of its shadows toward April.

However, this improvement might not be what the user could expect, so we should examine how the PVGIS data compare with our measurement, as shown in Figure 1.28 which plots the difference between daily PVGIS irradiation, raw and compensated, and our measurement. The good news is that the plot shows a significant improvement to the solar estimate accuracy and this is confirmed by visualising the boxplot of this daily deviation. With the raw data (PVGIS/Mes(NoShade) in Figure 1.29), the daily PVGIS irradiation is on average 69% higher than the daily measured irradiation with a maximum of 462% and a minimum of -17%. After shading was considered (PVGIS/Mes(+Shade) in Figure 1.29), the median fell to 8% with a maximum of 197% and a minimum of -35%.

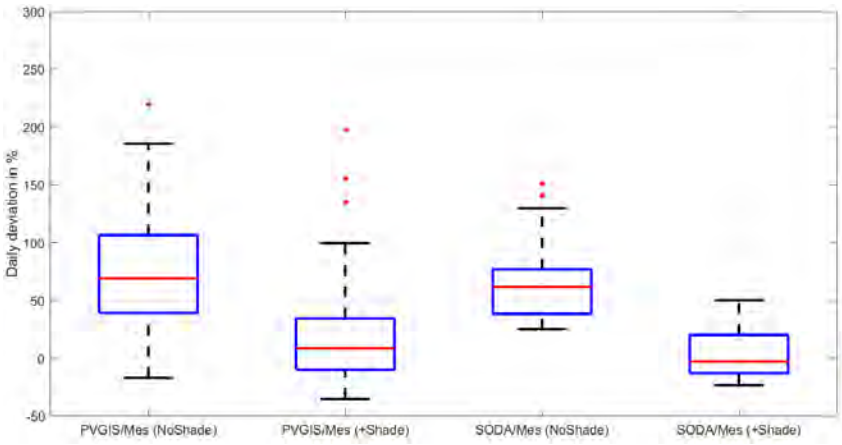


Figure 1.29: Boxplots of the daily averaged irradiance differences.

An overview of Figure 1.29 makes it clear that the shading forecast works as intended with both PVGIS's and SoDa's irradiation estimation converging towards our measurement after the shading information was taken into account. However, we could also see that the deviation between PVGIS data and our measurement has a wider dispersion than the deviation between the SoDa data and our measurement. This is most likely due to the effect of varying weather being more significant in short time windows.

1.6 Summary of user procedure

This section is dedicated to explaining the procedures that a user needs to follow if they wish to use the program. There are only two simple steps to acquire all the information required by the program as shown in Figure 1.30. There is a small final step to run the program that generates the solar energy estimation.

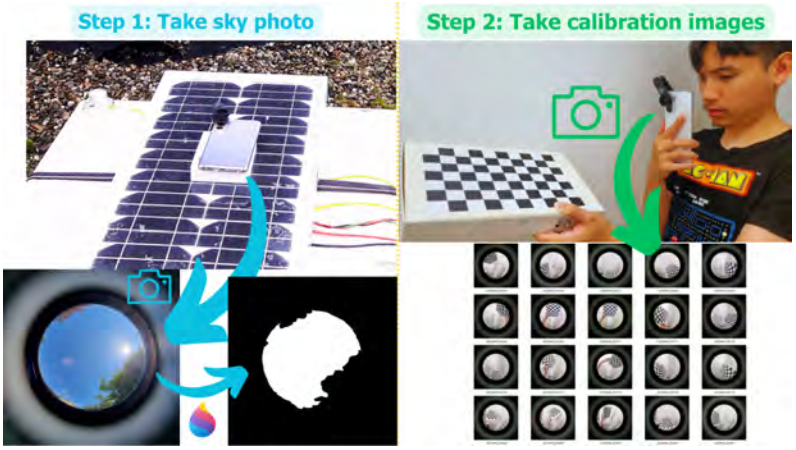


Figure 1.30: Taking sky photo above the potential deployment and take calibration images.

1.6.1 Taking the sky photo

The first step is to capture the sky image at the desired location using the smartphone's camera with a clip-on fisheye lens. It is important to choose a smartphone that can capture the entirety of the fisheye FOV in the image for best result. Furthermore, avoid camera lens with optical image stabilization because its movement compensation perturbs the optical axis and makes correct calibration impossible. It is important to change the setting so that the capture photo is a square because this is required by the program. Finally, the user needs to perform a post-processing with the photo where they have to draw the sky as white and everything else black. This could be done using any paint tool available on their computer like Paint on Windows.

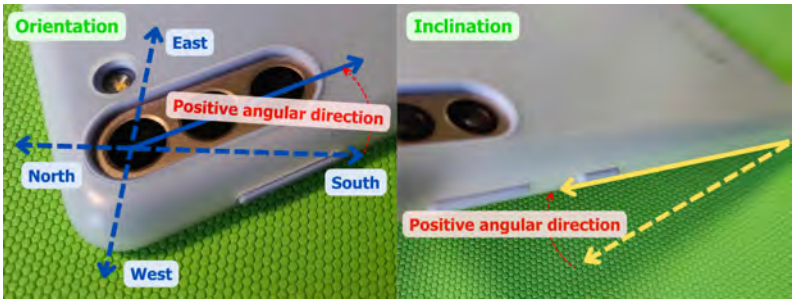


Figure 1.31: Positive angular direction convention for orientation and inclination angles.

Good precision on the phone's orientation and inclination is a must to properly evaluate the solar position in the photo. The positive angular direction convention for these quantities can be found in Figure 1.31. For the orientation of the photo, most smartphones should have a relatively reliable built-in magnetometer that we could use. We need to record which way the bottom of the phone is pointing. When using the compass application, it is important to use **true heading** instead of **magnetic heading** because our chosen library gives solar position in true heading. However, as we have seen in the previous section, getting a correct orientation measurement could be a bit tricky. Since there is a step to calibrate the phone's magnetometer, the user should perform multiple calibrations and measurements to be on the safe side. Then, the inclination of the phone could be determined using the phone's accelerometer. Most likely, there is no built-in app equivalent to a bubble level, but it is easy to acquire a free one via the phone's app store. The photo could be taken at arbitrary orientation and inclination, depending on the need and convenience, but this information has to be recorded to be later used in our solar position calculation procedure. If the fisheye camera is of good quality with an FOV above 140°, it is likely that one photo of the sky above the panel's potential placement will suffice.

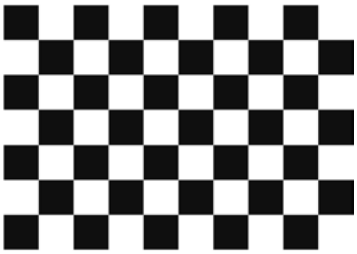


Figure 1.32: Example of a calibration pattern with six by nine vertices.

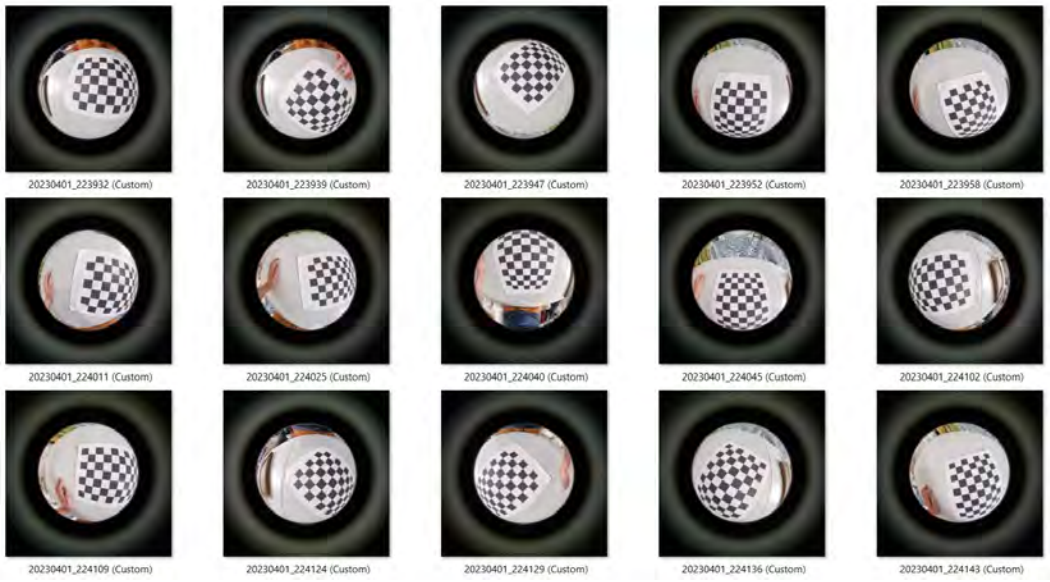
1.6.2 Taking the calibration images

After taking a photo of the site, we proceed to capture the checkerboard pattern to be later used for camera calibration. We used a pattern having six by nine vertices, but it can have any number of vertices as long as the overall calibration pattern is not a square. An example of patterns is found in Figure 1.32. For best result, the photo set must have our pattern covering every visible sector of the image. The number of images is not important as long as the previous requirement is met, but it is further recommended to take several photos of slightly different orientations for a specific pattern orientation so that in case the automatic checkerboard detection fails, the program still has some other samples to try without compromising the complete calibration data coverage. An example of a good and insufficient image set can be found in Figure 1.33.

As for each individual calibration photo, there are some recommendations for optimal result:

- Ensure good lighting on the pattern without flares from harsh light sources or deep shadows.
- Orient the pattern so that all vertices are clearly visible on the image.
- For photos that cover the edge of the image, move the checkerboard so that its outer squares go slightly beyond the FOV while keeping the vertices clearly visible in the image.
- Consider capturing these photos on a clean background like a white wall.
- Verify that the orientation of the photo does not change when preparing the image. This is because most smartphones have the auto-rotate feature, where it detects whether the user is taking the photo in landscape or portrait mode. However, we need to keep the potential lens misalignment the same throughout the data set for an optimal calibration result.

(a) Example of a good calibration image set. All images are properly taken and we have the pattern covers the entire camera's FOV.



(b) Example of an insufficient calibration image set. While each individual image is up to standard, the upper right portion of the FOV does not have pattern coverage. Furthermore, this set contains some photos in landscape and some photos in portrait, making the accurate estimation of the principal point impossible.

Figure 1.33: Example to help taking a good calibration photo set.

1.7 Python script for system energy forecast

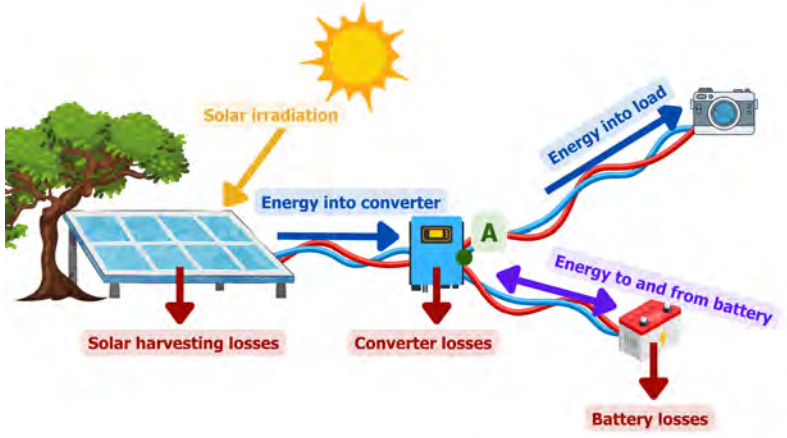


Figure 1.34: A simplified energy flow description of an autonomous solar harvesting energy.

The program outputs an hourly irradiance profile that could be used as is or extended to suit a user need. Here, we provide an example that was implemented in our Python script to generate the estimated hourly state of charge profile of a battery in an autonomous solar harvesting system. Note that there is yet an experimental validation to verify the accuracy of the results generated by this script. Furthermore, the efficiencies such as the solar harvesting efficiency, the converter efficiency, the battery's charge and discharge efficiencies are all assumed to be constant and not dependant on environment parameters like temperature, dust build-up, ageing, etc. The simplified energy flow and the description of the system can be found in Figure 1.34. At each timestamp k (note that this timestamp is, in fact, a duration), the solar panel of size Sm^2 receives a solar irradiation of $R^k_{solar} Whm^{-2}$ and has an efficiency of η_{solar}. This energy is converted by a converter with an efficiency of $\eta_{conversion}$. We could then deduce the energy generated by the converter ${E_{conv}}^k$ in Wh at every timestamp using equation 1.9.

$$ {E_{conv}}^k = \eta_{conversion} \times R^k_{solar} \times S \times \eta_{solar} \tag{1.9} $$

Looking at output node A of the converter found in Figure 1.34, we could see that the energy flows in from the converter ${E_{conv}}^k$ in Wh, the energy ${E_{load}}^k$ in Wh flows out to the load, the energy ${E_{batt}}^k$ in Wh that is positive when it is charging and negative then it is discharging. The battery discharges with efficiency $\eta_{battout}$ and charges with efficiency η_{battin}. The formula for when the converter the energy going toward the battery

$$ {E_{batt}}^k = \begin{cases} \eta_{battin} \times ({E_{conv}}^k - {E_{load}}^k) & \text{if } E^k_{conv} \geq {E_{load}}^k \\ \eta_{battout} \times ({E_{conv}}^k - {E_{load}}^k) & \text{if } E^k_{conv} < {E_{load}}^k \end{cases} \tag{1.10} $$

To estimate the state of charge of the battery SOC^k at each timestamp k, we need the battery capacity C_{batt} in Ah, the nominal voltage of the battery $V_{battnom}$, and the state of charge of the previous time stamp SOC_{k-1}. The battery has a configurable maximum state of charge SOC_{max} and minimum state of charge SOC_{min}. This was done in equation 1.11.

$$SOC^k = \begin{cases} \frac{{E_{batt}}^k}{C_{batt} \times V_{battnom}} + SOC^{k-1} & \text{if } SOC_{max} \geq SOC^k \geq SOC_{min} \\ SOC_{max} & \text{if } SOC_{max} < SOC^k \\ SOC_{min} & \text{if } SOC^k < SOC_{min} \end{cases} \tag{1.11}$$

To show this in action, we provide the result in Figure 1.35. The example is the state of charge of the system battery from 01 February 2015 to 07 February 2015. The irradiation data were taken from PVGIS. The solar panel has a size of $S = 0.2m^2$ and an efficiency of $\eta_{solar} = 18\%$. The converter has an efficiency of $\eta_{conv} = 95\%$. The load consumes 0.5Wh before 18h every day and 1Wh after 18h every day. The battery has a capacity of $C_{batt} = 8Ah$, a nominal voltage of $V_{battnom} = 3.7V$, a discharge efficiency of $\eta_{battout} = 95\%$, a charge efficiency of $\eta_{battin} = 98\%$, a maximum state of charge $SOC_{max} = 90\%$ and a minimum state of charge $SOC_{min} = 10\%$.

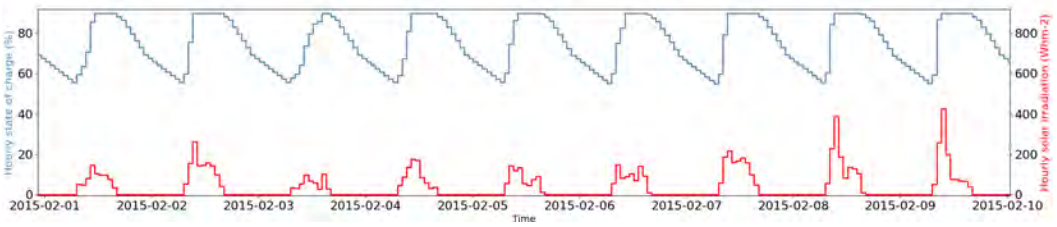


Figure 1.35: The evolution of the state of charge of the battery and the hourly irradiation received by the solar panels for an example from 01 February 2015 to 09 February 2015. The irradiation is the shading compensated irradiance data from PVGIS.

We then graph the daily minimum state of charge for two years 2014 and 2015, with the raw irradiance data from PVGIS and with the data from PVGIS after shading compensation in Figure 1.36. To highlight how shading inclusion impacted daily solar energy availability, we also plotted daily solar irradiation in Wh throughout the period. We notice that the irradiance during winter is much lower after shading compensation, which is consistent with the fact that the trees are toward the south side of the sky. We could also see several dips in the minimum state of charge profile during the period around December 2014 and January 2015 when shading was considered, highlighting the importance of considering the shadows. In this context, the dip is not significant and does not impact the continuous operation, but we could imagine a case where a tighter tolerance (lower solar harvesting surface or lower battery capacity) could lead to a good result when shading is not considered, but a bad result when shading is considered.

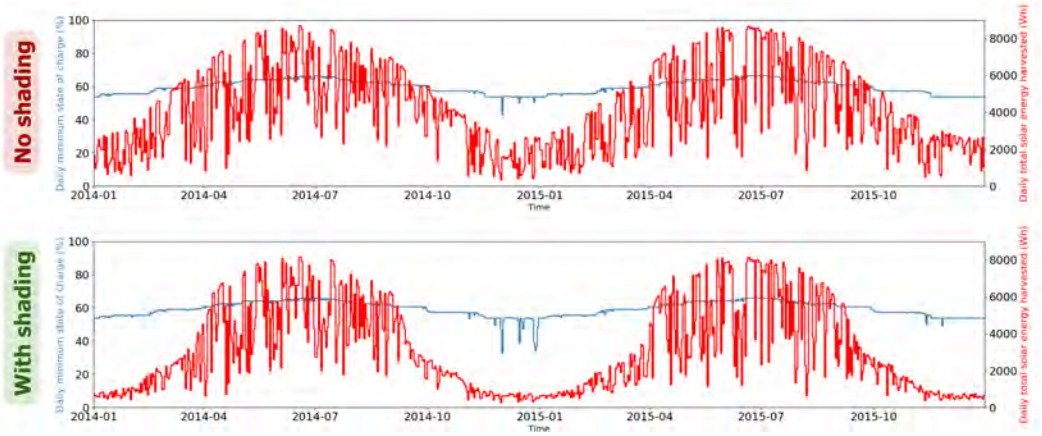


Figure 1.36: The daily minimum state of charge of the battery and daily solar energy available from 2014 to 2015.

Seeing this in action, we could potentially extend the program to provide a hardware recommendation to the user for the appropriate solar panel to use or the correct sizing of their energy storage solution. This could be done by iterating through parameters such as C_{batt}, $V_{battnom}$, Sm^2, η_{solar}, and assessing the state of charge profile to detect whether it reaches minimum. Of course, the user could also configure a charge and discharge limit to increase the life span of the battery if they wish, or add a list of components with pricing and a price limit so that the program could pick out the optimal solution. This is the strength of a highly maintainable and open-source script that should be fully utilised.

1.8 Conclusion and perspective of Chapter 1

In this section, we looked at three paradigms of solar estimation, chosen a set of existing tools, and proposed a solar estimation method. By providing an example of a system energy forecast, we highlight the **importance of accurate hourly solar irradiation**. We have also shown through preliminary testing that our method is **very effective with an improvement in accuracy of more than 50%** (Figure 1.29) and provided a concrete example of how these better solar estimates could be used to evaluate the energy flow of the autonomous power supply. The project could be found open access in this GitHub repository: "github.com/ckbk123/ShadingCompensation". The code source, a packaged executable for Windows, as well as a detailed user guide can all be found in this link. While continued maintenance of the repository could not be guaranteed (update, bug fixes, etc.), we will provide support if possible when contacted via GitHub. This was achieved while satisfying all the constraints set out at the beginning of the chapter as follows:

- **Inexpensive**: the only tools required are an inexpensive clip-on fisheye for smartphone photography and an A4 printed calibration pattern. We assumed that a smartphone and a computer capable of running Python are accessible to everyone.
- **Easy to use**: the user procedure is to take the photo of the sky while recording the phone's orientation and inclination, paint the sky white and the obstructions black in the fisheye photo, take a set of calibration photo, and request the software for the solar estimation within any time frame they wish.
- **Highly maintainable**: the script is in Python, a very intuitive and basic programming language. All libraries and enquired databases are open source.

However, there are still several critiques that could be made about our solar estimation method. Below is a full list of drawbacks and potential improvements that we found throughout the work.

- We had some difficulty in obtaining a reliable measurement of the orientation of the phone using the internal magnetometer. So far, our solution to this is to enlarge the circle that estimates the solar position on the image. But a better and potentially more accurate way would be to introduce an error margin to the orientation where we perform multiple passes of the estimation, but with different orientations. The result is then presented as a solar estimate with a confidence interval.
- The fisheye does not have a perfect 180°, but this is a limitation of the low-cost criteria. A solution would be to spend more on a better fisheye lens. Furthermore, the camera sensor size is tiny compared to the area of a solar panel, so we may need to know before hand where the solar panel would be placed and then take multiple photos to better assess the shading pattern.

- There is an inconvenient step where the user has to paint the sky white and the obstruction black on the acquired fisheye photo. This could be remedied by introducing a function that automatically processes the fisheye photo in Python using machine learning techniques.
- The use of PVGIS irradiance data was an arbitrary choice until we found that the data ended in 2016 if requested via the Python API. In the future, it is better to switch to another up-to-date service, such as the NASA POWER project database.
- We have neglected the reflected component, whose contribution is dependient on the scenery around the solar panel.
- For areas with trees, it works best with evergreen trees, because they do not shed leaves during winter. Furthermore, the script also does not account for risks such as bird droppings, dust build-ups, solar panel's temperature, etc., all of which have a non-zero impact on their power generation. If a statistical analysis of these unforseen circumstances is available, it could be integrated into the script for a more reliable estimation.

Chapter 2

PV simulation for studying GMPPT

This chapter will be the first of two dedicated to discussing the impact of shadows on solar panels, often called **partial shading conditions (PSCs)** in the literature. But before answering why this is necessary, we first need to discuss the basics of solar panels and clarify some nomenclatures.

Monocrystalline silicon cells are by far the most widely used as of 2023, both for industrial and commercial applications, accounting for around 95% of the global production of solar cells [58]. First developed in the 1950s, they are made from single crystal silicon wafers, which, although being more expensive to manufacture than polycrystalline silicon solar cells, make up for this with their superior efficiency [59]. For example, the cells by Maxeon Solar (formerly Sunpower) that we used have an efficiency of around 23%, better than the average 18-20% of polycrystalline silicon cells. The quest for more efficient solar harvesting material has led us to numerous new technologies which are summarised in Figure 2.1.

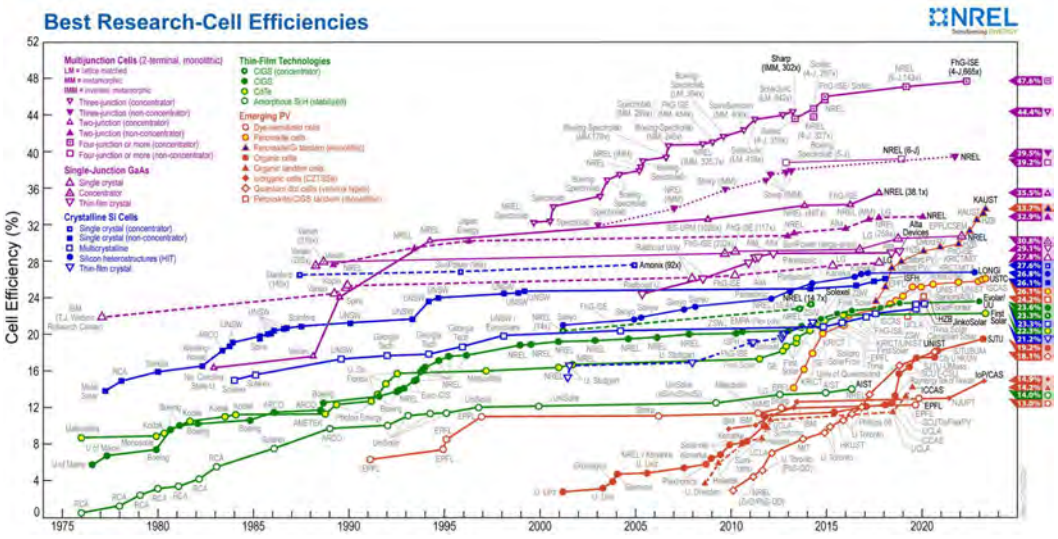


Figure 2.1: NREL's compilation of best research-cells throughout the years [60].

Each solar panel manufacturer provides the set of specifications for their product and to ensure a fair comparison, the industry have a standard test condition (STC) for all measurements which is at $\mathbf{G_{ref} = 1000W.m^{-2}}$ of solar irradiance

on the **AM1.5** solar spectrum at $\mathbf{T_{ref} = 298.15K}$ ($25°C$). The two most basic specifications are the **nominal open-circuit voltage** of the solar panel, $\mathbf{V_{ocn}}$, and its **nominal short-circuit voltage**, $\mathbf{I_{scn}}$. The term nominal clarifies that the values are taken at STC. The next set of characteristics are the **temperature coefficients**, which tell us how the **open circuit voltage** $\mathbf{V_{oc}}$ and **short circuit current** $\mathbf{I_{sc}}$ deviates from nominal values when the temperature is not $298.15K$, given as $\mathbf{K_v}$ in $mV.K^{-1}$ and $\mathbf{K_i}$ in $mA.K^{-1}$, respectively. Finally, we have the voltage and power at the MPP, $\mathbf{V_{mpp}}$ and $\mathbf{P_{mpp}}$, respectively, which provide the solar panel's voltage for optimal power generation and its peak power output. Sometimes, a temperature coefficient for power at MPP in $\%.K^{-1}$ is also provided, which allows for a rough estimation of peak power under non-STC conditions. The electrical characteristics of a PV module are usually visualised with a current over voltage graph (**I-V graph**) or a power over voltage graph (**P-V graph**) (Figure 2.2). In the current profile, we could roughly see three different regions being the large current plateau at voltages below $80\%V_{oc}$, the current roll-off around $80\%V_{oc}$, and the steep current slope toward V_{oc}. With the power plot, we could see the module's V_{mpp} and P_{mpp}, which occur when the current rolls off around $80\%V_{oc}$. It is important not to confuse "P-V", which is an acronym of "**P**ower over **V**oltage graph" and "PV" which is an abbreviation of "**p**hoto**v**oltaic" throughout the discussion in this thesis.

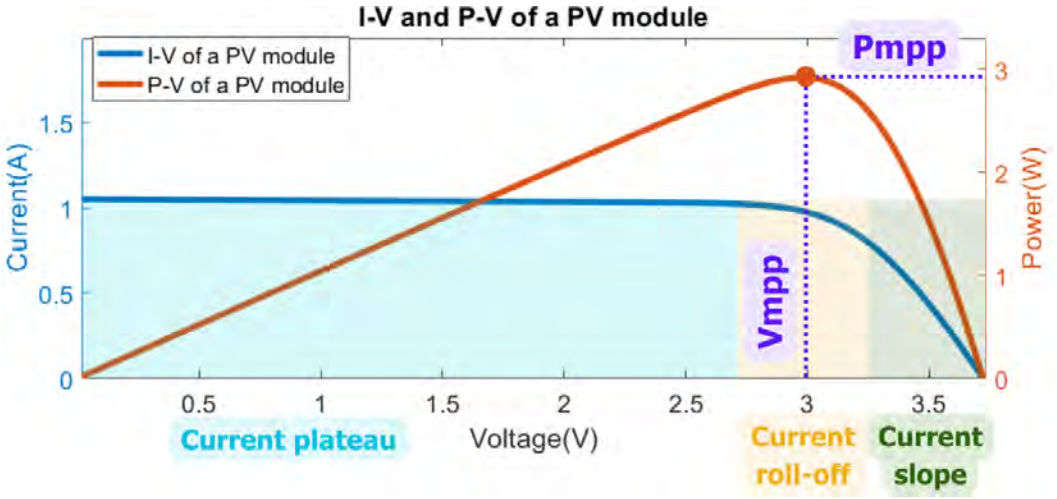


Figure 2.2: I-V and P-V graph of a PV module.

We now discuss the important PV nomenclatures and symbols used in this thesis, summarised in Figure 2.3. The most basic component, a **PV cell**, is a single photovoltaic junction. Next, when multiple cells are connected and encapsulated to isolate them from the weather, we have a **PV module**. When the modules are exclusively connected in series, it is called a **PV string**. Connecting m strings of n PV modules in parallel produces a **PV array** often described as a $nSmP$ array. This prevalent cabling method is known as the **series-parallel (SP)** configuration.

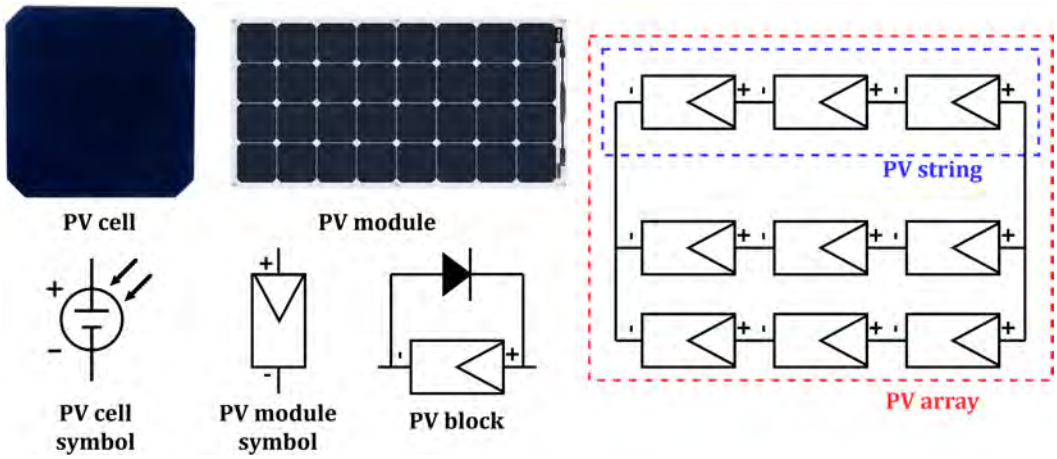


Figure 2.3: A summary of PV terminologies and architectures.

With the basics out of the way, we could now discuss the impact of PSC. When a PV module is shaded, it essentially behaves like a reversed biased diode with high parasitic parallel resistance. If one panel in a string behaves like this, the current generated by the others would lead to hotspots that could destroy the shaded one prematurely [61]. The bypass diode provides a path around the shaded cells and mitigates this problem. We would specifically call any group of PV cells protected by one bypass diode a **PV block**.

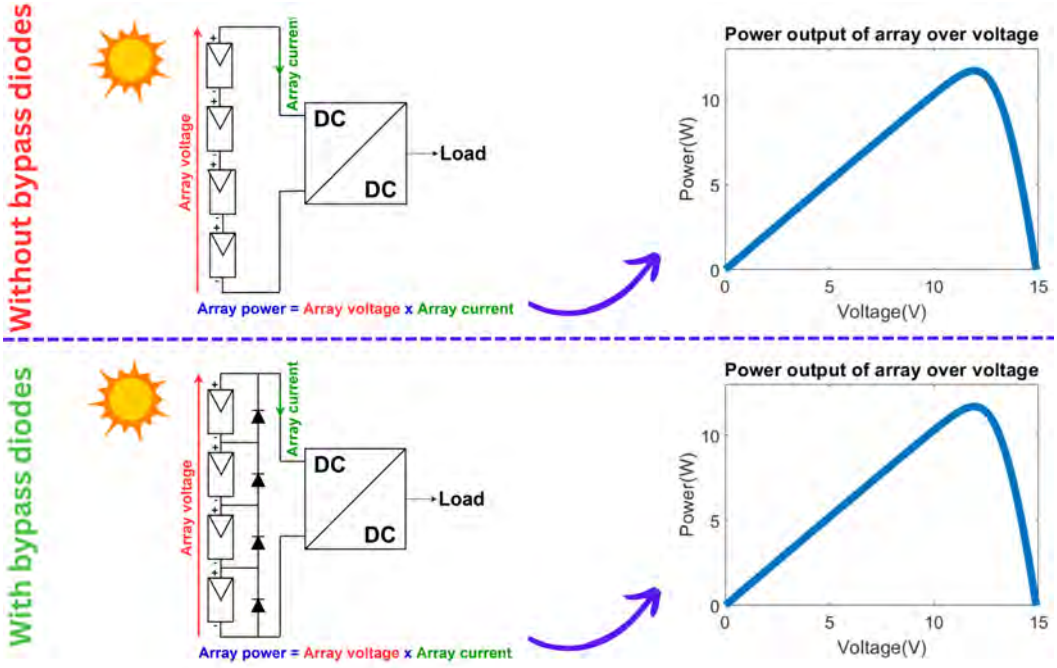


Figure 2.4: Behaviour of solar array, with and without bypass diodes, under even solar irradiance.

Having bypass diodes also added some benefits to power harvesting which we would see by studying the P-V characteristics of two PV arrays: one of four PV modules and one of four PV blocks. Figure 2.4 shows their similar behaviour under even solar irradiance where one MPP is present. But when one module is partially shaded as in Figure 2.5, the array without bypass diodes has a significantly lower maximum power output than its counterpart with bypass diodes. However, the former's P-V curve now has two local maximum power peaks (LMPP), with one being the global maximum power peak (GMPP), making optimal power harvesting more challenging. This is our motivation to propose a lightweight and fast GMPPT algorithm suitable for challenging situations where a lot of PSCs are expected.

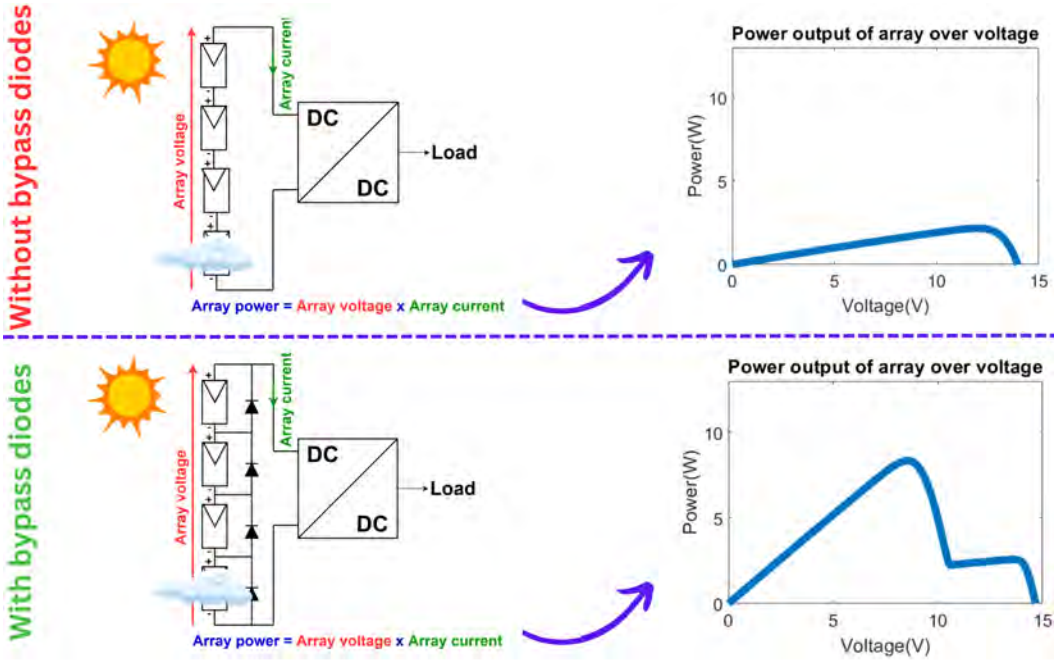


Figure 2.5: Behaviour of solar array, with and without bypass diodes, under partial shading.

This chapter has two main objectives. The first is to **graph the distribution of GMPPs**, which means plotting the occurrence of voltage at GMPP V_{gmpp}. It is assumed in the GMPPT literature (e.g. [62], [63], [64]) that they should occur in distinct zones on the voltage range as illustrated in Figure 2.6, but this was not yet verified by any quantitative analysis. To achieve this, we simulated a PV array under millions of different **irradiance and temperature (G-T)** conditions and determined the GMPP, meaning calculating both the power at GMPP P_{gmpp} and the voltage at GMPP V_{gmpp}, for each of these conditions. However, given the large number of G-T conditions, a section would be dedicated to compute optimisations where we progressively introduce improvements to reduce the runtime of the program and quantify their impacts.

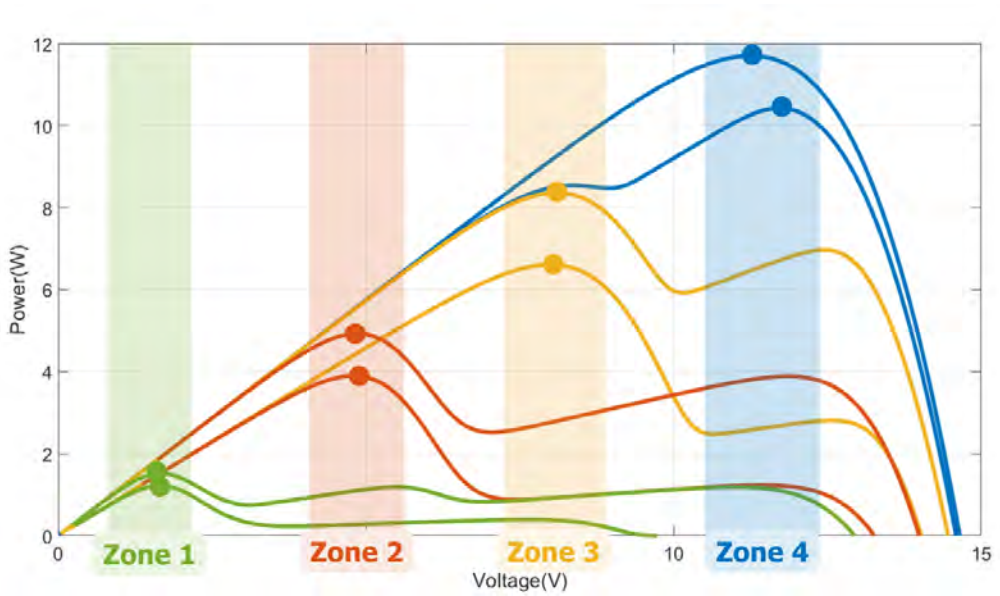


Figure 2.6: P-V curves of eight different irradiance conditions on a string of four PV blocks showing how GMPPs may be distributed in distinct zones on the voltage range. The electrical characteristics of the PV modules used to generate this example graph is found in Figure 2.2

.

The second objective is to build an **electrical simulation of our solar harvesting system** consisting of a PV array with bypass diodes, a buck converter, and a stable voltage load (e.g. Li-ion battery). The work was done on an existing low-cost, low-power hardware from a previous project in the laboratory, since it already matches the target application type. Although we could directly carry out all studies in experimental setups, the software environment provides several advantages. Having a 1000W light source is not very comfortable, and physically replicating the same PSCs is difficult. It also facilitates fine-tuning the algorithm before it is deployed to the microcontroller.

However, before tackling these two objectives, the first section will discuss **modelling a PV array under partial shading conditions**. We first start from the PV cell, then build up to a full PV array, and finally discuss how to determine the model parameters.

Contents

2.1 Modelling a PV array under PSCs 49
2.1.1 PV cell modelling 49
2.1.2 PV module modelling 50
2.1.3 PV block modelling 53
2.1.4 Modelling a string of PV blocks 55
2.1.5 Determining model parameters 56
2.1.5.1 DC parameters of the PV module 56
2.1.5.2 DC parameters of the bypass diode 60
2.2 Graph the distribution of GMPPs 62
2.2.1 Selecting the set of irradiance for the G-T sweep 62
2.2.2 Selecting the set of temperature for the G-T sweep 63
2.2.3 Studying the GMPP distribution from the G-T sweeps 64
2.3 Compute optimisations 67
2.3.1 Two basic G-T sweep scripts 67
2.3.2 Improvement one: Reducing program instructions 70
2.3.3 Improvement two: Vectorisation 72
2.3.4 Improvement three: Look-up table (LUT) 72
2.3.5 Improvement four: Parallel computing 72
2.3.6 Evaluating the execution time impact of the optimisations . . 73
2.4 Electrical simulation of solar harvesting system 74
2.4.1 The basics of software GMPPT algorithm 74
2.4.2 Electrical model of the PV string 76
2.4.2.1 Electrical model of the PV module 76
2.4.2.2 Adding the bypass diodes to model the PV string . 77
2.4.3 Electrical model of the converter board 77
2.4.3.1 Modelling the buck converter section 78
2.4.3.2 Modelling the controller section 80
2.4.4 Simple battery and load model 81
2.4.5 Evaluating the Simulink model 82
2.5 Conclusion and perspective of Chapter 2 84

2.1 Modelling a PV array under partial shading conditions

In this section, our goal is to model a partially shaded array with bypass diodes. We start from the cell, move up to the module, and finally integrate the bypass diode to simulate a PV block. From there, we discuss how to obtain the I-V for a string of these blocks.

2.1.1 PV cell modelling

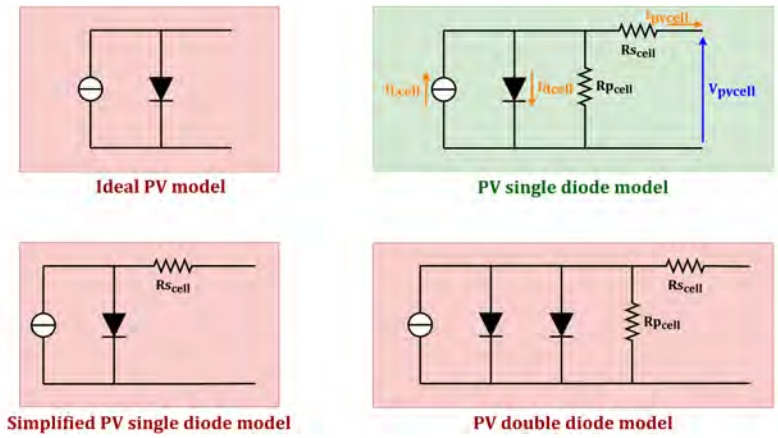


Figure 2.7: A summary of PV cell models.

Among existing works, Villalva et al. [65] seem to provide the most comprehensive approach to photovoltaic modelling. In its most basic form, an ideal solar cell is a current generator in parallel with a single diode (Figure 2.7 Ideal PV model). The equivalent current generator is linearly dependent on the solar irradiance and is influenced by the cell's temperature, whereas the diode is modelled using the Shockley diode equation.

Although the simplicity of the ideal photovoltaic model is a plus, it does not consider the effect of parasitic elements that may be significant when we want to model multiple cells connected together [66]. The practical single-diode model in Figure 2.7 reflects this, introducing a parallel resistance $R_{p_{cell}}$ and a series resistance R_{scell}, to account for imperfections such as the leakage current through the semiconductor or the resistance of the soldering joints connecting them to the bus bars. Its mathematical representation is found in equation 2.1 where $\mathbf{I_{pv_{cell}}}$ and $\mathbf{V_{pv_{cell}}}$ are the current and voltage of the PV cell, respectively, $\mathbf{I_{Lcell}}$ the photocurrent generated by the cell, $\mathbf{I_{0cell}}$ the reverse saturation current of the internal diode formed by the p-n junction in the solar cell, $\mathbf{I_{dcell}}$ the current traversing this diode, $\mathbf{k_B}$ the Boltzmann constant, $\mathbf{A_{cell}}$ the ideality factor of the diode of the cell, $\mathbf{q}$ the elementary charge, $\mathbf{T}$ the cell temperature, and $\mathbf{G}$ is the irradiance received by the cell.

Several authors have also chosen to neglect the effect of the parallel resistance whose value is usually very high, resulting in a simplified single-diode model where only the series resistance R_{scell} is left (Figure 2.7). This simplification is justified because the parallel path is usually very limited due to the nature of the p-n junction and R_{pcell} are usually three to four orders of magnitude higher than R_{scell}. Our equation 2.1 has also partially included this simplification where in the formula for I_{Lcell}, $\frac{R_{scell}+R_{pcell}}{R_{pcell}} I_{scncell}$ was simplified to just $I_{scncell}$ because $\frac{R_{scell}+R_{pcell}}{R_{pcell}} \approx 1$.

$$
\begin{aligned}
I_{pvcell} &= I_{Lcell} - I_{dcell} - \frac{V_{pvcell} + I_{pvcell} R_{scell}}{R_{pcell}} \\
I_{Lcell} &= \frac{G}{G_{ref}} (I_{scncell} + K_{icell}(T - T_{ref})) \\
I_{dcell} &= I_{0cell} (e^{\frac{q(V_{pvcell} + I_{pvcell} R_{scell})}{A_{cell} \cdot k_B \cdot T}} - 1) \\
I_{0cell} &= \frac{I_{scncell} \; + \; K_{icell}(T - T_{ref})}{e^{(\frac{q}{A_{cell} \cdot k_B \cdot T}(V_{ocncell} + K_{vcell}(T - T_{ref})))} - 1}
\end{aligned}
\tag{2.1}
$$

Finally, several works in the literature add an additional diode in parallel to account for carrier recombination losses, making it a two-diode model (Figure 2.7). This model is mostly used for high-precision modelling, most notably in the PV material science community. However, it is difficult to properly determine the parameters needed to simulate a double diode model [67] and it also suffers from high computational complexity without an explicit solution [68].

Our goal is to have a good compromise between complexity and precision. Having a simple model will reduce simulation time, while a "good enough" precision should be sufficient to evaluate GMPPT algorithms in the laboratory setup. Therefore, we chose the single diode model which ticks all of these boxes.

2.1.2 PV module modelling

Figure 2.8: PV module used throughout our work.

To describe a PV module which usually is an array of cells, we need to consider N_s, the number of cells in series, and N_p, the number of strings in parallel. Experimentally, we use this small 3W module for solar hobbyists shown in Figure 2.8 that has six cells in series cut from a single Maxeon solar cell. These modules were used because they were available from a previous project and the existing solar

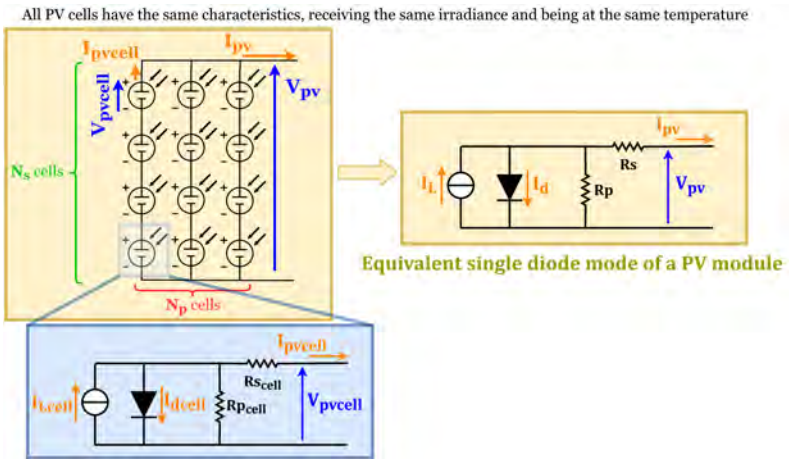


Figure 2.9: A PV module consisting of N_p strings of N_s cells.

harvesting board was designed to work of them in mind. Assuming that each PV cell comprising the array has the characteristics described in equation 2.1 and that they have the same irradiance and temperature (illustrated in Figure 2.9), we have $I_{pv} = N_p I_{pv_{cell}}$ and $V_{pv} = N_s V_{pv_{cell}}$. Applying these basic equalities to equation 2.1, the module's current I_{pv} and voltage V_{pv} could be described as equation 2.2.

$$
\begin{aligned}
I_{pv} &= N_p I_{pv_{cell}} = N_p (I_{Lcell} - I_{dcell} - \frac{\frac{V_{pv}}{N_s} + \frac{I_{pv}}{N_p} R_{scell}}{R_{p_{cell}}}) \\
I_{Lcell} &= \frac{G}{G_{ref}} (I_{scncell} + K_{icell}(T - T_{ref})) \\
I_{dcell} &= I_{0cell}(e^{\frac{q(V_{pv_{cell}} + I_{pv_{cell}} R_{s_{cell}})}{A_{cell} \cdot k_B \cdot T}} - 1) = I_{0cell}(e^{\frac{q(\frac{V_{pv}}{N_s} + \frac{I_{pv}}{N_p} R_{s_{cell}})}{A_{cell} \cdot k_B \cdot T}} - 1) \\
I_{0cell} &= \frac{I_{scncell} + K_{icell}(T - T_{ref})}{e^{(\frac{q}{A_{cell} \cdot k_B \cdot T}(V_{oncell} + K_{vcell}(T - T_{ref})))} - 1}
\end{aligned}
\tag{2.2}
$$

The above equations are not very practical because we cannot characterise each of the encapsulated cells in the solar panel. However, it has been proven that it is possible to use the single diode model to describe this array of PV cells, as shown by Nguyen Ngoc Ban [69] and illustrated in Figure 2.9. To show the mathematical equivalence, we first transform equation 2.2 by distributing the N_p term in I_{pv} to the other equations, and multiply the numerator and denominator of all exponents by N_s. This results in equation 2.3 where we could now see the equivalent photocurrent I_L, the equivalent diode reverse saturation current I_0, and the equivalent diode current I_d.

$$
\begin{aligned}
I_{pv} &= I_L - I_d - \frac{V_{pv} + I_{pv}\frac{N_s}{N_p}R_{scell}}{\frac{N_s}{N_p}R_{pcell}} \\
I_L &= \frac{G}{G_{ref}}(N_p I_{scncell} + N_p K_{icell}(T - T_{ref})) \\
I_d &= I_0(e^{\frac{q(V_{pv}+I_{pv}\frac{N_s}{N_p}R_s)}{A_{cell}.N_s.k_B.T}} - 1) \\
I_0 &= \frac{N_p I_{scncell} + N_p K_{icell}(T - T_{ref})}{e^{(\frac{q}{A_{cell}.N_s.k_B.T}(N_s V_{ocncell} + N_s K_{vcell}(T - T_{ref})))} - 1}
\end{aligned}
\tag{2.3}
$$

Let the module's equivalent series resistance be $R_s = \frac{N_s}{N_p}R_{scell}$, equivalent parallel resistance be $R_p = \frac{N_s}{N_p}R_{pcell}$, equivalent nominal open circuit voltage be $V_{ocn} = N_s V_{ocncell}$, equivalent nominal short circuit current be $I_{scn} = N_p I_{scncell}$, equivalent current temperature coefficient be $K_i = N_p K_{icell}$, equivalent voltage temperature coefficient be $K_v = N_s K_{vcell}$, and equivalent diode ideality factor be $A = N_s A_{cell}$. The transformation gives us equation 2.4 which accurately describe a single diode model in Figure 2.9.

$$
\begin{aligned}
I_{pv} &= I_L - I_d - \frac{V_{pv} + I_{pv}R_s}{R_p} \\
I_L &= \frac{G}{G_{ref}}(I_{scn} + K_i(T - T_{ref})) \\
I_d &= I_0(e^{\frac{q(V_{pv}+I_{pv}R_s)}{AkT}} - 1) \\
I_0 &= \frac{I_{scn} + K_i(T - T_{ref})}{e^{(\frac{q}{AkT}(V_{ocn} + K_v(T - T_{ref})))} - 1}
\end{aligned}
\tag{2.4}
$$

IMPORTANT NOTICE: Using the equation 2.4 to describe a complete PV module is **only valid if all PV cells of the PV module are equally irradiated and have the same temperature**. This assumption will be applied throughout this work.

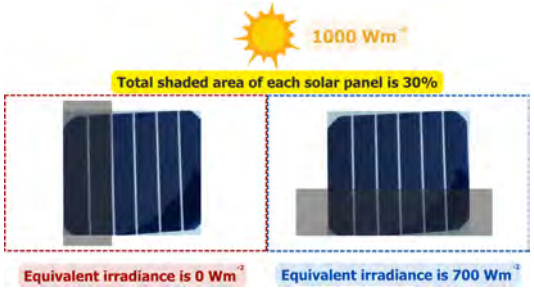


Figure 2.10: Equivalent irradiance received by the PV module when being proportionally shaded the same (30%) but the shading patterns are different.

However, a short discussion of the situations in which a shadow casting proportionally the same area on a solar panel but creating vastly different I-V characteristics is worth considering. Generally, **it is the most shaded PV cell in the string of PV cells that impose the I-V of the string**. In Figure 2.10 we present a situation where a solar panel consisting of six cells in series is shaded by 30% but with different shading patterns. We could see that the pattern on the left completely covers two of its six cells, and as we have discussed, the most shaded cells impose the current of the entire string. Therefore, this solar panel behaves just like a completely shaded panel. This can be seen in Figure 2.11 where we observe that the current of the module is almost zero after having shaded only one cell. However, on the right, we see that the six cells are equally shaded, so it is equivalent to a PV module receiving $700Wm^{-2}$. Therefore, a better way to accurately model would be to analyse the arrangement of the PV cells in a module and the shading pattern, determine the most shaded PV cell, and use its shading factor to apply to the whole PV module.

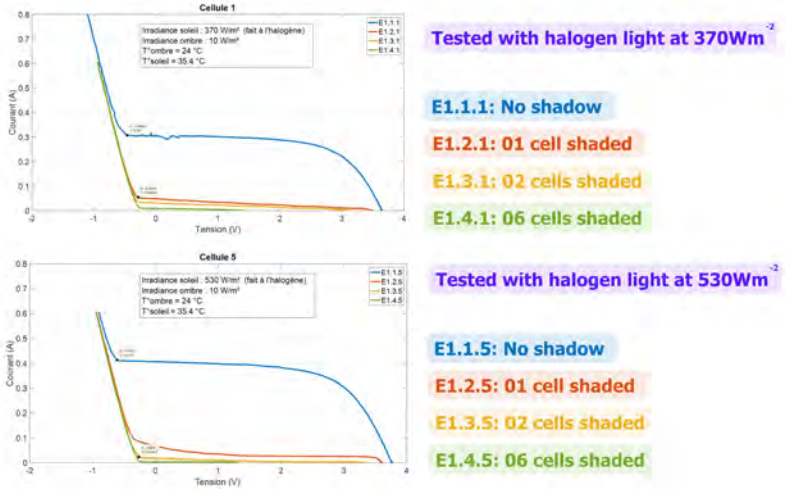


Figure 2.11: Testing result showing how one completely shaded cell among the six in series impacts the power generation of the entire PV module.

2.1.3 PV block modelling

To describe a PV block, the bypass diode is added in parallel to the single diode model as shown in Figure 2.12. To facilitate the discussion, we refer to the set of equations 2.4 using two simplified representations, $I_{pv} = f_m(V_{pv})$ or $V_{pv} = g_m(I_{pv})$. We could then write the current output I_{block} of the block as Equation 2.5.

$$I_{block} = f_m(V_{pv}) + I_{db} \tag{2.5}$$

There are two possible approaches to model a diode. The first is the Shockley equation 2.6 where I_{byp} is the diode's reverse saturation current and A_{byp} its ideality

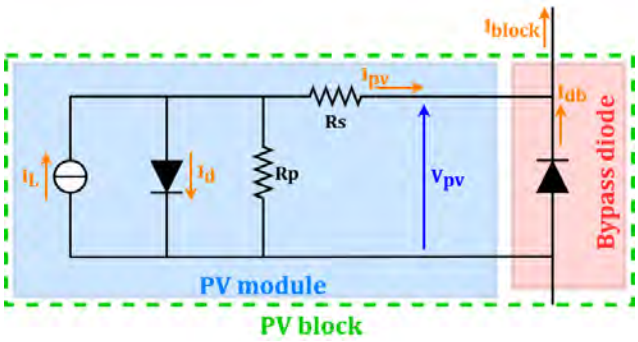


Figure 2.12: PV block model

factor. The second is the piecewise linear equation 2.7 usually found in electrical simulations where V_f is the diode's forward voltage and R_{don} its on-resistance. While there is also the more complex two-diode model, we did not see the higher accuracy to be necessary. Again, for simplicity, we refer to equation 2.5 using two simplified forms, $I_{block} = f(V_{pv})$ and $V_{pv} = g(I_{block})$, in the sections below.

$$I_{db} = I_{byp}(e^{\frac{-qV_{pv}}{A_{byp}KT}} - 1) \tag{2.6}$$

$$I_{db} = \begin{cases} 0 & \text{if } -V_{pv} < V_f \\ \frac{(-V_{pv} - V_f)}{R_{don}} & \text{if } -V_{pv} \geq V_f \end{cases} \tag{2.7}$$

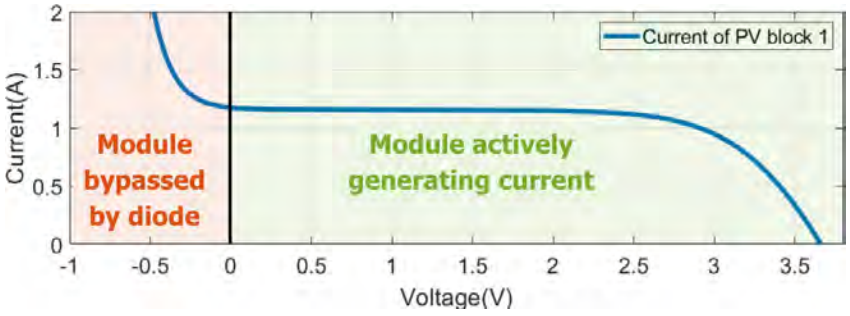


Figure 2.13: I-V of a PV block with its two operating regimes being highlighted.

When we plot the I-V based on equation 2.5, we get Figure 2.13 where we could observe two distinct operating regions of a PV block. When the current passing through it is lower than the module's I_{sc}, the module would be actively generating current and the block's voltage would be positive. However, when the current traversing the PV block is higher than what the module could deliver such as during a partial shading condition, the bypass diode becomes active and the block's voltage becomes slightly negative due to the diode's forward voltage.

2.1.4 Modelling a string of PV blocks

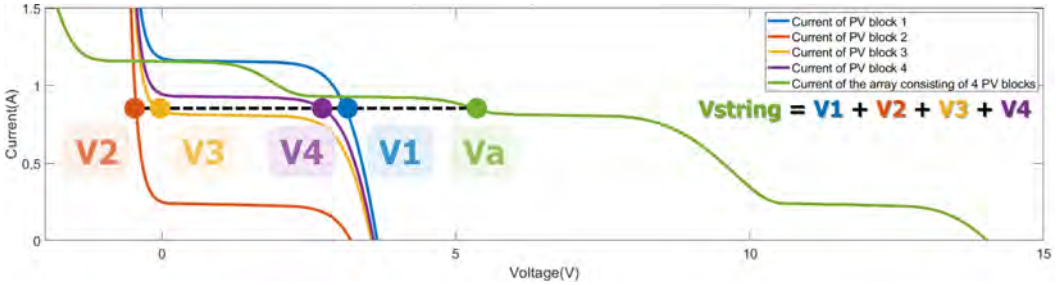


Figure 2.14: Illustration of how the PV string's global current is calculated from the current of its composing PV blocks.

Kirchhoff's current law dictates that the current traversing any electrical components in series is the same, so the voltage V_{string} of a string of n blocks would be the sum of each block's voltage under the same current as shown in equation 2.8 with $g_k(), k \in 1..n$ describing the I-V characteristics of each block k. Figure 2.14 illustrates how the I-V of four PV blocks in series are constructed, a configuration that would be used throughout our thesis. We could also see the two operating regions of a PV block. Above 0V, the module is actively generating current, and below 0V, the module is inactive since the bypass diode is conducting the current generated by other modules. It also shows that the bypass current never exceeds the highest irradiated module's current, which is useful knowledge to select and model the bypass diode. Finally, with the I-V of the string at hand, its P-V could be obtained by simply multiplying the currents and voltages together.

$$V_{string} = \sum_{k=1}^{n} g_k(I_{block}) \tag{2.8}$$

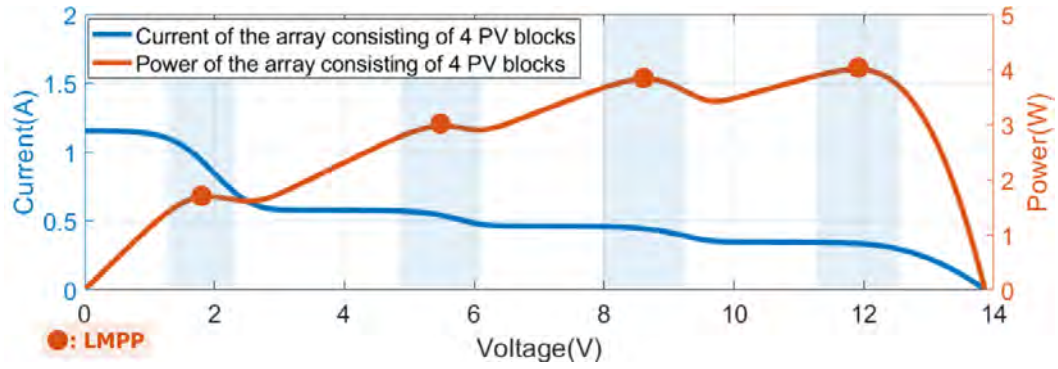


Figure 2.15: I-V and P-V curve of four PV blocks in series to showcase the relationship between LMPPs and the current roll-off regions (highlighted in blue).

Now we can discuss the assumption that GMPPs occur in distinct **regions** on the voltage range found in several GMPPT works such as [62], [63], [64], etc. First, knowing that the MPP of a module is located around its current roll-off region, we could deduce that the LMPPs of several PV blocks in series should be found in its multiple current roll-off regions (Figure 2.15). Since GMPP is just the maximum among the LMPPs, we could conclude that the GMPP of n blocks in series should also occur in these n current roll-off regions. Most of the work in the GMPPT literature estimates that the different **GMP region** i, $i \in 1..n$ counted from low to high voltage, would be around a **GMPP estimate** of $i \times V_{mpp}$ with V_{mpp} being the voltage at the MPP of the PV modules. However, we believe that this is not always valid. For example, in Figure 2.16, we could clearly see that the current roll-off occurs much lower than the expected GMPP estimate $1 \times V_{mpp}$ for region one because there are three active bypass diodes, which reduce the voltage of the only contributing module. Therefore, the GMPP estimate $i \times V_{mpp}$ for the GMPP regions i would only be valid if $(n-1)V_f$ is negligible compared to V_{mpp}. Otherwise, we must consider the effect of the forward voltage of the bypass diode, and a GMPP estimate of $i \times V_{mpp} - (n-i) \times V_f$ would be a better approximation. We will study the accuracy of these GMPP estimates in a later section.

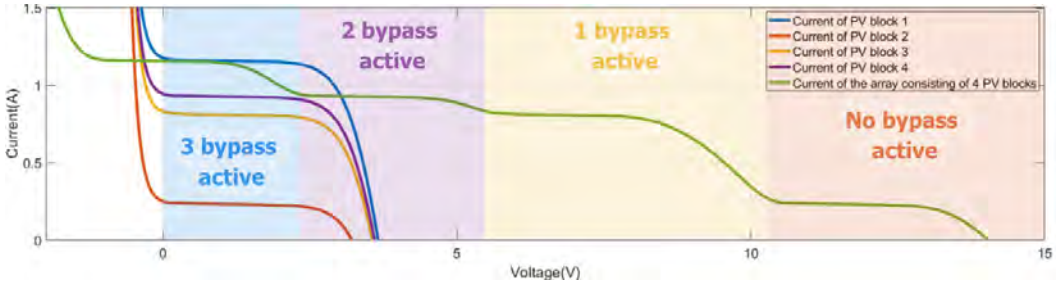


Figure 2.16: I-V of four PV blocks in series with different operating regions highlighted based on the number of active bypass diodes. Each region's color is consistent with which module's MPP is contributing to the array's LMPP in that region.

2.1.5 Determining model parameters

2.1.5.1 DC parameters of the PV module

All symbols in equation 2.4 could be divided into 4 groups as the **variables** (I_{pv}, V_{pv}, G, T), the **constants** (k_B, q, G_{ref}, T_{ref}), the **specifications** (V_{ocn}, I_{scn}, K_v, K_i), and the **modelling parameters** (R_s, R_p, A).

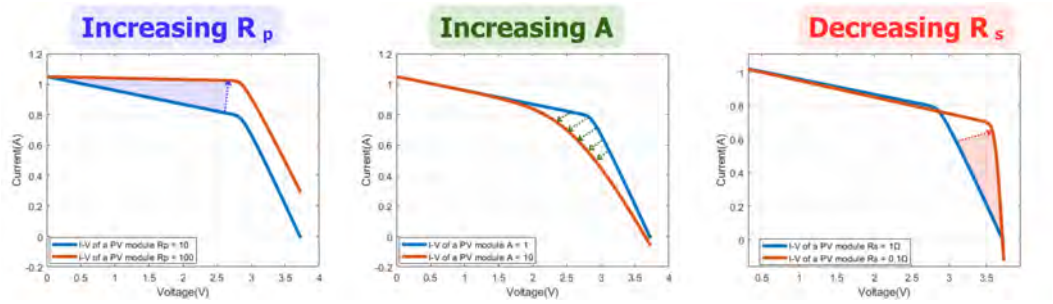


Figure 2.17: Impact of the model parameters R_s, R_p, and A on the I-V of the PV module.

In our research, a module consisting of six PV cells in series cut from a single Maxeon Solar PV cell was used (Figure 2.8), and specifications are easily acquired through the manufacturer's documentation. As for the modelling parameters, these are easily acquired for most commercial solar panels in NREL's SAM database [21]. However, data for our hobby-grade module are not available, so we characterised them ourselves. The procedure involves tracing the module's I-V under a known irradiance and temperature. We used the MP-165 from EKO Instruments to trace the I-V, which has pyranometer and thermocouple inputs to measure solar irradiance and panel temperature, respectively. The tests were conducted by putting the solar panels out on a sunny day during summer 2021 with no clouds overhead and we proceeded to trace the I-V of the module. For more context, the irradiance and temperature received by panels one to four are $967.26 Wm^{-2}$ and $32.6°C$, $967.02 Wm^{-2}$ and $32.2°C$, $966.78 Wm^{-2}$ and $32.4°C$, $966.78 Wm^{-2}$ and $33.1°C$, respectively. While the the measurement condition were not properly controlled, these parameters are sufficient to simulate the P-V of the modules to compare with the experimental results. Next, we progressively fine-tuned the parameters based on the effect of each component on the I-V profile which are summarised in Figure 2.17. If a more analytical approach is preferred, a good reference to consider is Cotfas et al. [67] where the authors compiled 34 different ways to determine the DC parameters of solar panels in detail. All identified numerical values of our PV module are summarised in Table 2.1.

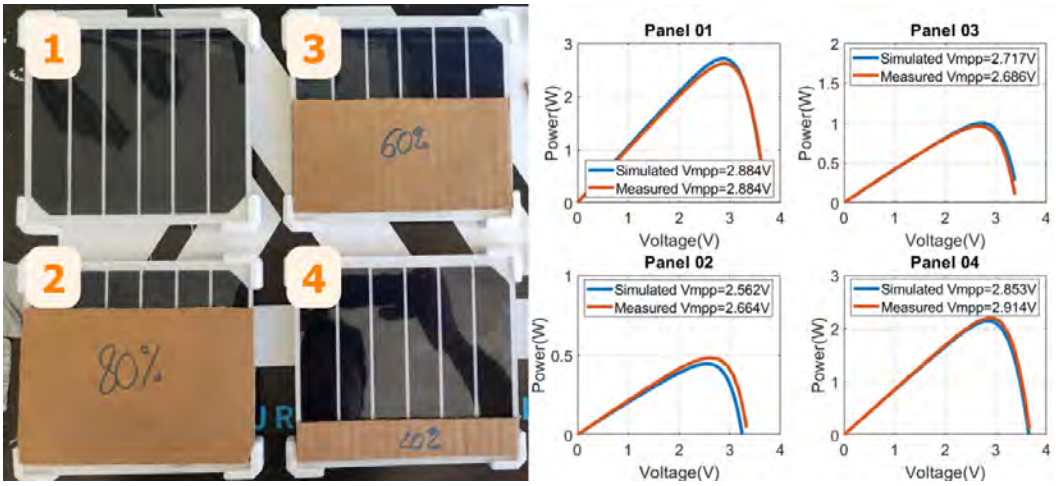


Figure 2.18: Measurement configuration of four solar panels and their measured P-V curves and V_{mpp} compared to simulation result.

In MPPT research context, a low V_{mpp} error as well as a good correlation between the measured and simulated P-V curves are desirable. Furthermore, it is important to verify that the model works under partial shading. Therefore, we traced the P-V of four identical solar panels under the same global irradiance and temperature, with three of them artificially shaded by 80%, 60%, and 20%. It is important to point out that we have intentionally placed the cardboard so that all individual cells are equally covered. This is because using the single diode model requires equal irradiance and temperature for all composing PV cells, and this assumption would be applied to our work. If they are unequally shaded, the resulting P-V profile would be much more complex, as shown in [70].

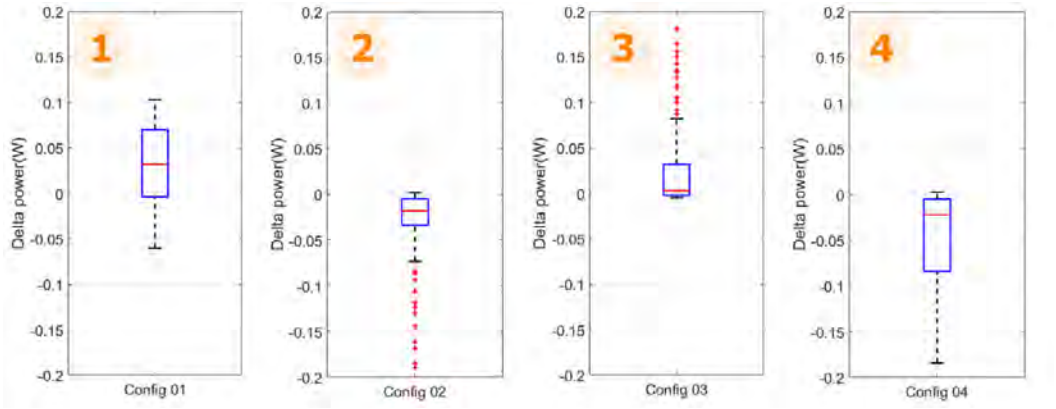


Figure 2.19: Dispersion of difference between the measured power minus the simulated power at each voltage of four solar panels.

The test configuration as well as each panel's measured and simulated P-V plots are shown in Figure 2.18. To quantify the visual difference, we calculated the

correlation factor of each dataset and obtained 0.9995 for configuration one, 0.9598 for configuration two, 0.9928 for configuration three, and 0.9975 for configuration four. Overall, the model provides a very good estimate. We also determined the relative error on V_{mpp} and found that it never exceeds 3.8%. Next, we graphed the dispersion of absolute power error between the measured and simulated results in Figure 2.19. While the median errors are mostly satisfactory with the absolute value never exceeding 4mW, the dispersion is relatively wide and there are a lot of outliers in test configurations two and three. Overall, despite these imperfections, these parameters would be accurate enough for studying GMPPT.

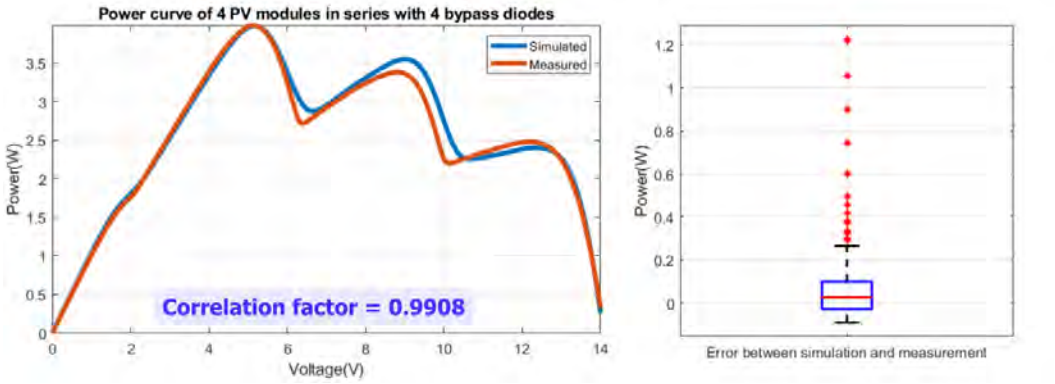


Figure 2.20: P-V graph of the string of four PV blocks where four modules are shaded like in the test configuration in Figure 2.18. The characteristics of the bypass diodes would be discussed in the following section.

Finally, we provided in Figure 2.20 the P-V graph obtained when we characterise the string of four PV blocks shaded similarly to the test conditions in Figure 2.18. The test was carried out similarly to the test with four individual solar panels, where the system is placed outside on a sunny day with no cloud overhead, and we measured the irradiance at $961.03 W m^{-2}$ when the measurement was taken. As for their temperature, we leave the four solar panels without the shading cardboards for sometimes out under the sun so that their temperature reaches equilibrium (they all reached $46.6°C$). Then, we put on the cardboard shadings and quickly performed the measurement so that their temperature have not varied. Overall, although we did not achieve a perfect match between simulation and reality, this result still confirms that we could model the string of four PV blocks with an acceptable accuracy.

Parameter	Value	Unit
V_{ocn}	3.725	V
I_{scn}	1.05	A
V_{mpp}	3	V
I_{mpp}	0.98	A
K_v	-11	mVK^{-1}
K_i	3	mAK^{-1}
R_p	1200	Ω
R_s	0.2	Ω
A	9.5	Unitless

Table 2.1: Summary of our PV module's DC parameters.

2.1.5.2 DC parameters of the bypass diode

Parameter	Value	Unit
I_{byp}	0.0076	A
A_{byp}	3.38	Unitless
V_f	0.26	V
R_{don}	0.18	Ω

Table 2.2: Summary of our bypass diode's DC parameters.

This section would discuss how to determine the bypass diode's parameters for both the Shockley model in equation 2.6 and the piecewise model in equation 2.7. We traced the I-V of four SL42 diodes from 0A to 1.2A using the N6705A DC Power Analyser. This test was carried out on the individual diodes without having them connected to the PV modules in the laboratory with no exact control of the component's temperature. The range was chosen because the module's current cannot exceed 1.2A. The diode parameters are determined using MATLAB's $cftool()$ curve fitting toolbox and are summarised in Table 2.2.

Visually in Figure 2.21A, we observe that both models, the Shockley equation and the piecewise equation, are equally valid due to the variance of the diodes for currents above 0.2A (green region highlighted in Figure 2.21). Below that, the piecewise model underestimates and the Shockley equation overestimates the current (red region highlighted in Figure 2.21). The errors in this region are quantified in Figure 2.21B where we see a higher median error on the Shockley model and more significant outliers on the piecewise model. Since there is no convincing evidence to choose one model over the other nor do we require a very high accuracy, we simply chose the convenient option depending on the context.

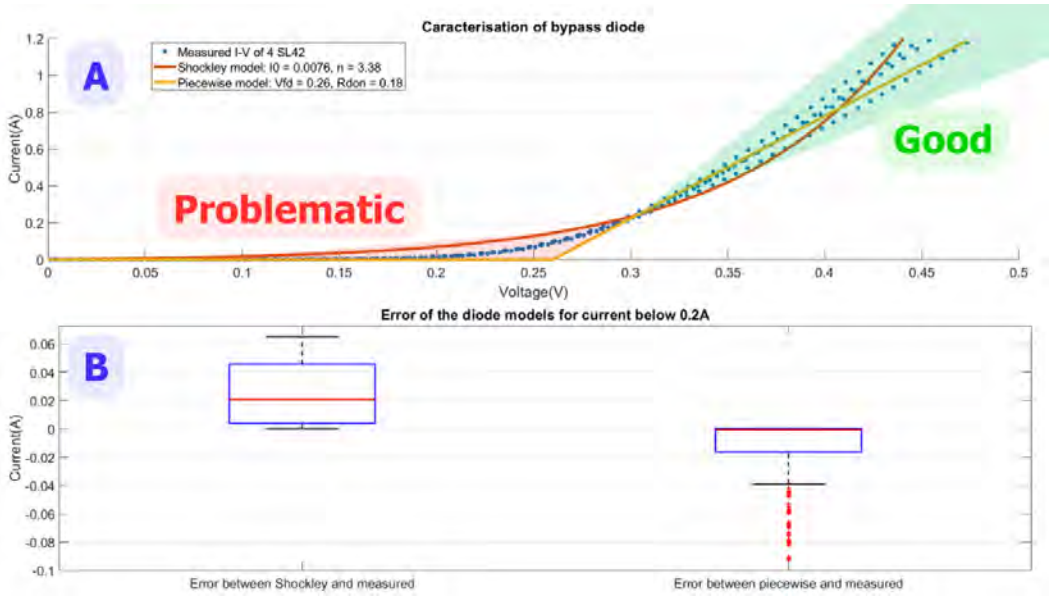


Figure 2.21: Bypass diode characterization result. Section A: Graph showing the I-V of four SL42 diodes measured in laboratory setup, the estimated I-V of the diodes from Shockley equation model, and the estimated I-V of the diodes from piecewise equation model. Section B: Boxplot graph showcasing the current error between our two diode models and the measurement for the "problematic" region in section A where the current is below 0.2A.

2.2 Graph the distribution of GMPPs

In this section, we concretely plot the GMPP distribution to validate our estimation of the GMPP regions. To achieve this, we first describe the string of four PV blocks using the model obtained above in MATLAB with the parameters found in Table 2.1 and Table 2.2. We then iterated through millions of different G-T conditions and determined the GMPP of each simulated P-V in a process called **G-T sweep**. The higher the **resolution** of this sweep, meaning more irradiance conditions or more temperature conditions, the more detailed the result would be but the program would also take longer to run and might not actually provide any additional useful information.

There are some terminologies to clarify before we proceed. An **irradiance condition** is the four irradiance values that our four PV blocks receive, an **set of irradiance** is a list of irradiance conditions, and the **irradiance resolution** is the number of irradiance conditions in a set. The same logic apply to **set of temperature**, **temperature condition**, and **temperature resolution**. The total number of iterations per G-T sweep would then be the irradiance resolution times the temperature resolution. Each G-T sweep generates a **distribution dataset** containing all the V_{gmpp} and P_{gmpp} that was recorded, and to specify which distribution dataset, we would refer to it using the name of the set of irradiance and the set of temperature used in the sweep.

2.2.1 Selecting the set of irradiance for the G-T sweep

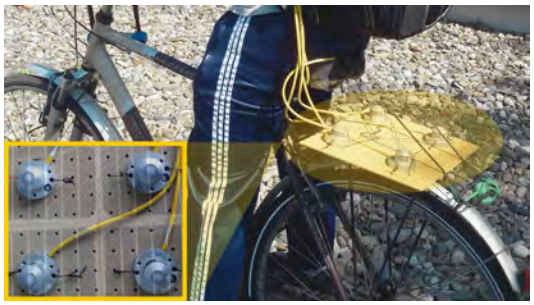


Figure 2.22: Photo of our biking measurement trip, carried out from 10h34 to 11h22 on the 21/06/2021 in Toulouse, France. The positioning of the pyranometers was inspired by how the four PV modules could be positioned.

Given that the irradiance of each module could vary independently, the first option is to sweep all possible combinations of four values of irradiances with repetition and assume that they all have equal probability of occurring. A combination with repetition of x items from a set of y items means that we pick out x items from this set where order does not matter and that the same item from y could appear multiple times in the combination of x items. This is because when we look back at equation 2.8 and Figure 2.14, the I-V of a string would not change if we permute the irradiances of the PV blocks. Therefore, we iterate through all combinations of

four irradiances with repetition from a set of $10Wm^{-2}$ to $1000Wm^{-2}$ at a step of $10Wm^{-2}$. We call this the set of **equal probability irradiance**. In total, this set contains 4,421,275 irradiance conditions.

However, it would be interesting to observe the distribution in a more realistic and challenging use case. For this, we rode a bicycle around the city while measuring solar irradiance using four SP Lite2 pyranometers as shown in Figure 2.22. This test situation was somewhat arbitrary because our initial research objective was to study what happens to a solar array under unstable and rapidly changing irradiance profiles and measuring irradiance on a mobile system was the first thing that came to mind. However, while we could have studied any other situations like oscillating shadows cast by the branches of a tree or solar panels on the wings of an airplane, measuring the irradiance on the back of a bike provides the quickest way to obtain a lot of different shadow dynamics. Each channel is sampled at 5kHz by a NI-6009 DAQ and the raw acquisitions are then filtered to four irradiance profiles, each effectively sampled at 100Hz. We then round the acquired irradiance values to the nearest $10Wm^{-2}$ and use this measurement as our second irradiances set that we call the set of **bicycle irradiance**. In total, this set contains 201,808 irradiance conditions, where 38,802 are distinct irradiance conditions.

2.2.2 Selecting the set of temperature for the G-T sweep

As for the set of temperature, we want a more limited resolution because of the massive amount of irradiance conditions already at hand. Furthermore, it is unreasonable to simply sweep the temperature of each module from $-10°C$ to $80°C$ because we would simulate some situations where one block is operating at $-10°C$ while the other is operating at $80°C$. Therefore, we would make several loose assumptions on the temperature of our PV blocks, as shown in Figure 2.23. However, the **goal here is not to have a highly accurate representation of their temperature**, but rather to put them in some challenging context so that we could observe their impact on the GMPP distribution.

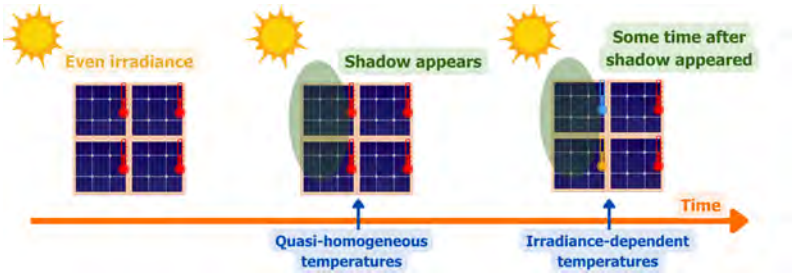


Figure 2.23: Illustration of the contexts to estimate the temperature of the PV modules.

Imagine that we want to observe the P-V of four PV blocks in series being evenly irradiated when they are suddenly shaded by an object resulting in one of the irradiance conditions. We first consider what the temperature of the modules would

be when the PV of the string was supposedly measured immediately after the shading event. Since all PV modules were evenly irradiated, it is reasonable to say that each module's temperature is close to the string's average temperature with some variance, and their temperature could not have varied instantaneously after the shading event. Next, we need to consider the temperature of the bypass diodes since their characteristics are also temperature dependent, and for the sake of simplicity we would assume that they share their respective module's temperature because they are installed directly in the modules. So, in this context, it is safe to assume an average temperature for the string, and the temperature of each module would be this average value plus some uniformly distributed random temperature delta between $\pm 5°C$. For the averaged string temperature, we chose $-10°C$ for a very cold operating condition, $25°C$ for the optimal operating condition, and $60°C$ for a very hot operating condition. This is called the set of **quasi-homogeneous temperature** with three temperature conditions.

However, if we wait some time after the shading event before observing the P-V of the string, we could assume that the more irradiated modules are hotter than the less irradiated ones. Of course, this is not always the case because the heat that each PV module receives also depends on solar spectrum, wind speed, humidity, structure of the module, etc. But, for the sake of simplicity, we suppose that its temperature is linearly dependent on its irradiance. In this context, the temperature of each PV block would be the ambient temperature plus a positive temperature delta linearly dependent on the irradiance it receives. We also assume here that the bypass diode's temperature is equal to its respective PV module. We then need a reasonable estimate for the temperature deltas at the two extremes of the irradiance values, $0Wm^{-2}$ and $1000Wm^{-2}$, and from there we linearly interpolate the temperature delta for all other irradiance levels. First, a completely shaded module should not have any sunlight heating it up, so its temperature would be at ambient. Second, based on our solar panel characterisations and information from residential solar energy forums, we concluded that a $25°C$ higher than ambient temperature delta is a reasonable estimate for a $1000Wm^{-2}$ irradiated module. An example of linear interpolation is that a module receiving $500Wm^{-2}$ would be $12.5°C$ hotter than ambient temperature. We decided on three values of ambient temperature, $40°C$ for a very hot day, $25°C$ for a good day and $-10°C$ for a very cold day. This is called the set of **irradiance-dependent temperature** with three temperature conditions.

2.2.3 Studying the GMPP distribution from the G-T sweeps

In total, we have four distinct G-T sweeps that could be done with our two set of irrdiance and two sets of temperature. First, let us look at the two distribution datasets obtained when running the set of equal probability irradiance with the set of quasi-homogeneous temperature and the set of irradiance-dependent temperature. Both datasets have 13,263,825 recorded GMPP. Figure 2.24 plots the occur-

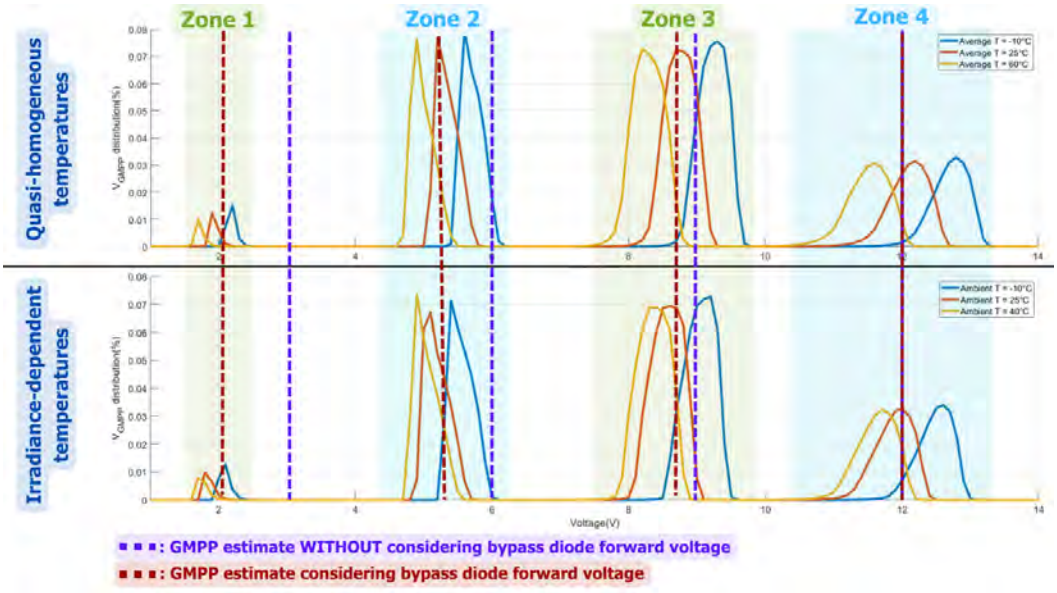


Figure 2.24: V_{gmpp} distribution from the G-T sweep where we have the equal probability irradiance set combined with the quasi-homogeneous temperature set and the irradiance-dependent temperature set. Different GMPP regions are highlighted as zone one to four. Dotted line shows the GMPP regions estimate value with and without considering the effect of bypass diodes.

rence rate of distinct V_{gmpp} as a percentage of total GMPP at each temperature condition. Overall, we could clearly distinguish the four GMPP regions confirming the assumption that GMPPs should occur in distinct zones. Furthermore, we also show that the GMPP region estimate is better when the effect of the bypass diode's forward voltage is considered than when it is neglected. How these GMPP estimate values are calculated could be found in Section 2.1.4. Furthermore, while we could see some differences between the distributions under the two different temperature assumptions, the small temperature coefficients of the panels mean that their effect is not overly significant (meaning that the regions do not overlap even under extreme temperatures).

However, the regions are less clear when we visualise the two distribution datasets obtained when running the set of bicycle irradiance with the set of quasi-homogeneous temperature and the set of irradiance-dependent temperature. While there is still separation between regions one and two and three, there is no clear delimitation between regions three and four. To see why this is the case, we now must consider the power at GMPP P_{gmpp} and we extracted two specific subsets of the distribution for this in Figure 2.24. It shows that for the same temperature condition, we have many lower values P_{gmpp} with the set of bicycle irradiance. Therefore, we could conclude that the assumption of distinct GMPP regions is only valid if the array is not frequently heavily shaded.

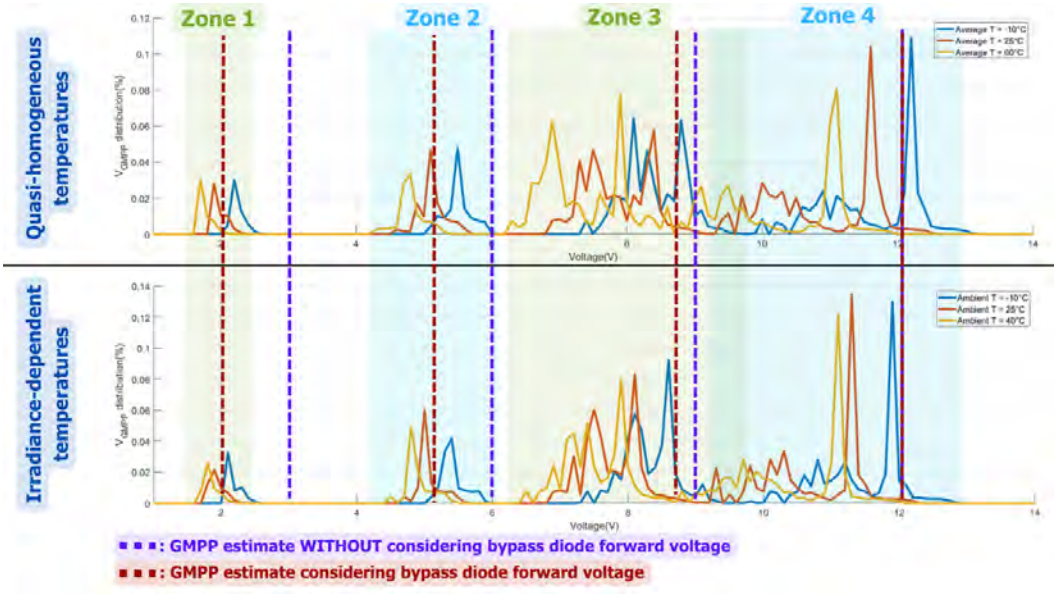


Figure 2.25: V_{gmpp} distribution from the G-T sweep where we have the bicycle irradiance set combined with the quasi-homogeneous temperature set and the irradiance-dependent temperature set. Different GMPP regions are highlighted as zone 1 to 4 in the figure. Dotted line shows the GMPP regions estimate value with and without considering the effect of bypass diodes.

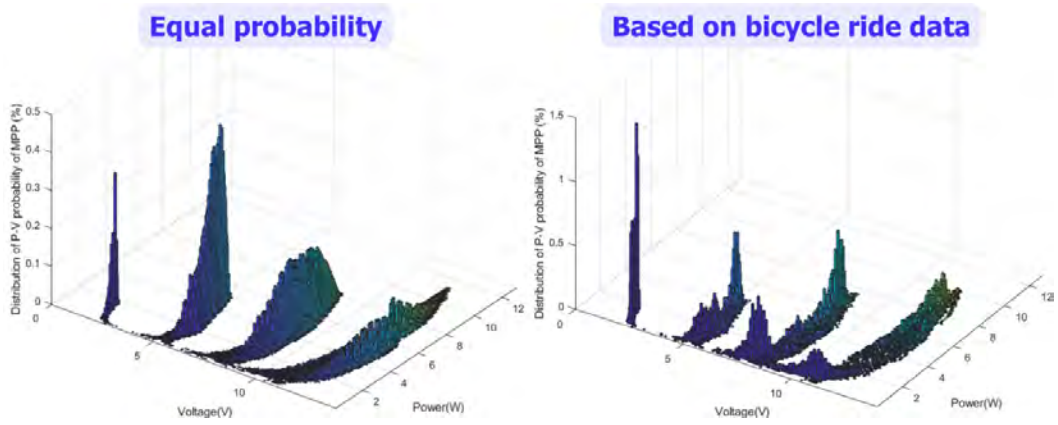


Figure 2.26: V_{gmpp}, P_{gmpp} occurrence. On the left was data taken from the distribution dataset generated by the set of equal probability irradiance and quasi-homogeneous temperature at $25°C$. On the right was data taken from the distribution dataset generated by the set of bicycle irradiance and quasi-homogeneous temperature at $25°C$.

2.3 Compute optimisations

Since the idea of examining all possible P-Vs to see how the GMPPs are distributed is relatively simple, the lack of quantitative analysis in the literature could be attributed to the long computation time caused by the complexity of the PV equation. Therefore, we find it important to present the optimisations needed to simulate the massive set of G-T conditions in a reasonable timeframe. The goal is to evaluate how much time is needed to calculate the GMPP of the array per G-T condition and see the improvement provided by each optimisation. For this, we will time each simulation of 1000 different G-T conditions and plot the runtime in a boxplot to see the average and variance. It must be noted that the flow of the discussion followed our logic at the time, so some subsequent optimisations might render some previous ones redundant.

2.3.1 Two basic G-T sweep scripts

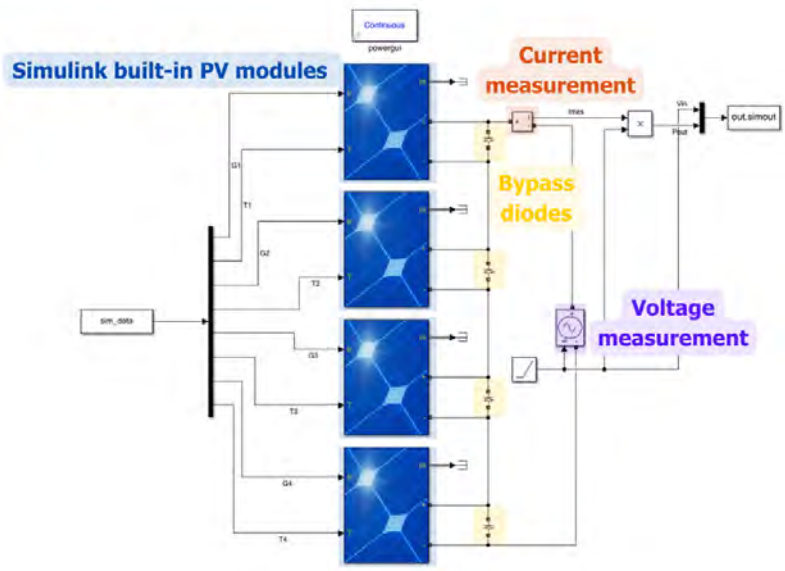


Figure 2.27: Simulink schema to sweep voltage and current. The PV modules and diodes are built-in Simulink models.

To set a baseline for execution time, we crafted two simple scripts to find the GMPP of four PV blocks in series that take no longer than 30 minutes to setup. The hardware specification of the computer used for the program is a Ryzen 7 3700X, 32GB of RAM at 3200MT/s, and 1TB of NVME SSD. These specifications influence, respectively, the computing speed, the number of variables to be kept in quick access during the execution of the program, and the initial loading time of the model.

Our first simple MATLAB program will call a Simulink model where we sweep the voltage of a PV array while measuring the power at each operating point (Figure

2.27). The bypass diode model used by Simulink is based on the piecewise equation 2.7. After simulation, the Simulink I-V graph is transferred to MATLAB to determine V_{gmpp} and P_{gmpp}.

The first step of the second method involves solving the set of equations 2.4 with the help of MATLAB's built-in function *solve*() to acquire the I-V of the module. It finds the symbolic solution to which we could then apply a whole vector of input values without the need to iterate through each I-V pair. We need to choose a form of solution for our PV module which could be $I_{pv} = f_m(V_{pv})$ or $V_{pv} = g_m(I_{pv})$. However, there is not much of a choice, since $V_{pv} = g_m(I_{pv})$ is unusable because its explicit form contains the LambertW function, which causes double precision overflow during intermediate computation steps. Note that while the explicit solution of $I_{pv} = f_m(V_{pv})$ also has the LambertW function, it does not lead to double precision overflow. This non-linear function would be further explored when we use it to solve the set of equations 2.4 ourselves to reduce the computation overhead. After getting the I-V of a module, the next step would be to add in the bypass diodes to get the I-V of a PV block, this time modelled with the Shockley equation, because it is slightly more convenient to write a single equation. At this stage, we have the I-V of a PV block in the form $I_{block} = f(V_{pv})$, which is quite inconvenient since the next step would be to construct the string voltage in the form $V_{string} = \sum_{k=1}^{n} g_k(I_{block})$ found in equation 2.8. Since this would be a recurrent problem when mathematically simulating the PV string with bypass diodes, let us discuss it and find a solution.

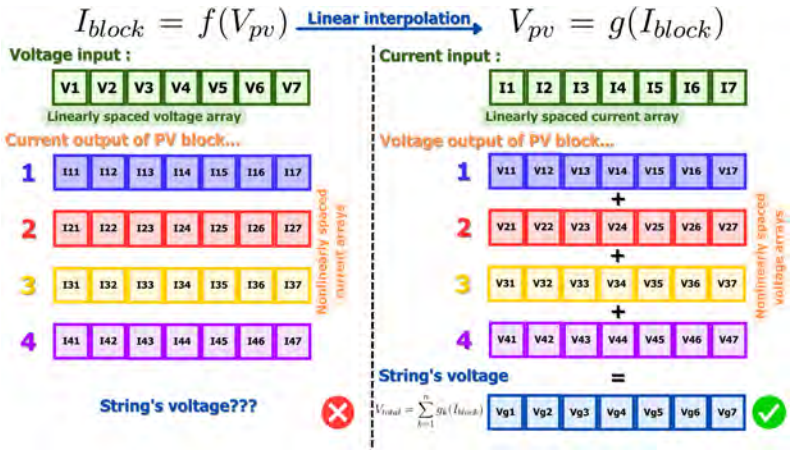


Figure 2.28: How the PV block's explicit form impacts computation.

Computationally, obtaining an I-V from a PV block in the form $I_{block} = f(V_{pv})$ means giving it a linearly spaced array of voltages V_{pv}^{linear} and receiving an unknown spaced array of corresponding currents $I_{block}^{nonlinear}$ as output. On the other hand, we need $V_{pv} = g(I_{block})$ to work in the form $V_{string} = \sum_{k=1}^{n} g_k(I_{block})$ is. In this form, we would obtain an array of linearly spaced currents I_{block}^{linear} tied to an unknownly spaced array of voltages $V_{pv}^{nonlinear}$. To better understand this, we made a visualisation

in Figure 2.28. To fix this, we utilised the *interp*1() function in MATLAB that interpolates a V_{pv}^{linear} tied to $I_{block}^{nonlinear}$ to I_{block}^{linear} tied to $V_{pv}^{nonlinear}$. From there, it would be trivial to get $V_{string} = \sum_{k=1}^{n} g_k(I_{block})$ and determine the GMPP.

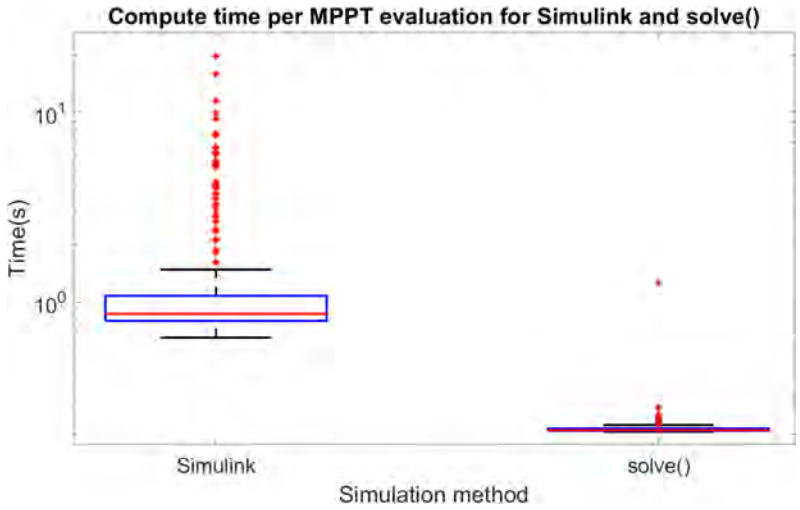


Figure 2.29: Execution time of our two simple MPP evaluation method as baseline measurement.

We now assess the execution time of these two methods and graph the boxplot of the simulation time of 1000 different G-T conditions in Figure 2.29. The median runtime using Simulink is 0.865s while the median runtime using *solve*() is 0.209s. From the above results, the use of Simulink is inefficient and inconsistent. Furthermore, the solver frequently fails to converge, which is a problem that will be addressed later when we model our system in Simulink. Regarding the use of the *solve*() function, the extra work put into describing the PV block paid off and the simulation was completed in a reasonable timeframe, but there are still some inconsistencies. However, when we start considering how many iterations could be fit into a specific timeframe, it paints a different picture. We plotted the number of iterations over the runtime of the program in Figure 2.30 and took 24h to be an arbitrary limit to the maximum run time of the program. In this period, we could calculate a total of 414720 GMPPs, which could be exhausted by simulating four PV blocks at a resolution of 12 irradiance conditions and six temperature conditions. This leaves little room for more PV blocks or a higher sweep resolution. Therefore, optimisation is essential.

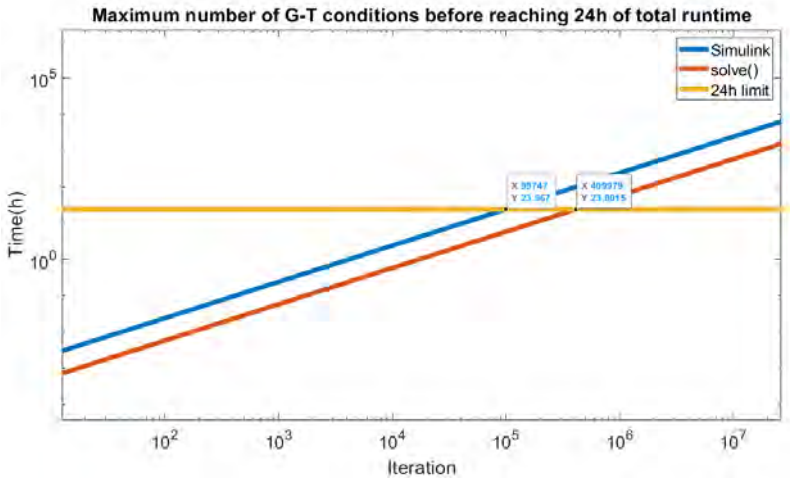


Figure 2.30: The number of iterations completable within the 24h time frame of the 02 methods to simulate GMPP.

Since the previous methods are very primitive, there is much room for improvement. By looking into different aspects of the program, we see four potential time saves that are investigated in this section. Although these are standard knowledge in computer science, it is still worth going into detail and evaluating the improvements they brought.

2.3.2 Improvement one: Reducing program instructions

A program with more machine instructions will generally run longer than one with less instructions, therefore the runtime would be reduced if we directly provide the explicit PV model solution instead of using $solve()$. Although the PV model of equation 2.4 is non-linear due to the presence of exponents, it is possible to arrive at an explicit solution using the $LambertW$ function [71] and its definition is found in equation 2.9. In the PV literature, there were works using $LambertW$ to determine the parameters of the PV model [72][73], to perform MPPT in real time [74], and there were also discussions on its optimisation for PV applications [75][76].

$$ye^y = x \quad then \quad y = LambertW(x) \tag{2.9}$$

Step by step, we describe how to arrive at the solution $I_{block} = f(V_{pv})$ to describe a PV block. But we have to start at the PV module level where we first transform the equation 2.4 as follows to consolidate the terms I_{pv} and V_{pv}:

$$
\begin{aligned}
I_{pv}\ (1 + \frac{R_s}{R_p}) =&\ (I_L + I_0 - \frac{V_{pv}}{R_p}) - I_0 e^{\frac{q}{AkT} V_{pv}} e^{\frac{q}{AkT} I_{pv} R_s} \\
I_L =& \frac{G}{G_{ref}}(I_{scn} + K_i(T - T_{ref})) \\
I_0 =& \frac{I_{scn}\ +\ K_i(T - T_{ref})}{e^{(\frac{q}{AkT}(V_{ocn} + K_v(T - T_{ref})))} - 1}
\end{aligned}
\tag{2.10}
$$

Let us now have the following terms:

$$
\begin{aligned}
K =& \frac{q}{AkT} \\
X =& \frac{(I_L + I_0 - \frac{V_{pv}}{R_p})}{(1 + \frac{R_s}{R_p})} \\
Y =& \frac{I_0 e^{KV_{pv}}}{(1 + \frac{R_s}{R_p})}
\end{aligned}
\tag{2.11}
$$

With both equations 2.10 and 2.11, we arrive at this simplified form:

$$
I_{pv}\ =\ X - Y e^{KR_s I_{pv}} \tag{2.12}
$$

We continue to modify the equation reach the form $ye^y = x$:

$$
KR_s(X - I_{pv}) e^{KR_s(X - I_{pv})} =\ KYR_s e^{KXR_s} \tag{2.13}
$$

By applying the Lambert W function to equation 2.13, we have the following:

$$
KR_s(X - I_{pv}) =\ LambertW(KYR_s e^{KXR_s}) \tag{2.14}
$$

By simply rearranging the terms, we now reach the desired form $I_{pv} = f_m(V_{pv})$:

$$
I_{pv} =\ X - \frac{LambertW(KYR_s e^{KXR_s})}{KR_s} \tag{2.15}
$$

Since the above equation just describe the I-V of the PV module, we now include the bypass diode's current giving us the PV block's characteristics in equation 2.16. To implement in MATLAB, we replaces K, X and Y with their definition in equation 2.11.

$$
I_{block} =\ X - \frac{LambertW(KYR_s e^{KXR_s})}{KR_s} + I_{byp}(e^{\frac{-qV_{pv}}{A_{byp}KT}} - 1) \tag{2.16}
$$

2.3.3 Improvement two: Vectorisation

Vectorisation, also known as array programming, is a programming paradigm that leverages operation between arrays of values instead of iterating through each element and performing the calculation individually. This feat is possible due to the computing power we have today, where all processors have vector accelerating units, and computer memory is not a significant concern. The idea is simple: **replaces for loops with matrix computation if possible**. We have partially vectorised the computation of I_{block} up to now, but it is also possible to reduce the four for loops used to compute I_{block} with different irradiances with vectorisation.

2.3.4 Improvement three: Look-up table (LUT)

The program up to now has been calculating the I-V of each PV block every time a new G-T condition is checked. But this is very redundant because their I-V under the same irradiance and same temperature does not change. Therefore, we should precompute the I-V characteristics of each PV block for all potential G-T conditions and store them in a LUT, meaning that we have all the possible $V_{pv} = g(I_{block})$ stored in memory. When the program iterates through the different G-T conditions of the string, it could construct the I-V of the string using the quick addition $V_{string} = \sum_{k=1}^{n} g_k(I_{block})$ on the already computed I-V profiles it needs. However, the LUT takes up large amounts of system memory, but in our use case it is negligible at around 200MB. To time the impact of this optimisation, we would not include the LUT table compute time, because it is constant and does not significantly impact the global program runtime.

2.3.5 Improvement four: Parallel computing

Next on our agenda would be irreducible loops because we are evaluating a range of G-T conditions. With today's computing horsepower, we could leverage multi-core processors to execute them in parallel. In MATLAB specifically, $parfor$ could distribute the loop content to run on different threads (called workers in MATLAB), effectively dividing the total run time of the program by the number of **true parallel threads**. We used this term because the x86 architecture powering the vast majority of PCs and servers is hyper-threaded (a term coined by Intel). It means that a single physical core has two parallel threads, but paralleling the code on two threads of a single core does not half the program's runtime. This is furthermore complicated by the fact that the gain from parallelism depends on the processor's thread count and core count, the occupancy level of the operating system, the data distribution to workers' overhead, latency due to shared memory access, etc. Therefore, we decided to approximate that the number of true parallel threads is about 1.25 times the number of hyperthreaded physical cores. This means that the program's runtime will be divided by 1.25 times the number of hyperthreaded physical cores, assuming that the number of loops is large enough to offset the initial data distribution time and that the OS does not have a lot of other tasks running. Since

we could not simulate a single G-T condition in parallel, timing this optimisation would be done slightly differently. For each of the 1000 G-T conditions, we would perform the same calculation eight million times in a parallel loop and take the total loop time divided by eight million to be the run-time of that G-T condition.

2.3.6 Evaluating the execution time impact of the optimisations

We graphed the box plots of the 1,000 run-times per each optimisation in Figure 2.31. Overall, we have reduced the runtime by about five orders of magnitude with all of these improvements. However, not all of them are equally impactful. First, by solving the equation ourselves rather than using the *solve*() function, the median simulation time of one G-T condition drops to 0.6ms from the previous 209ms. However, given that we are still computing on demand for every G-T condition, this means that vectorisation simply replaces a loop of four separate I-V compute of the four PV blocks. Therefore, we see that the compute time only drops to 0.38 ms from 0.6 ms. On the other hand, using the look-up table provides a massive time reduction from 380μs to 4μs. This is due to the removal of the costly interpolation function *interp*1() from the loops. We could see that if we started using the LUTs directly, solving the PV equation and using vectorisation would be fairly redundant. However, they do contribute to computing the LUT much more efficiently, reducing the time from minutes to less than 1s. But this is still inconsequential because the program itself would still take around 5 to 10 minutes to iterate through the massive number of G-T conditions. Finally, the parallelisation creates an effective time reduction from 4μs to 0.6μs, but this result is machine-dependent, so a more powerful machine should make the compute time even faster.

By plotting the theoretical number of iterations that could be completed in a time frame shown in Figure 2.32, we could expect to complete around 145 billion G-T conditions in 24 h or around six billion G-T conditions per hour. Therefore, we could comfortably say that these optimisations make the study of a larger photovoltaic array very feasible. For example, taking one hour to simulate a random set of six billion G-T conditions of an array of 20 modules with bypass diodes should give a good insight into how the GMPPs are distributed. Of course, the more modules we have, the more additions we have per loop. However, these calculations are very efficient on modern machines, so we do not expect a substantial time increase here.

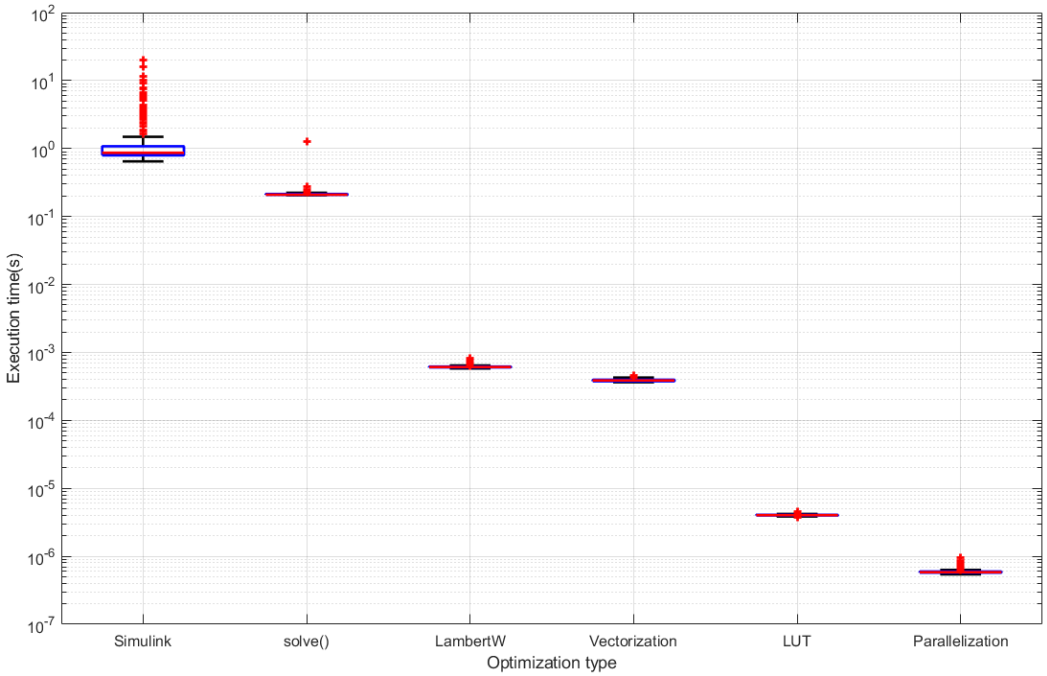


Figure 2.31: Boxplot of execution time for each optimization.

2.4 Electrical simulation of solar harvesting system

2.4.1 The basics of software GMPPT algorithm

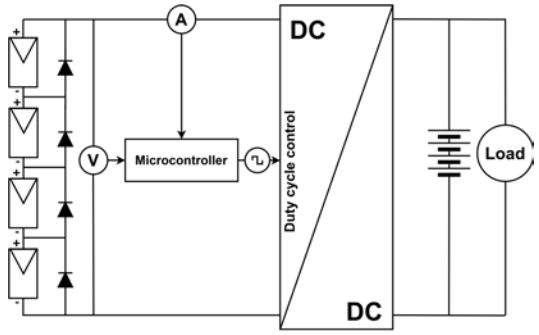


Figure 2.33: Block diagram of our solar harvesting system.

Figure 2.33 shows the block diagram of our solar harvesting system: a string of four PV blocks, a microcontroller measuring the voltage and current of the string, a buck converter, a battery, and a load. Why this architecture was kept instead of a new solution will be explained in the next chapter. For now, let us proceed with the **core principle of software MPPT** and the **the modelling objective**.

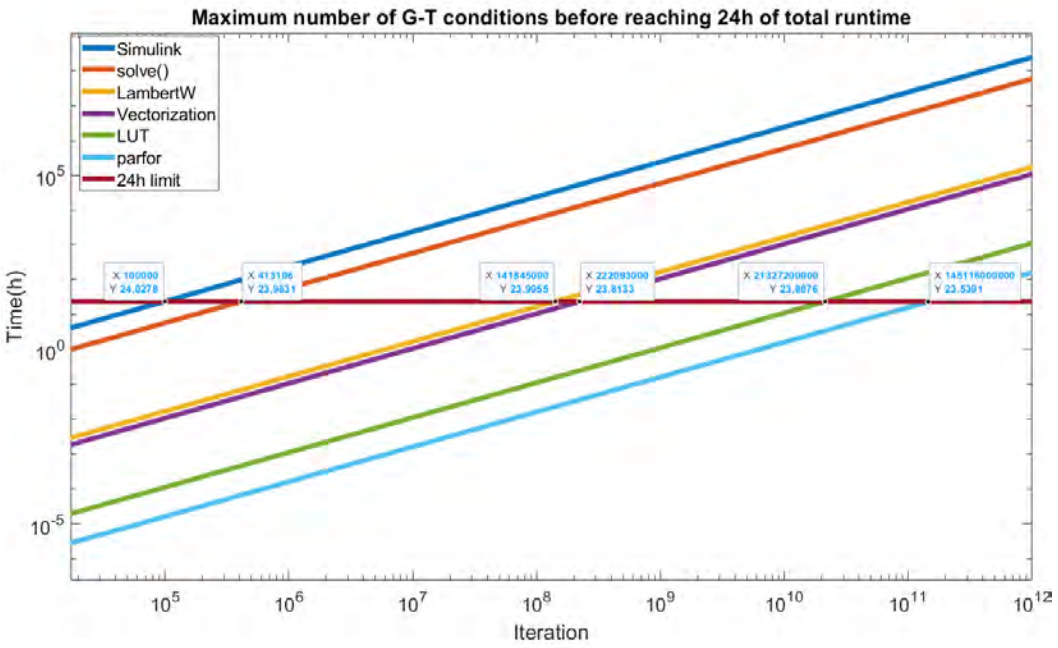


Figure 2.32: Theoretical number of iterations that could be completed over time based on the average execution time.

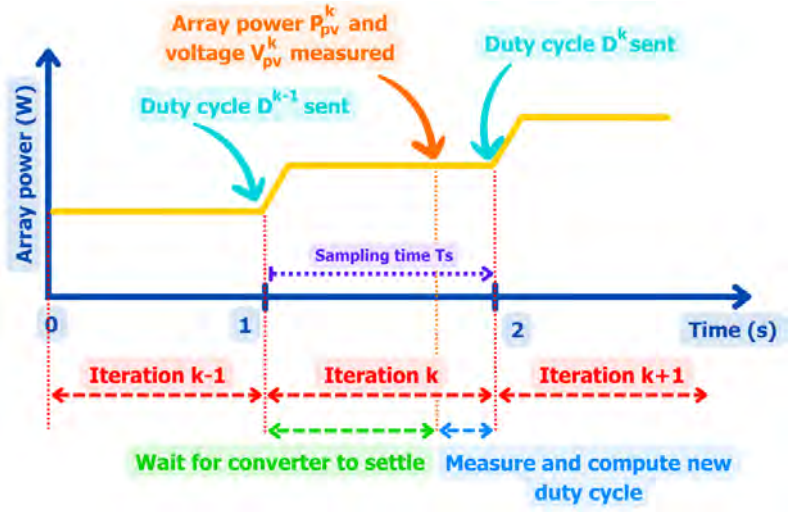


Figure 2.34: Time diagram showing the power evolution of the PV array when MPPT is active.

The overarching principle of MPPT is that the microcontroller measures the power obtained at each duty cycle and makes duty cycle adjustments based on those previous measurements to increase the power output of the array. This is repeated until the maximum power point is reached, which is called **convergence**. To better understand this concept, as well as establish important nomenclatures for later

discussion, we provide a simple time diagram of the power evolution of a photovoltaic array when the system is tracking maximum power point in Figure 2.34. At each **iteration** k, the microcontroller waits for the converter to settle from the input of the duty cycle D^{k-1} from the previous iteration $k-1$, then measures the voltage V_{pv}^k of the array and the power P_{pv}^k of the array to update a new duty cycle D^k driving the DC/DC. The **sampling time** T_s as implemented in the microcontroller is the time between the previous update of the duty cycle D^{k-1} and the current measurement of voltage V_{pv}^k and power P_{pv}^k. However, since the measurement and computation phase is mostly negligible (exaggerated in Figure 2.34 for better clarity), iteration k could be considered to last exactly one sampling time. Nevertheless, keep in mind that it might not be the case if the compute time on the microcontroller is important.

The **modelling objective** of this section is to represent our system in Simulink so that we could study the GMPPT algorithms before deploying them in a physical setup. Since we already have a physical system at hand, the model is primarily used for preliminary testing of the algorithm flow to avoid any surprises that might be harder to debug on a physical system. Furthermore, it could help us pre-tune the algorithm's operational parameters before experimental deployment in C because repeatedly flashing a microcontroller without a good starting point is not the best experience. Therefore, we want a fast and lightweight model where accuracy is not paramount.

2.4.2 Electrical model of the PV string

We start this section with a brief background on the PV string modelling process. In the previous section on computational optimisations, we showed that Simulink provides a built-in PV array block that could conveniently simulate any PV module or array without bypass diodes (Figure 2.27). We also discussed how we ran into solver errors when simulating the string of four PV blocks with this Simulink model, which is how we initially modelled them to test our algorithms. Believing that these errors were caused by the way the built-in block is written, we set out to develop our own single diode model based on equation 2.4. After finishing our model, the solver errors still occurred, but we found through trial and error that adding a parasitic capacitance to the modules fixed the problem. This is the reason we explicitly modelled the module in Simulink and did not use the built-in model. Furthermore, we could not have used the optimised computation using the *LambertW* function because it could not be used in Simulink. By the time we realised that an LUT table could be used to accelerate the Simulink model by replacing the entire string with a voltage-current-dependent source, we had proceeded past the simulation phase.

2.4.2.1 Electrical model of the PV module

The model requires the SimScape Electrical toolbox, which allows for electrical signal simulations. The implementation is relatively simple since we just need to

follow the block diagram of the single-diode model in Figure 2.7 coupled with the mathematical representation found in equation 2.4 as shown in Figure 2.35. To select a reasonable parasitic capacitance, we consulted [77] which stated that the capacitance of a silicon solar cell is around $40nFcm^{-2}$. This number has yet to include the effect of frequency, but we are looking only for a rough estimate so that the model no longer runs into errors. Each individual cell that makes up our photovoltaic module is $\frac{1}{6}$ of an original $12.5cm \times 12.5cm$ solar cell, so we estimate that each cell would have a capacitance of around $1\mu F$. And these six cells are connected in series to make up our module, so we estimate the module's capacitance to be around $330nF$.

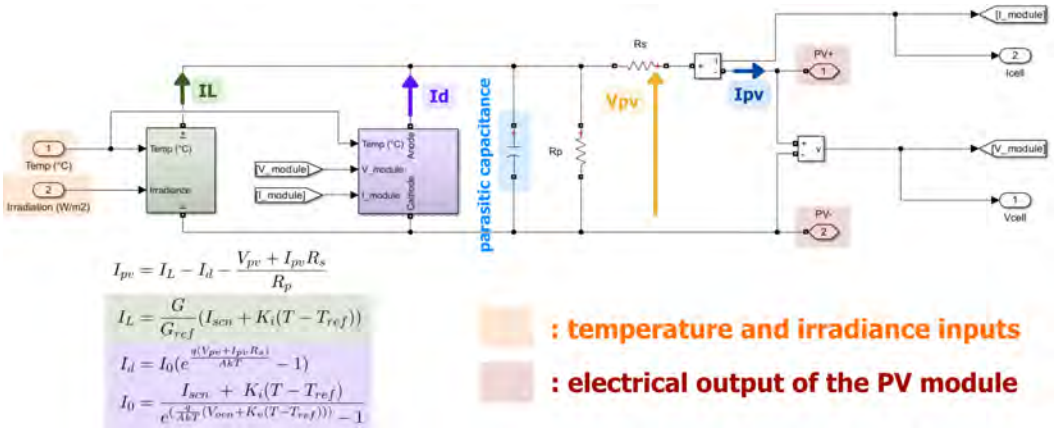


Figure 2.35: Our electrical model of one PV module in Simulink.

2.4.2.2 Adding the bypass diodes to model the PV string

To create our string of four PV blocks, we need bypass diodes. For these, we conveniently used the built-in Simscape Electrical toolbox's which is modelled using the piecewise equation and the parameters in Table 2.2. Although it is sufficiently close to our Shockley mathematical model used in the mathematical model in Section 2.1.4, it does cause some very small discrepancy. Nevertheless, we observed that the I-V of the PV string was not significantly impacted, so we proceeded with this choice. The final model of the string in Simulink is found in Figure 2.36.

2.4.3 Electrical model of the converter board

In this section, we study how our existing converter board in Figure 2.37 could be modelled in Simulink. We have extracted all relevant components and present a simplified schematic of the board in Figure 2.38 where we could see a **controller section** and the **buck converter section**. These two blocks are simulated as separate submodules in Simulink, so we start with the buck converter section followed by the controller section.

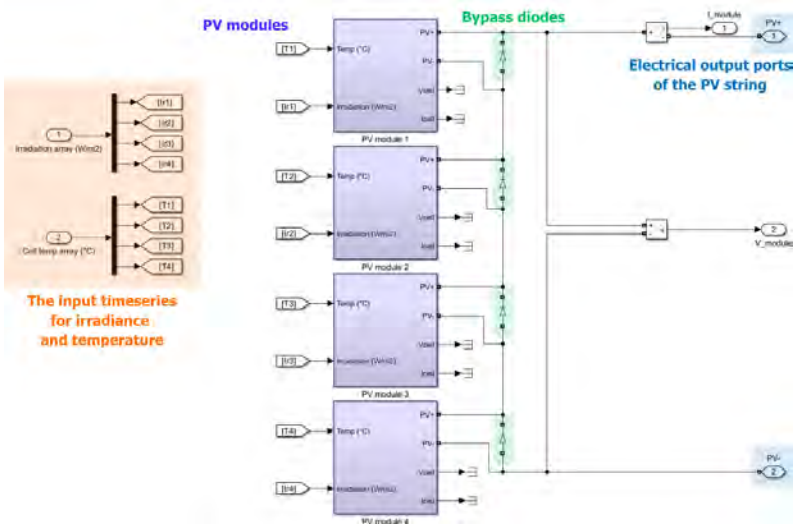


Figure 2.36: Simulink model of the PV string.

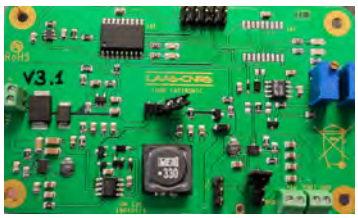


Figure 2.37: Photo of the converter board we used. The board was originally designed for a previous project in the laboratory.

2.4.3.1 Modelling the buck converter section

For this section, we extracted the core electrical components of the buck converter and connected it to a simplified representation of the PV string and a battery in Figure 2.39A to facilitate the discussion. First, our buck is a **synchronous buck converter** that replaces the usual low side diode with a MOSFET to reduce loss. Second, we remark on the presence of a diode D_0 between the converter's output and the battery. This diode serves to block a flow of current from the battery back into the PV module. Without it, there is a low resistance path from the battery through the ground via the body diode D_1 of the high-side MOSFET S_1, the very small R_s equivalent series resistance of the PV string and the equivalent internal diode of the PV string. This diode is responsible for most of the power losses in the conversion, so it was considered in the model. Losses in the input capacitor C_1 and the output capacitor C_2, the inductance L, and both MOSFETs S_1 and S_2 would be neglected because they are not important enough to be relevant.

We first address the filter circuit and the LTC6992 PWM generator in Figure 2.38. A 10kHz PWM signal is sent from our microcontroller that is filtered to an analogue voltage by the simple low-pass filter made up of R_1, R_2, and C_3. This circuit

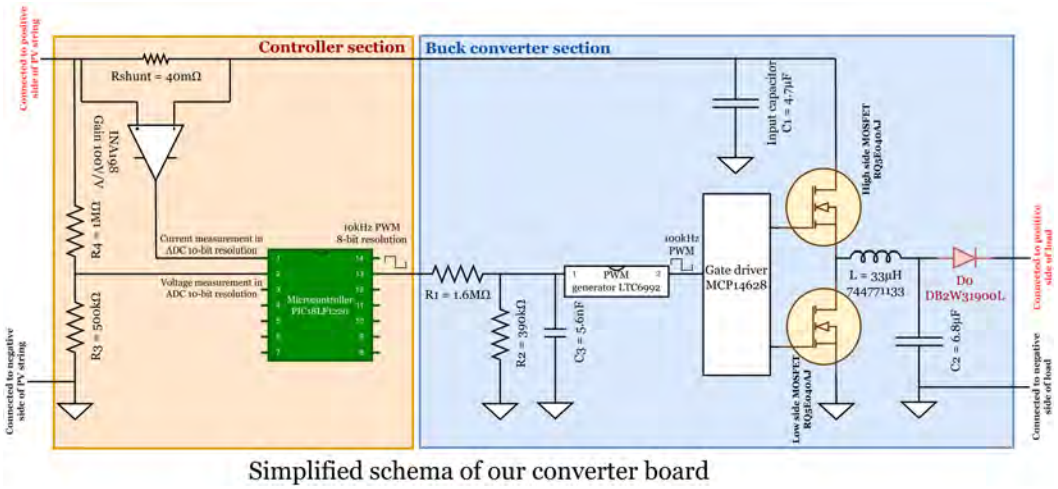


Figure 2.38: A simplified electrical schema of our controller board with 2 distinct sections, the controller section and the buck converter section.

has a cutoff frequency of 100Hz. Then, the PWM generator LT6992 reads that analogue voltage and generates a 100kHz PWM signal with the same duty cycle as the original duty cycle sent out by the microcontroller, and this 100kHz signal is fed to the MCP14628 gate driver that controls the MOSFETs.

The buck converter is a simple chopper converter that generates a lower voltage output V_{out} from a higher voltage input V_{in} and generates a higher current output I_{out} from a lower current input I_{in}. It is controlled by a duty cycle D who tells us the percentage of time where the **high-side MOSFET** is **ON**. The most obvious solution with the Simscape Electrical Toolbox at hand would be to use the built-in MOSFET, inductance, and capacitor models and practically copy-paste their parameters from their respective datasheets. However, with a switching frequency at 100kHz, the Simulink model must simulate at a much lower time step than $\frac{1}{100000} = 10ms$. This is not suitable for rapid and speedy simulation. Therefore, we turn to a **averaged buck converter model**. Its principle could be further examined in this article by Gragger et al. [78]. The idea is to separate the operation of the buck converter into two clear phases, as shown in Figure 2.39B: D where the highside MOSFET is ON and the lowside MOSFET is OFF, and $1 - D$ where the highside MOSFET is OFF and the lowside MOSFET is ON. We added node A denoting the point between the two MOSFETs and the inductance L for clarity. During D of the time when the high-side MOSFET is ON, node A is shortcircuited to the input voltage; therefore, the current I_{s1} is equal to I_{out} and the voltage of node A V_A is equal to V_{in}. Next, during $1 - D$ of the time when the low-side MOSFET is ON, we have a short circuit from node A to ground, so $V_A = 0V$. By adding these two operating regimes together, we get the averaged buck model in which the high-side MOSFET is represented as a current-dependent source whose value is DI_L and the low-side MOSFET is represented as a voltage-dependent source whose value is DV_{in}.

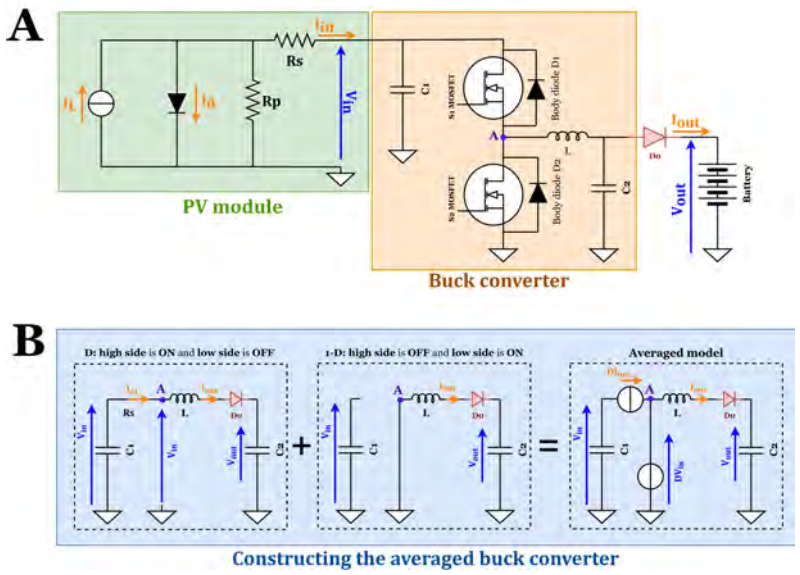


Figure 2.39: A: a simplified electrical model of the PV string (we neglected the bypass dioes so the string could now be modelled homogeneously using the single diode model), the buck converter, and the battery. B: illustrating how to obtained the averaged buck model.

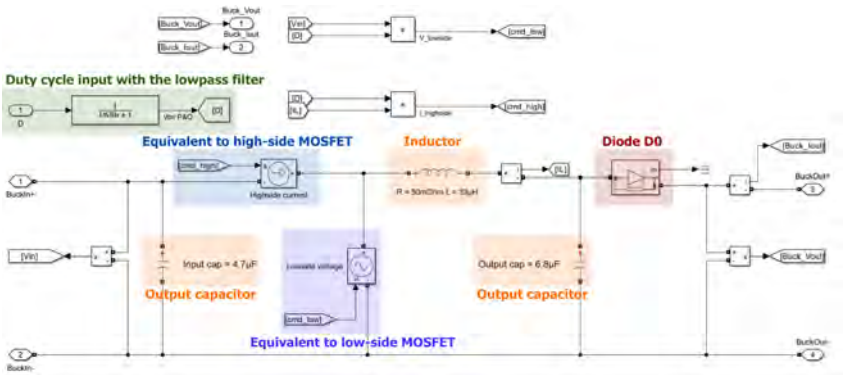


Figure 2.40: Model of the buck converter section of the converter board in Simulink.

2.4.3.2 Modelling the controller section

The main component of the controller section is the low-power 8-bit microcontroller PIC18LF1220 from Microchip, where we are utilising two of its ADC inputs and one PWM output. The ADC outputs have a full-scale range from 0V to 5V and a resolution of 10 bits. The PWM on the other hand controls the duty cycle from 0% to 100% with a resolution of 8 bits. Therefore, when the duty cycle D is implemented in the microcontroller code, it is represented by an 8-bit **duty cycle variable** that varies from 0 to 255. When necessary, in addition to the duty cycle value from 0% to 100%, we will also provide the value of the duty cycle variable over 255 to provide the reader with further implementation information. Next, we have the voltage measurement which is the output of a resistor divider consisting

of $R_3 = 500k\Omega$ and $R_4 = 1.6M\Omega$ which gives a gain of $\frac{1}{3}$. This is designed so that voltages up to 15V are still within the ADC's full-scale range. Finally, we have the current measurement based on a $R_{shunt} = 40m\Omega$ and the INA198 100V/V differential amplifier, which means that the overall structure has a gain of 4V/A. Because the array's current never exceeds 1.25A, this was designed so that we could use the ADC's full-scale range. We model these components in Simulink as shown in Figure 2.38. Each ADC measurement is modelled as a gain, followed by a saturator to clamp the value between 0V and 5V, and a simulated ADC that converts the input to an integer between 0 and 1023. The microcontroller is modelled as a function that takes the digitised input voltage and current and generates an 8-bit duty cycle. Finally, there is a gain converting this 8-bit duty cycle variable to a floating value between 0 and 1 for compatibility with the buck converter model.

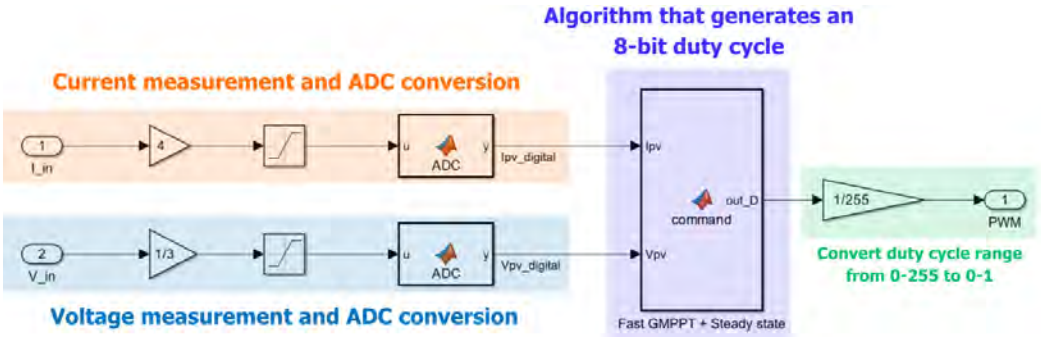


Figure 2.41: Model of the controller section of the converter board in Simulink.

2.4.4 Simple battery and load model

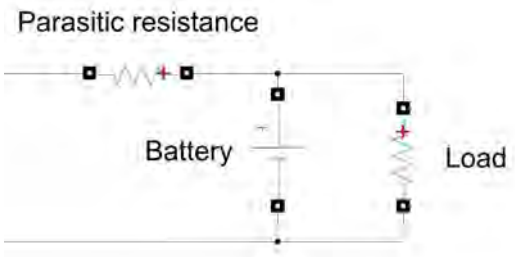


Figure 2.42: Model of the battery and load in Simulink.

As the structure of the battery and load shown in Figure 2.33 behaves essentially like a voltage source with a resistance in short time frames, we decided to use a simple voltage source and a resistor to simulate them (Figure 2.42). This choice was aimed at accelerating the simulation because Simscape proposes a good battery model, but its accuracy makes the simulation very slow. However, if the voltage source is connected directly to the load of the buck converter, it sometimes causes problems with the solver, so we added parasitic resistance to fix it.

2.4.5 Evaluating the Simulink model

We present **two test results** to see how accurate our model is compared to the physical board. The **first test** involves a **input voltage sweep from 4V to 10V of the buck converter at a fixed duty cycle and observes the output voltage under a constant load of 1A**. This tells us where we would get the same output voltage for the same duty cycle value in the experiment and in the simulation. This test was carried out for five duty cycles from 30% to 70% with a 10% step, and the result is presented in Figure 2.43. We observe that the estimation is very good at $D = 50\%$, but the model underestimates the output voltage for duty cycles above 50% and overestimates the output voltage for duty cycles below 50%. However, as previously discussed, this discrepancy is not detrimental to our work because it does not interfere with the algorithm's operation.

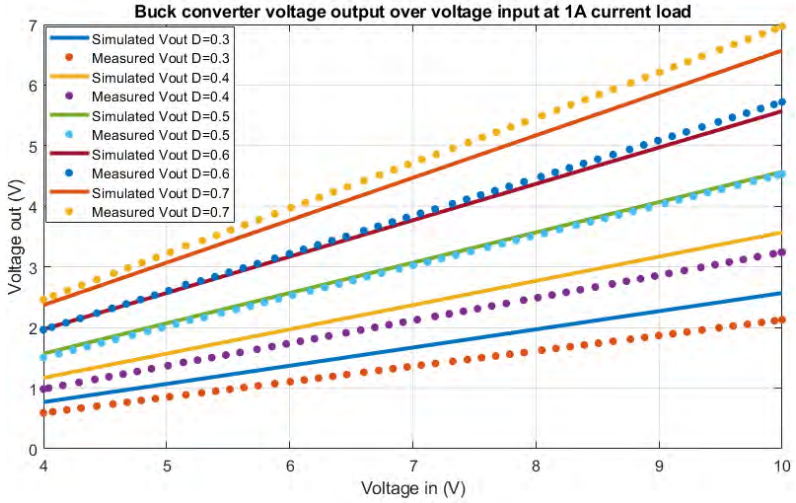


Figure 2.43: Validation result of the buck converter model for an input voltage sweep and a constant current load of 1A.

The second test involves comparing the response time of the buck converter when connected to the PV string, the battery, and the load. Specifically, we study the response of the input voltage of the converter to two duty cycle steps, from 40% to 90% and from 90% to 40%. However, this response is not entirely dependent on the converter as it also depends on the characteristics of the PV string. The input voltage response is dictated by the voltage on the input capacitor, meaning that the response time is heavily impacted by the current coming in to charge up this capacitor. Therefore, we put the PV string in a low irradiance condition where all four PV modules receive only $200Wm^{-2}$ and their temperatures are arbitrarily fixed at $25°C$. The input voltage response times for both duty cycle steps, in experiment and simulation, are shown in Figure 2.44. We highlighted the 8ms time frame in each result where we could consider that the response has settled. This value was chosen because its the minimum value that could cover the response time while being convenient to implement in the microcontroller. We notice that the response time to

the duty cycle step from 90% down to 40% is slightly slower than the response time to the duty cycle step from 40% up to 90%. This was primarily caused by the fact that the current generated by the PV string is higher at lower voltage (toward the current plateau) and lower at higher voltages (toward the current roll-off). So when the voltage approaches the string's open-circuit voltage, less current is generated to charge up the input capacitor, leading to a slightly slower input voltage response. Also, we should point out that a lower duty cycle induces higher input voltage, and a higher duty cycle induces lower input voltage because we are operating with a fixed voltage load. This is important to keep in mind for the later discussion on the operation of MPPT algorithms.

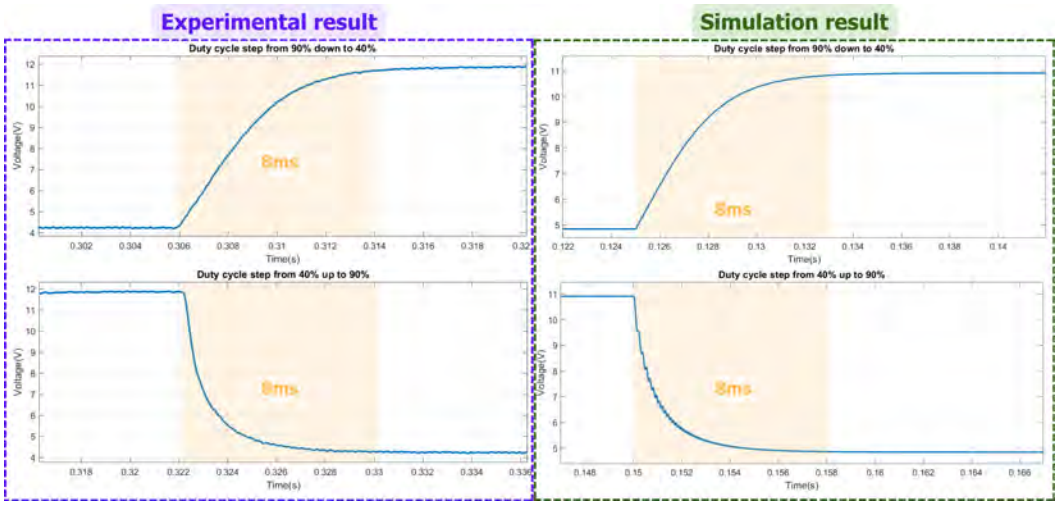


Figure 2.44: Validation result of the buck converter model's input voltage response time when a duty cycle step is applied. The converter is connected to a weakly generating PV string, a 4V battery, and a load.

Generally, this is not a highly accurate model, but it does satisfy our need. However, the simulation of a specific time frame is still longer than in real life, for example, a 2s sequence still takes more than 2s. It fluctuates depending on the operating point of the PV string. This aspect will influence one of our testing decisions in the next chapter.

2.5 Conclusion and perspective of Chapter 2

Generally, this chapter focused on the mathematical modelling of a PV module. We resume everything that was done and the information that we obtained below.

- We studied the various models of a PV cell, chosen the single diode model to proceed because it provides good compromise between accuracy and ease of use.
- We provided mathematical proof that if all PV cells composing a PV module are under the same irradiance and all at the same temperature, it is possible to model it using an equivalent single diode model. This assumption is important to keep in mind for the next chapter as well.
- The method to acquire the necessary modelling parameters was discussed, and an accuracy analysis was presented.
- We highlight two different operating regimes of a PV block, one in which the module is actively generating current and one in which the module is bypassed. This trivial fact will also be important in the next chapter.
- We propose a better way to arrive at a GMPP estimate by also considering the forward voltage of the bypass diodes. Our graph of GMPP distributions proved definitively that GMPP regions do occur in distinct zones and that our GMPP estimate is more accurate. However, it is only valid when there are more than three cells in series per bypass diode. This is because fewer cells lead to a lower V_{mpp} of the block, and therefore the regions will be less seperated.
- All optimisations to mathematically simulate the I-V characteristic of the PV string were given so that it could be applied to even larger PV arrays with an arbitrary number of PV cells per PV block and an arbitrary number of PV blocks in series or parallel. The two most impactful improvements were using the *LambertW* function to explicitly solve equation 2.4 and using a LUT to quickly construct the I-V of the PV string. This is a novel use case and unlocks the ability to quickly visualise in high detail the GMPP distribution of any arbitrary PV array.
- We built and discussed a Simulink model of a photovoltaic chain consisting of a PV string with bypass diodes, a buck converter, and a battery. It is used to quickly simulate the operation of an MPPT algorithm to fine-tune them before deploying them in C to a microcontroller. Although the accuracy analysis showed that the model is not exactly a digital twin of the physical system, we perceive that it was sufficient given the task at hand.

However, there are still some limitations to this work that could be summarised as follows:

- The assumption that all PV cells composing a PV module are receiving even irradiance and all at the same temperature is not highly accurate under partial shadings. This is because we would have to assume that the shadows are distributed equally among the PV cells. For a better model, we should consider the arrangement of the PV cells and compute, for a particular shading pattern, which cell is most shaded. Then, its shading factor is applied to the entire PV module.
- The measurements to acquire the model parameters were not performed in an optimal and controlled environnment so most error sources could not be accurately accertained.
- Although we discussed computing optimisations, it was not used to accelerate the electrical simulation in Simulink, mainly because the LambertW function was not available in the Simulink environment. However, there could be workarounds by computing those profiles as LUTs in MATLAB workspace and transferring them into the Simulink workspace to use as a voltage-current-dependent source.
- The distribution results were still mostly in theoretical situations except for the irradiance measured during the bicycle ride. We intend to expand the analysis to more shadowing situations to provide a better overview of potential shading patterns.

Chapter 3

Global maximum power point tracking algorithms

We should consider the context of work being small-scale solar harvesting as discussed in the Introduction and that we are trying to mitigate the effect of **unexpected shadows moving at varying speeds**. In the spirit of keeping the cost and complexity very low for the end user, we want to extract the maximum amount of solar energy from the most basic hardware possible. This is why despite having a capable microcontroller, we are seeking a **lightweight** solution so that it could be implemented even on the weakest microcontrollers. Furthermore, it is also the reason why we proceeded with an existing unoptimised hardware.

In general, there are two paradigms of shading management: **hardware-based** and **software-based**. We will look at some hardware-based shading management in this introductory section, but the main objective of this chapter is software-based shading management, which is **maximum power point tracking (MPPT)** in case the PV array only has one singular power peak, or **global maximum power point tracking (GMPPT)** in case the PV array could have multiple power peaks.

Let us now explore some hardware-based shading management methods. The basic PV system consists of an array of PV modules (with or without bypass) in **series-parallel configuration** connected to a single **central converter**, usually DC-DC for standalone applications and DC-AC for grid-connected, responsible for harvesting optimal power from the array. However, there is the possibility to modify the **configuration of the array** and **the architecture of the converter** for better shading management that was explored in the literature.

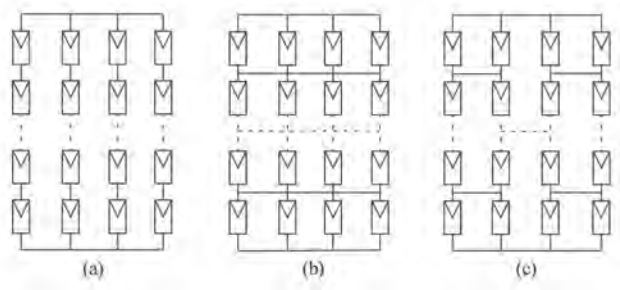


Figure 3.1: Wiring diagrams of three solar arrays configuration: the classical SP configuration (a), total-cross-tied (b), and bridge-linked (c). Figure extracted from [79].

First, we discuss some studies on modifying the array's configuration. Bidram et al.[79] have compiled two novel panel arrangements, total-cross-tied and bridge-linked (Figure 3.1). These are proposed to mitigate the potential damage to solar panels, as well as the power loss caused by uneven irradiance. Expanding on this idea, there are several works that implement a switch matrix to dynamically rearrange the array's configuration such as Velasco-Queseda et al. [80], thesis by Youssef El Basri [81], and thesis by Luiz Fernando Lavado Villa [82]. Despite the promising results, they do require more wiring and sophisticated algorithms to optimise power generation [80].

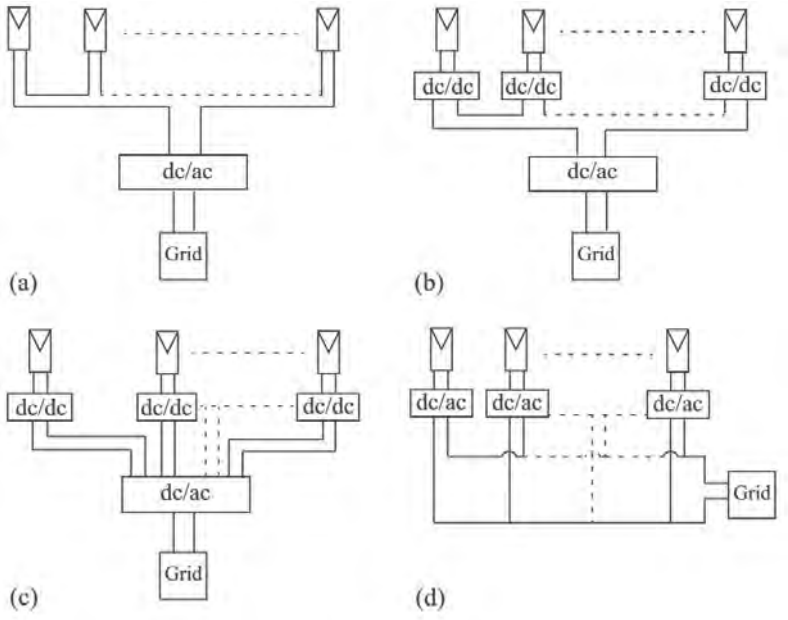


Figure 3.2: Distributed MPPT architectures: the conventional central inverter architecture (a), series-connected DC-DC to a central inverter (b), parallel-connected DC-DC to a central inverter (c), and micro-inverters (d). Figure extracted from [79].

Next, it is possible to replace the single central converter with multiple converters in a schema called **distributed architecture** where each converter handles the MPPT of each module in the array. These could be found in studies such as Solorzano et al. [61], Saranrom et al. [83], Gao et al. [84]; thesis by Stéphane Petibon [85], thesis by Cédric Cabal [86], and thesis by Angel Cid Pastor [87]. Distributed architecture is a good choice for large installations with hundreds or more solar panels because a single converter handling hundreds of potential power peaks is simply not practical. Because of this, they are more frequently applied in AC grid applications, and currently there are three commonly used schemas: series connected DC-DC to a central inverter, parallel connected DC-DC to a central inverter, and micro-inverters (Figure 3.2). Of course, these architectures are not AC exclusive and could be easily adapted to supply a DC output. This shading management technique is very effective because each converter performs a highly

efficient MPPT, and it also comes with an additional benefit. In Chapter 2, we have seen that the I-V of a string of PV blocks is not affected when we permute the order of the shaded modules, and we only know that shading is indeed present. Applying the same logic to module failure, it would manifest itself in the I-V, but it is impossible to pinpoint which one has failed without characterising each module. Considering that a large-scale solar panel installation could have up to thousands of solar panels, the task of identifying the failed module would be extremely tedious. Distributed architecture would provide the diagnostic capability to easily find the failed module.

We believe that this should be used when possible. As we shall see at the end of this chapter, GMPPT algorithms and operation under partial shadings have some inherent challenges that are hard to overcome. But distributing to every single panel would also not be very practical in terms of cost and complexity, so combining distribution and GMPPT for a distributed GMPPT schema could be a good compromise.

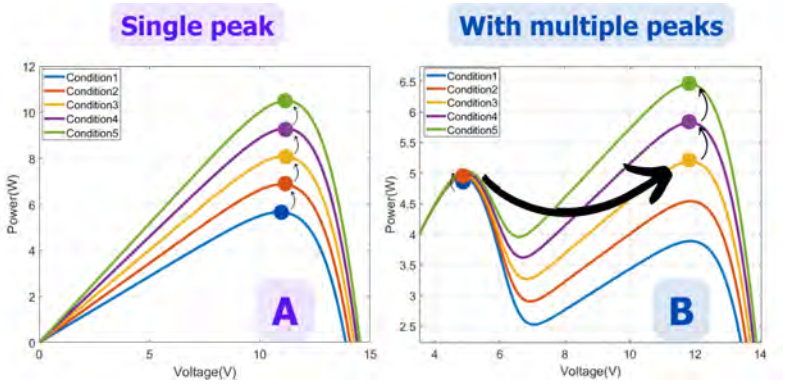


Figure 3.3: Illustration of how MPP evolves under varying irradiance when there could only be a single peak and when multiple peaks may be present. We suppose here that the power output changes progressively from condition one to condition five.

There is a fundamental problem in the characteristics of a partially shaded PV string with bypass diodes that needs to be discussed before we proceed. This is illustrated in Figure 3.3. A maximum power point **tracking** algorithm operates when the array only has one power peak. Under varying irradiance and temperature, this peak will stay in a relatively narrow voltage zone, as shown in Figure 3.3A. This means that during searches, the algorithm is **not required to depart** from this zone and therefore does not suffer a significant power loss during the search. But when partial shading comes into play, we have situations that resemble Figure 3.3B where the maximum power peak suddenly occurs at a distant voltage point. Therefore, all GMPPT algorithms **are required to search a wide range of voltages** to detect the maximum power peak, which is called **global searches**. During this phase, the algorithm will inevitably land on various non-optimal operating points

and suffer power drops. This information will be important to evaluate the efficacy of GMPPT by the end of this chapter.

We start with a review of the literature in which we explore, classify, and critique the various MPPT and GMPPT algorithms. From this evaluation, we propose our own lightweight GMPPT algorithm based on the information gathered in Chapter 2. Seeing that there is currently no procedure that fits our research context of varying partial shadows, a novel testing configuration will be proposed to put these algorithms under **constantly fluctuating shadows over time**. Finally, we test our algorithm against three other algorithms in the literature and provide some insight into the GMPPT problem as a whole.

Contents

3.1 Literature review 92
3.1.1 Overview of software MPPT 92
3.1.2 In-depth analysis of MPPT algorithms 93
3.1.2.1 Perturb and Observe - Hill Climbing (P&O/HC) . . 93
3.1.2.2 Incremental Conductance (InC) 95
3.1.2.3 Beta parameter 96
3.1.2.4 Equivalent model 98
3.1.2.5 Fractional open circuit voltage (fVoc) 98
3.1.2.6 Temperature and irradiance measurement 98
3.1.2.7 Fuzzy logic controller (FLC) 99
3.1.2.8 Neural network (NN) 100
3.1.3 In-depth analysis of GMPPT algorithms 101
3.1.3.1 Voltage scanning 101
3.1.3.2 nVoc method 101
3.1.3.3 Probabilistic approach 102
3.1.3.4 Optimisation-based algorithms 102
3.1.3.5 Deterministic Particle Swarm Optimisation 103
3.1.3.6 Grey Wolf Optimisation 106
3.1.4 State of the literature 107
3.1.4.1 Simulation results 108
3.1.4.2 Experimental results 109
3.1.4.3 Steady state phase and irradiance perturbation detection 109
3.1.4.4 Choice of test profiles 111
3.1.4.5 Result presentation 112
3.2 Probabilistic GMPPT algorithm 114
3.2.1 Algorithm description 114
3.2.2 Theoretical evaluation of the search mechanism 116
3.3 Validation of our algorithm proposal 118
3.3.1 Test profiles 118
3.3.2 Test setups 120
3.3.3 Results for a sequence of irradiance steps 121
3.3.4 Results for under varying irradiance profiles 123
3.4 Conclusion and perspective of Chapter 3 128

3.1 Literature review

3.1.1 Overview of software MPPT

This review consists of **duty cycle control** and **online** methods that are applicable to **standalone harvesting using a DC-DC converter**. Duty control means that the algorithm generates a duty cycle directly to drive the DC-DC chopper instead of a reference voltage value. This negates the need for an analogue PID controller to regulate the input voltage of the converter, but we do not exclude the use of simple digital proportional controller. Being online means that the algorithm must be performed without disconnecting the solar panels. The standalone harvesting architecture that we target is a string of PV modules in series with bypass diodes harvested by a buck converter to charge a stable voltage load (i.e. supercapacitor, battery).

We first set out to classify existing algorithms into clear groups and superficially compare them. The first criterion is the nature of the algorithm. A method is said to be **characteristics-based** (CB) when inspired by the physical properties of solar panels such as the form of the P-V curve or the negative temperature coefficient K_v of Si-based solar panel. Algorithms that are not based on the characteristics of the array could be **intelligent control** (IntelC) or **optimisation algorithms** (OptA). Next, we are interested in their ability to track the global power peak, which we call being **multiple peak capable** (MPC). GMPPT algorithms are MPC while MPPT ones are not. Another important metric is whether the algorithm is **parameter-dependant** (PD), meaning that they require fine-tuning from a user before deployment. Consequently, its tracking quality may degrade when PV modules are damaged or age. Therefore, being non-PD would be more advantageous. The set of measurements needed for the algorithm is also of interest, since some measurements are harder than others and monitoring too many variables may also introduce more noise. We compile the characteristics of several methods in Table 3.1 and a good paper to study them in detail. Since most methods have multiple contributions, we only provided reference to one that best explained their operating principle.

Generally, CB algorithms are good at tracking, but if the PV array has damaged modules, the tracking may not work properly, with the exception of voltage scanning. IntelC methods have a very fast convergence time, but they are highly dependent on the array's parameters and require extensive fine tuning before deployment. Finally, OptA algorithms are a great choice when multiple peaks are present and they also boast fast convergence time, but they do not guarantee convergence to GMPP because of their metaheuristic nature. To go deeper than this superficial analysis, we need to look into how each of them operates and discuss their merits and demerits. All technical examples will be based on a string of four PV blocks connected to a buck converter that delivers current to a stable voltage load similar

to Figure 2.33. The characteristics of the photovoltaic modules are shown in Table 2.1.

MPPT method basic comparison table				
Algorithm name	Nature	MPC	PD	Sensor data
Perturb and Observe / Hill Climbing [88]	CB	No	No	Voltage, Current
Incremental Conductance (InC) [89]	CB	No	No	Voltage, Current
Fractional open circuit voltage [90]	CB	No	Yes	Voltage
β-parameter [91]	CB	No	Yes	Voltage, Current
Equivalent model [92]	CB	No	Yes	Voltage, Current, Temperature
Temperature based [93]	CB	No	Yes	Voltage, Temperature
Neural network [94]	IntelC	No	Yes	Voltage, Current
Fuzzy logic controller (FLC) [95]	IntelC	No	Yes	Voltage, Current
nVoc method [63]	CB	Yes	Yes	Voltage, Current
Voltage scanning [96]	CB	Yes	No	Voltage, Current
Probabilistic approach [97]	CB	Yes	Yes	Voltage, Current
Particle Swarm Optimisation [98]	OptA	Yes	No	Voltage, Current
Differential Evolution Optimisation [99]	OptA	Yes	No	Voltage, Current
Grey Wolf Optimisation [100]	OptA	Yes	No	Voltage, Current
Artificial Bee Colony Optimisation [101]	OptA	Yes	No	Voltage, Current
Dragonfly Optimisation [102]	OptA	Yes	No	Voltage, Current
Grasshopper Optimisation [103]	OptA	Yes	No	Voltage, Current
Flower Pollination Optimisation [104]	OptA	Yes	No	Voltage, Current
Ant Colony Optimisation [105]	OptA	Yes	No	Voltage, Current
Flashing Fireflies Colonies Optimisation [106]	OptA	Yes	No	Voltage, Current
Cuckcoo Search [107]	OptA	Yes	No	Voltage, Current
Student Psychology [108]	OptA	Yes	No	Voltage, Current
Most Valuable Player Algorithm [109]	OptA	Yes	No	Voltage, Current
Teaching-Learning Based [110]	OptA	Yes	No	Voltage, Current
Fibonacci Search [111]	OptA	Yes	No	Voltage, Current
Simulated Annealing [112]	OptA	Yes	No	Voltage, Current
Henri Gas Optimisation [113]	OptA	Yes	No	Voltage, Current

Table 3.1: Basic compairison of existing MPPT methods.

3.1.2 In-depth analysis of MPPT algorithms

3.1.2.1 Perturb and Observe - Hill Climbing (P&O/HC)

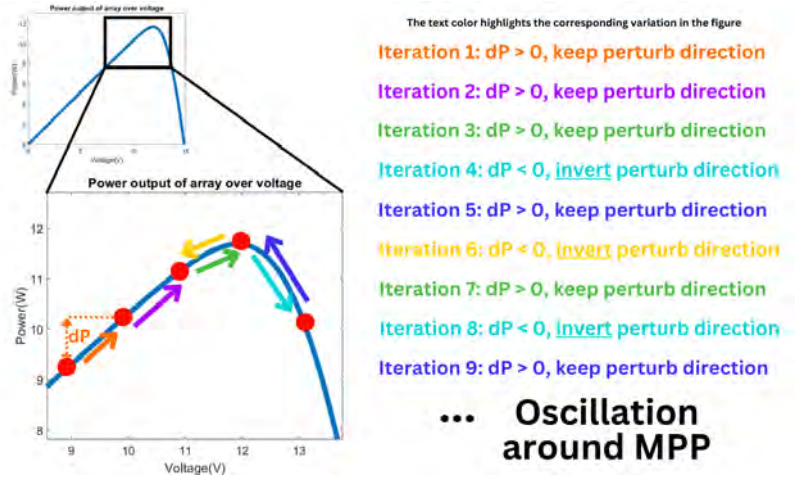


Figure 3.4: Illustration of how P&O/HC track MPP. The P-V characteristics is that of four PV modules in series with four bypass diodes under even $1000W^{-2}$ irradiance.

This is by far the most widely used and discussed method. Its working principle is to vary the duty cycle of the converter in a particular direction (increasing or decreasing) to slightly perturb the operating point and measure the newly acquired

power. If the array's power has increased, the controller continues the perturbation in the same direction, and inverts the direction if it has decreased. While P&O and HC are essentially the same, their naming difference lies in the control signal that is perturbed: the voltage of the array for P&O and duty cycle for HC. Since duty cycle control is our focus, the name HC will be used from this point on. An illustration of how the algorithm tracks the MPP is provided in Figure 3.4 with an exagerated voltage variation for better clarity.

Overall, this is a very simple MPPT algorithm, only requiring a comparison between the power samples from the previous and current iterations to make a decision. Furthermore, it is non-PD so it works with any PV array once the controller was flashed.

However, it is not capable of tracking the global peak, so this algorithm may not be suitable for situations where multiple local peaks are present [64], illustrated in Figure 3.5.

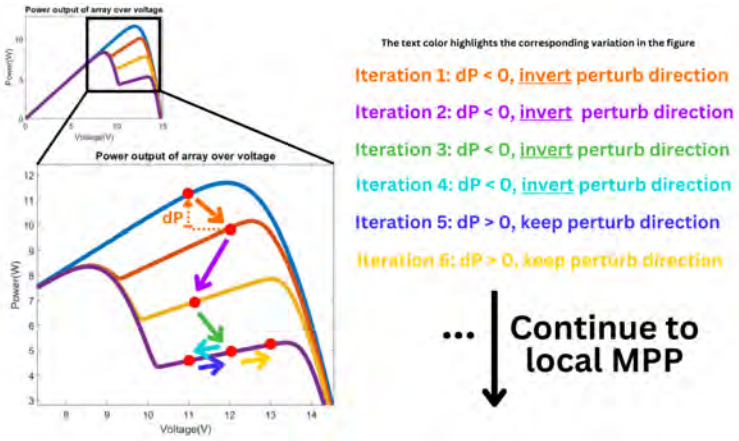


Figure 3.5: Illustration of how P&O/HC fails to locate MPP. The P-V characteristics belongs to a string of four PV blocks under $1000W^{-2}$ irradiance, but one module is progressively shaded to 800→600→$400W^{-2}$ creating a partial shading condition.

The first problem is the oscillation around MPP previously shown in Figure 3.4. Studies by Ahmed et al. [114] and Killi et al. [115] proposed a variable duty cycle step size that is gradually reduced every time an oscillation is detected, resulting in a much more stable power output.

The second problem is the slow convergence time if the starting operating point is far away from the MPP. The solution could be to limit the voltage search window such as proposed by Ahmed et al. [114] or Scarpa et al. [93]. A slight inconvenience is that the added voltage search window requires knowledge of the PV array to implement, making it PD. A more advanced way of achieving this is to add another algorithm that is faster in locating the general MPP region before initiating hill climbing (e.g. β-parameter by Jain et al. [91] (PD), PSO-aided by Lian et al. [116]

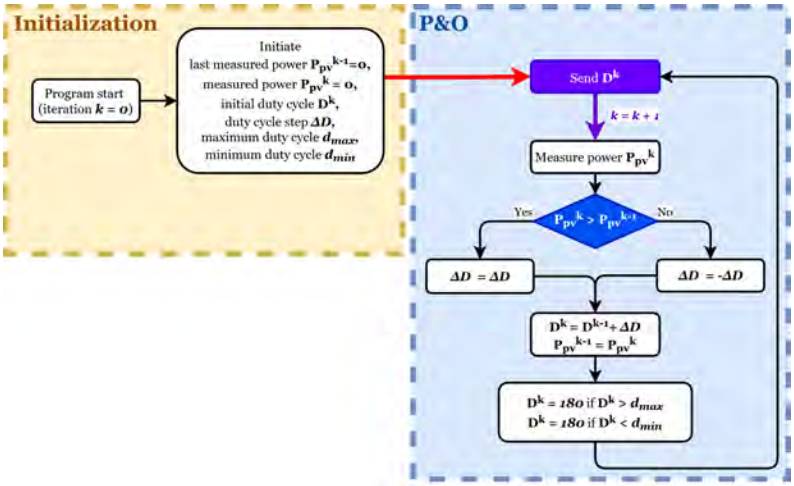


Figure 3.6: Detailed flowchart of HC as we implemented.

(non-PD), ABC-aided by Pilakkat et al. [117] (non-PD)). These are called **hybrid algorithms** and the overall schema could also be made multiple peak capable.

The third problem with HC is the loss of tracking in a rapidly increasing irradiance, where the algorithm momentarily diverges away from the MPP region [115]. This is because the controller cannot distinguish whether the higher power output is caused by its perturbation or by the externally increasing irradiance. Two proposed solutions involved adding an additional measurement cycle between two duty cycle updates (Sera et al. [118]) and adding an evaluation of the change in the current of the array (Killi et al. [115]). Neither of these improvements adds any significant drawback.

Since HC is very widely used, it will be included to compare with our algorithm despite not being a GMPPT algorithm. The detailed flow chart of HC is shown in Figure 3.6. We fix the duty cycle step $\Delta D = 0.78\%$ ($\frac{2}{255}$). d_{min} and d_{max} are the minimum and maximum values of the duty cycle, respectively, fixed around 5% and 95% of the full duty cycle range. Seeing that reaching these limits likely means that HC has lost track of MPP, we reset the duty cycle to 70% ($\frac{180}{255}$) and a new search is initiated.

3.1.2.2 Incremental Conductance (InC)

InC's core principle relies on the derivative of power over voltage of the P-V characteristic as shown in Figure 3.7. The algorithm seeks to increase the voltage operating point when $\frac{dP}{dV} > 0$, decrease it when $\frac{dP}{dV} < 0$ and stop the search when $\frac{dP}{dV} = 0$. Due to measurement errors, the principle in practice is rather to increase the voltage operating point when $\frac{dP}{dV} > \varepsilon$, decrease it when $\frac{dP}{dV} < -\varepsilon$ and stops the search when $-\varepsilon < \frac{dP}{dV} < \varepsilon$, where $\varepsilon > 0$ is higher than the noise level of the measurement.

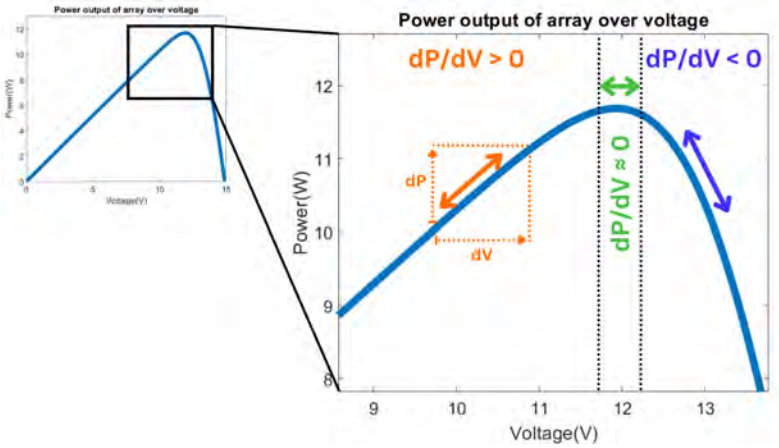


Figure 3.7: Derivative of power over voltage in three different zones across the voltage range of the PV array.

$$\frac{dP}{dV} = \frac{d(IV)}{dV} = \frac{d(I)V}{dV} + \frac{d(V)I}{dV} = V\frac{dI}{dV} + I \tag{3.1}$$

Since the algorithm has a steady state, a problem arises where the current changes due to varying irradiance, causing a change to dP yet $dV = 0$. This leads us to the actual implementation of the algorithm in practice. A mathematical transform is made as shown in equation 3.1. The first check for voltage variation is performed, in which case the algorithm will proceed with evaluating $\frac{dP}{dV}$. If the voltage was instead constant since the last update, a check on the current variation dI is performed. If $dI \neq 0$, then a duty cycle step will be applied according to the sign of dI.

InC is only slightly more complex and HC, requiring some extra decision making, and is also non-PD. But it shares the same three demerits with HC: being non-MPC, having slow convergence time, and may momentarily diverge from the MPP under varying irradiance. The mitigations for these problems are also similar to HC such as introducing variable step size (Liu et al. [89]) or some other techniques to help limit the search window (Hsieh et al. [119]). However, given that its performance is similar to HC, we struggle to see any inherent advantage of InC because it requires more computation and is more sensitive to measurement errors.

3.1.2.3 Beta parameter

$$\begin{aligned} \beta &= ln(\frac{I}{V}) + c * V \\ c &= \frac{q}{AkT} \end{aligned} \tag{3.2}$$

This is an intermediary tracking variable first proposed by Jain et al. [91] that is used to appropriately size the duty cycle step. They first introduce a linear variable to the array voltage β with the mathematical expression found in equation 3.2,

where q is the electronic charge in $A.s$, k is Boltzmann's constant in $m^2.kg.s^{-2}.K^{-1}$, A is the ideality factor of the PV module and T is the ambient temperature 298.15K. It has the interesting property of being highly dependent on temperature and mostly independent of irradiance (Figure 3.8). When operating at MPP, we can expect β_{mpp} to fall in a range of β_{min} and β_{max}. The authors have determined that the best choice for these two parameters would be β_{mpp} at the lowest and highest expected temperature and irradiance, respectively (corresponding to $\beta 3$ and $\beta 2$ Figure 3.8). A value of β_g within this range is chosen so that it corresponds to the most probable operating temperature of the PV module. By measuring β on each sample, we can use $\beta_{error} = \beta - \beta_g$ to size an appropriate duty cycle step in the form of $k\beta_{error}$, where k is an appropriate proportional coefficient [120]. Overall, the parameter proved to be effective in reducing the convergence speed to MPP when used in conjunction with another method such as HC [121] or FLC [95].

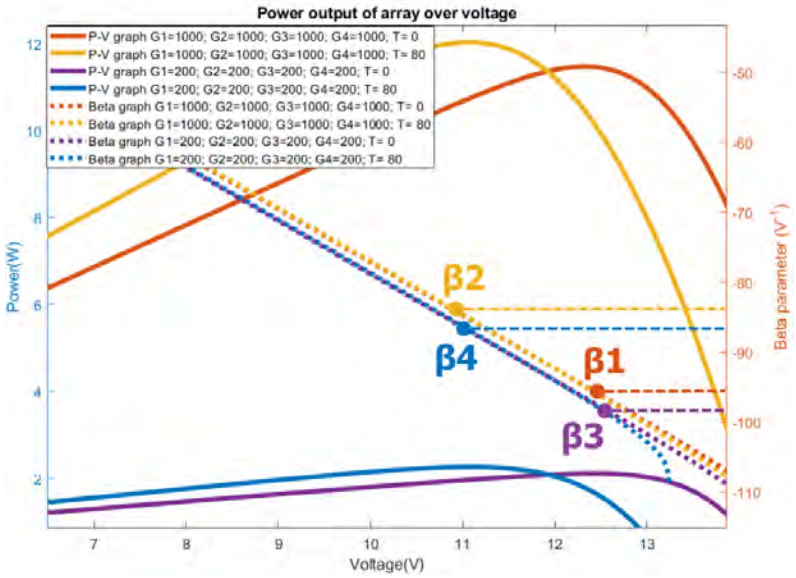


Figure 3.8: Graph displaying the value of β parameter when operating at MPP for four conditions of temperature and irradiance. Irradiance of the four PV modules (G1, G2, G3, G4) are given in Wm^2 and their temperature T is given in $°C$.

There are several flaws with the β parameter approach. First, we graphed the β parameter of our string under partial shading conditions and confirmed that it is indeed linear even under partial shading, so it could not help us detect the local peaks. Second, determining the β_{min} and β_{max} requires an extra step to determine the parameters of the solar panel, in which case we could just determine the minimum MPP voltage V_{mpp_min} and maximum MPP voltage V_{mpp_max}. V_{mpp} has the same properties as β_{mpp}, both of which are highly dependent on temperature and mostly independent of irradiance, β to also linear with voltage of the array. We can then choose a V_{mpp_g} that corresponds to the most likely operating temperature of the PV module, measuring its distance from the current operating point $V_{error} = V - V_{mpp\_g}$ and use this quantity to size an appropriate duty cycle step with a

simple proportional controller. This is actually the principle behind the variable step size HC proposed by Kollimalla and Mishra [122].

3.1.2.4 Equivalent model

The almost flat current plateau where the voltage of the array is below V_{mpp} is the main inspiration for this technique. With the knowledge of the current and voltage, as well as the module temperature, it is possible to reconstruct the profile and determine the voltage at the MPP. Farivar et al. [92] have implemented this method with the help of the Lambert function, and the result is an immediate convergence to MPP after one single measurement. Of course, this great capability comes with great complexity. It is heavily parameter-dependent, sensitive to parameter errors as well as measurement errors, requires temperature measurement of the PV module, which is not trivial, and the algorithm generates a reference voltage value for the converter to regulate instead of a direct duty cycle. To directly generate a duty cycle, we also now need to know the exact characteristics of the converter and the load. Furthermore, it requires complex computation, albeit very few cycles, due to its high convergence speed. A simpler option would be to use a Thevenin equivalent model of the entire system as proposed by Moradi et al. [123] to directly determine the duty cycle to put the system in MPP, but the process still requires an offline measurement phase and characterisation not only of the photovoltaic array but also of the converter and load.

3.1.2.5 Fractional open circuit voltage (fVoc)

Seeing that the MPP is typically around 80% of Voc, we can set the controller to always force the voltage of the array to this value. Of course, this means that the energy efficiency is not the best. However, there are situations where fractional open circuit voltage is useful. First, when irradiance is very low, the signal-to-noise ratio dramatically decreases, leading to unreliable measurements. At these conditions, any algorithms depending on accurate measurements could lose tracking of the MPP anyway, and therefore forcing a constant array voltage would be a better option. Second, it is more suitable for ultra-low cost harvesting applications that do not require the highest efficiency [90]. There are also several simple commercial MPPT converters for ultra low power harvesting that employed fVoc such as the BQ25570 [124] and the SPV1050 [125].

3.1.2.6 Temperature and irradiance measurement

Measuring the irradiance and temperature for MPPT purposes is a logical choice, given that MPP depends directly on these variables. Examples include the study by Lei et al. [126] where MPPT was performed with irradiance measurement and the study by Moradi et al. [127] where MPPT was performed with temperature measurement. However, the MPP efficiency relies heavily on parameter's accuracy and the algorithm requires extra sensors. Furthermore, accurately determining the

temperature of an encapsulated PV cell is difficult, and obtaining a good irradiance reading, for which the equipment is already costly, might still not accurately represent the irradiance received by the PV modules under PSC.

3.1.2.7 Fuzzy logic controller (FLC)

Fuzzy logic was inspired by the way we humans make decisions. Instead of assigning a binary TRUE or FALSE to how cold it is outside, we describe it as a range, like "a bit cold", "cold", or "very cold". Numerically, this is represented by a range of values from 0 to 1 called **logic values**. A fuzzy logic controller receives a set of input variables and outputs a set of control signals to drive the system, similar to a PID controller. The Mamdani rule-based schema is the most widely used for MPPT applications and works in three steps: **fuzzification**, **inference with a rule base**, and **defuzzification** [95].

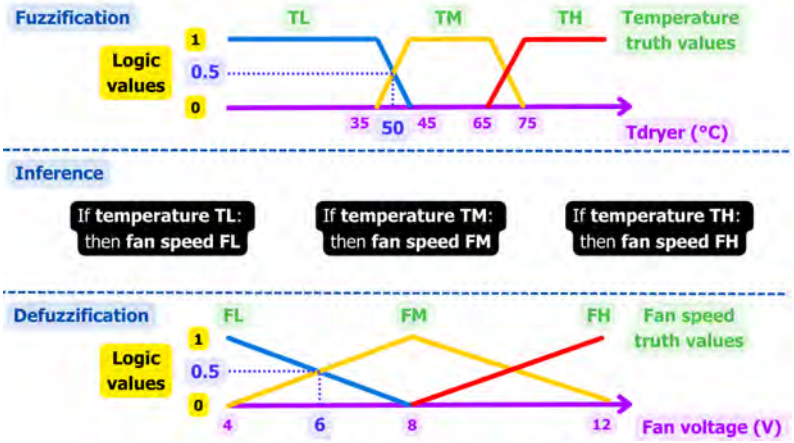


Figure 3.9: Representation of the three steps in a FLC: fuzzification, inference, and defuzzification.

Let us take a simple example to better understand how an FLC works (Figure 3.9). Imagine a simple hair dryer with a fan and a heating coil powered by a constant current, where we measure the temperature of the air and control the fan speed. When using the hair dryer, the user could perceive the air temperature to be either just warm (TL, Temperature Low), hot (TM, Temperature Medium) or too hot (TH, Temperature High). Fuzzification means assigning a "rating" to the measured temperature according to these "feelings", mathematically called truth values. For example, $40°C$ air could be considered a bit more than warm, but still not quite hot, so its mathematical representation could be "0.5TL and 0.5TM". We also have a set of truth values for our fan speed, like low (FL, fan low), medium (FM, fan medium), and high (FH, fan high). When the temperature is not high enough, we would like the fan to spin down for a hotter air flow. This is where **inference with a rule base** comes in. When the temperature is high, we want the fan speed to be high, when the temperature is medium, we want the fan to spin at medium, and when the temperature is low, we want the fan to be low. Continuing with

our example, a measurement of "0.5TL and 0.5TM" becomes a desired output of "0.5FL and 0.5FM" after inference. However, the fan speed is driven by an analogue voltage signal, so we have to convert "0.5FL and 0.5FM" to an analogue value. This is **defuzzification**, where we map the range of desired fan speed truth values to a continuous range of the analogue fan control voltage. Building the appropriate controller consists of selecting the number of truth values for input and output, as well as the appropriate functions for each of the three steps.

In general, FLC gives good tracking results as reported by Alajmi et al. 2011 [128], Alajmi et al. 2013 [129], El Khateb et al. [130], Shah et al. [131], Kumar et al. [132], Yilmaz et al. [133], and Li et al. [95]. However, the showcased results are mostly on par with simpler techniques such as variable step HC by Ahmed et al. [114] or variable step InC by Liu et al. [89], making the added complexity hard to justify, more so that the method itself is not MPC and is highly PD. Although FLC applied to MPPT does not require complex mathematical operations, determining the optimal parameters for the fuzzification and defuzzification process is, however, very tedious.

3.1.2.8 Neural network (NN)

Machine learning technique is also frequently explored for MPPT applications, in this case specifically the feedforward neural network. A simple description of this network is a set of layers, with one input layer receiving sensor information, one or more hidden layers, and one output layer that generates the control signal to our converter (Figure 3.10). The sensor readings are processed through the layers until it reaches the output, hence the name feedforward. They are useful for non-linear and abstract problems, which are indeed the characteristics of a solar panel's P-V curve [134]. The performance of using NN is well proven by the results from studies by Meng et al. [134], Rahman et al. [94], Agha et al., [135], and Jyothy et al. [136].

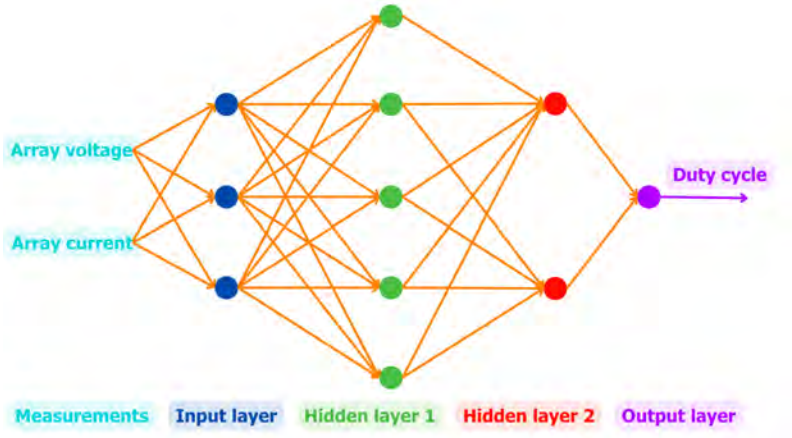


Figure 3.10: Schema of a neural-network-based MPPT controller with an input layer, an output layer, and 2 hidden layers.

Just like FLCs, we perceive the gain in convergence speed and efficiency to be not worth the complexity added to the system, since implementing the NN requires an accurate and large dataset to train and a capable microcontroller. This means that NN is highly PD and is very time consuming to implement on a new set of PV array. From a data science point of view, it is important to point out some demerits of these NN MPPT proposals. First, some lack the justification for why a specific type of model was used [134] or for how the number of neurones was chosen [136]. Second, some models were trained with a small dataset that could not guarantee the reliability of the result, such as [135] and [94]. Finally, although the problem was indeed non-linear and abstract, we do have the full mathematical description of the P-V curve. Given that NN requires high computation, good knowledge of the characteristics of the array and the converter, it could be simpler to use the Lambert function to solve for MPP where Farivar et al. [92] also achieved instant MPP convergence.

3.1.3 In-depth analysis of GMPPT algorithms

3.1.3.1 Voltage scanning

This schema marks the passage to GMPPT algorithms where they are considered mulitple peak capable. Voltage scanning means that the controller periodically or contextually initiates a full range duty cycle variation from 0% to 100% (or more likely 5% to 95% in practice) to determine which duty cycle yields maximum power. It is not parameter dependent, does not require any significant computation, and guarantees convergence toward GMPP. It is also the only method that provides data for a potential diagnosis of the state-of-health of the photovoltaic array if necessary. However, there is a trade-off: a smaller duty cycle step ensures higher tracking accuracy but takes longer to converge, and vice versa. Voltage scanning is rarely used alone, but rather combined with another method like HC (Ghasemi et al. [96]) or FLC (Shah et al. [131]).

3.1.3.2 nVoc method

Improving on voltage scanning by considering the PV array's parameters, we have nVoc method. It is a common understanding in the GMPPT research community that GMPP occurs at regular intervals on the PV array's voltage range, a fact that we have also visualised in the previous chapter. Its name was inspired by the simple GMPP estimate found in Section 2.1.4. By initiating a MPPT algorithm in different zones of the voltage range, the controller can evaluate the multiple local peaks and select the global maximum power point. Therefore, nVoc is not a standalone algorithm, but rather an extension to MPPT algorithms to handle PSC. It was used with HC by Ramyar et al. [62] and with InC by Tey et al. [64] with good experimental results. Although the search range is narrower than voltage scanning, letting the MPPT algorithm finish in each zone before moving on to the next still results in an overall slow convergence time toward GMPP.

3.1.3.3 Probabilistic approach

This approach was first found in a research work by Tang et al. [137] where the authors drastically reduced the voltage scan to only 24 points and the one having the highest power was chosen as the seed to begin the HC phase. While the paper in itself is not interesting with limited simulation testing and no experimental setup, it is the only other work besides our own in this category, as far as we can find. Our contributions to this category were documented in two conference papers: [138] with simulation results and [97] with experimental results, which will be discussed in detail later. In general, these algorithms are parameter-dependent and have a fast convergence time. However, it might sometimes not converge toward GMPP due to the probabilistic nature, which is a demerit compared to nVoc and voltage scanning.

3.1.3.4 Optimisation-based algorithms

Optimisation is a process that determines the optimal value of a function and is used in almost all scientific fields. In the GMPPT context, it is determining the GMPP of the PV array with the mathematical representation found in equation 3.3 where P is the power of the array and D is the duty cycle.

$$maxP = f(D) \tag{3.3}$$

The principles of all algorithms based on optimisation are similar and could be distinguished into two different phases: **evaluation** and **decision**. The evaluation phase consists of gathering information on the power profile by sending a set of random duty cycles, mathematically called **candidate solutions**, and observing how much power is harvested at these points. Based on these measurements, a decision is then made on whether to continue the search with a new set of duty cycles or to stop the search. Each completed evaluation phase is called a **search iteration** (shortened to **s-iteration**) of the algorithm, and the decision phase occurs at the transition from one iteration to the next. This is to distinguish it from an iteration which we have defined in Figure 2.34 in the previous chapter. Since only one candidate solution is checked per sampling cycle, the time to complete each s-iteration is the number of candidate solutions times the sampling period. Throughout the evaluation decision cycle, the maximum power achieved is recorded among other parameters that different algorithms require. A stopping criterion is checked at every decision phase, and if met, the search is terminated, and the converter will then be driven by the duty cycle yielding maximum power.

We proceed with an example to see how an optimisation algorithm works in the context of MPPT (Figure 3.11). This generic algorithm will be performed with three candidate solutions per s-iteration, and a maximum of three s-iterations is the termination criterion. This means that each s-iteration takes three sampling periods to complete. S-iteration 1 starts with an evaluation of three randomly

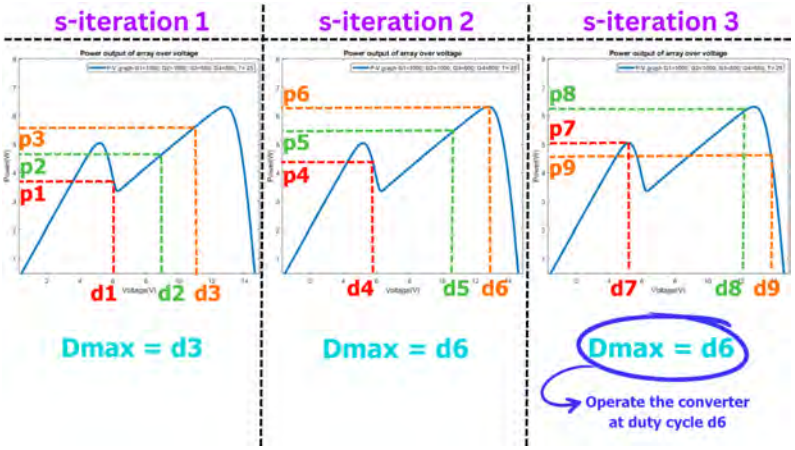


Figure 3.11: An illustration for 3 steps of a generic optimisation algorithm.

chosen duty cycles $\{d_1, d_2, d_3\}$ and the controller observes three associated power values $\{p_1, p_2, p_3\}$. The decision phase is performed where a set of new duty cycles $\{d_4, d_5, d_6\}$ is calculated. At s-iteration two, this new set is then sent, and the controller observes three associated power values $\{p_4, p_5, p_6\}$. The process is then repeated until we finish iteration three. At this point, the termination criterion is reached, and the search is stopped.

The biggest advantage of optimisation-based MPPT is their independence from the characteristics and parameters of the PV array. However, they could be computationally complex, caused by the calculations performed at each decision phase. Furthermore, their random nature means that they sometimes do not converge toward GMPP ([112]).

In the following sections, two specific optimisation algorithms will be discussed: deterministic particle swarm optimisation (DPSO) and grey wolf optimisation (GWO) ([98] and [139] respectively). There are several reasons for this choice, in addition to the fact that they are both well documented, simple algorithms with good simulated and experimental results. First, they have different termination criteria. Second, one is deterministic, while the other one is randomised. This gives us a diversified view of how some characteristics would impact the algorithm's performance. For improved clarity, a lot of descriptive language would be used, and their mathematical representation would be adapted to our specific MPPT context. These optimisation algorithms were chosen because there seems to be a lot of interest in this technique and the result seems to always be very promising.

3.1.3.5 Deterministic Particle Swarm Optimisation

The first optimisation-based MPPT algorithm was introduced in [140] by Miyatake et al. based on PSO. A descriptive way to summarise its search mechanism is by imagining a set of agents moving throughout the range of duty cycles and each one

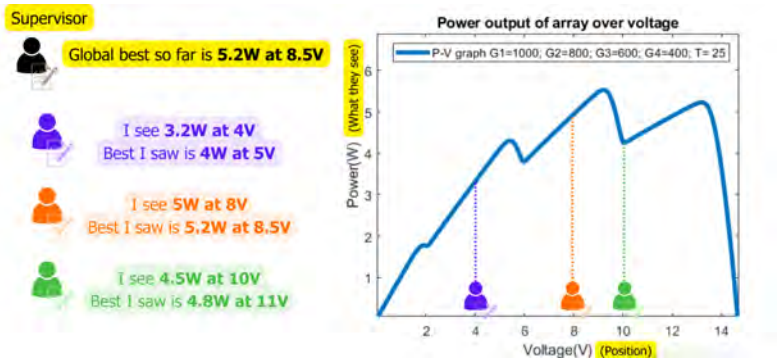


Figure 3.12: A descriptive way to visualize the PSO's mechanism.

reporting the best power value observed (Figure 3.12). An agent i among n agents at s-iteration x moves to its assigned position at duty cycle d_x^i and records a power value of p_x^i. The controller surveys the personal best power p_x^i of each agent at s-iteration x (and the associated duty cycle db_x^i at this personal best power) as well as the global best power g_{best} achieved by all agents (and the associated duty cycle d_{best} at global best power). Finishing the iteration, each agent moves toward their new duty cycle value d_{x+1}^i (equation 3.5) with a step of v_{x+1}^i (equation 3.4) where w is the momentum coefficient, c_1 and c_2 are acceleration coefficients, and r_1, r_2 are uniformly distributed random numbers between 0 and 1. w, c_1, c_2 are subjected to fine-tuning before deployment for the optimal search result.

$$v_{x+1}^i = wv_x^i + c_1r_1(db_x^i - d_x^i) + c_2r_2(d_{best} - d_x^i) \tag{3.4}$$

$$d_{x+1}^i = d_x^i + v_{x+1}^i \tag{3.5}$$

$$v_{x+1}^i = wv_x^i + db_x^i + d_{best} - 2d_x^i \tag{3.6}$$

Seeing that the MPPT problem does not necessarily benefit from a randomised process, the authors of [98] proposed to make the algorithm deterministic where c_1, r_1, c_2, r_2 equals 1 and equation 3.4 becomes equation 3.6. There are numerous advantages to DPSO: it is simpler to fine-tune by the user with only w left to be determined and it is very lightweight. Furthermore, being deterministic makes the convergence time and rate consistent. However, eliminating randomisation limits the exploration capacity of each agent. Therefore, if somehow all of them missed the MPP region, recovery would be impossible. To combat this, the authors suggested an upper limit to v_{x+1}^i that we called ΔD_{max} and its value was fixed to about 8% ($\frac{20}{255}$) of the full duty cycle range .

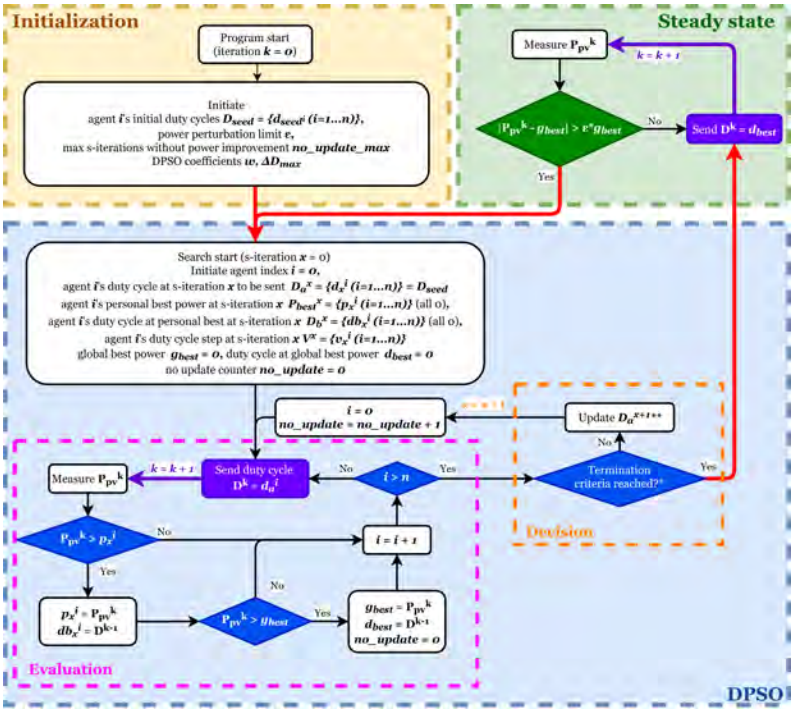


Figure 3.13: Detailed flowchart of DPSO as we implemented. *: either when equation 3.7 or 3.8 is reached. **: using equation 3.5 and 3.6

The detailed flow chart of the algorithm is provided in Figure 3.13. Since three candidate solutions were the choice of the authors, we also implemented ours with three candidate solutions. w was fixed to 1 because it works well and has the added benefit of reducing computation steps. The initial search duty cycle are chosen far apart on the duty cycle range. The search is terminated when the absolute differences of at least two of the candidate solutions are smaller than db_{limit} (equation 3.7) or when a maximum number of s-iterations without any power improvement no_update_max is reached (equation 3.8). Equation 3.7 is used to accelerate the convergence time since reaching it usually means that the GMPP was found. Most OptA algorithms that do not have an s-iteration cap usually relies on the second criterion based on equation 3.8.

$$\exists a, b \in 1..n, |db_x^a - db_x^b| < db_{limit} \tag{3.7}$$

$$no\_update > no\_update\_max \tag{3.8}$$

The algorithm has a steady state phase in which the microcontroller sets the converter at the global maximum power point found P_{gmpp} and only monitors the power output P_{pv}^k for any changes $|P_{gmpp} - P_{pv}^k|$. When the power variation over the power at global maximum power point exceeds a certain ε (shown in equation

3.9), the controller will initiate a new global search. Note that the P_{gmpp} notation of equation 3.9 depends on the flow chart of each algorithm. We chose $\varepsilon = 1.5\%$ for all algorithms in this work.

$$\frac{|P_{gmpp} - P_{pv}^{k}|}{P_{gmpp}} > \varepsilon \tag{3.9}$$

3.1.3.6 Grey Wolf Optimisation

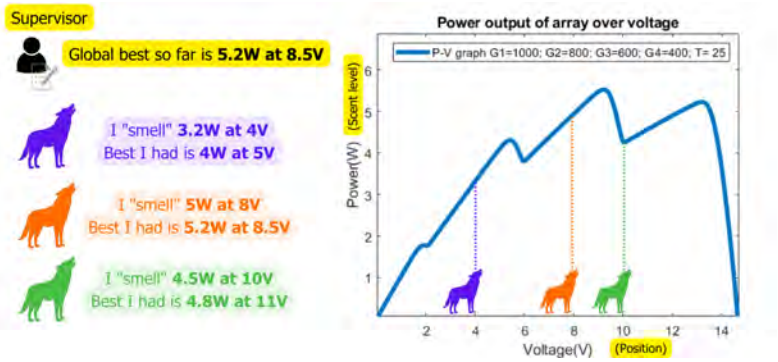


Figure 3.14: A descriptive way to visualize the GWO's mechanism.

This algorithm was inspired by how wolves move in a pack to find food. They use their sense of smell to sniff out a potential catch and the closer they are to their target, the higher the scent of the prey could be detected (Figure 3.14). To be efficient, the pack converges toward the position closest to the prey while still maintaining some sense of exploration to make sure that they are indeed approaching the right target.

Transferring to our MPPT context, the amount of scent that each wolf is detecting is equivalent to the power measured, and the position of the wolf is equivalent to the duty cycle. At s-iteration x, each wolf i among n wolves is at a position d_x^i and detects a certain amount of scent p_x^i from its prey. Only the results of the two best wolves, alpha and beta, are recorded. The alpha's position is d_α and the scent it detects is p_α, the beta's position is d_β and the scent it detects is p_β. The position of each wolf for the next s-iteration d_{x+1}^i is determined using the set of equations 3.10 where r_1 and r_2 are random numbers from 0 to 1, a_x is a linearly decreasing variable through each iteration from 2 to 0, A_1 and A_2 are A with two differently randomised r_1, C_1 and C_2 are C with two differently randomised r_2. The voltage with highest detected power is recorded throughout the process, and the algorithm stops searching when a_x reaches 0. GWO was proven to be competitive in GMPP tracking capability versus HC and PSO in the experimental results by Mohanty et al. [139].

GWO would be included in our algorithm performance comparison and the detailed flow chart can be found in Figure 3.15. All operations were implemented exclusively

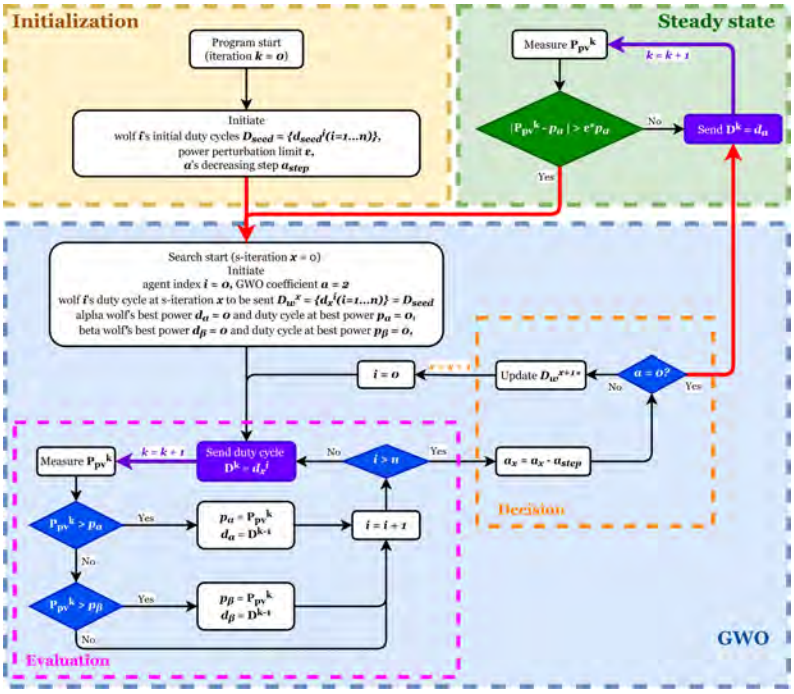


Figure 3.15: Detailed flowchart of GWO as we implemented. *: using equation 3.10

using integer maths. The randomisation of r_1 and r_2 are taken from an analogue measurement from a floating ADC pin of the microcontroller. The decreasing rate of a_x was chosen so that the algorithm would finish after eight s-iterations. The steady state phase is implemented similarly to DPSO.

$$\begin{aligned} A &= (2a_x - 1)r_1 \\ C &= 2r_2 \\ D_\alpha &= |C_1 d_\alpha - d_x^i| \\ D_\beta &= |C_2 d_\beta - d_x^i| \\ d_A &= d_\alpha - A_1 D_\alpha \\ d_B &= d_\beta - A_2 D_\beta \\ d_{x+1}^i &= \frac{d_A + d_B}{2} \end{aligned} \tag{3.10}$$

3.1.4 State of the literature

Before proceeding with our algorithm, it is important to look at how MPPT and GMPPT algorithms are presented in the literature and contemplate the various aspects of the algorithms so that we could come up with a good validation setup.

A total of 88 papers from 2000 to 2021 were evaluated from both journals and conference publications, giving us a total of 92 algorithms where 51 are MPC (55%).

Here is the exhaustive list of all papers that we included for this review: Sridhar et al. 2021 [103], Pervez et al. 2021 [109], Pal et al. 2021 [108], Lodhi et al. 2021 [102], Riquelme-Dominguez and Martinez 2020 [88], Mirza et al. 2020 [141], Mansoor et al. 2020 [142], Rahman et al. 2019 [94], Priyadarshi et al. 2019 [143], Pilakkat and Kanthalakshmi 2019 [117], Meng et al. 2019 [134], Li et al. 2019 [95], Yilmaz et al. 2018 [133], Tey et al. 2018 [144], Pei et al. 2018 [145], Kumar et al. 2018 [132], Jyothy et al. 2018 [136], Titri et al. 2017 [146], Rezk and Fathy 2017 [110], Ramyar et al. 2017 [62], Prasanth Ram and Rajesekar 2017 [104], Mohanty et al. 2017 [139], Fang et al. 2017 [63], Cherukuri et al. 2017 [147], Agha et al. 2017 [135], Veerasamy et al. 2016 [148], Shah et al. 2016 [131], Manickam et al. 2016 [149], Lyden et al. 2016 [112], Li et al. 2016 [121], Ghasemi et al. 2016 [96], Ahmed et al. 2016 [114], Ahmed et al. 2016 [150], Tang et al. 2015 [137], Sundareswaran et al. 2015 [101], Sundareswaran et al. 2015 [151], Seyedmahmoudian et al. 2015 [152], Li et al. 2015 [120], Killi et al. 2015 [115], Kermadi et al. 2015 [153], Dahhani et al. 2015 [154], Benyoucef et al. 2015 [155], Tey et al. 2014 [99], Tey et al. 2014 [64], Tajuddin et al. 2014 [156], Sundareswaran et al. 2014 [106], Lian et al. 2014 [116], Kollimalla et al. 2014 [122], Jiang et al. 2014 [105], Faraji et al. 2014 [157], Elnosh et al. 2014 [158], El Khateb et al. 2014 [130], Boztepe et al. 2014 [159], Ahmed and Salam 2014 [107], Moradi et al. 2013 [123], Ishaque et al. 2013 [98], Hsieh et al. 2013 [119], Hosseini et al. 2013 [160], Farivar et al. 2013 [92], Alajmi et al. 2013 [129], Liu et al. 2012 [161], Koutroulis et al. 2012 [162], Ishaque et al. 2012 [163], Zhou et al. 2011 [164], Petrone et al. 2011 [165], Moradi et al. 2011 [127], Miyatake et al. 2011 [140], Mei et al. 2011 [166], Ji et al. 2011 [167], Alajmi et al. 2011 [128], Abdelsalam et al. 2011 [168], Taheri et al. 2010 [169], Roy Chowdhury and Saha 2010 [170], Li et al. 2010 [90], Dzung et al. 2010 [171], Coelho et al. 2010 [172], Scarpa et al. 2009 [93], Sera et al. 2008 [118], Liu et al. 2008 [89], Pandey et al. 2007 [173], Khaehintung et al. 2006 [174], Femia et al. 2005 [175], Miyatake et al. 2004 [111], Jain et al. 2004 [91], Ho et al. 2004 [176], Hua and Lin 2003 [177], Koutroulis et al. 2001 [178], Zhang et al. 2000[179].

3.1.4.1 Simulation results

Simulation testing is generally preferred, with 73 out of 92 (79%) articles having a set of simulated validations, 32 out of 92 (35%) having only a set of simulated results without hardware experiment. Software simulation using tools like MATLAB/Simulink is convenient for testing the general behaviour of the algorithm, but they neglect experimental challenges such as measurement errors. This accelerated testing capability was fully used by Liu et al. [161], Benyoucef et al. [155], Sundareswaran et al. [101], and Lyden et al. [112] to evaluate the convergence rate of their metaheuristic algorithm by repeating the same condition multiple times and seeing how often the algorithm converges on the GMPP. On the other hand, we have an example of simulation's shortcomings such as when working with Incremental Conductance (InC). Its operating principle requires a perfectly clean measurement

of the voltage and current of the array, which is not a problem in simulation but essentially impossible to obtain in practice as shown by Femia et al. [175].

3.1.4.2 Experimental results

Experimental testings are included for 60 of 92 (65%) of the proposals, 25 of which (42%) are tested using solar simulators with the other 35 (58%) using PV modules or having an unknown setup. To study the effect of moving shadows and continuous irradiance variations, it would be preferable to use laboratory instruments that could accurately and consistently recreate a sequence of P-V characteristics multiple times to compare different shading mitigation algorithms (e.g. experimental results by Li et al. [95], Tey et al. [144], and Mohanty et al. [139]). The complexity of accurately replicating varying shading conditions using PV modules is probably why most papers with this setup type only look at the algorithm's response under stable irradiance (e.g. experimental results of Tey et al. [144], Ramyar et al. [62], and Lyden et al. [112]).

3.1.4.3 Steady state phase and irradiance perturbation detection

This section is relevant because of the way hill climbing works. Most HC implementations do not stop the search after convergence, leading to an oscillation around MPP that causes some efficiency reduction [88]. But there is a reason why HC was implemented this way. When there could only be one power peak, a continuous search ensures that the converter closely follows the MPP that stays in a relatively small voltage window even when irradiance and temperature change (Figure 3.3A).

However, when the power output is stable, oscillations would affect efficiency. Furthermore, if HC was implemented with a global search to tackle partial shading conditions, this continuous tracking is also not useful because the peak could jump to a far away operating point as shown in Figure 3.3B that is inaccessible by HC. Therefore, this oscillation is frequently referred to by many authors as a setback that needs to be rectified.

As for GMPPT algorithms, they perform global searches and consequently spend a lot of time at non-optimal operating points during this phase, so letting them search indefinitely is also not good for efficiency. Due to the above drawbacks, most algorithms implement a **steady state phase** where after determining the MPP, the microcontroller sets the converter at the optimal operating point found and monitors power generation for any changes.

Having a steady state means that there must be an **irradiance perturbation detection mechanism** to initiate a new search when necessary. Miyatake et al. [140] proposed a simple criterion that we used in equation 3.9 when describing our implementation of Deterministic Particle Swarm Optimisation.

This detection mechanism is widely adopted used in the literature, where among 33 of the 51 MPC algorithm proposals (65%) discussing their irradiance perturbation

detection mechanism, 27 (82%) used the same criterion, and the other six being essentially its slight variations. There is a compromise to consider when choosing the threshold: a tighter limit would ensure that even the slightest perturbations are detected, but the risk of initiating a search due to measurement noise would also increase.

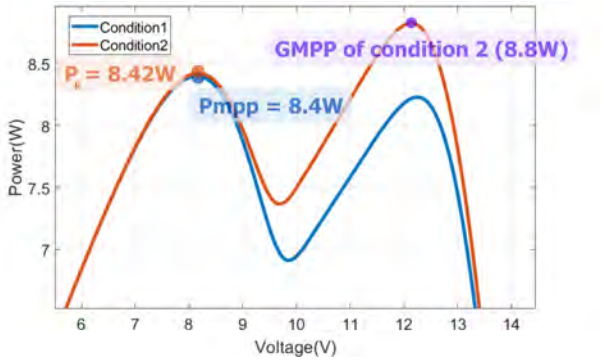


Figure 3.16: A case study where the shading detection proposed by Miyatake et al. [140] might fail.

Although the above criterion works every time a shadow causes a power drop or when the global irradiation changes, it might sometimes fail to detect a shadow moving away from the PV array. To illustrate the problem, we provide in Figure 3.16 two P-V curves of four PV blocks in series under partial shading. We assume that the array is handled by an MPC algorithm with a steady-state phase and a tight power threshold $\varepsilon = 1\%$. Initially, a PV module is underperforming due to shadows (condition one) and the search converges correctly toward $P_{mpp} = 8.4W$ at 8.2V where the algorithm enters steady state. The shadow then moves slightly away, resulting in the P-V curve of condition two. The microcontroller observes $P_k = 8.42W$ at the same 8.2V operating point and calculates $|P_{mpp} - P_k| = 0.02W < 0.084W$, so it does not initiate a GMPP search. However, we know that the optimal operating point has increased to 8.8W at 12.2V in condition two. Nevertheless, it must be noted that we cherry-picked this situation to point out the problem, and while it did occasionally occur in our tests (see Figure 3.17), this blind spot is not a significant concern.

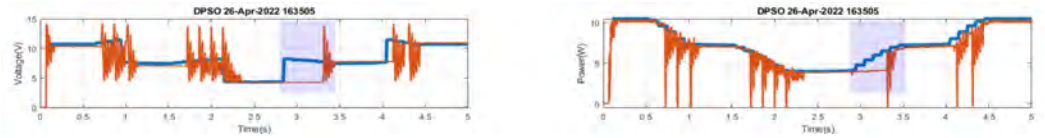


Figure 3.17: An example of where the irradiance perturbation detection mechanism failed in our tests with DPSO. On the left is the voltage of the array during experiment in orange and the expected voltage at GMPP in blue. On the right is the power of the array during experiment in orange and the expected power at GMPP in blue. We highlighted the region where the GMPP suddely jump to another voltage operating point and optimal power is changing but the algorithm did not catch the variation at its current operating point.

3.1.4.4 Choice of test profiles

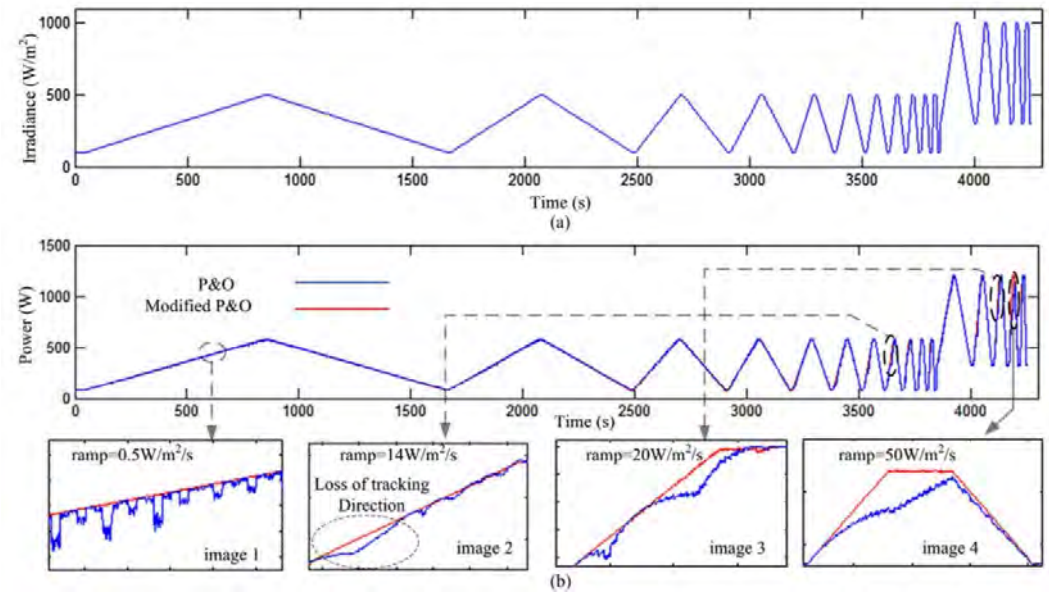


Figure 3.18: Extract of test results using the EN 50530 MPPT standard from [114].

Regarding testing profiles, the vast majority of proposals, 80 out of 92 (87%), rely exclusively on irradiance steps. This means that a particular I-V characteristic was initially set and an immediate change to a new profile was applied (i.e., simulation and experimental result by Li et al. [95], Tey et al. [144], Mohanty et al. [139]). This test is useful for evaluating the response time of the algorithm, as well as its accuracy after the response has settled. However, this procedure is not representative of real-world applications. Operating in the field, irradiance received by the solar panels may decrease over a span of time, from 10ms (fast moving shadows) to hours (static object shadow throughout the day).

Testing algorithms under continuous shading variations or discretised continuous shading variations using solar simulators is not yet widespread in the literature. We found a varying irradiance profile based on the European Norm for grid-connected

solar inverter efficiency, EN 50530 [180], and Ahmed et al. [114] have tested their MPPT algorithm proposal using this standard (Figure 3.18). No proposal GMPPT algorithm was tested according to this standard, as far as we know (only claimed by Lian et al. in [116]). But the norm could only account for global irradiance changes and not partial shading conditions, which are much more complex in nature. Overall, this is a great gap in the research on GMPPT under partial shadings.

3.1.4.5 Result presentation

Which parameters to present are inconsistent in the literature. Usually, this is not a problem, but for comparison and performance evaluation purposes, many frequently reported metrics are not very useful. A good case study would be the PSO convergence time metric, which ranges from 2s in the studies of Miyatake et al. [140] to 7s in Liu et al. [161], and simulation results could go as low as 0.04s in the studies by Mohanty et al. [139]. These differences are due to different sampling rates being chosen for each test which depends on the microcontroller sampling time. A better metric to report would be the number of sampling cycles necessary before convergence as presented by Benyoucef et al. [155], or to explicitly state the sampling rate, available in 31 of the 60 experimental results (52%) and in 43 of the 73 simulation results (59%).

Another metric is the **efficiency result**, presented in 42 of the 92 proposals examined (46%). This could be power efficiency, the measured power over theoretical maximum power in $\frac{W}{W}$ found in 26 of those 42 (62%), or energy efficiency, the measured energy over theoretical maximum energy in $\frac{Wh}{Wh}$ in 16 of those 42 (38%). Among these two, we believe that energy efficiency is the better metric. As previously discussed, being MPC means having a global search phase spending time at non-optimal operating points that causes power loss. If it is called frequently enough, energy efficiency would be greatly impacted. These two metrics are only similar when the irradiance is relatively stable, and, consequently, the global searches are called more infrequently. In situations of constantly varying irradiance, it is better to focus on energy efficiency, such as the theoretical estimate performed by Mirza et al. [113] or the 10 hour test profile performed by Ishaque et al. [98]. Energy effciency figures actually account for other aspects of the algorithm, such as its irradiance perturbation mechanism and convergence rate as well.

Next, we have the **convergence rate**, i.e. the consistency of algorithm's capability to converge toward MPP. As mentioned previously, noise in measurements could throw the algorithm off course and it may fail to converge. This is even more necessary for the metaheuristic OptA algorithms due to their random nature. In general, we only found simulated evaluations such as those by Liu et al. [161], Benyoucef et al. [155], Sundareswaran et al. [101], and Lyden et al. [112], while experimental results were not available.

Algorithm complexity is difficult to objectively quantify, and there are not many papers that include this metric. Most evaluations are limited to simple statements

such as high or low cost (e.g. Erdem et al. [181], Mohapatra et al. [182]) without a set of clear guidelines on how the conclusion was made. A concrete way to objectively compare the complexity could be the computation time necessary to perform a duty cycle update, as well as the amount of memory usage of the algorithm on the microcontroller (both ROM and RAM), as presented by Liu et al. in [161]. While this is insignificant most of the time since computing power has evolved so much in the last several decades, there are still ultra-low power microcontroller running at very low frequencies that may struggle even with floating point operations because of the lack of a dedicated floating point unit, let alone functions like exponentials and sinusoidal.

3.2 A probabilistic GMPPT algorithm based on GMPP distribution

3.2.1 Algorithm description

Figure 3.19: The general idea of our proposed simple GMPPT algorithm.

Knowing that GMPP occurs in distinct zones, we suggest a very limited search in which a single power point is evaluated in each GMPP region and then the maximum of these is chosen as the starting point for hill climbing (Figure 3.19). We call this algorithm **fast GMPPT** because on paper it seems to trade efficiency for a faster convergence time. These voltage values to evaluate are called **voltage targets**. Although having the concrete distribution results found in Figure 2.24 and Figure 2.25 would be nice to choose these targets, using the GMPP estimates shown in Section 2.1.4 should suffice.

Our proposed algorithm consists of four main phases as shown in the flow chart in Figure 3.6: initialisation of variables, voltage search to find the initial seed for HC, improved HC, and steady state. The initialisation phase is where all the parameters are loaded into the program memory, and the steady state phase is implemented similarly to DPSO and GWO. Therefore, we have two important phases to discuss, the voltage search phase and the improved HC phase.

In the voltage search phase, n voltage targets are evaluated and the maximum is chosen as a seed for the subsequent improved HC phase. While in this work we are only targeting the GMPP regions, n could be any number of points, but too many points would defeat the purpose of a limited global search. Due to measurement noise, we relaxed the "point" requirement of each voltage target i to a "narrow voltage window". In the program, this is represented by the optimal point v^i_{target}, the upper limit v^i_{up}, and the lower limit v^i_{low}. As long as the array's voltage is in this window, the voltage target is considered reached. However, since we are not directly controlling the voltage of the string because we are in duty cycle control, we added a simple proportional controller in the form of $D^k = D^{k-1} + p(V^k_{pv} - v^i_{target})$ where D^k is the duty cycle to be sent for the current iteration, D^{k-1} is the duty cycle from the previous iteration, V^k_{pv} is the measurement from the current iteration, and p is

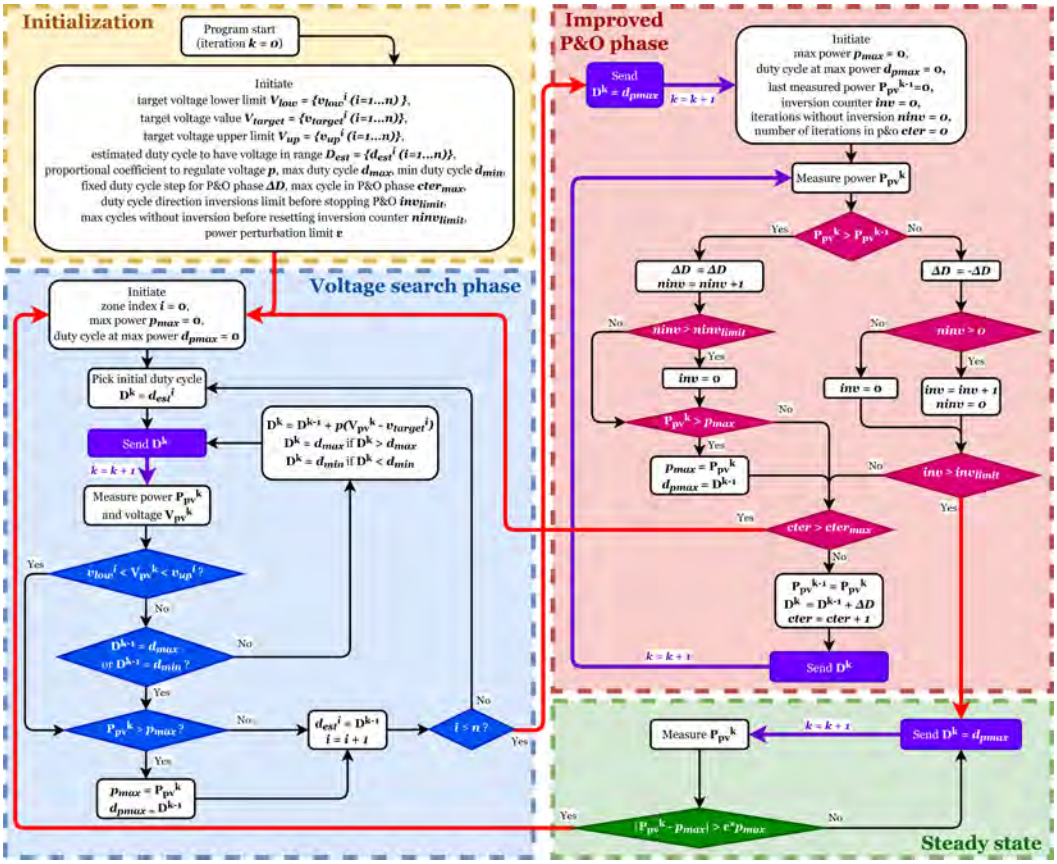


Figure 3.20: Flowchart of our algorithm proposal.

the proportional coefficient. To refresh about the algorithm's iteration convention, refer to Figure 2.34. An array of initial guessed duty cycles was given as d_{est}^i and constantly updated at every voltage search phase with the duty cycle that gets to the voltage target to accelerate subsequent searches.

Next, we present the improved HC phase where we addressed two main drawbacks of the basic HC algorithm: the oscillation around the peak and the potential loss of tracking. To remove the oscillation, we could detect when it happens and force the system to a steady state at MPP. Figure 3.21 shows the voltage evolution of our PV string when HC has converged. We observe that frequent inversion of duty cycle variation with one sample in between is a sign of oscillation. However, this is a relatively perfect example because measurement noise might sometimes lead to some extra steps between the inversions. Furthermore, inversion could also occur when the power continuously changes, as shown in Figure 3.5. Therefore, we must examine how many times the duty cycle variation is inverted inv as well as the streak of samples without inversion $ninv$. When $ninv$ exceeds a limit of $ninv_{limit}$, we could conclude that the algorithm is in the search phase or that the irradiance is varying, so inv is reset to 0. When an inversion occurs, inv is incremented and $ninv$ is reset to 0 only if $ninv$ is non-zero, otherwise we are seeing a continuous

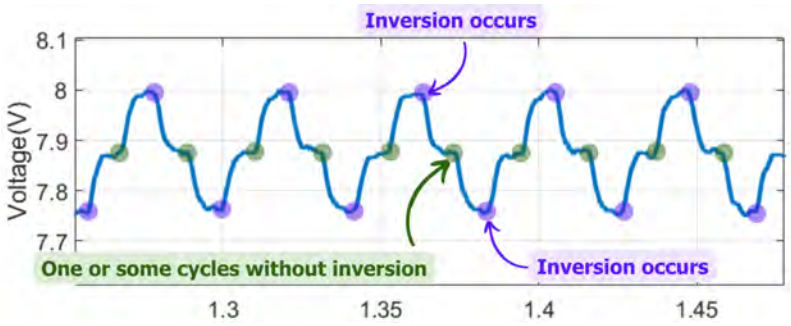


Figure 3.21: Voltage graph showing the behavior of oscillation around MPP.

inversion which suggests continuous power variation and inv is reset. Finally, the oscillation is confirmed when inv exceeds a certain limit inv_{limit}. We determined that $ninv_{limit} = 2$ and $inv_{limit} = 3$ are optimal. Regarding tracking loss, we added a simple iteration counter $cter$ in the HC phase, and the algorithm reverts back to the sweep phase when it exceeds $cter_{limit} = 100$. This value could be arbitrarily chosen as long as it is not too low so that hill climbing could have enough time to converge in normal operation.

3.2.2 Theoretical evaluation of the search mechanism

We used our accelerated simulation capabilities shown in the previous chapter to roughly estimate how often this GMPPT schema successfully tracks GMPP. Knowing that HC is only successful when the power between the starting point and the GMPP increases strictly, we used this as a simple criterion to evaluate the theoretical success rate of our algorithm (Figure 3.22). So, we need is to go through a lot of P-V characteristics of our string, calculate the power at these voltage targets, and see if in theory the algorithm could track toward GMPP.

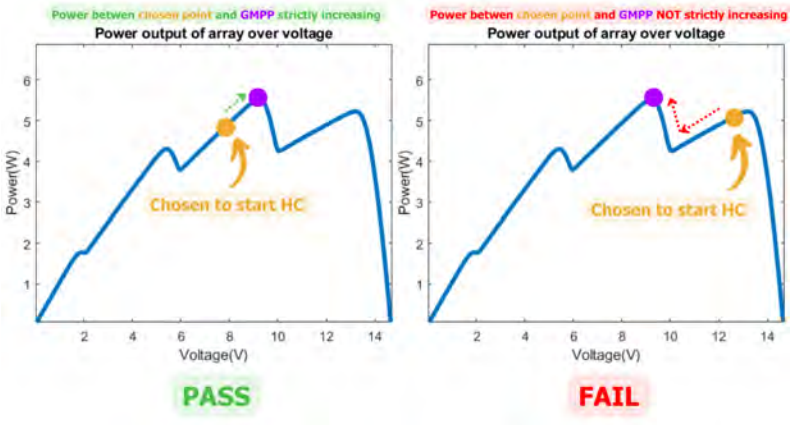


Figure 3.22: Illustration of the criterion to theoretically evaluate the capability of our algorithm.

However, the voltage targets in our implementation are rather narrow voltage windows instead of points, so this dispersion should be considered. First, we start with the target voltage values that are 5.4V, 8.7V, and 12V based on the GMPP estimates. Note that we do not consider 2.1V because our buck converter could not reach this string voltage because the output being limited to around 4V. Next, in our implementation, the upper bound is $v^i_{up} = v^i_{target} + 16$ and the lower bound in $v^i_{lower} = v^i_{target} - 16$. Since the microcontroller's ADC has a resolution of 10-bit on a full-scale range of 0 to 5V, so this delta of 16 translates to a 0.08V delta on the ADC's measurement. But, this measurement is only $\frac{1}{3}$ of the string's voltage as shown in Section 2.4.3, so this delta on the voltage of the string would be 0.24V. Rounding the limits, we need to evaluate the success rate when the target voltages are in $5.4V \pm 0.3V$, $8.7V \pm 0.3V$, and $12V \pm 0.3V$. It is worth noting that our G-T sweep generates the I-V curve of the PV string that has an array of unknownly spaced voltage array tied to a linearly spaced current array. We need to first interpolate the data back to a linearly spaced voltage array with a step of 0.1V tied to an unknownly spaced current array. From there, we could consider seven voltages per each of these voltage target windows (e.g. $5.4V \pm 0.3V = 5.1V, 5.2V, ...5.7V$). If each of the three voltage targets could have seven values, the total combination of possible voltage targets would be $7^3 = 343$.

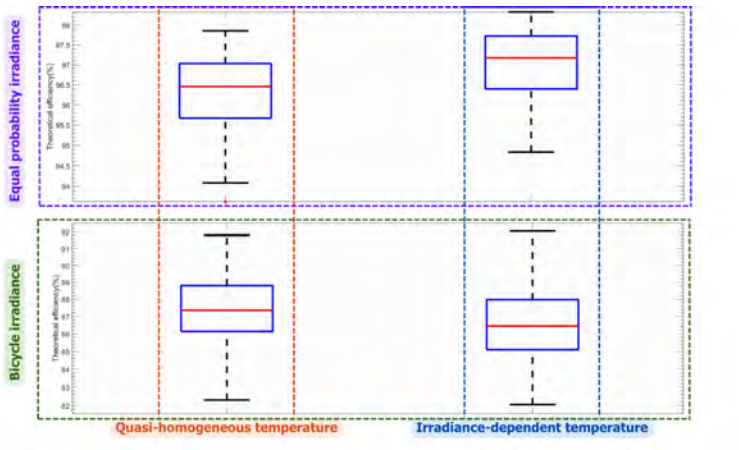


Figure 3.23: Theoretical estimation of successful MPPT rate.

We modified our four G-T sweeps from section 2.2 to add in a calculation of the power of the string at the voltage targets and evaluate the power gradient between the maximum power points among them and the GMPP of the string. We plot the success rate of each of the 343 sets of voltage points in each G-T sweep in Figure 3.23. We observe that the probability of successful MPPT tracking ranges from 86% to 97%, with the worst case scenario being 82%. Notice how the result is worse with the I-Vs from the set of bicycle irradiance, where there are many low power conditions.

3.3 Validation of our algorithm proposal

To evaluate our algorithm, we compared it with three others that shares the same relatively lightweight structure, namely **Hill Climbing (HC)**, **Deterministic Particle Swarm Optimisation (DPSO)**, and **Grey Wolf Optimisation (GWO)**. All tests were carried out on the string of four PV blocks whose characteristics are found in Table 2.1 and Table 2.2.

First, we want to observe the **convergence time**, but not to compare the convergence time of each algorithm. Rather, they give insight into the algorithm's operating principle, which is important to discuss their capability. Second, the **energy efficiency** must be assessed because the ultimate goal of having a GMPPT algorithm is the optimal extraction of solar energy.

3.3.1 Test profiles

The best scenario to observe the operating principle of each algorithm in action, as well as its convergence time, is a **sequence of irradiance steps**. Figure 3.24 shows the five conditions with which we tested the algorithms, and the irradiance and temperature of each PV block are summarised in Table 3.2 where G_k is the irradiance received by module k and T is the same temperature for all modules. The test sequence consists of conditions one to five sent in that order, and each condition lasts one second. Note that this is a small showcase of the results as we have performed many more of these irradiance steps during development, but this limited set should be sufficient to present their operating principle and to give a general idea of the convergence time. We present the results from both the simulated environment and the experimental setup.

Condition	$G_1(Wm^{-2})$	$G_2(Wm^{-2})$	$G_3(Wm^{-2})$	$G_4(Wm^{-2})$	$T(^{\circ}C)$
1	900	900	900	900	60
2	900	900	900	200	60
3	900	900	200	200	60
4	900	700	700	700	60
5	900	700	700	200	60

Table 3.2: Summary of the five irradiance-temperature conditions used in the sequence of irradiance steps.

Regarding energy efficiency, we believe that the optimal test would be to put the algorithms to work in various **varying irradiance profiles** since they would put all aspects of the algorithm under scrutiny such as convergence rate, convergence accuracy, its irradiance perturbation detection mechanism and its resilience to varying irradiance during global search. Therefore, the idea is to simulate the irradiance of each PV module over time as an object passes by them while maintaining a constant global irradiance.

We devised a simplified mathematical model to achieve this, as shown in Figure 3.25. This shading profile creator first has the string of four square solar panels

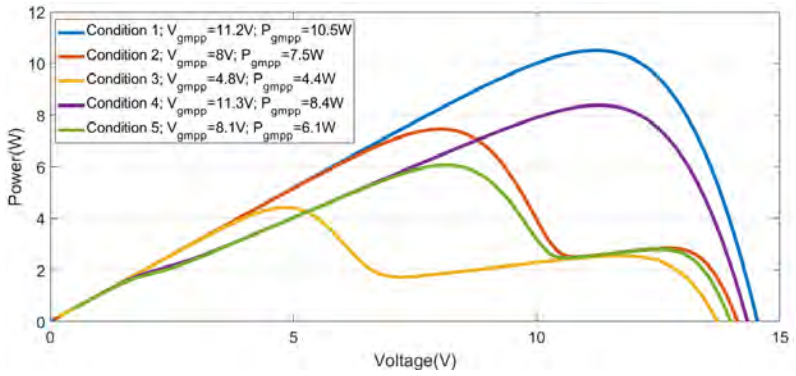


Figure 3.24: P-V curves of the five test conditions in the sequence of irradiance steps. Their respective V_{gmpp} and P_{gmpp} are provided in the figure.

placed in a square formation on the Oxy plane. For convenience, the length $1l$ of the sides of these square solar panels is used as a unit basis in this reference frame. They are receiving even G_{global} irradiance and all at the same temperature T_{global}. A shading object with arbitrary width w_{shade} and height h_{shade} starting from an arbitrary position (x_{shade}, y_{shade}) moves across the plane at a velocity described by v_{shade} and its angle relative to Ox θ_{shade}. At each timestamp, we calculate the overlap between the shading object and the solar panels to obtain their instantaneous irradiance. Note that the shading factor of a photovoltaic module is assumed to be applied equally to all of its individual cells (ref Section 2.1.2). By changing the global irradiance, global temperature, and how the shading object moves, we could conveniently create a list of different varying irradiance profiles, each lasting an arbitrarily chosen 8s. The list of different parameters is summarised in Table 3.3 and we combined them all for a total of 288 different varying irradiance profiles. Since simulating these varying irradiance profiles is very slow, we exclusively performed this test on the experimental setup.

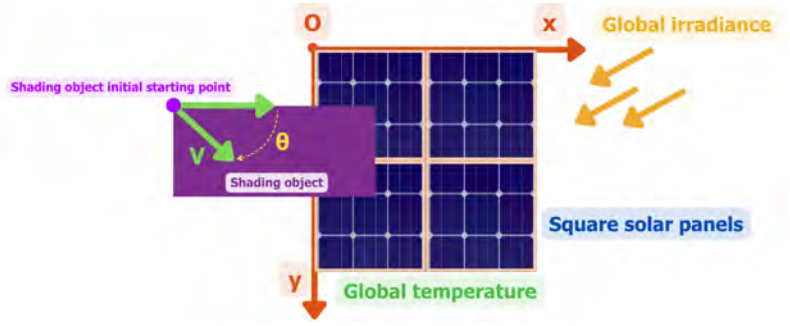


Figure 3.25: Illustration of our shading profile creator, a simplified mathematical model to estimate how the irradiance of each PV module could vary over time.

Parameter	List of values	Unit
G_{global}	800, 600, 400	Wm^{-2}
T_{global}	45, 65	$^{\circ}C$
w_{shade}	1, 1.5	l
h_{shade}	1, 3	l
x_{shade}	equals $-h_{shade}$	l
y_{shade}	-0.25, -0.75	l
v_{shade}	0.1, 0.5	ls^{-1}
θ_{shade}	0, 30, 60	$degrees$

Table 3.3: List of different values for each parameter in our shading profile creator.

3.3.2 Test setups

The simulation test setup used was already discussed in Section 2.4.3 and we provide the complete model in Figure 3.26. The model corresponding to each algorithm are called from the MATLAB workspace, the irradiance and temperature timeseries are sent, the model is simulated, and the data are retrieved in the workspace to be processed.

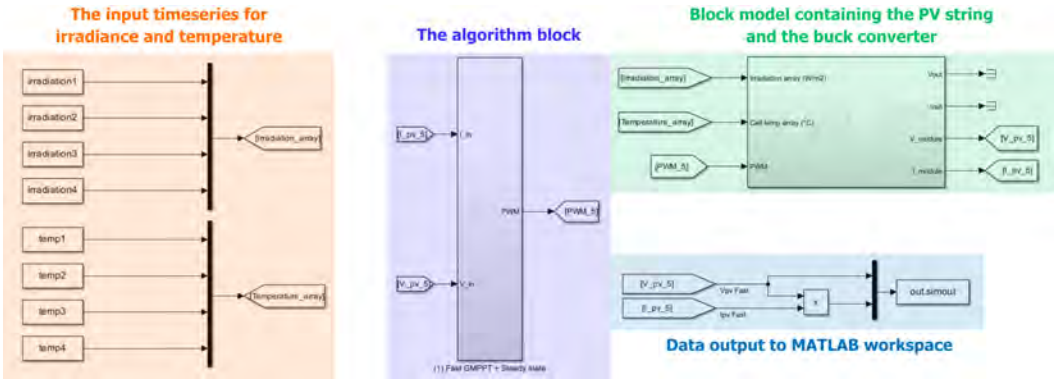


Figure 3.26: Simulink model of the system of test the algorithms. The detailed description of the blocks could be found in Section 2.4.3.

As for the experimental setup, it is based on the Agilent E4360A solar simulator, the voltage and current measurements were acquired using the Keysight DSOX3014A oscilloscope, and the battery was emulated using N6705B. The current was measured using a Tektronix A622 current probe with 10 wiring loops for a current gain of 10V/A. The probe was calibrated before each irradiance step sequence and before running the whole sequence of varying irradiance profiles. A MATLAB interface interacts with the solar simulator to send out the different I-V characteristics. For the irradiance steps, the MATLAB program communicates a new I-V characteristics to the solar simulator every second. For the varying irradiance profile test, a new I-V characteristic corresponding to a new G-T condition is communicated every 100ms to the simulator which is its hardware limit. The oscilloscope captures the measurements in 1s segments and they are retrieved via USB to be processed in MATLAB.

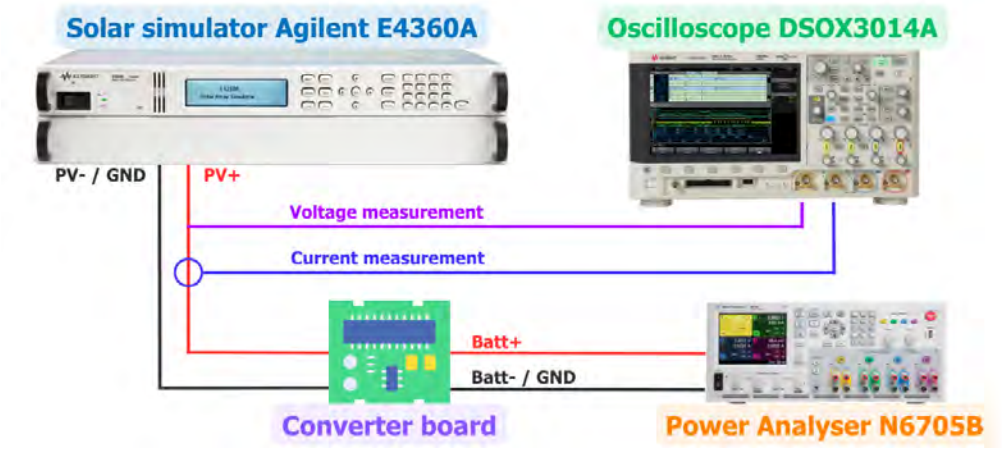


Figure 3.27: Block diagram of the experimental setup to evaluate the algorithms' performances.

3.3.3 Results for a sequence of irradiance steps

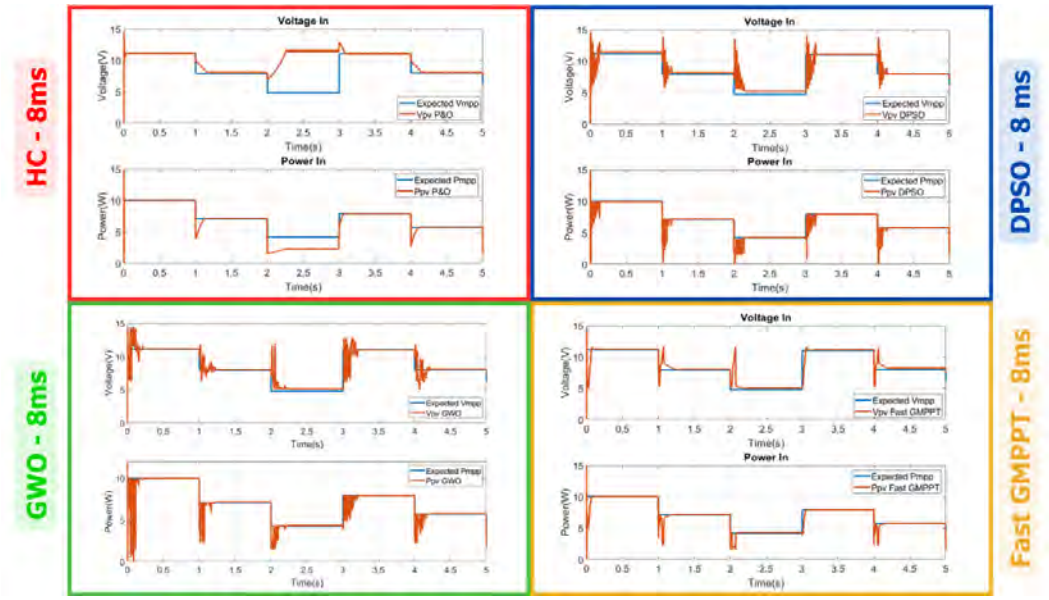


Figure 3.28: Sequence of irradiance steps results of four algorithms in **simulation**: HC, DSPO, GWO and fast GMPPT. The sampling time T_s are given to put each algorithm's response time in perspective.

The simulated and experimental tests results for the sequence of irradiance steps are presented in Figure 3.28 and Figure 3.29 respectively. All algorithms were simulated with a sampling time $T_s = 8ms$. Note that in the simulated result, the V_{gmpp} and P_{gmpp} estimates were calculated using Shockley equation bypass diode model which is slightly different from the piecewise model used in Simulink. However, the discrepancy is slight enough that we could still conclude on the algorithm's ability

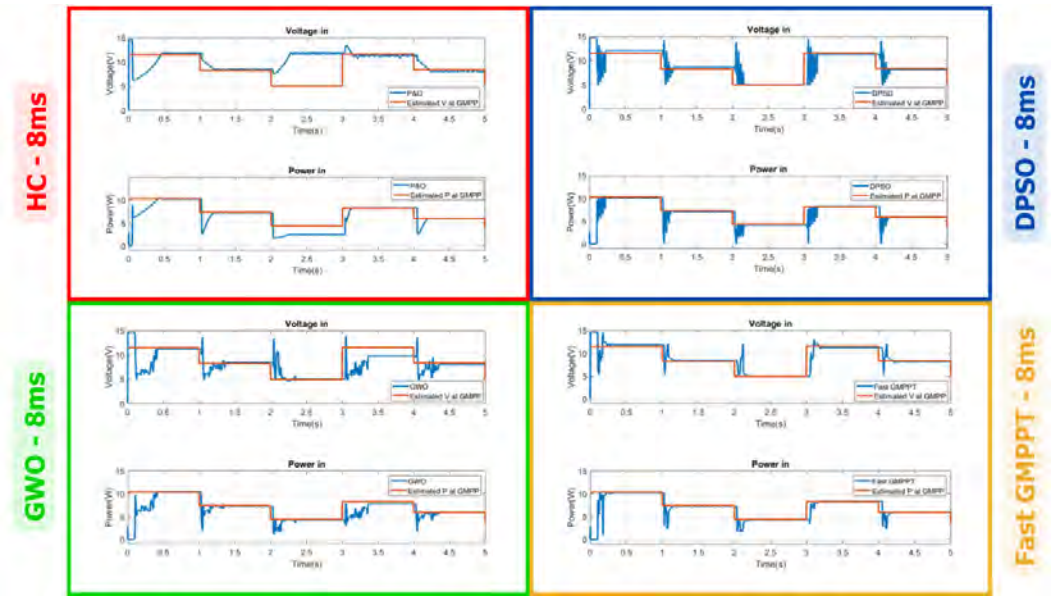


Figure 3.29: Sequence of irradiance steps results of four algorithms in **experiment**: HC, DSPO, GWO and fast GMPPT. The sampling time are given to put each algorithm's response time in perspective.

to converge. As for the experimental setup, the V_{gmpp} and P_{gmpp} are taken from the same P-V profile sent to the simulator. In general, the agreed simulated and experimental results show that the algorithms were consistently implemented. Furthermore, the parameters we tuned via the simulation also performed very well in the experiment. But there is a small delay of 50ms between the expected GMPP and the experimental measurement because of the initialisation phase in the microcontroller which is removed when we discuss the algorithm's convergence.

First, the result of HC showcases its inconsistency in partial shading conditions where it failed to correctly track toward GMPP in one condition. Its convergence time varies widely from a very low 08 iterations up to 34 iterations (64ms to 272ms). There is a significant difference between the convergence time at condition one in the simulation and in experimental setup caused by the different initial duty cycles. We remark then that its convergence time and successful tracking capability heavily depends on the initial search position.

Next, we have the DPSO's response that progressively converges toward the GMPP between 04 to 06 s-iterations (96ms to 144ms). No significant response time difference was detected between the simulated and experimental result indicates that the algorithm was not heavy on the microcontroller. As for GWO, it should consistently converge after 08 s-iterations, but the simulated result shows a consistent 192ms convergence time while the experimental results ranges from 200ms to 250ms. By zooming in, we noticed that the duty cycle updates are applied at varying times indicating that the randomised number generation using the ADC ports has impacted

its performance in experiment. Furthermore, we could see its failure to converge with high accuracy under condition four in the experimental result, a drawback of its randomised nature. Overall, the two OptA methods boast good tracking time and good accuracy, but cause a lot of power jitters.

Finally, we have fast GMPPT correctly tracks toward GMPP in all five conditions and its convergence time ranges from 20 to 32 iterations (160ms - 256ms). Generally, our method causes less power jitter than the OptA algorithms, but its convergence time is slightly slower than DPSO and comparable to GWO. However, the slower convergence time is mostly contributed by the hill climbing phase that was already near the GMPP, so losses are minimised. The only significant power loss is caused during the global search phase, where it searches for the voltage targets.

Generally, under heavy partial shading, GMPPT algorithms will perform better than HC which is understandable given how they operate. But we further comment on this result to add to our previous point made in Section 3.1.4.5. Truth be told, this sequence of irradiance steps could be manipulated to draw any conclusion that we would like. For example, we could have made HC fail every condition to make the other algorithms look even better. If we want a high energy efficiency, we could increase the test length and reduce the number of steps, which significantly downplays the effect of the global searches. Our own result itself has a lot of blind spots. For example, the experimental convergence rate of GWO is much worse than what our graphs suggest, probably due to the poor random generator on the microcontroller. Furthermore, we did not showcase conditions where fast GMPPT failed, not to avoid bad results, but because we had to actively find these edge cases.

3.3.4 Results for under varying irradiance profiles

This is where the algorithms are evaluated as a whole from convergence time, convergence rate, convergence accuracy, its irradiance perturbation mechanism, and the most important metric of all, energy efficiency. A raw data figure is provided in Figure 3.30. We present in Figure 3.31 the distribution of energy efficiency of the four algorithms under the 288 varying irradiance profiles. The median, lowest, and highest energy efficiency are compiled in Table 3.4.

Algorithm	Median	Lowest	Highest
HC	93.64%	56.2%	98.35%
Fast GMPPT	94.74%	72.68%	97.74%
DPSO	90.68%	75.20%	97.42%
GWO	86%	71%	96.97%

Table 3.4: Compilation of energy efficiency figures of the four tested algorithms. The worst and best result in each category is highlighted in red and green, respectively.

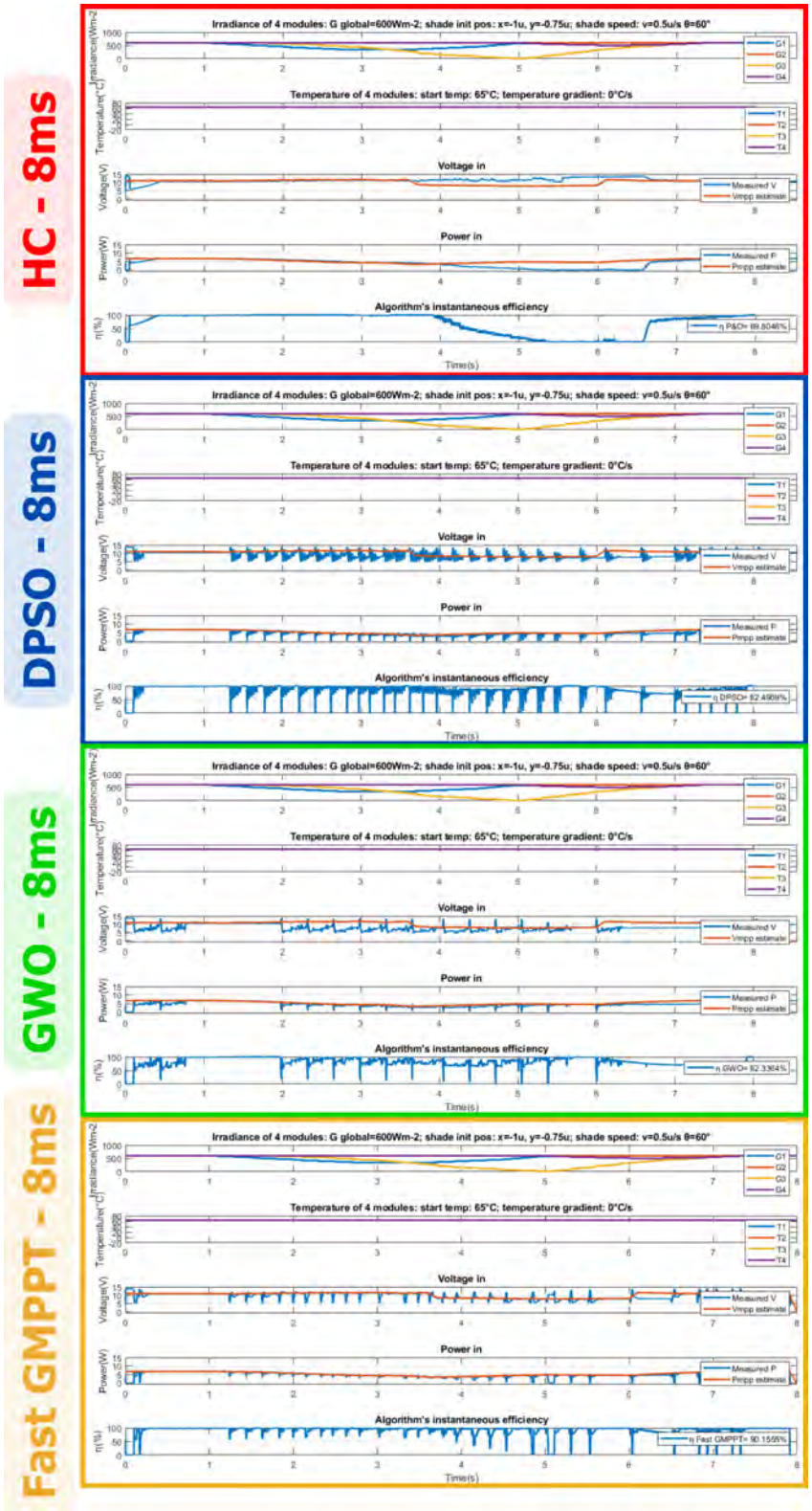


Figure 3.30: One varying irradiance profile raw result from testing four algorithms.

HC having the worst lowest energy efficiency of 56.2% demonstrates that it lost GMPP tracking under certain conditions, but its highest energy efficiency of 98.35% is also the best among the four. We observe that fast GMPPT, DPSO, and GWO all have better lowest energy efficiencies but slightly worse highest energy efficiencies compared to HC. This highlights the advantages, but also drawbacks, of global search. In challenging situations where hill climbing failed, they managed to converge and extract more power. This could be seen in the first 4s of the test profile given in Figure 3.30 where the HC stuck very close to the peak while the other three caused some power drops. However, in more mildly varying situations, HC tracks the peak without any perturbations, while the MPC algorithms inevitably initiate global searches that cause energy losses. This could be seen in the last 4s of the test profile given in Figure 3.30 where HC was unable to track GMPP accurately.

Fast GMPPT has the best overall median energy efficiency at 94.74%, followed by HC at 93.64%, then DPSO at 90.68%, and finally GWO at 86%. Among the GMPPT algorithms, fast GMPPT, DPSO, and GWO, we could see that limiting the global search phase to only where GMPP could appear is very advantageous. However, this is just a compromise to be made; by considering the characteristics of the PV string, the search becomes more optimal, but it also means that the algorithm is now parameter-dependent.

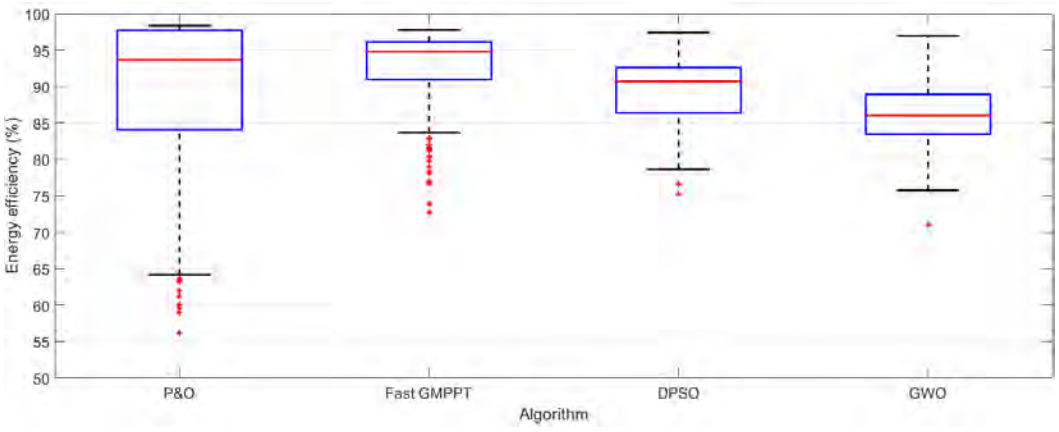


Figure 3.31: Average energy efficiency of each algorithm (Hill Climbing, Fast GMPPT, Deterministic Particle Swarm Optimisation, and Grey Wolf Optimisation) under the 288 varying irradiance profiles.

We now focus on some specific shadow speed to further comment on fast GMPPT performance. Note that our solar panels have a dimension of $10cm \times 10cm$, so $v_{shade} = 0.1ls^{-1} = 1cms^{-1}$ and $v_{shade} = 0.5ls^{-1} = 5cms^{-1}$. First, we examine a condition where there is a slow shadow $v_{shade} = 0.1ls^{-1} = 1cms^{-1}$ moving across the solar panels, which is the situation provided in Figure 3.32. Since slower shadows would behave the same given the convergence speed of the algorithms (around 200ms), this could be considered equivalent to a shadow cast by a static object throughout the day. In this context with light partial shading, HC perfectly tracks

the peak without any disturbance and yields an energy efficiency of 98.15%, while fast GMPPT tries to trigger global searches, causes power losses, and ends the test with an energy efficiency of 96.54%. Of course, when solar panels are more heavily shaded, fast GMPPT will perform better.

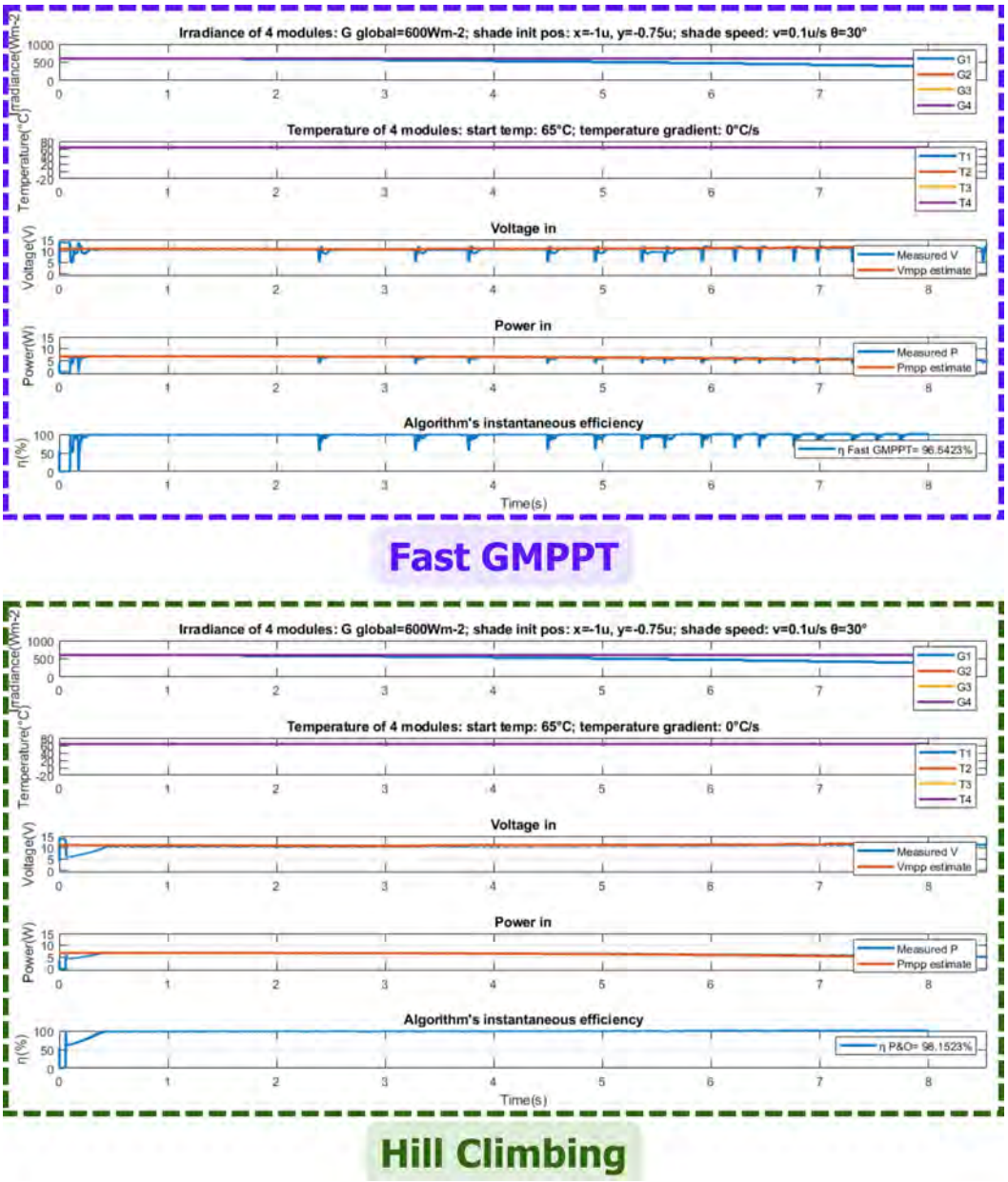


Figure 3.32: Result of HC and fast GMPPT in a very slow moving shadow and light partial shading situation.

We now look at the tracking capacity of fast GMPPT compared to HC with the faster variation speed that we had in our data set, which means $v_{shade} = 0.5ls^{-1} = 5cms^{-1}$. This speed is equivalent to shadows cast by moving objects, such as

swinging branches. We compiled a set of efficiency numbers for this shadow speed, with different shadow directions and all other parameters being the same, in Table 3.5. We could perceive that, generally, our algorithm tracks relatively well under these faster varying shadows, and also better when the partial shading conditions become more complex when the direction of the shadow changes. Usually, in the literature, we discuss how well an algorithm tracks under specific variations, but this shows that its tracking capacity also depends on the direction of the shadow across the array.

Algorithm	$\theta_{shade} = 0°$	$\theta_{shade} = 30°$	$\theta_{shade} = 60°$
HC	97.58%	82.68%	87.68%
Fast GMPPT	95.06%	90.68%	89.63%

Table 3.5: Compilation of efficiency figure from HC and Fast GMPPT under a fast varying shadow but different shadow direction.

From the above results, we could conclude some strong points of fast GMPPT:

- In theory, it should correctly track toward the global power peak in around 86% to 97% of the time with this limited global search.
- It could track GMPP very well under stable irradiance conditions, as proved by the step irradiance result.
- It is slightly less efficient than HC under slow varying shadows with light partial shading because it still tries to initiate global searches, but the limited global search significantly improves its overall energy efficiency compared to DSPO and GWO.
- It is more efficient than HC under fast varying irradiance with heavy partial shading because of its improved tracking capability. It is also better than DPSO and GWO for the same reason, which is having more efficient global searches.

3.4 Conclusion and perspective of Chapter 3

In general, we looked at the current literature of MPPT and GMPPT, did an extensive deep dive into the mechanism of several algorithms of both types, discussed how algorithms were studied in the literature, proposed our own algorithm called fast GMPPT, and compared it to several other methods with promising results. Going a bit further into the details, we compiled several key notes.

- There are a lot of MPPT and GMPPT algorithms that we struggle to find a practical use case for. Although not all research is about having a concrete use case, there are some proposals whose inherent advantages we could not determine. An example could be Incremental Conductance, where it is hard to justify how it is better than the basic Hill Climbing algorithm. Or β-parameter where we provided an example of how its supposed advantage could be replicated without having the parameter itself.
- We discussed a core challenge of GMPPT where a peak could suddenly appear elsewhere on the voltage range. We speculated that the global searches of GMPPT will inevitably cause a lot of power loss and our experimental result eventually proved it.
- Seeing that there is no testing method in the literature that fits our research context, we presented our own procedure to better simulate fluidly changing irradiance. We argued that this novel testing method is better at stressing both MPPT and GMPPT algorithms to assess their true performance when deployed.
- We developed fast GMPPT inspired by nVoc method and a probabilistic approach where we also provided its theoretical efficiency via intensive simulation. Its operating principle is simple requiring no complex computation, and all the parameters needed to get one running are easily accessible via the documentations of the components used. We compared our algorithm against Hill Climbing, Deterministic Particle Swarm Optimisation, and Grey Wolf Optimisation. The end results show that fast GMPPT is the overall best option for fast but complex shading patterns and a competitive option for slow but light partial shadings.

However, this work still has several limitations that we could have improved. They are not overlooked, but rather neglected for several reasons, including out of scope or time constraints. We first discuss some drawbacks of our novel testing procedure involving varying irradiance profiles. Improving these setbacks will make testing MPPT and GMPPT algorithms even more robust, and we could have more confidence in the performance assessment.

- Throughout this work, we are using the simplified assumption in Section 2.1.2 to simulate our string of PV blocks. Accurate modelling the I-V of a solar module by determining the minimum irradiated cell should fix this. There

are also untouched subjects, such as more strings in parallel, module ageing, module damage, etc. that could change the characteristics of the PV array.

- It is worth evaluating the different shading patterns, how the PV cells are arranged within the PV modules, how the modules are arranged in the array, etc. with the more accurate I-V so that that result better represents real-world applications.
- The varying irradiance profiles are still discrete with a sampling step of 100ms limited by the solar simulator, and we have not tested the algorithm with physical photovoltaic modules. However, it would be very difficult to recreate consistently varying irradiance profiles in this situation to accurately assess the performance of different algorithms. We did have an experimental bench of oscillating wings to simulate varying irradiance but it could not yet consider partial shadings.

Next, although fast GMPPT performed very well, it is not without several demerits.

- The varying irradiance did not include very low irradiance conditions, and the minimum global irradiance was $400 W m^{-2}$. These situations cause the signal-to-noise ratio to drastically increase, which could potentially throw the algorithms off course. Generally, algorithms should include a phase in which we implement a **fractional open circuit voltage** because at least harvesting something is preferable over a search that just jitters around due to measurement errors.
- Although the chosen test condition indeed represents some use case, it does not cover every possible situation.
- The test scenario is limited to four PV modules with bypass diodes, and we did not test with more PV modules in series, nor did we test with parallel PV strings. Our algorithm achieved this performance because the global search was limited to a few potential GMPP regions. If the number of potential GMPP regions increases, its advantage over others, such as DPSO or GWO, could be less significant.

We now address the fact that hill climbing performed, on average, better than two other GMPPT algorithms under varying irradiance profiles. This is surprising because it is frequently called out in the literature as being unsuitable for partial shading conditions. In our varying irradiance profiles, which included many PSCs as shown in Figure 3.30, even fast GMPPT performs barely better than HC. This puts into question the effectiveness of global searches under varying shadows; they are very inefficient, and being frequently called during a long varying irradiance situation, also shown in Figure 3.30, did cause significant energy loss.

Therefore, we conclude that GMPPT is not the definitive answer to our problem. This conclusion may sound strange, given that fast GMPPT provides numerous advantages and could be considered the most well-rounded solution among the four

algorithms that we tested. But it only performed this well because the search is limited to only three voltage targets. Having more potential peaks would almost certainly diminish its only strong point. Although it is true that designing a better buck converter that provides cleaner signals to spend less samples to digitally filter the measurements and responds quicker to duty cycle steps will improve the algorithm's performance significantly, this is slightly out-of-scope. We stuck to this basic and rough system because it is inexpensive and easy to obtain. If these constraints are completely removed, there are many better options. This would be an important point for the end user to consider. For example, if complexity and cost are not of concern, we recommend distributing the array into smaller portions and running multiple MPPT algorithms to track the peak of individual modules, or GMPPT algorithms with targeted search such as fast GMPPT to track the peak of a small subgroup of modules.

Conclusion

To conclude, we are going back to the original research context, which is **solar harvesting and optimisation for autonomous low to medium power supplies between 10W and 100W**, and see how this work fits into that framework. Furthermore, there is also the constraint of keeping the solution inexpensive and simple for the end user to consider.

The first question is **how could we estimate the solar availability of the PV system in these challenging situations with relative accuracy?** which was answered by the studies presented in Chapter 1. We have proposed a procedure that consists only of taking a **fisheye photo of the sky** above the intended deployment site and **calibrating the camera** using a simple printed calibration pattern. With this information, the shading information is inferred via the detected obstructions and the estimated solar position throughout the day in the image. It is then incorporated into the meteorological data from PVGIS to generate a solar estimate that, in the studied use case, is **50% more accurate** than when shading was not considered. This shading compensated irradiation is then used to simulate the energy profile of a solar harvesting system, where we graphed the evolution of the system battery's minimum state of charge, showcasing the importance of having a good solar estimate to ensure continuous operation of the autonomous power supply. All of this was done within the constraints of the work. First, **only inexpensive tools** were used: a fisheye lens from e-commerce sites, a calibration pattern printed on A4 paper. Second, the **simple user procedure** involves only two steps as described in Chapter 1 that anyone could use, even those who are not familiar with solar energy. We believe that our biologist colleagues will be satisfied with this toolbox for this next wave of sensor deployment. And third, the script is **entirely open-source** and provided on GitHub so that anyone could extend the work to match their use case.

The second question we asked was **what is the impact of the partial shading conditions on the available power of the PV system?** Chapter 2 has answered the reason why a series of cells without bypass diodes is a bad idea; the most shaded cell limit the current of the string, leading to significant power loss when the string is only slightly covered. Although bypass diodes did help, they lead to multiple local power peaks among which we could find the global maximum power point. To better understand how these are distributed so that a good tracking algorithm could be deployed, we mathematically model the **PV cell, PV module, PV block (which is a module and a bypass diode), and a string of PV blocks**. We discussed in detail how they could be simulated efficiently by presenting the compute optimisations, the two most significant improvements being the usage of the **LambertW**() function and a pre-computed **look-up table** for all current-voltage characteristics of each PV block under all potential irradiance and

temperature conditions. This toolbox unlocks the capability to analyse the GMPP distribution of an arbitrarily designed PV array in a reasonable time frame. We presented the GMPP distribution of a string of four PV blocks in two specific contexts, one where we assumed an equal probability between all irradiance and temperature conditions, and one where the irradiance was measured by a matrix of pyranometers based behind a moving bicycle. Also in Chapter 2, we build a Simulink model for the photovoltaic harvesting system consisting of a string of PV blocks, a buck converter, and a battery, and present experimental results to validate the model. Although not having the highest level of accuracy, it was good enough that we could port the tuned parameters from the simulation directly to the microcontroller and have the algorithm work consistently with what was observed in the simulation.

The final question is **how could we extract the maximum energy from the system under these unstable irradiance conditions caused by unexpected shadowing events?** and Chapter 3 was where we answered this question. To choose the right tool for the right solution, we performed a deep analysis of many algorithms in the literature and analysed how they are presented to critically evaluate the algorithms better. From there, we saw the need to propose a novel testing methodology if we are to really tackle the question of varying partial shadows, which resulted in a simplified mathematical shading profile creator. Next, based on various methods in the literature and the constraints of simplicity, we proposed **fast GMPPT** algorithm based on the nVoc method, but with an extremely limited global search. We theoretically analysed its tracking accuracy to be around 86% to 97%, a very good result for such a limited search range. We then compared it with three other algorithms, MPPT Hill Climbing, GMPPT Deterministic Particle Swarm Optimisation, and GMPPT Grey Wolf Optimisation algorithm. It tracks very well in our sequence of irradiance steps, proving that it was capable of tracking GMPPs and it does so while causing little power jitters, unlike DPSO and GWO. Furthermore, its 94.74% energy efficiency came out on top when tested in our varying shading profile. Looking deeper into the data, we confirmed that fast GMPPT performs well when faced with fast and complex shading patterns, but is only competitive with HC when faced with simple and slow shadows. In general, we believe that fast GMPPT was a great solution for very basic hardware and could be implemented even on random hobby grade electronics. In general, it is a great solution given the constraint, but it would not always be the optimal solution if cost and complexity were no longer a significant concern. Rather, a distributed architecture should be considered so that a fast GMPPT does not have to handle more than three PV blocks in series.

The thesis overall has resulted in some scientific productions that are referenced in [183], [184], [138], [97], [185]. Although this work has made several interesting contributions, it is not without several drawbacks and simplified assumptions that were already discussed in each chapter. Despite that, we have reason to believe that the work here could be applied to systems with higher power output. For example, the solar estimation method in Chapter 1 is not exclusive to low-power solar

harvesting; any user with a clip-on fisheye lens for their smartphone could follow the procedure presented to estimate how much energy their home solar installation will produce. Of course, there are precautions to follow such as taking multiple photos at different places across the intended surface of the solar panels so that the program could come up with a shading pattern or trying to get a good orientation measurement to ensure higher solar estimate accuracy.

As for fast GMPPT, it only makes sense in a certain context, like for a string of less than four PV blocks where moving shadows are expected. However, it could be great when coupled with a distributed architecture. Let us provide an example where we have a system of 12 solar panels with 12 bypass diodes to be deployed among nature for an autonomous pumping application and we want to mitigate the effect of partial shadings. Here are the three options:

- Option A: A single converter that handles all 12 converters. They could run Hill Climbing or fast GMPPT.
- Option B: Three converters, each for four solar panels. They could run either HC or fast GMPPT.
- Option C: One converter per panel for a total of 12 converters. Each converter run HC.

If cost, complexity and logistic problems are not of concern, option C will be the best. However, if the bare minimum is prefered, we could go for option A with fast GMPPT, at least it will handle fast and complex shading patterns better than HC but the longer global search may be inefficient in slower and simpler shading patterns. The optimal solution would be option B where each converter runs fast GMPPT for a group of four solar panels. If partial shadings are very light, even HC could suffice. In general, the best solution is the one that suits the need. If anyone is in search of the absolute best solution, we will recommend not having shadows on the solar panels in the first place. This is an interesting analysis, and with the strong foundation we have set in this thesis, we believe that a future project could eventually provide the best solution for each given use case.

Bibliography

[1] CNRS, "ECONECT – Ecosystèmes connectés Sentinelles de l'Environnement - https://econect.cnrs.fr/." (Cited in page 1.)

[2] B. Genet, V. Boitier, Y. Briere, and F. Defay, "Forecasting of the Photovoltaic Electricity Production on a Sail Ship by Taking Account Shadow Effects," *Renewable Energy and Power Quality Journal*, vol. 17, pp. 288–293, July 2019. (Cited in page 2.)

[3] C. N. Obiora, A. Ali, and A. N. Hasan, "Forecasting Hourly Solar Irradiance Using Long Short-Term Memory (LSTM) Network," in *2020 11th International Renewable Energy Congress (IREC)*, pp. 1–6, Oct. 2020. ISSN: 2378-3451. (Cited in page 7.)

[4] N. ur Rehman, M. Hijazi, and M. Uzair, "Solar potential assessment of public bus routes for solar buses," *Renewable Energy*, vol. 156, pp. 193–200, Aug. 2020. (Cited in page 7.)

[5] F. Guarino, S. Longo, C. Hachem Vermette, M. Cellura, and V. La Rocca, "Life cycle assessment of solar communities," *Solar Energy*, vol. 207, pp. 209–217, Sept. 2020. (Cited in page 7.)

[6] E. Crenna, M. Gauch, R. Widmer, P. Wäger, and R. Hischier, "Towards more flexibility and transparency in life cycle inventories for Lithium-ion batteries," *Resources, Conservation and Recycling*, vol. 170, p. 105619, July 2021. (Cited in page 7.)

[7] A. Niewienda and F. Heidt, "SOMBRERO: A PC-tool to calculate shadows on arbitrarily oriented surfaces," *Solar Energy*, vol. 58, pp. 253–263, Oct. 1996. (Cited in page 10.)

[8] J. R. C. EU, "JRC Photovoltaic Geographical Information System (PVGIS) - European Commission - https://re.jrc.ec.europa.eu/pvg_tools/en/." (Cited in pages 10 and 11.)

[9] PVSyst, "PVsyst – Logiciel Photovoltaïque." (Cited in page 10.)

[10] NREL, "PVWatts Calculator - https://pvwatts.nrel.gov/." (Cited in page 10.)

[11] SolarGIS, "SolarGIS - https://solargis.com/." (Cited in page 10.)

[12] R. n. Canada, "RETScreen Expert - https://ressources-naturelles.canada.ca/cartes-outils-et-publications/outils/outils-modelisation/retscreen/7466," Mar. 2010. Last Modified: 2023-05-30 Publisher: Ressources naturelles Canada. (Cited in page 10.)

[13] C. S.r.L., "BlueSol Design 4 - http://www.bluesolpv.com/dnnsite/default.aspx." (Cited in page 10.)

[14] A. Inc., "HelioScope - https://helioscope.aurorasolar.com/." (Cited in page 10.)

[15] G. Valentin Software, "PV*SOL – https://pvsol.software/en/." (Cited in page 10.)

[16] G. Valentin Software, "PV*SOL online - https://pvsol-online.valentin-software.com/." (Cited in page 10.)

[17] G. S. T. Energie, "Solar Design Software | Solarius PV | https://www.accasoftware.com/en/solar-design-software." (Cited in page 10.)

[18] L. Laplace Systems Co., "Solar Pro - https://www.lapsys.co.jp/english/products/pro.html#Download." (Cited in page 10.)

[19] L. F-Chart Software, "PV F-CHART: Photovoltaic Systems Analysis | https://fchartsoftware.com/pvfchart/." (Cited in page 10.)

[20] V. S. AG, "Polysun – https://www.velasolaris.com/?lang=en." (Cited in page 10.)

[21] NREL, "System Advisor Model (SAM) - https://sam.nrel.gov/." (Cited in pages 10 and 57.)

[22] H. Energy, "HOMER Grid | https://www.homerenergy.com/." (Cited in page 10.)

[23] T. S. International, "archelios™ PRO - https://www.archelios.com/." (Cited in page 10.)

[24] INES, "AutoCalSol - https://autocalsol.ines-solaire.org/." (Cited in page 10.)

[25] D. D. Milosavljević, T. S. Kevkić, and S. J. Jovanović, "Review and validation of photovoltaic solar simulation tools/software based on case study:," *Open Physics*, vol. 20, pp. 431–451, Jan. 2022. Publisher: De Gruyter Open Access. (Cited in pages 10 and 25.)

[26] V. Energy, "Virginia Energy - Geology and Mineral Resources - LiDAR - https://www.energy.virginia.gov/geology/Lidar.shtml." (Cited in page 13.)

[27] USGS, "What is a geographic information system (GIS)? | U.S. Geological Survey - https://www.usgs.gov/faqs/what-geographic-information-system-gis." (Cited in pages 13 and 28.)

[28] O. Isabella, R. Santbergen, H. Ziar, A. Calcabrini, J. C. O. Lizcano, E. G. Goma, P. Nepal, V. Schepel, and M. Zeman, "Advanced modelling of E/UIPV

systems from location to load," in *2018 IEEE 7th World Conference on Photovoltaic Energy Conversion (WCPEC) (A Joint Conference of 45th IEEE PVSC, 28th PVSEC & 34th EU PVSEC)*, pp. 2691–2696, June 2018. ISSN: 0160-8371. (Cited in page 13.)

[29] D. A. Mira, M. Bartholomäus, S. Poessl, P. B. Poulsen, and S. V. Spataru, "Comparing the Accuracy of Horizon Shade Modelling Based on Digital Surface Models Versus Fisheye Sky Imaging," in *2022 IEEE 49th Photovoltaics Specialists Conference (PVSC)*, pp. 060–065, June 2022. (Cited in page 13.)

[30] GéoServices, "LiDAR HD | Géoservices - https://geoservices.ign.fr/lidarhd." (Cited in page 14.)

[31] C. D. Sánchez-Segura, L. Valentín-Coronado, M. I. Peña-Cruz, A. Díaz-Ponce, D. Moctezuma, G. Flores, and D. Riveros-Rosas, "Solar irradiance components estimation based on a low-cost sky-imager," *Solar Energy*, vol. 220, pp. 269–281, May 2021. (Cited in pages 14 and 15.)

[32] PG&E, "Taking a Fisheye Photograph." (Cited in page 14.)

[33] T. S. D. Company, "PanoramaMaster - The Solar Design Company - https://www.solardesign.co.uk/panoramamaster.php." (Cited in page 15.)

[34] Meteonorm, "Horicatcher - Imaging Tool for Meteonorm - https://meteonorm.com/produkt/horicatcher." (Cited in page 15.)

[35] Solmetric, "User guide for SunEye 210," 2011. (Cited in page 15.)

[36] S. Pathfinder, "User guide to SolarPathfinder Unit," 2016. (Cited in page 15.)

[37] PG&E, "User guide for PG&E SunPath software." (Cited in page 15.)

[38] G. Dawson, "Sun Seeker - https://sun-seeker.soft112.com/." (Cited in page 15.)

[39] T. Moder, "SUNNYTRACK App - Plan the Sun's Position and Shadows - https://www.sunnytrack.app/." (Cited in page 15.)

[40] S. Duluk, A. Kwok, and N. Heather, "Comparison of Solar Evaluation Tools: From Learning to Practice," 2021. (Cited in page 15.)

[41] A. Orioli and A. D. Gangi, "An improved photographic method to estimate the shading effect of obstructions," *Solar Energy*, vol. 86, pp. 3470–3488, Nov. 2012. (Cited in page 15.)

[42] M. J. Oliveira Panão, R. F. Carreira, and M. C. Brito, "Determining the shading correction factor using a smartphone camera with a fisheye lens," *Solar Energy*, vol. 190, pp. 596–607, Sept. 2019. (Cited in pages 15, 16, and 22.)

[43] J. A. Ranalli, "Solar Survey: Development and validation of a smartphone-based solar site assessment tool," *Solar Energy*, vol. 122, pp. 1199–1213, Dec. 2015. (Cited in page 15.)

[44] F. Wang, Z. Xuan, Z. Zhen, Y. Li, K. Li, L. Zhao, M. Shafie-khah, and J. P. Catalão, "A minutely solar irradiance forecasting method based on real-time sky image-irradiance mapping model," *Energy Conversion and Management*, vol. 220, p. 113075, Sept. 2020. (Cited in page 15.)

[45] Y. Chu, M. Li, H. T. Pedro, and C. F. Coimbra, "A network of sky imagers for spatial solar irradiance assessment," *Renewable Energy*, vol. 187, pp. 1009–1019, Mar. 2022. (Cited in page 15.)

[46] N. B. Blum, S. Wilbert, B. Nouri, J. Lezaca, D. Huckebrink, A. Kazantzidis, D. Heinemann, L. F. Zarzalejo, M. J. Jiménez, and R. Pitz-Paal, "Measurement of diffuse and plane of array irradiance by a combination of a pyranometer and an all-sky imager," *Solar Energy*, vol. 232, pp. 232–247, Jan. 2022. (Cited in page 15.)

[47] G. Terrén-Serrano and M. Martínez-Ramón, "Deep learning for intra-hour solar forecasting with fusion of features extracted from infrared sky images," *Information Fusion*, vol. 95, pp. 42–61, July 2023. (Cited in page 15.)

[48] J. Hardy, R. Melloh, G. Koenig, D. Marks, A. Winstral, J. Pomeroy, and T. Link, "Solar radiation transmission through conifer canopies," *Agricultural and Forest Meteorology*, vol. 126, pp. 257–270, Nov. 2004. (Cited in page 15.)

[49] Y. Wang, "A new parameterization of canopy spectral response to incident solar radiation: case study with hyperspectral data from pine dominant forest," *Remote Sensing of Environment*, vol. 85, pp. 304–315, May 2003. (Cited in page 15.)

[50] R. Gunasagaran, L. Kamarudin, E. Kanagaraj, A. Zakaria, and A. Shakaff, "Internet of Things: Solar power under forest canopy," in *2016 IEEE Student Conference on Research and Development (SCOReD)*, pp. 1–4, Dec. 2016. (Cited in page 15.)

[51] OpenCV, "OpenCV: Custom Calibration Pattern for 3D reconstruction - https://docs.opencv.org/3.4." (Cited in page 22.)

[52] OpenCV, "OpenCV Project - https://opencv.org/." (Cited in page 22.)

[53] Scaramuzza, "Davide Scaramuzza - OCamCalib: Omnidirectional Camera Calibration Toolbox for Matlab (https://sites.google.com/site/scarabotix/ocamcalib-omnidirectional-camera-calibration-toolbox-for-matlab." (Cited in page 22.)

[54] NASA, "NASA POWER | Prediction Of Worldwide Energy Resources | https://power.larc.nasa.gov/." (Cited in page 25.)

[55] H. Hottel and A. Whillier, "Evaluation of flat-plate solar collector performance," *Trans. Conf. Use of Solar Energy; ()*, vol. 3 (Thermal Processes) Part 2, Jan. 1955. (Cited in page 26.)

[56] S. O. S. electronic, "ET-M53620 ED | ETSOLAR." (Cited in page 29.)

[57] SoDa, "SoDa - Solar Radiation Data - Solar Energy Services for Professionals - https://www.soda-pro.com." (Cited in page 31.)

[58] D. S. Philipps, F. Ise, W. Warmuth, and P. P. GmbH, "Photovoltaics Report," (Cited in page 43.)

[59] "Monocrystalline - an overview | ScienceDirect Topics." (Cited in page 43.)

[60] NREL, "Best Research-Cell Efficiency Chart - https://www.nrel.gov/pv/cell-efficiency.html." (Cited in page 43.)

[61] J. Solórzano and M. Egido, "Hot-spot mitigation in PV arrays with distributed MPPT (DMPPT)," *Solar Energy*, vol. 101, pp. 131–137, Mar. 2014. (Cited in pages 45 and 88.)

[62] A. Ramyar, H. Iman-Eini, and S. Farhangi, "Global Maximum Power Point Tracking Method for Photovoltaic Arrays Under Partial Shading Conditions," *IEEE Transactions on Industrial Electronics*, vol. 64, pp. 2855–2864, Apr. 2017. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 46, 56, 101, 108, and 109.)

[63] G.-J. Fang and K.-L. Lian, "A maximum power point tracking method based on multiple perturb-and-observe method for overcoming solar partial shaded problems," in *2017 6th International Conference on Clean Electrical Power (ICCEP)*, pp. 68–73, June 2017. ISSN: 2474-9664. (Cited in pages 46, 56, 93, and 108.)

[64] K. S. Tey and S. Mekhilef, "Modified Incremental Conductance Algorithm for Photovoltaic System Under Partial Shading Conditions and Load Variation," *IEEE Transactions on Industrial Electronics*, vol. 61, pp. 5384–5392, Oct. 2014. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 46, 56, 94, 101, and 108.)

[65] M. G. Villalva, J. R. Gazoli, and E. R. Filho, "Comprehensive Approach to Modeling and Simulation of Photovoltaic Arrays," *IEEE Transactions on Power Electronics*, vol. 24, pp. 1198–1208, May 2009. Conference Name: IEEE Transactions on Power Electronics. (Cited in page 49.)

[66] N. J. P. Laboratory, *Solar Cell Array Design Handbook Vol I.* Oct. 1976. (Cited in page 49.)

[67] D. Cotfas, P. Cotfas, and S. Kaplanis, "Methods to determine the dc parameters of solar cells: A critical review," *Renewable and Sustainable Energy Reviews*, vol. 28, pp. 588–596, Dec. 2013. (Cited in pages 50 and 57.)

[68] A. Dehghanzadeh, G. Farahani, and M. Maboodi, "A novel approximate explicit double-diode model of solar cells for use in simulation studies," *Renewable Energy*, vol. 103, pp. 468–477, Apr. 2017. (Cited in page 50.)

[69] N. B. Nguyen, *Modeling and Simulation of Photovoltaic Generator*. other, Laboratory on Plasma and Conversion of Energy (LAPLACE), UMR 5213, 2 rue Charles Camichel, BP 7122, 31071 Toulouse Cedex 7, 2014. Pages: 74. (Cited in page 51.)

[70] M. Alonsogarcia, J. Ruiz, and F. Chenlo, "Experimental study of mismatch and shading effects in the – characteristic of a photovoltaic module," *Solar Energy Materials and Solar Cells*, vol. 90, pp. 329–340, Feb. 2006. (Cited in page 58.)

[71] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth, "On the LambertW function," *Advances in Computational Mathematics*, vol. 5, pp. 329–359, Dec. 1996. (Cited in page 70.)

[72] P. Upadhyay, S. Pulipaka, R. Kumar, and M. Sharma, "Parameter extraction of solar photovoltaic system using Lambert-W function for different environmental condition," in *2016 IEEE Region 10 Conference (TENCON)*, pp. 1884–1888, Nov. 2016. ISSN: 2159-3450. (Cited in page 70.)

[73] B. Romero, G. del Pozo, and B. Arredondo, "Exact analytical solution of a two diode circuit model for organic solar cells showing S-shape using Lambert W-functions," *Solar Energy*, vol. 86, pp. 3026–3029, Oct. 2012. (Cited in page 70.)

[74] Y. Ma, H. Wen, and X. Li, "A Novel Photovoltaic String Model Based on the Lambert w Function for Partial Shading Conditions," in *2018 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES)*, pp. 1–6, Dec. 2018. (Cited in page 70.)

[75] E. Roibás-Millán, J. L. Cubero-Estalrrich, A. Gonzalez-Estrada, R. Jado-Puente, M. Sanabria-Pinzón, D. Alfonso-Corcuera, J. M. Álvarez, J. Cubas, and S. Pindado, "Lambert W-function simplified expressions for photovoltaic current-voltage modelling," in *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I CPS Europe)*, pp. 1–6, June 2020. (Cited in page 70.)

[76] E. Batzelis, G. Anagnostou, C. Chakraborty, and B. Pal, "Computation of the Lambert W function in photovoltaic modeling," May 2019. (Cited in page 70.)

[77] P. Merhej, E. Dallago, and D. Finarelli, "Effect of capacitance on the output characteristics of solar cells," in *6th Conference on Ph.D. Research in Microelectronics & Electronics*, pp. 1–4, July 2010. (Cited in page 77.)

[78] J. V. Gragger, A. Haumer, and M. Einhorn, "Averaged Model of a Buck Converter for Efficiency Analysis," 2010. (Cited in page 79.)

[79] A. Bidram, A. Davoudi, and R. S. Balog, "Control and Circuit Techniques to Mitigate Partial Shading Effects in Photovoltaic Arrays," *IEEE Journal of Photovoltaics*, vol. 2, pp. 532–546, Oct. 2012. Conference Name: IEEE Journal of Photovoltaics. (Cited in pages 87 and 88.)

[80] G. Velasco-Quesada, F. Guinjoan-Gispert, R. Pique-Lopez, M. Roman-Lumbreras, and A. Conesa-Roca, "Electrical PV Array Reconfiguration Strategy for Energy Extraction Improvement in Grid-Connected PV Systems," *IEEE Transactions on Industrial Electronics*, vol. 56, pp. 4319–4331, Nov. 2009. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in page 88.)

[81] Y. El Basri, *Architecture de puissance distribuée reconfigurable dédiée à l'optimisation de l'énergie photovoltaïque.* These de doctorat, Toulouse 3, Jan. 2013. (Cited in page 88.)

[82] L. F. Lavado Villa, *Architectures de puissance et commandes associées pour la gestion des ombrages dans les installations photovoltaïques. Power Architectures and Control Systems Associated to the Management of Shadows in Photovoltaic Plants.* These de doctorat, Grenoble, Oct. 2013. (Cited in page 88.)

[83] W. Saranrom and S. Polmai, "The efficiency improvement of series connected PV panels operating under partial shading condition by using per-panel DC/DC converter," in *The 8th Electrical Engineering/ Electronics, Computer, Telecommunications and Information Technology (ECTI) Association of Thailand - Conference 2011*, pp. 760–763, May 2011. (Cited in page 88.)

[84] L. Gao, R. A. Dougal, S. Liu, and A. P. Iotova, "Parallel-Connected Solar PV System to Address Partial and Rapidly Fluctuating Shadow Conditions," *IEEE Transactions on Industrial Electronics*, vol. 56, pp. 1548–1556, May 2009. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in page 88.)

[85] S. Petibon, *Nouvelles architectures distribuées de gestion et de conversion de l'énergie pour les applications photovoltaïques.* These de doctorat, Toulouse 3, Jan. 2009. (Cited in page 88.)

[86] C. Cabal, *Optimisation énergétique de l'étage d'adaptation électronique dédié à la conversion photovoltaïque.* These de doctorat, Toulouse 3, Jan. 2008. (Cited in page 88.)

[87] A. Cid Pastor, *Conception et réalisation de modules photovoltaïques électroniques.* These de doctorat, Toulouse, INSA, Jan. 2006. (Cited in page 88.)

[88] J. M. Riquelme-Dominguez and S. Martinez, "Comparison of Different Photovoltaic Perturb and Observe Algorithms for Drift Avoidance in Fluctuating Irradiance Conditions," in *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I CPS Europe)*, pp. 1–5, June 2020. (Cited in pages 93, 108, and 109.)

[89] F. Liu, S. Duan, F. Liu, B. Liu, and Y. Kang, "A Variable Step Size INC MPPT Method for PV Systems," *IEEE Transactions on Industrial Electronics*, vol. 55, pp. 2622–2628, July 2008. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 93, 96, 100, and 108.)

[90] W. Li, Y. Zheng, W. Li, Y. zhao, and X. He, "A smart and simple PV charger for portable applications," in *2010 Twenty-Fifth Annual IEEE Applied Power Electronics Conference and Exposition (APEC)*, pp. 2080–2084, Feb. 2010. ISSN: 1048-2334. (Cited in pages 93, 98, and 108.)

[91] S. Jain and V. Agarwal, "A new algorithm for rapid tracking of approximate maximum power point in photovoltaic systems," *IEEE Power Electronics Letters*, vol. 2, pp. 16–19, Mar. 2004. Conference Name: IEEE Power Electronics Letters. (Cited in pages 93, 94, 96, and 108.)

[92] G. Farivar, B. Asaei, and S. Mehrnami, "An Analytical Solution for Tracking Photovoltaic Module MPP," *IEEE Journal of Photovoltaics*, vol. 3, pp. 1053–1061, July 2013. Conference Name: IEEE Journal of Photovoltaics. (Cited in pages 93, 98, 101, and 108.)

[93] V. V. R. Scarpa, S. Buso, and G. Spiazzi, "Low-Complexity MPPT Technique Exploiting the PV Module MPP Locus Characterization," *IEEE Transactions on Industrial Electronics*, vol. 56, pp. 1531–1538, May 2009. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 93, 94, and 108.)

[94] M. M. Rahman and M. S. Islam, "Artificial Neural Network Based Maximum Power Point Tracking of a Photovoltaic System," in *2019 3rd International Conference on Electrical, Computer Telecommunication Engineering (ICECTE)*, pp. 117–120, Dec. 2019. (Cited in pages 93, 100, 101, and 108.)

[95] X. Li, H. Wen, Y. Hu, and L. Jiang, "A novel beta parameter based fuzzy-logic controller for photovoltaic MPPT application," *Renewable Energy*, vol. 130, pp. 416–427, Jan. 2019. (Cited in pages 93, 97, 99, 100, 108, 109, and 111.)

[96] M. A. Ghasemi, H. M. Foroushani, and M. Parniani, "Partial Shading Detection and Smooth Maximum Power Point Tracking of PV Arrays Under PSC," *IEEE Transactions on Power Electronics*, vol. 31, pp. 6281–6292, Sept. 2016. Conference Name: IEEE Transactions on Power Electronics. (Cited in pages 93, 101, and 108.)

[97] K. B. K. Cao and V. Boitier, "A novel simple GMPPT method based on probability distribution of global maximum power point under partial shading conditions," in *IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society*, pp. 1–6, Oct. 2022. ISSN: 2577-1647. (Cited in pages 93, 102, and 132.)

[98] K. Ishaque and Z. Salam, "A Deterministic Particle Swarm Optimization Maximum Power Point Tracker for Photovoltaic System Under Partial Shading Condition," *IEEE Transactions on Industrial Electronics*, vol. 60, pp. 3195–3206, Aug. 2013. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 93, 103, 104, 108, and 112.)

[99] K. S. Tey, S. Mekhilef, H.-T. Yang, and M.-K. Chuang, "A Differential Evolution Based MPPT Method for Photovoltaic Modules under Partial Shading Conditions," *International Journal of Photoenergy*, vol. 2014, p. e945906, May 2014. Publisher: Hindawi. (Cited in pages 93 and 108.)

[100] S. Mohanty, B. Subudhi, and P. K. Ray, "A New MPPT Design Using Grey Wolf Optimization Technique for Photovoltaic System Under Partial Shading Conditions," *IEEE Transactions on Sustainable Energy*, vol. 7, pp. 181–188, Jan. 2016. Conference Name: IEEE Transactions on Sustainable Energy. (Cited in page 93.)

[101] K. Sundareswaran, P. Sankar, P. S. R. Nayak, S. P. Simon, and S. Palani, "Enhanced Energy Output From a PV System Under Partial Shaded Conditions Through Artificial Bee Colony," *IEEE Transactions on Sustainable Energy*, vol. 6, pp. 198–209, Jan. 2015. Conference Name: IEEE Transactions on Sustainable Energy. (Cited in pages 93, 108, and 112.)

[102] E. Lodhi, P. Yang, L. Wang, M. A. Khan, Z. Lodhi, U. Javed, and Q. Saleem, "Dragonfly Optimization-based MPPT Algorithm for Standalone PV System under Partial Shading," in *2021 IEEE International Conference on Emergency Science and Information Technology (ICESIT)*, pp. 277–283, Nov. 2021. (Cited in pages 93 and 108.)

[103] R. Sridhar, C. Subramani, and S. Pathy, "A grasshopper optimization algorithm aided maximum power point tracking for partially shaded photovoltaic systems," *Computers & Electrical Engineering*, vol. 92, p. 107124, June 2021. (Cited in pages 93 and 108.)

[104] J. Prasanth Ram and N. Rajasekar, "A Novel Flower Pollination Based Global Maximum Power Point Method for Solar Maximum Power Point Tracking," *IEEE Transactions on Power Electronics*, vol. 32, pp. 8486–8499, Nov. 2017. Conference Name: IEEE Transactions on Power Electronics. (Cited in pages 93 and 108.)

[105] L. L. Jiang and D. L. Maskell, "A uniform implementation scheme for evolutionary optimization algorithms and the experimental implementation of an

ACO based MPPT for PV systems under partial shading," in *2014 IEEE Symposium on Computational Intelligence Applications in Smart Grid (CIASG)*, pp. 1–8, Dec. 2014. ISSN: 2326-7690. (Cited in pages 93 and 108.)

[106] K. Sundareswaran, S. Peddapati, and S. Palani, "MPPT of PV Systems Under Partial Shaded Conditions Through a Colony of Flashing Fireflies," *IEEE Transactions on Energy Conversion*, vol. 29, pp. 463–472, June 2014. Conference Name: IEEE Transactions on Energy Conversion. (Cited in pages 93 and 108.)

[107] J. Ahmed and Z. Salam, "A Maximum Power Point Tracking (MPPT) for PV system using Cuckoo Search with partial shading capability," *Applied Energy*, vol. 119, pp. 118–130, Apr. 2014. (Cited in pages 93 and 108.)

[108] R. S. Pal and V. Mukherjee, "A novel population based maximum point tracking algorithm to overcome partial shading issues in solar photovoltaic technology," *Energy Conversion and Management*, vol. 244, p. 114470, Sept. 2021. (Cited in pages 93 and 108.)

[109] I. Pervez, I. Shams, S. Mekhilef, A. Sarwar, M. Tariq, and B. Alamri, "Most Valuable Player Algorithm based Maximum Power Point Tracking for a Partially Shaded PV Generation System," *IEEE Transactions on Sustainable Energy*, vol. 12, pp. 1876–1890, Oct. 2021. Conference Name: IEEE Transactions on Sustainable Energy. (Cited in pages 93 and 108.)

[110] H. Rezk and A. Fathy, "Simulation of global MPPT based on teaching–learning-based optimization technique for partially shaded PV system," *Electrical Engineering*, vol. 99, pp. 847–859, Sept. 2017. (Cited in pages 93 and 108.)

[111] M. Miyatake, T. Inada, I. Hiratsuka, H. Zhao, H. Otsuka, and M. Nakano, "Control characteristics of a fibonacci-search-based maximum power point tracker when a photovoltaic array is partially shaded," in *The 4th International Power Electronics and Motion Control Conference, 2004. IPEMC 2004.*, vol. 2, pp. 816–821 Vol.2, Aug. 2004. (Cited in pages 93 and 108.)

[112] S. Lyden and M. E. Haque, "A Simulated Annealing Global Maximum Power Point Tracking Approach for PV Modules Under Partial Shading Conditions," *IEEE Transactions on Power Electronics*, vol. 31, pp. 4171–4181, June 2016. Conference Name: IEEE Transactions on Power Electronics. (Cited in pages 93, 103, 108, 109, and 112.)

[113] A. F. Mirza, M. Mansoor, and Q. Ling, "A novel MPPT technique based on Henry gas solubility optimization," *Energy Conversion and Management*, vol. 225, p. 113409, Dec. 2020. (Cited in pages 93 and 112.)

[114] J. Ahmed and Z. Salam, "A Modified P&O Maximum Power Point Tracking Method With Reduced Steady-State Oscillation and Improved Tracking Ef-

ficiency," *IEEE Transactions on Sustainable Energy*, vol. 7, pp. 1506–1515, Oct. 2016. Conference Name: IEEE Transactions on Sustainable Energy. (Cited in pages 94, 100, 108, 111, and 112.)

[115] M. Killi and S. Samanta, "Modified Perturb and Observe MPPT Algorithm for Drift Avoidance in Photovoltaic Systems," *IEEE Transactions on Industrial Electronics*, vol. 62, pp. 5549–5559, Sept. 2015. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 94, 95, and 108.)

[116] K. L. Lian, J. H. Jhang, and I. S. Tian, "A Maximum Power Point Tracking Method Based on Perturb-and-Observe Combined With Particle Swarm Optimization," *IEEE Journal of Photovoltaics*, vol. 4, pp. 626–633, Mar. 2014. Conference Name: IEEE Journal of Photovoltaics. (Cited in pages 94, 108, and 112.)

[117] D. Pilakkat and S. Kanthalakshmi, "An improved P&O algorithm integrated with artificial bee colony for photovoltaic systems under partial shading conditions," *Solar Energy*, vol. 178, pp. 37–47, Jan. 2019. (Cited in pages 95 and 108.)

[118] D. Sera, R. Teodorescu, J. Hantschel, and M. Knoll, "Optimized Maximum Power Point Tracker for Fast-Changing Environmental Conditions," *IEEE Transactions on Industrial Electronics*, vol. 55, pp. 2629–2637, July 2008. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 95 and 108.)

[119] G.-C. Hsieh, H.-I. Hsieh, C.-Y. Tsai, and C.-H. Wang, "Photovoltaic Power-Increment-Aided Incremental-Conductance MPPT With Two-Phased Tracking," *IEEE Transactions on Power Electronics*, vol. 28, pp. 2895–2911, June 2013. Conference Name: IEEE Transactions on Power Electronics. (Cited in pages 96 and 108.)

[120] X. Li, H. Wen, and C. Zhao, "Improved beta parameter based MPPT method in photovoltaic system," in *2015 9th International Conference on Power Electronics and ECCE Asia (ICPE-ECCE Asia)*, pp. 1405–1412, June 2015. ISSN: 2150-6086. (Cited in pages 97 and 108.)

[121] X. Li, H. Wen, L. Jiang, E. G. Lim, Y. Du, and C. Zhao, "Photovoltaic Modified -Parameter-based MPPT Method with Fast Tracking," *Journal of Power Electronics*, vol. 16, no. 1, pp. 9–17, 2016. Publisher: The Korean Institute of Power Electronics. (Cited in pages 97 and 108.)

[122] S. K. Kollimalla and M. K. Mishra, "Variable Perturbation Size Adaptive P&O MPPT Algorithm for Sudden Changes in Irradiance," *IEEE Transactions on Sustainable Energy*, vol. 5, pp. 718–728, July 2014. Conference Name: IEEE Transactions on Sustainable Energy. (Cited in pages 98 and 108.)

[123] M. H. Moradi, S. M. Reza Tousi, M. Nemati, N. Saadat Basir, and N. Shalavi, "A robust hybrid method for maximum power point tracking in photovoltaic systems," *Solar Energy*, vol. 94, pp. 266–276, Aug. 2013. (Cited in pages 98 and 108.)

[124] T. Instruments, "BQ25570 data sheet, product information and support | TI.com." (Cited in page 98.)

[125] STMicroelectronics, "SPV1050 - Ultra low power energy harvester and battery charger with embedded MPPT and LDOs - STMicroelectronics." (Cited in page 98.)

[126] M. Lei, S. Yaojie, L. Yandan, B. Zhifeng, T. Liqin, and S. Jieqiong, "A high performance MPPT control method," in *2011 International Conference on Materials for Renewable Energy & Environment*, vol. 1, pp. 195–199, May 2011. (Cited in page 98.)

[127] M. H. Moradi and A. R. Reisi, "A hybrid maximum power point tracking method for photovoltaic systems," *Solar Energy*, vol. 85, pp. 2965–2976, Nov. 2011. (Cited in pages 98 and 108.)

[128] B. N. Alajmi, K. H. Ahmed, S. J. Finney, and B. W. Williams, "Fuzzy-Logic-Control Approach of a Modified Hill-Climbing Method for Maximum Power Point in Microgrid Standalone Photovoltaic System," *IEEE Transactions on Power Electronics*, vol. 26, pp. 1022–1030, Apr. 2011. Conference Name: IEEE Transactions on Power Electronics. (Cited in pages 100 and 108.)

[129] B. N. Alajmi, K. H. Ahmed, S. J. Finney, and B. W. Williams, "A Maximum Power Point Tracking Technique for Partially Shaded Photovoltaic Systems in Microgrids," *IEEE Transactions on Industrial Electronics*, vol. 60, pp. 1596–1606, Apr. 2013. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in pages 100 and 108.)

[130] A. El Khateb, N. A. Rahim, J. Selvaraj, and M. N. Uddin, "Fuzzy-Logic-Controller-Based SEPIC Converter for Maximum Power Point Tracking," *IEEE Transactions on Industry Applications*, vol. 50, pp. 2349–2358, July 2014. Conference Name: IEEE Transactions on Industry Applications. (Cited in pages 100 and 108.)

[131] N. Shah and C. Rajagopalan, "Experimental evaluation of a partially shaded photovoltaic system with a fuzzy logic-based peak power tracking control strategy," *IET Renewable Power Generation*, vol. 10, no. 1, pp. 98–107, 2016. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1049/iet-rpg.2015.0098. (Cited in pages 100, 101, and 108.)

[132] R. Kumar, B. Kumar, and S. D., "Fuzzy Logic based Improved P&O MPPT Technique for Partial Shading Conditions," in *2018 International Conference*

on Computing, Power and Communication Technologies (GUCON), pp. 775–779, Sept. 2018. (Cited in pages 100 and 108.)

[133] U. Yilmaz, A. Kircay, and S. Borekci, "PV system fuzzy logic MPPT method and PI control as a charge controller," *Renewable and Sustainable Energy Reviews*, vol. 81, pp. 994–1001, Jan. 2018. (Cited in pages 100 and 108.)

[134] X. Meng, Y. An, H. Wang, Q. Yao, and C. Liang, "Tracking the Maximum Power Point of Photovoltaic Power Generation Based on Self-coding Neural Network," in *2019 Chinese Control And Decision Conference (CCDC)*, pp. 592–597, June 2019. ISSN: 1948-9447. (Cited in pages 100, 101, and 108.)

[135] H. S. Agha, Z.-u. Koreshi, and M. B. Khan, "Artificial neural network based maximum power point tracking for solar photovoltaics," in *2017 International Conference on Information and Communication Technologies (ICICT)*, pp. 150–155, Dec. 2017. (Cited in pages 100, 101, and 108.)

[136] L. P. Jyothy and M. R. Sindhu, "An Artificial Neural Network based MPPT Algorithm For Solar PV System," in *2018 4th International Conference on Electrical Energy Systems (ICEES)*, pp. 375–380, Feb. 2018. (Cited in pages 100, 101, and 108.)

[137] L. Tang, W. Xu, and C. X. Mu, "Maximum power point tracking strategy for photovoltaic system based on probability," in *2015 IEEE International Conference on Applied Superconductivity and Electromagnetic Devices (ASEMD)*, pp. 60–61, Nov. 2015. (Cited in pages 102 and 108.)

[138] K. Cao and V. Boitier, "A novel simple GMPPT method based on probability distribution of global maximum power point under partial shading conditions," *Renewable Energy and Power Quality Journal*, vol. 20, pp. 240–245, Sept. 2022. (Cited in pages 102 and 132.)

[139] S. Mohanty, B. Subudhi, and P. K. Ray, "A Grey Wolf-Assisted Perturb & Observe MPPT Algorithm for a PV System," *IEEE Transactions on Energy Conversion*, vol. 32, pp. 340–347, Mar. 2017. Conference Name: IEEE Transactions on Energy Conversion. (Cited in pages 103, 106, 108, 109, 111, and 112.)

[140] M. Miyatake, M. Veerachary, F. Toriumi, N. Fujii, and H. Ko, "Maximum Power Point Tracking of Multiple Photovoltaic Arrays: A PSO Approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, pp. 367–380, Jan. 2011. Conference Name: IEEE Transactions on Aerospace and Electronic Systems. (Cited in pages 103, 108, 109, 110, and 112.)

[141] A. F. Mirza, M. Mansoor, and Q. Ling, "A novel MPPT technique based on Henry gas solubility optimization," *Energy Conversion and Management*, vol. 225, p. 113409, Dec. 2020. (Cited in page 108.)

[142] M. Mansoor, A. F. Mirza, Q. Ling, and M. Y. Javed, "Novel Grass Hopper optimization based MPPT of PV systems for complex partial shading conditions," *Solar Energy*, vol. 198, pp. 499–518, Mar. 2020. (Cited in page 108.)

[143] N. Priyadarshi, V. K. Ramachandaramurthy, S. Padmanaban, and F. Azam, "An Ant Colony Optimized MPPT for Standalone Hybrid PV-Wind Power System with Single Cuk Converter," *Energies*, vol. 12, p. 167, Jan. 2019. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute. (Cited in page 108.)

[144] K. S. Tey, S. Mekhilef, M. Seyedmahmoudian, B. Horan, A. T. Oo, and A. Stojcevski, "Improved Differential Evolution-Based MPPT Algorithm Using SEPIC for PV Systems Under Partial Shading Conditions and Load Variation," *IEEE Transactions on Industrial Informatics*, vol. 14, pp. 4322–4333, Oct. 2018. Conference Name: IEEE Transactions on Industrial Informatics. (Cited in pages 108, 109, and 111.)

[145] T. Pei, X. Hao, and Q. Gu, "A Novel Global Maximum Power Point Tracking Strategy Based on Modified Flower Pollination Algorithm for Photovoltaic Systems under Non-Uniform Irradiation and Temperature Conditions," *Energies*, vol. 11, p. 2708, Oct. 2018. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute. (Cited in page 108.)

[146] S. Titri, C. Larbes, K. Y. Toumi, and K. Benatchba, "A new MPPT controller based on the Ant colony optimization algorithm for Photovoltaic systems under partial shading conditions," *Applied Soft Computing*, vol. 58, pp. 465–479, Sept. 2017. (Cited in page 108.)

[147] S. K. Cherukuri and S. R. Rayapudi, "Enhanced Grey Wolf Optimizer Based MPPT Algorithm of PV System Under Partial Shaded Condition," *International Journal of Renewable Energy Development*, vol. 6, pp. 203–212, Nov. 2017. Publisher: Center of Biomass & Renewable Energy, Diponegoro University. (Cited in page 108.)

[148] B. Veerasamy, A. R. Thelkar, G. Ramu, and T. Takeshita, "Efficient MPPT control for fast irradiation changes and partial shading conditions on PV systems," in *2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA)*, pp. 358–363, Nov. 2016. (Cited in page 108.)

[149] C. Manickam, G. R. Raman, G. P. Raman, S. I. Ganesan, and C. Nagamani, "A Hybrid Algorithm for Tracking of GMPP Based on P&o; PSO With Reduced Power Oscillation in String Inverters," *IEEE Transactions on Industrial Electronics*, vol. 63, pp. 6097–6106, Oct. 2016. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in page 108.)

[150] J. Ahmad, F. Spertino, P. Di Leo, and A. Ciocia, "A Variable Step Size Perturb and Observe Method Based MPPT for Partially Shaded Photovoltaic Arrays," in *PCIM Europe 2016; International Exhibition and Conference for*

Power Electronics, Intelligent Motion, Renewable Energy and Energy Management, pp. 1–8, May 2016. (Cited in page 108.)

[151] K. Sundareswaran, V. Vignesh kumar, and S. Palani, "Application of a combined particle swarm optimization and perturb and observe method for MPPT in PV systems under partial shading conditions," *Renewable Energy*, vol. 75, pp. 308–317, Mar. 2015. (Cited in page 108.)

[152] M. Seyedmahmoudian, R. Rahmani, S. Mekhilef, A. Maung Than Oo, A. Stojcevski, T. K. Soon, and A. S. Ghandhari, "Simulation and Hardware Implementation of New Maximum Power Point Tracking Technique for Partially Shaded PV System Using Hybrid DEPSO Method," *IEEE Transactions on Sustainable Energy*, vol. 6, pp. 850–862, July 2015. Conference Name: IEEE Transactions on Sustainable Energy. (Cited in page 108.)

[153] M. Kermadi and E. M. Berkouk, "A Hybrid PSO-PI based Maximum Power Point Tracking algorithm using adaptive sampling time strategy," in *2015 4th International Conference on Electrical Engineering (ICEE)*, pp. 1–6, Dec. 2015. (Cited in page 108.)

[154] O. Dahhani, A. E. Jouni, B. Sefriti, and I. Boumhidi, "Optimal perturb and observe control for MPPT based on least square support vector machines algorithm," in *2015 Intelligent Systems and Computer Vision (ISCV)*, pp. 1–7, Mar. 2015. (Cited in page 108.)

[155] A. s. Benyoucef, A. Chouder, K. Kara, S. Silvestre, and O. A. sahed, "Artificial bee colony based algorithm for maximum power point tracking (MPPT) for PV systems operating under partial shaded conditions," *Applied Soft Computing*, vol. 32, pp. 38–48, July 2015. (Cited in pages 108 and 112.)

[156] M. F. N. Tajuddin, S. M. Ayob, and Z. Salam, "Global maximum power point tracking of PV system using dynamic population size differential evolution (DynNP-DE) algorithm," in *2014 IEEE Conference on Energy Conversion (CENCON)*, pp. 254–259, Oct. 2014. (Cited in page 108.)

[157] R. Faraji, A. Rouholamini, H. Naji, R. Fadaeinedjad, and M. Chavoshian, "FPGA-based real time incremental conductance maximum power point tracking controller for photovoltaic systems," *IET Power Electronics*, vol. 7, pp. 1294–1304, May 2014. (Cited in page 108.)

[158] A. Elnosh, V. Khadkikar, W. Xiao, and J. L. Kirtely, "An improved Extremum-Seeking based MPPT for grid-connected PV systems with partial shading," in *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)*, pp. 2548–2553, June 2014. ISSN: 2163-5145. (Cited in page 108.)

[159] M. Boztepe, F. Guinjoan, G. Velasco-Quesada, S. Silvestre, A. Chouder, and E. Karatepe, "Global MPPT Scheme for Photovoltaic String Inverters Based on Restricted Voltage Window Search Algorithm," *IEEE Transactions on*

Industrial Electronics, vol. 61, pp. 3302–3312, July 2014. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in page 108.)

[160] S. H. Hosseini, A. Farakhor, and S. Khadem Haghighian, "Novel algorithm of MPPT for PV array based on variable step Newton-Raphson method through model predictive control," in *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, pp. 1577–1582, Oct. 2013. ISSN: 2093-7121. (Cited in page 108.)

[161] Y.-H. Liu, S.-C. Huang, J.-W. Huang, and W.-C. Liang, "A Particle Swarm Optimization-Based Maximum Power Point Tracking Algorithm for PV Systems Operating Under Partially Shaded Conditions," *IEEE Transactions on Energy Conversion*, vol. 27, pp. 1027–1035, Dec. 2012. Conference Name: IEEE Transactions on Energy Conversion. (Cited in pages 108, 112, and 113.)

[162] E. Koutroulis and F. Blaabjerg, "A New Technique for Tracking the Global Maximum Power Point of PV Arrays Operating Under Partial-Shading Conditions," *IEEE Journal of Photovoltaics*, vol. 2, pp. 184–190, Apr. 2012. Conference Name: IEEE Journal of Photovoltaics. (Cited in page 108.)

[163] K. Ishaque, Z. Salam, M. Amjad, and S. Mekhilef, "An Improved Particle Swarm Optimization (PSO) based MPPT for PV with Reduced Steady State Oscillation," *IEEE Transactions on Power Electronics*, vol. 27, pp. 3627–33638, Aug. 2012. (Cited in page 108.)

[164] L. Zhou, Y. Chen, K. Guo, and F. Jia, "New Approach for MPPT Control of Photovoltaic System With Mutative-Scale Dual-Carrier Chaotic Search," *IEEE Transactions on Power Electronics*, vol. 26, pp. 1038–1048, Apr. 2011. Conference Name: IEEE Transactions on Power Electronics. (Cited in page 108.)

[165] G. Petrone, G. Spagnuolo, and M. Vitelli, "A Multivariable Perturb-and-Observe Maximum Power Point Tracking Technique Applied to a Single-Stage Photovoltaic Inverter," *IEEE Transactions on Industrial Electronics*, vol. 58, pp. 76–84, Jan. 2011. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in page 108.)

[166] Q. Mei, M. Shan, L. Liu, and J. M. Guerrero, "A Novel Improved Variable Step-Size Incremental-Resistance MPPT Method for PV Systems," *IEEE Transactions on Industrial Electronics*, vol. 58, pp. 2427–2434, June 2011. Conference Name: IEEE Transactions on Industrial Electronics. (Cited in page 108.)

[167] Y.-H. Ji, D.-Y. Jung, J.-G. Kim, J.-H. Kim, T.-W. Lee, and C.-Y. Won, "A Real Maximum Power Point Tracking Method for Mismatching Compensation in PV Array Under Partially Shaded Conditions," *IEEE Transactions on Power Electronics*, vol. 26, pp. 1001–1009, Apr. 2011. Conference Name: IEEE Transactions on Power Electronics. (Cited in page 108.)

[168] A. K. Abdelsalam, A. M. Massoud, S. Ahmed, and P. N. Enjeti, "High-Performance Adaptive Perturb and Observe MPPT Technique for Photovoltaic-Based Microgrids," *IEEE Transactions on Power Electronics*, vol. 26, pp. 1010–1021, Apr. 2011. Conference Name: IEEE Transactions on Power Electronics. (Cited in page 108.)

[169] H. Taheri, Z. Salam, K. Ishaque, and Syafaruddin, "A novel Maximum Power Point tracking control of photovoltaic system under partial and rapidly fluctuating shadow conditions using Differential Evolution," in *2010 IEEE Symposium on Industrial Electronics and Applications (ISIEA)*, pp. 82–87, Oct. 2010. (Cited in page 108.)

[170] S. Roy Chowdhury and H. Saha, "Maximum Power Point Tracking of Partially Shaded Solar Photovoltaic Arrays," *Solar Energy Materials and Solar Cells - SOLAR ENERG MATER SOLAR CELLS*, vol. 94, pp. 1441–1447, Sept. 2010. (Cited in page 108.)

[171] P. Q. Dzung, L. D. Khoa, H. H. Lee, L. M. Phuong, and N. T. D. Vu, "The new MPPT algorithm using ANN-based PV," in *International Forum on Strategic Technology 2010*, pp. 402–407, Oct. 2010. (Cited in page 108.)

[172] R. F. Coelho, F. M. Concer, and D. C. Martins, "A MPPT approach based on temperature measurements applied in PV systems," in *2010 IEEE International Conference on Sustainable Energy Technologies (ICSET)*, pp. 1–6, Dec. 2010. ISSN: 2165-4395. (Cited in page 108.)

[173] A. Pandey, N. Dasgupta, and A. K. Mukerjee, "A Simple Single-Sensor MPPT Solution," *IEEE Transactions on Power Electronics*, vol. 22, pp. 698–700, Mar. 2007. Conference Name: IEEE Transactions on Power Electronics. (Cited in page 108.)

[174] N. Khaehintung, T. Wiangtong, and P. Sirisuk, "FPGA Implementation of MPPT Using Variable Step-Size P&O Algorithm for PV Applications," in *2006 International Symposium on Communications and Information Technologies*, pp. 212–215, Oct. 2006. (Cited in page 108.)

[175] N. Femia, G. Petrone, G. Spagnuolo, and M. Vitelli, "Optimization of perturb and observe maximum power point tracking method," *IEEE Transactions on Power Electronics*, vol. 20, pp. 963–973, July 2005. Conference Name: IEEE Transactions on Power Electronics. (Cited in pages 108 and 109.)

[176] B. Ho, H. Chung, and W. Lo, "Use of system oscillation to locate the MPP of PV panels," *IEEE Power Electronics Letters*, vol. 2, pp. 1–5, Mar. 2004. Conference Name: IEEE Power Electronics Letters. (Cited in page 108.)

[177] C. Hua and J. Lin, "An on-line MPPT algorithm for rapidly changing illuminations of solar arrays," *Renewable Energy*, vol. 28, pp. 1129–1142, June 2003. (Cited in page 108.)

[178] E. Koutroulis, K. Kalaitzakis, and N. Voulgaris, "Development of a microcontroller-based, photovoltaic maximum power point tracking control system," *IEEE Transactions on Power Electronics*, vol. 16, pp. 46–54, Jan. 2001. Conference Name: IEEE Transactions on Power Electronics. (Cited in page 108.)

[179] L. Zhang, A. Al-Amoudi, and Y. Bai, "Real-time maximum power point tracking for grid-connected photovoltaic systems," in *2000 Eighth International Conference on Power Electronics and Variable Speed Drives (IEE Conf. Publ. No. 475)*, pp. 124–129, Sept. 2000. (Cited in page 108.)

[180] E. Standards, "EN 50530." (Cited in page 112.)

[181] Z. Erdem, "A Review of MPPT Algorithms for Partial Shading Conditions," *Acta Physica Polonica A*, vol. 132, pp. 1128–1133, Sept. 2017. (Cited in page 113.)

[182] A. Mohapatra, B. Nayak, P. Das, and K. B. Mohanty, "A review on MPPT techniques of PV system under partial shading condition," *Renewable and Sustainable Energy Reviews*, vol. 80, pp. 854–867, Dec. 2017. (Cited in page 113.)

[183] K. B. K. Cao and V. Boitier, "Probability distribution of GMPP under different temperatures and irradiation conditions," in *Journées Nationales sur la Récupération et le Stockage de l'Energie 2021*, (Grenoble (Online), France), June 2021. (Cited in page 132.)

[184] K. B. K. Cao and V. Boitier, "Probability distribution of GMPP under different irradiation and temperature conditions for GMPP tracking algorithm," in *PowerMEMS 2021*, (Exeter (on line), United Kingdom), Dec. 2021. (Cited in page 132.)

[185] K. B. K. Cao and V. Boitier, "Simple and low-cost horizon estimation method for solar harvesting systems," June 2023. (Cited in page 132.)

Abstract: This thesis focuses on the resource estimation and optimisation of autonomous power supplies ranging from 10 to 100W, powered by photovoltaic (PV) panels in complex environments (such as forests, bicycles, drones, etc.)

The first chapter of this thesis deals with the estimation of available solar energy for stationary systems. Estimation is straightforward under simple shadows (such as those cast by buildings or utility poles). However, for complex shading scenarios, shadow forecast becomes more challenging and requires a detailed understanding of obstructions that cannot be easily modelled with simple shapes. For such cases, a study was conducted using an inexpensive fisheye lens attached to a smartphone. A photo taken toward the sky captures the obstructions at the intended site. A Python script processes the photo, infers the shading information from the solar position in the photo to compensate for the meteorological data from the site, and generates an hourly irradiation estimation over the desired period. Experimental measurements validated the approach and demonstrated its use case. These irradiation estimates are then used to model an autonomous system (solar panel, battery, loads, charger), providing insight into the evolution of the minimum state of charge of the battery, which is a clear indicator of its capacity to ensure continuous operation.

To achieve the desired voltage level, solar cells are connected in series. Under uniform irradiance, the power-voltage characteristics of the PV array have a single maximum. However, under partial shading conditions, multiple local maxima emerge. The second chapter addresses the distribution of these maxima' positions on the full voltage range of the array under different irradiance and temperature conditions. To achieve this, the PV module and string are modelled and optimised with the help of the Lambert function, as well as other optimisation techniques. This helps to simulate its characteristics under a multitude of irradiance and temperature conditions in a reasonable time frame. Analysing the results, assuming equiprobability of possible conditions and based on experimental data from a mobile system, yields the distribution of the PV panel's global maximum power point (GMPP). In Simulink, a comprehensive model of a simple autonomous system is established, comprising a solar panel with four bypass diodes associated with a buck converter to recharge a battery. The microcontroller measurement inputs and the buck converter's control are also modelled. This model was validated to have reasonable accuracy with experimental measurements, and it was used to quickly simulate shading management algorithms in the next section.

The third chapter of this thesis delves into the real-time optimisation of solar power under partial shading conditions, particularly focussing on software-based global maximum power point (GMPPT) algorithms. A thorough literature review and critical examination of these GMPPT controls were performed. Seeing a gap of validation methodology for the research context at hand, a new approach is proposed to simulate the varying irradiance perceived by each solar panel over time under

moving shadows. A simple and lightweight algorithm is suggested and inspired by the distribution observed in the previous chapter. A comparative experimental analysis is conducted using various shading profiles, implementing this algorithm as well as some other algorithms with similar complexity on an actual converter connected to a computer-controlled physical solar simulator. The results highlight the merits and limitations of the proposed control approach and, more broadly, the GMPPT algorithms.

Keywords: photovoltaic, shading forecast, photovoltaic modelling, global maximum power point tracking (GMPPT)