SOLAR PREDICTOR

By
Andrei-Cristian ŞTEFAN

GRADUATION REPORT

Submitted to
Hanze University of Applied Science Groningen

in partial fulfilment of the requirements
for the degree of

Fulltime Honours Bachelor Advanced Sensor Applications

2018

ABSTRACT

SOLAR PREDICTOR

by

ANDREI-CRISTIAN ȘTEFAN

A detailed solar prediction is necessary in case of particular cloudiness-patterns causing undesirable large fluctuations in photovoltaic electricity production. By considering the project objectives and expected outcomes, as well as the fact that the motion of the clouds is the key element that influences solar irradiance, the project focused on predicting cloud motion at an intra-hour temporal resolution. This report describes possible solutions to develop a Solar Predictor system in order to find a suitable method for anticipating the times when solar irradiance is at its peak depending on the motion of the clouds. Furthermore, it provides details about the fact that Camera Detection technology in combination with background reduction and motion tracking algorithms is the most suitable prediction method, as well as it gives indications regarding its development and application in a real-life situation. The testing in a real environment allowed the implemented system to achieve predictions for time intervals up to one minute in advance and demonstrated that cloud motion and solar irradiance can be predicted at an intra-hour temporal resolution for time intervals up to one minute, even when lower resources and significantly less time for acquiring data and validation are allocated.

DECLARATION

I hereby certify that this report constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another.

I declare that the report describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Andrei-Cristian ŞTEFAN, 22nd of June 2018

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my graduation supervisor, Ms. Corina Vogt, Lecturer at Hanze University of Applied Sciences and Mr. Gerard Nanninga from Energy Academy Europe for their permanent support and guidance throughout the whole duration of the graduation project.

Furthermore, I would like to thank my mentor, Mr. Bryan Williams, as well as Dr. Felipe Nascimento Martins and Mr. Ronald van Elburg from Hanze University of Applied Sciences for their professional assistance in clarifying my questions in different phases of the project development.

I would also like to express my gratitude to Ms. Lies Oldenhof for giving me the opportunity to do the graduation project at Energy Academy Europe. Special thanks go to Mr. Jip Kosse for his availability to discuss and share ideas with regards to the project development.

Last but not least, I would like thank my parents for their permanent support and encouragement not only during the graduation project but for the whole duration of my bachelor studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1 – Rationale

Solar energy possesses the highest potential for electricity generation, in comparison to the other renewable energy sources (ex. wind, biomass), as an average amount of approximately 350 W/m^2 intensity reaches the Earth; from this amount, about 70% is available for harvesting [1]–[3]. During the day, the power demand grows as more energy is required by industry and business areas, making the power plants produce more electricity for the grid [4]. During the night less power is demanded, as there is less industrial activity and less energy is required by residences [4]. As a result, the cost of production rises with peak demand. Due to the fact that the peak load determines the size and complexity of a system, an important issue is related to finding ways to overcome the peak demands [4]; it can be done through the analysis of Photovoltaic (PV) outputs, by integrating and connecting it into the power grid, lowering in this way the overall cost [5].

Photovoltaic solar systems produce most of their power during summer, less during fall and spring and very little during winter. Furthermore, they do not produce at night and have a lower efficiency during the days when the sun is covered by clouds, thus requiring power compensations purchases from the grid [2], [3]. This leads to important challenges to grid-connected PV systems, such as the rapid output variations that occur as clouds pass overhead.

Modern systems for electrical network management named Cyber-Physical Systems – including the power grid – are extremely intertwined with each other, therefore one problem can cause cascading effects for many other connected systems, being practically impossible to keep defects isolated when they occur [6]–[8]. The Cyber-Physical System for optimal operation of the power system is called Smart Grid and it is an important subject of research in universities and industry [6], [7], [9], [10]. The Smart Grid technology allows a two-way communication between producers and consumers for exchange of electricity and information, aiming to control the operation of a huge number of sensors and computers interconnected in complex networks as well as to reroute electricity in order to compensate for defects or power fluctuations [4], [9]–[11].

The variability is a real challenge for grid operators who must be prepared to compensate when PV output drops and to reduce grid support as PV output recovers after clouds have passed, causing daily variations in PV output [12]. Generally speaking, the photovoltaic (PV) power output for a full day can be represented as a bell shape [13]. *Figure 1* shows the qualitative PV generation variability based on data extracted from the real power output of solar panels during a summer day in the State of Texas, U.S [13]. However, the contribution of solar energy to the grid during periods of peak demand is significant since it reduces demand to the grid through the addition of clean solar energy, thus helping to reduce electricity costs and increase the reliability of the energy network [14]. If there is no future information on when the grid will be used for power, power plants need to work uninterrupted to produce energy, so that there will be no shortage when demanded by consumers. In order to reduce the cost of running power plants during the time when solar power is available, a reliable generator switching system is required [15].

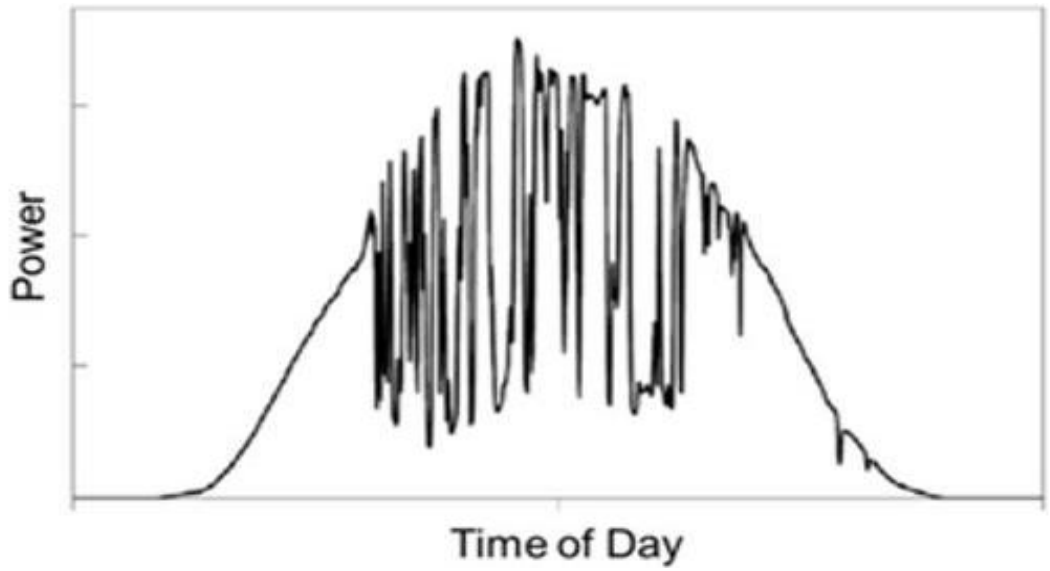


Figure 1: Variations in sunlight captured by solar panels, creating high and low peaks (Image source: [13]*)*

By integrating solar energy to the grid, the process known as peak shaving can occur where the amount of energy purchased from the energy providers during peak demand hours is reduced (*Figure 2*). This shifts the overall energy demand from midday to late in the evening [16].

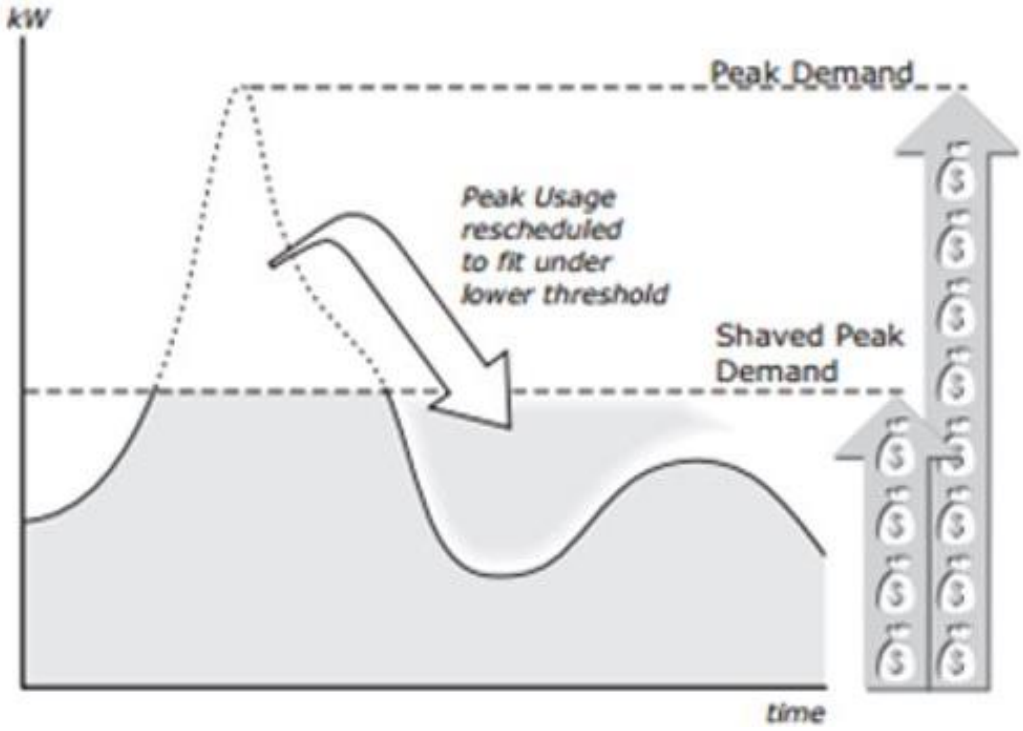


Figure 2: Qualitative Peak Shaving concept for one day (Image source: [17]*)*

There are a lot of ways to implement peak shaving, such as reducing consumption by turning off non-essential equipment during peak hours, as well as to install solar and battery solutions as soon as the peak demand occurs during peak PV output. Even though there are different solutions available, peak shaving methods require a lot of details, coordination and planning, including engineering support, utility company participation and energy producers [18]–[20]. Peak shaving can be a great option to reduce expenses and, as utility expenses and demand rises, it will become a more common way to reduce energy costs, leading to a lower peak demand and a cost price reduction [16].

Increasing energy efficiency and accelerating renewable energy production represents one of the top priorities for people and organisations around the world [21]–[24]. In order to achieve this goal, the implementation of Smart Grid systems plays an important role as they do not necessarily involve the replacement of the existing network, but it combines hardware and software elements to significantly improve the way the current system is operating while also offering the possibility of further upgrading [25]–[28].

Smart Grids can provide electricity using digital technology and can also integrate renewable energy giving the possibility to the consumers to reduce their consumption during peak hours by adapting the amount taken from the network to personal needs [21], [25]. Therefore, Smart Grid technology can revolutionize the industry by lowering power consumption by up to 30%, which also reduces the need to build new power plants [11].

As the fossil fuels are harmful to the environment by polluting not only the air, but also the soil, water, vegetation and buildings, renewable energy sources such as solar and wind energy are used more and more nowadays as they are environmentally friendly in comparison with the conventional energy sources [29]. However, because renewable energy sources are intermittent, Smart Grids are essential due to their flexibility, compatibility with the existing infrastructure, as well as safety and high efficiency [11].

In addition to the management of the grid to prevent power shortages, several technical issues – such as eliminating the solar energy fluctuations, linking consumption habits with the ability to collect renewable energy, setting the optimal price at which the energy gathered by an individual producer is sold to the Smart Grid – are required to be resolved to make Smart Grids more appealing and cost-effective [25], [30].

Fluctuations in the energy output are registered at the time when clouds pass over the solar panels, making them unable to deliver the required energy for running the desired appliances, thus having to rely on the grid to meet the energy demands [31]. Therefore, it is important to know when peak power generation occurs, so that the system can draw power directly from the cells using less electricity from the grid and reducing the overall cost of energy.

In this respect, PV technology in combination with Smart Grids encourage consumers to reduce their overall power consumption during peak time slots in order to minimise the costs of their electricity bill [30]. Moreover, Smart Grids can be efficiently used by coordinating the appliances used by each household, as well as managing the peak loads [32].

As an example of how a Smart Photovoltaic system can be used to manage domestic appliances depending on the peak solar irradiance, in 2016 *S. Rauf et al.* [30] published a scientific article in which an electrical load management system was proposed. In the mentioned article, three main electrical loads were identified: *Basic Load*, *Regular Load* and *Burst Load* (*Figure 3*). The *Basic Load* is represented by electrical devices that consume a low amount of electricity (ex. lights, fans), the *Regular Load* by the appliances that are always on (ex. refrigerator) and the *Burst Load* by those appliances with considerable energy consumption used only for a short period of time (ex. vacuum cleaner, washing machine).

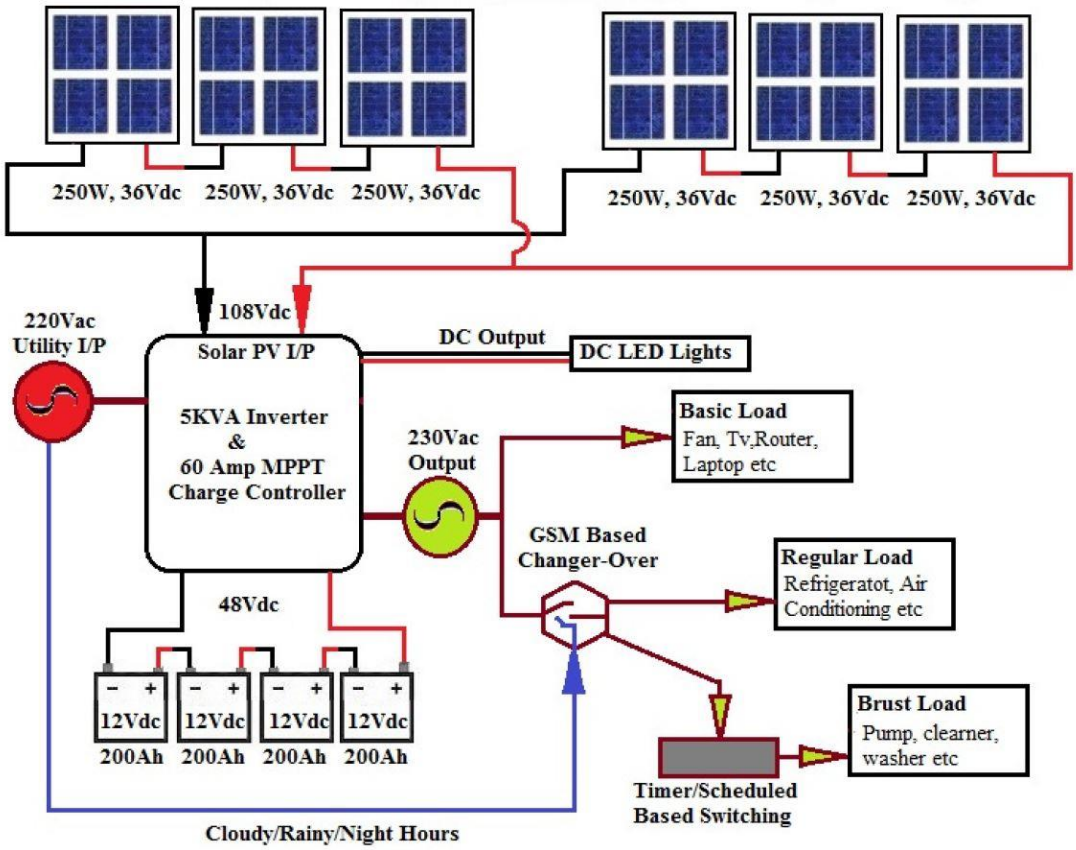


Figure 3: Smart PV system for a domestic user (Image source: [30]*)*

By predicting the peak irradiance, the appliances classified as part of the *Burst Load* can be scheduled in the periods when the solar panels produce the highest amount of electricity, thus reducing their operation cost [33]–[35]. Furthermore, this can be extended to the *Basic Load* and the *Regular Load* by including energy storage system to compensate for the variations in solar output [30].

The Solar Predictor offers the possibility to determine the peak solar irradiance (when solar panels produce the most electricity) before it happens – based on the motion, position and size of the clouds – encouraging consumers to reduce their overall power consumption during peak time intervals, thus minimising electricity costs [36]. Moreover, as the Solar Predictor can be used to schedule the use of appliances in each household, as well as to manage the peak loads, it grants the possibility to facilitate self-production and lower energy consumption.

Because of the limited information regarding the altitude of the clouds, satellite images offer insufficient spatial-temporal resolution for specific sites, not suitable for predictions at time intervals lower than one hour [37]–[39]. In this respect, in order to accurately schedule energy consumption, the need for a ground-based system able to predict intra-hour peak intervals arose.

A detailed solar prediction is necessary in case of particular cloudiness-patterns: e.g. blue sky combined with scattered clouds, causing undesirable large fluctuations in photovoltaic electricity production and/or opportunities for direct consumption of solar produced electricity.

Based on the client's targeted results, the main aim of the project was to perform a feasibility study based on the research of available scientific literature for the state-of-the-art solar prediction technology and to investigate the possibility to create a proof of concept for the prediction of solar irradiance at a high temporal resolution (5 minutes or less).

Due to the fact that the Solar Predictor is mainly meant for domestic applications, the device is to be placed on the roof of a house or building where solar panels are installed. This means that the field of view can vary from location to location depending on the surrounding objects such as trees or taller buildings. For the purpose of this project, the field of view is considered as approximately 5 km^2.

Regarding the accuracy, it was desired to achieve a prediction with an average margin of error of 5 seconds. However, this value depends on the wind speed as the lower the wind speed, the lower the cloud velocity is, making the allowed margin of error higher. This means that if a cloud moves at a slow pace, the margin of error can be higher as it will take longer for that particular loud to reach the sun and cover it completely so that the solar production is cut from the PV panels.

One of the main constrains of the project was finding a low-cost solution. In this respect, even the maximum allocated budget was of €1000, it was recommended by the client to keep the expenses as low as possible. Other important constrains were related to finding a reliable and easy-to-implement solution. This means that – besides the installation of the hardware and turning on the software, as well as the periodical maintenance check-ups – the system should not require any human supervision.

By considering the project objectives and its expected outcomes, as well as the fact that the motion of the clouds is the key element that influences solar irradiance [40], the following research question was derived:

"How can cloud motion and solar irradiance be predicted at an intra-hour temporal resolution?"

This led to several sub questions that needed to be taken into account as well:

- *"What is the state-of-the-art of the solar prediction technology?"*
- *"What is the most suitable and less resource-intensive method for an accurate prediction of solar irradiance?"*
- *"How can the system be made as accurate as possible?"*

This report describes possible solutions to develop a Solar Predictor system in order to find a suitable method for anticipating the times when solar irradiance is at its peak. Furthermore, this report provides details about the most suitable method based on literature research and project requirements, as well as gives indications regarding its development and application in a real-life situation.

Chapter 2 – Situational & Theoretical analysis

This chapter includes the research on the peak solar irradiance prediction topic, as well as the general description and analysis of the possible solutions based on the project requirements that led to the hypothesis that Camera Detection technology is the best approach for predicting solar irradiance at an intra-hour temporal resolution.

2.1. Pyranometer

Pyranometers are instruments that measure solar irradiance on a flat surface [41]. A pyranometer system is considered the most accurate way to measure the solar irradiation as it outputs readings for the flux density of the solar radiation [42]. They offer a hemispherical view on the surroundings and can be used to measure the total solar radiation on a given surface [43]. It consists of a flat sensor enclosed in a double hemispherical glass dome with high light transmission capacities, aimed to reduce errors (*Figure 4*).

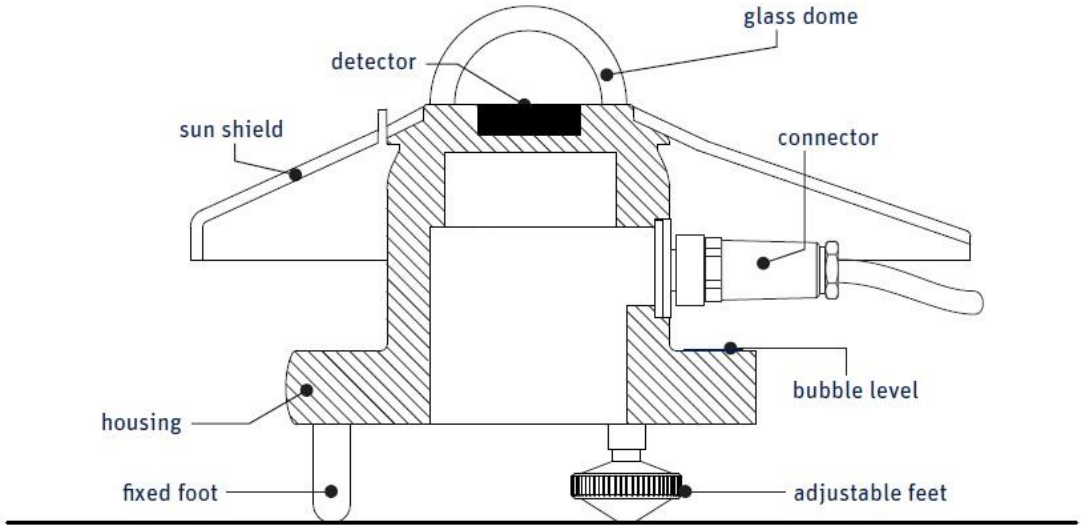


Figure 4: Component parts of a pyranometer (Image source: [44]*)*

The solar radiation (a broad range of wavelengths) that reaches the photo-sensitive cell is converted into a measurable current by the device. For most pyranometers, no power is required as the current is being generated only under illumination [41].

These devices can be classified into two major categories: Thermopile and Photovoltaic pyranometers. The first uses thermocouples to generate electricity depending on the temperature it reaches when illuminated, while the second is based on light-sensitive semiconductor chips, its working principle being similar with the one for solar cells. Between the two, the Photovoltaic pyranometer has a smaller band wavelength than the Thermopile pyranometer and it is therefore not suitable for precise measurements as it provides approximate readings [43].

A study performed in 2012 by *Chow et al.* [20] proposed a setup made out of two pyranometers for the prediction of intra-hour irradiance. Both pyranometers were placed in two locations on a horizontal plane facing the sky (*Figure 5*). Even though their relative distance from each other is not mentioned, the two pyranometers were separated by four solar panels.

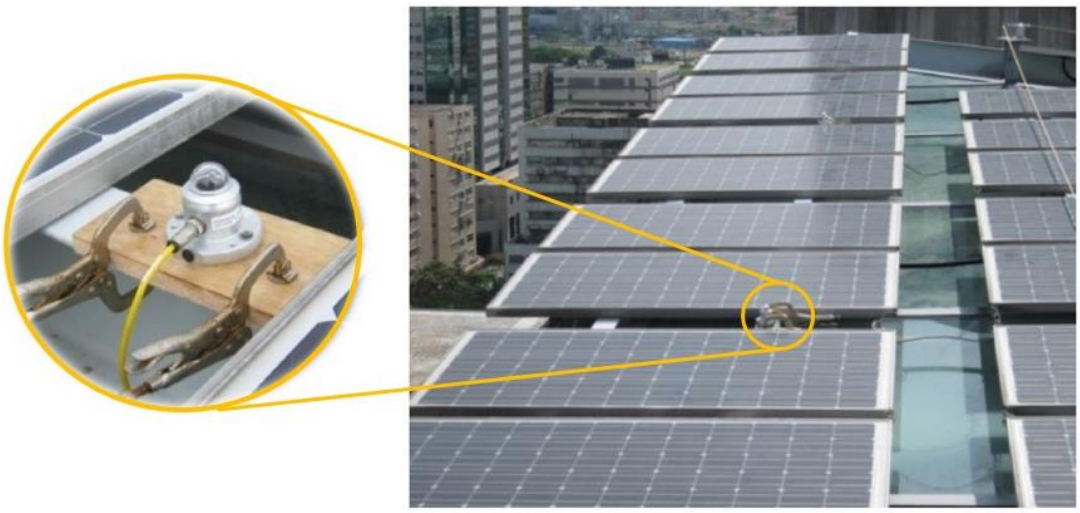


Figure 5: Positioning of the pyranometer for data acquisition (Image source: [20]*)*

In order to achieve irradiance predictions, the two pyranometers gathered data on the solar elevation angle and its azimuth angle, incident solar radiation on the sensor. Moreover, ambient temperature data was gathered as well. In order to analyse this set of data, an Artificial Neural Network (ANN) was created to compensate for the errors of the intra-hour predictions by using also historical data [20], [45], [46]. Using the two-pyranometer system, *Chow et al.* managed to achieve irradiance predictions of 20 minutes, with a recorder error of approximately 6.4% [20].

In 2014, *J.C. Baltazar et al.* [47] suggests that, in order to predict the accurate light intensity without the use of tracking devices, a multi-pyranometer array is required. The proposed design involved a four-pyranometer setup aimed in different directions: one on the horizontal plane and the rest set at azimuth angles of -60°, 0° and 60°, as shown in *Figure 6*. Furthermore, because of the fact that pyranometer are static sensors, they were mounted on a solar tracker device in order to allow the correction of the angle [47].

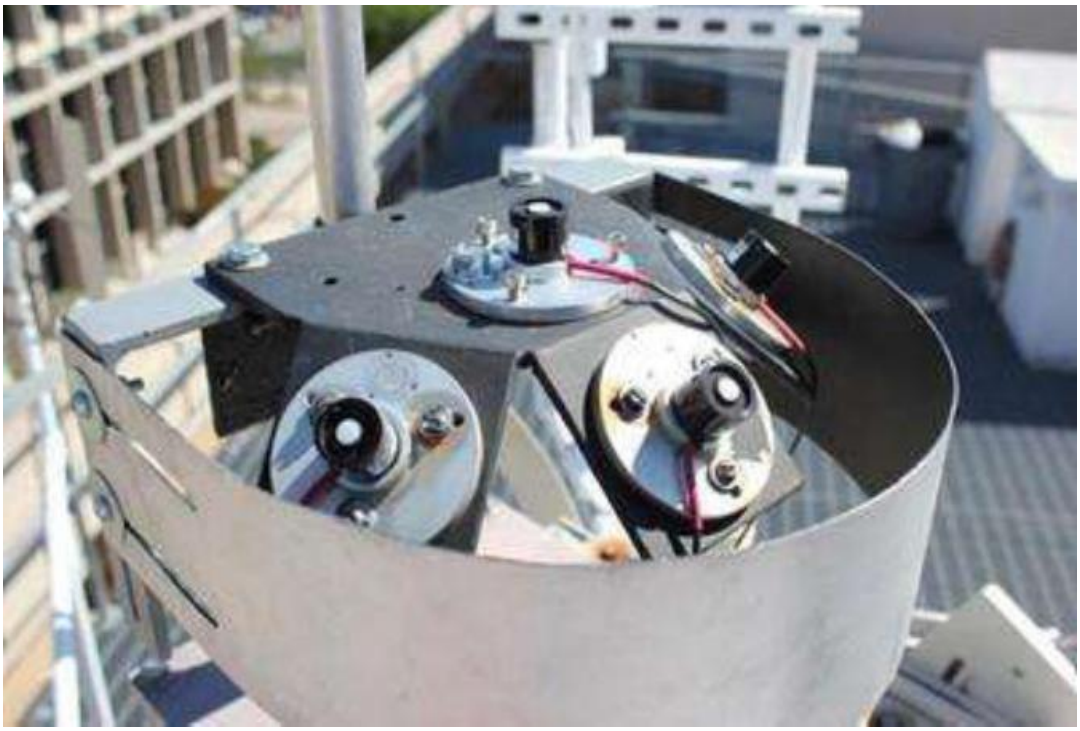


Figure 6: Pyranometer array setup (Image source: [47]*)*

The use of multiple pyranometers allows readings to be taken simultaneously in order to reduce the statistical variance of the measurements. Also, through algorithms used for estimating the irradiance on a tilted plane based on the global and diffused irradiance [48], the overall irradiance errors were reduced to ±10 W/m^2 which was considered acceptable [49]–[51].

Another study performed by *V. Srikrishnan et al.* [42] in 2015 also used a multiple pyranometers to predict the solar irradiance at intervals lower than an hour, using an approach based on neural networks, each sensor being considered as a node. Furthermore, as two pyranometer systems were tested, one with three and the other with five pyranometers, a comparison between the results obtained by the two systems was also made. For the two systems, the sensors were positioned in different configurations, similarly with the setup from *Figure 6*: the first configuration used five pyranometers where besides the one on the horizontal plane, the azimuth angle pyranometers were set 90° apart from each other, each facing a cardinal point. The second configuration uses only three pyranometers with the two azimuth ones facing South and West. Among their findings, was the fact that the five-pyranometer system showed an increase in the registered accuracy with approximately 2.5% in comparison with the three-pyranometer system [42]. This study concluded that, for a system with no moving parts, the results show lower errors if more pyranometers are used.

To sum up the findings from the examples above, in order to achieve solar irradiance predictions with low errors, one pyranometer is not enough. In order to use the pyranometer sensor in the Solar Predictor, a setup of at least five pyranometers would be needed. This would ensure a complete view on the surrounding area and even though it may be able to detect the irradiance, it will not be possible to accurately predict the movement of clouds that will block the sun rays until the moment when the sun is starting to be covered.

The advantages of the pyranometer consists of the fact that most pyranometers require no power to operate [43] and that it offers precise irradiance measurements (in W/m2) [42]. However, the drawbacks outweigh the advantages as hardware cost is high [52], constant monitoring is required in order to ensure accuracy [42], and – as the pyranometer is not designed to track cloud position – an array of pyranometer would be needed in order to reduce errors [42].

2.2. Digital Camera

Digital Cameras obtain an image (the frame) when the optical system is exposed to light; the image sensors convert the incoming light into electric signals and the different sections of the image sensor became charged proportional to the light intensity [53]. The image formed on the two-dimensional image detector array is converted into pixels, a process known as sampling.

There are two types of image sensors widely used in today's digital cameras: Charge Coupled Device (CCD) and Complementary Metal Oxide Semiconductor (CMOS). Even though both work in a similar fashion by converting light into electric signals, the main difference being the fact that CCD technology move the generated charge from pixel to pixel until it is converted to voltage at the output node, while CMOS technology converts the charge to voltage inside each of the pixels (*Figure 7*) [54].

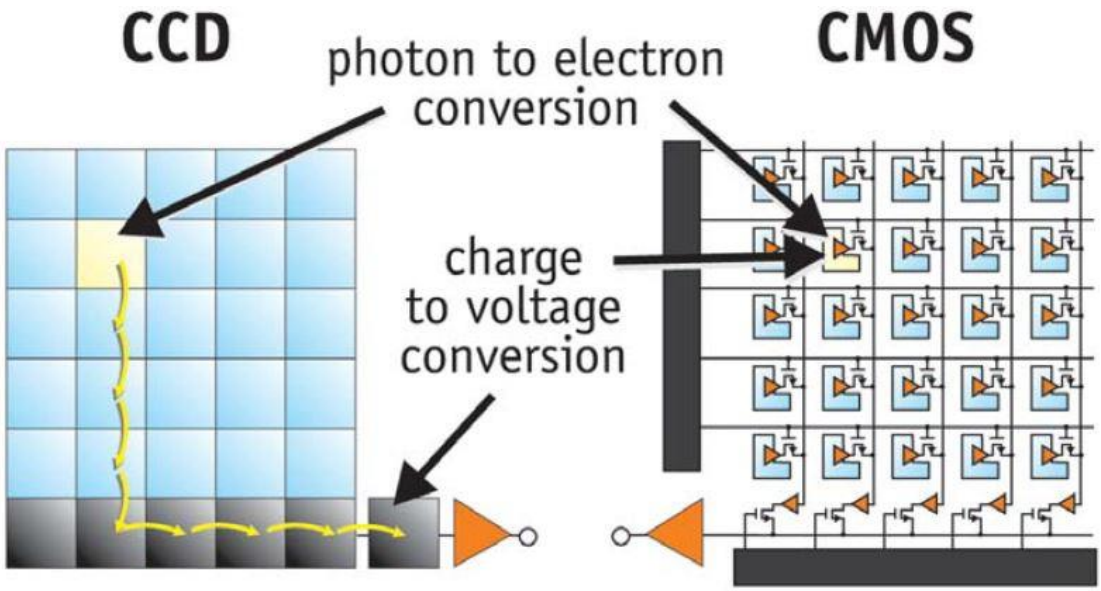


Figure 7: Difference between CCD and CMOS technologies (Image source: [54]*)*

Further comparing the two technologies, it is considered that CCD image sensors display a higher level of noise than CMOS image sensors because of the higher bandwidth used by CCD [54]. However, this difference is not noticeable for applications where high definition imagery is unnecessary.

In 2012, *Ghonima et al.* [38] developed a ground-based methodology to classify clouds by their optical thickness. Such a method can improve the accuracy of an intra-hour solar irradiance forecasting system because thicker clouds allow less light to pass through, in comparison to thinner clouds, having a noticeable effect on the solar irradiance that reaches the ground. For their proposed method, a *Total Sky Imager* device was used where a digital camera is pointed down at a spherical mirror reflecting the sky, in order to increase the radiometric resolution of the regions of interest. For the classification, methods such as the *red-to-blue ratio* [55], the *red-blue difference* [56] and the *normalised blue-red ratio* [57], [58] were tested to determine which clouds were optically thin and which were optically thick. However, the *red-to-blue ratio* method was regarded as most suitable. For a clear sky, it was noticed that the *red-to-blue ratio* has higher values around the area where the sun is located and gradually decreases with the *Sun-Pixel Angle*, the angle between the camera pixel and the direct solar beam (*Figure 8*). Furthermore, a neural network was trained with a dataset of 60 images, representative for different types of clouds. It was noted that the implemented algorithm performed better when the difference between *red-to-blue ratio* images and *clear sky* images was taken, being able to classify the cloud pixels as *optically thin* or *optically thick*, thus giving the opportunity to improve the detection accuracy of the short-term forecasting systems [38].

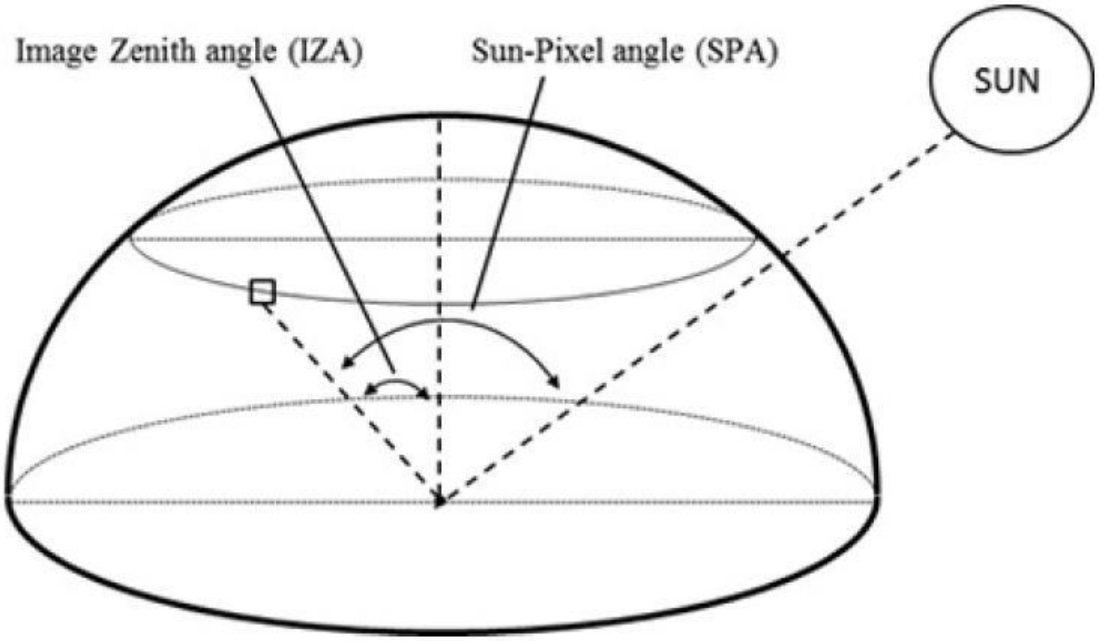


Figure 8: The determination of the Sun-Pixel angle (Image source: [38]*)*

A paper published by *Chu et al.* [59] in 2015 showed how cameras and image processing algorithms can be used to predict solar irradiance ten minutes in advance. The employed method combined a fish-eye digital camera with an artificial neural network and was divided into four parts: cloud identification, cloud indexing, cloud classification and performance assessment.

The first part, the identification method, identifies every image taken of the sky as either *clear* or *cloudy*. If the image is categorised as cloudy, a thresholding method created by *Li et al.* [57] in 2011 is used to further classify the image as either *overcast* or *partly cloudy*; the second part, the indexing of clouds, is used to obtain numerical pixel information for the clouds that move towards the sun; the third part, the classification of clouds, employs a multi-layered neural network to detect what influence will the clouds have on the solar light, based on the training data used for the network; the fourth part, the assessment of the results, was conducted by using statistical tests to evaluate the performance of the system, such as *mean bias error*, *root mean square error*, *forecasting skill* and *excess kurtosis* [60]–[62]. The research paper had positive results, succeeding in creating a real-time forecasting system with an accuracy of 65% able to predict when clouds will cover the sun for a time interval of ten minutes [59].

Another notable system based on Digital Camera technology was created by *R. Chauvin et al.* [63] in 2015. The main difference from the other systems presented above is that the approach of *R. Chauvin et al.* employs a thresholding technique on the sky images based on pixel identification performed by separating the cloud pixels from the clear sky pixels [57], [64]. This is done by calculating the optimal threshold based on light and colour [63]. Using the threshold, the background is successfully removed (*Figure 9*), allowing the implemented algorithm to detect the clouds. This type of method has a lot of potential to be implemented in the solar prediction, as it shows how – by using thresholding techniques – the background can be removed, so that clouds can be detected. With the detection of clouds, their trajectory as well as speed can be calculated, facilitating in this way the possibility to state predictions regarding the amount of time needed for the cloud to cover the sun, including the time it would take for the cloud to leave the sun's corona.

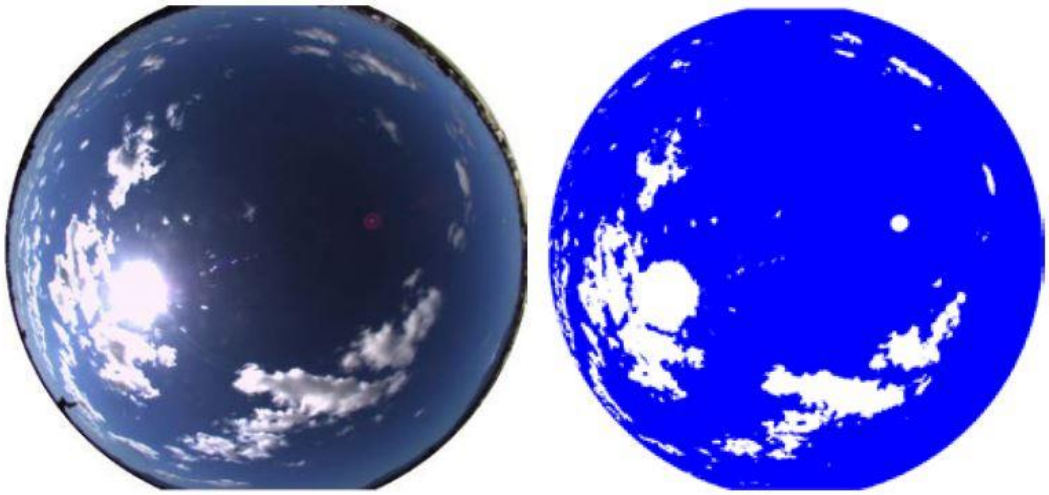


Figure 9: Background reduction algorithm applied to a sky image (Image source: [63]*)*

The systems presented above showed that using camera vision for intra-hour solar irradiance prediction offers a wide range of possibilities, and that various types of algorithms can be implemented. Digital cameras have the advantage of being able to perform detection in real-time [57] and having the flexibility of software implementation [57], [65]. However, as presented above, most camera-based systems are resource intensive [59] and there is the risk of damaging the image sensor when pointed directly at the sun [66]. In order to avoid damaging, special optical filter would be needed to reduce the intensity of the light that reaches the image sensor.

2.3. Thermographic Camera

Thermographic cameras, also known as infrared (IR) cameras, are used for the detection of infrared radiation in non-contact temperature measurements. In the electromagnetic spectrum (*Figure 10*), the IR interval ranges from 0.77 µm to 100 µm [67]. This spectrum – invisible to the human eye – can be detected by thermographic cameras as every physical body with a temperature larger than -273.15°C (absolute zero) radiates heat [68], [69].

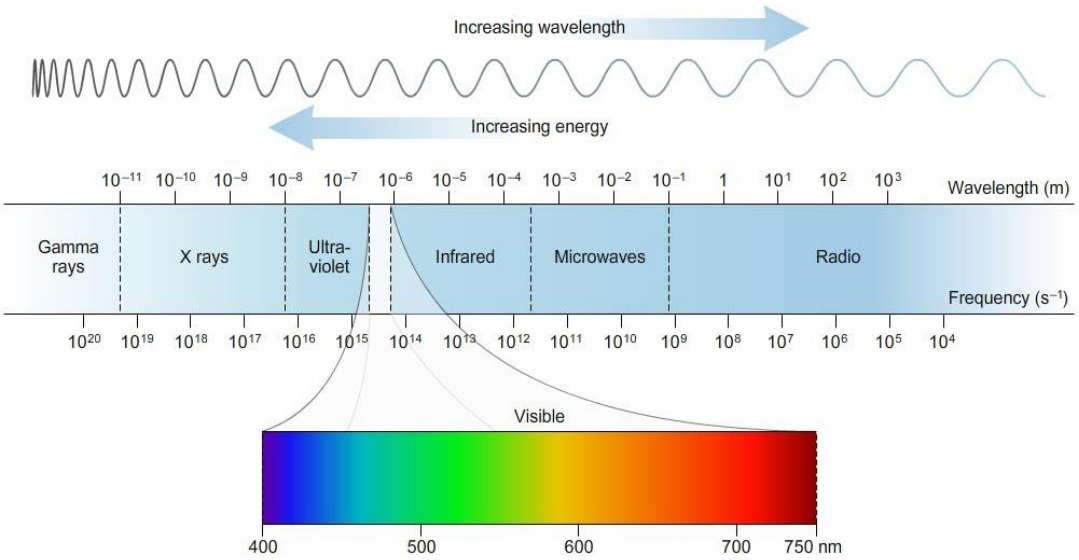


Figure 10: The Electromagnetic Spectrum (Image source: [70]*)*

As an object's temperature increases, the kinetic energy of the particles increases as well and more thermal radiation is produced [71]. This type of cameras uses the electromagnetic radiation to form an image, similar to how a normal camera uses visible light to create an image.

Because of the high spatial and temporal variability, it is a difficult task to accurately determine the radiative effects of clouds on solar irradiance [72]. This because clouds absorb a significant amount of radiation and also reflect back the radiation emitted from Earth's surface [73].

Scientific literature shows that little research has been done on using thermographic cameras for detecting clouds. This is mainly because of the high cost of a thermographic camera as well as of the fact that clouds can be detected using a "normal" digital camera. Furthermore, even if thermographic cameras could be used to detect clouds [73], its main atmospheric applications are the detection of volcanic plumes and masses emitted during eruptions [74]–[77].

However, one research paper published in 2008 by *S. Smith and R. Toumi* [78] used one such camera (ground-based) to measure cloud cover and brightness temperature and use the data to make irradiance predictions. The used camera constantly adapts the ambient temperature readings in order to detect the areas that have a colder or hotter temperature. Using the readings, a temperature threshold value was set and based on that, a Bit Mask was created with all the cloud pixels, which was then counted by an algorithm (*Figure 11*).

As mentioned above, "normal" digital cameras are preferred over thermographic cameras mainly because of the fact that "normal" cameras can be programmed to recognise clouds without the need for thermographic data. Moreover, the cost of a digital camera appropriate for cloud detection is far lower than the cost of an appropriate thermographic camera (tens to hundreds of euros for the digital camera, comparing to hundreds to thousands of euros for the thermographic camera). Therefore, the use of a thermographic camera is redundant for the creation of a Solar Predictor, a more viable approach being the use of "normal" digital cameras.

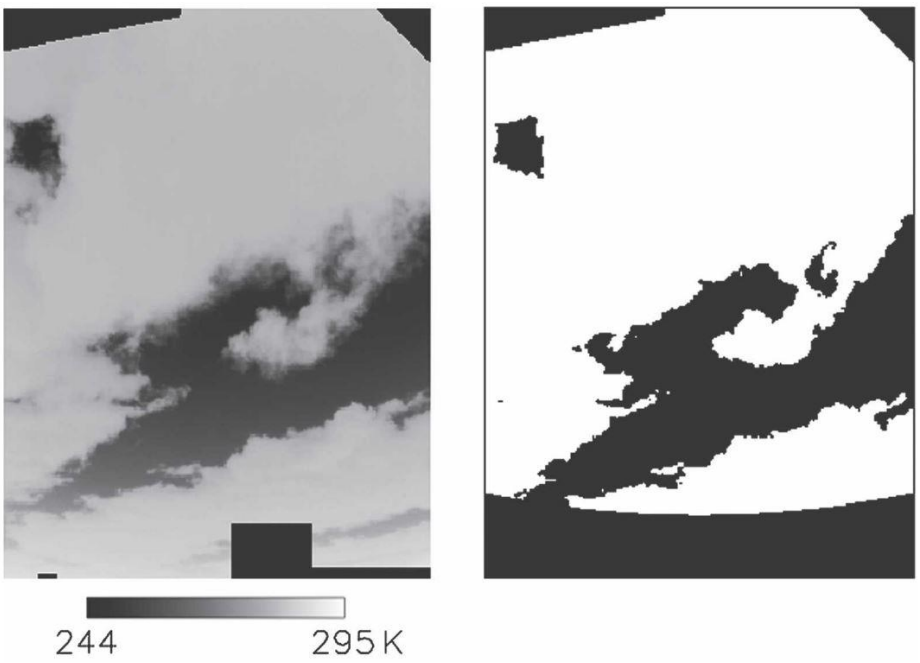


Figure 11: (Left) Image taken by thermographic camera. (Right) Detection of cloud pixels and creation of Bit Mask (Image source: [78]*)*

The paper published by *S. Smith and R. Toumi* [78] shows how a simple *background reduction* algorithm can be used to detect cloudiness. This type of algorithm can be implemented in the detection software of the Solar Predictor, without the use of an Artificial Neural Network. Despite the fact that in the mentioned article thermographic readings were used, the *background reduction* method can also be implemented for a "normal" digital camera, provided the fact that the creation of the Bit Mask is achieved by other means. For example, a colour filter can be implemented in the software to make the distinction between the sky and the clouds, based on a set threshold. However, as the accuracy of the created system is not clearly shown in the article, the Solar Predictor may require several other software methods to be implemented for a high accuracy. Even though the thermographic camera may identify clouds easier than a normal camera eliminating the background without the use of specialised software method [78], the high cost of hardware as well as the fact that the readings can be influenced by changes in the ambient temperature [79], [80], makes it not suitable for being used in the Solar Predictor.

Chapter 3 – Conceptual model

As mentioned in *Chapter 1*, finding a low-cost, easy-to-implement and reliable solution were the main constraints for this project, the research led to the conclusion that the Camera Detection technology is the most appropriate for the mentioned problem. This because, in comparison to the other technologies mentioned in *Chapter 2*, it has the advantage of being a relatively simple device that is able to offer enough flexibility for software implementation.

Table 1: Comparison overview of the methods in relation to the project requirements

	Advantages	**Drawbacks**
Pyranometer	- No power required to operate [43] - Precise solar irradiance measurements [42] - Real-time detection [42]	- High cost of hardware [52] - Not designed to track cloud position [42] - An array of pyranometers is necessary (one is not enough) [42], [47] - Constant monitoring required to ensure accuracy [42]
Digital Camera	- Low cost of hardware - Flexibility for software implementation [57], [65] - Real-time detection [57]	- Reliability dependent on the software quality [57], [65] - The amount of necessary processing power depends on the software efficiency [59] - risk of damage when pointed directly at the sun [66]
Thermographic Camera	- Background removed without specialised software method [78] - Real-time detection [78]	- High cost of hardware - Readings influenced by changes in the ambient temperature [79], [80]

Besides this type of approach, there is little information in literature regarding systems based on other technologies. The only other technology that differs from Camera Detection and that was used for similar purposes was the one based on pyranometers [42], [47]. Even though pyranometers may be able to accurately detect the solar irradiance, it will not be possible to detect the movement of clouds that will block the sun rays until the moment when the sun is covered. This means that pyranometers are not a good choice for solar prediction technologies with respect to the project requirements.

For topics involving solar irradiance prediction and cloud tracking such as the one that the Solar Predictor intends to cover, most scientific research papers use a digital cameras and employ systems based on digital cameras in combination with image processing techniques such as blob detection, colour filters, motion tracking and Artificial Neural Networks (ANN) [37], [59], [81]–[83].

Several scientific articles refer to the use of a *background reduction* algorithm to detect cloudiness and to its ability to predict the peak irradiance 5 minutes in advance [37], [78]. The advantage of implementing this type of algorithm in the detection software is that it can be done without the use of an Artificial Neural Network, thus without the need to create a complex hardware system able to perform the tasks. As an example of the amount of resources that might be needed for an ANN, the system created by *Chu et al.* in 2015 used 10 computer cores and had a duration for the training of the neural network of approximately 24h [59]. Given that the system created by *Chu et al.* in 2015, as well as similar systems, require a large amount of resources both in terms of hardware and software, in order to comply with the project requirements and budget limitations, a less resource-intensive approach is needed. In this respect, the use of *background reduction* algorithm is the most suitable. Moreover, as mentioned in *Part 2.3.*, the

Solar Predictor requires the implementation of other software methods in order to achieve a high prediction accuracy.

In conclusion, taking into account the project constrains, the information from the literature research combining existing weather/solar prediction technology, as well as the state-of-the-art technology and programming methods, Camera Detection is chosen to be used for the Solar Predictor project. With regard to the software, cloud detection algorithms such as *background reduction* and *motion tracking* can be implemented to predict the movement of clouds.

Furthermore, even though a single camera is considered accurate, it may prove difficult to exploit its full potential, as it may not be possible to point it directly at the sun without causing damage to its image sensor; this because the lens can act like a magnifying glass and focus the rays of the sun on the camera's image sensor. Therefore, a special optical filter is required to be placed on top of the camera's lens in order to protect it from damage.

Chapter 4 – Research design

This chapter presents the design approach as well the work done and the step-by-step technical process used for the Solar Predictor. In order to answer to the research question, the project work was distributed over three main phases: Data Acquisition, Data Processing and Data Prediction. The Data Acquisition phase highlights the procedure in which the hardware was created, the programming language used and the reasons why it was chosen. The Data Prediction section describes the implemented software architecture, including the filtering process and the detection algorithm. Data Prediction explains the reasoning behind the prediction algorithm as well as its gradual implementation within the software.

4.1. Data Acquisition

For the Solar Predictor, a webcam [84] facing the sky was used in a place that is not obstructed by objects such as houses or trees. During the testing, the camera was placed on the roof of Hanze University of Applied Sciences (53°00'16.1"N; 6°34'12.6"E) at a height of approximately 10m with an unobstructed sky field of view of approximately 5 km^2.

As one of the requirements for the Solar Predictor was to find a low-cost solution, the chosen materials needed to have a balance between cost and quality, the predominant factor being the quality. This is reflected in the decision to buy an optical light filter of a good quality, instead of settling for an inferior one, even though its cost was higher than the used camera (*Appendix A*).

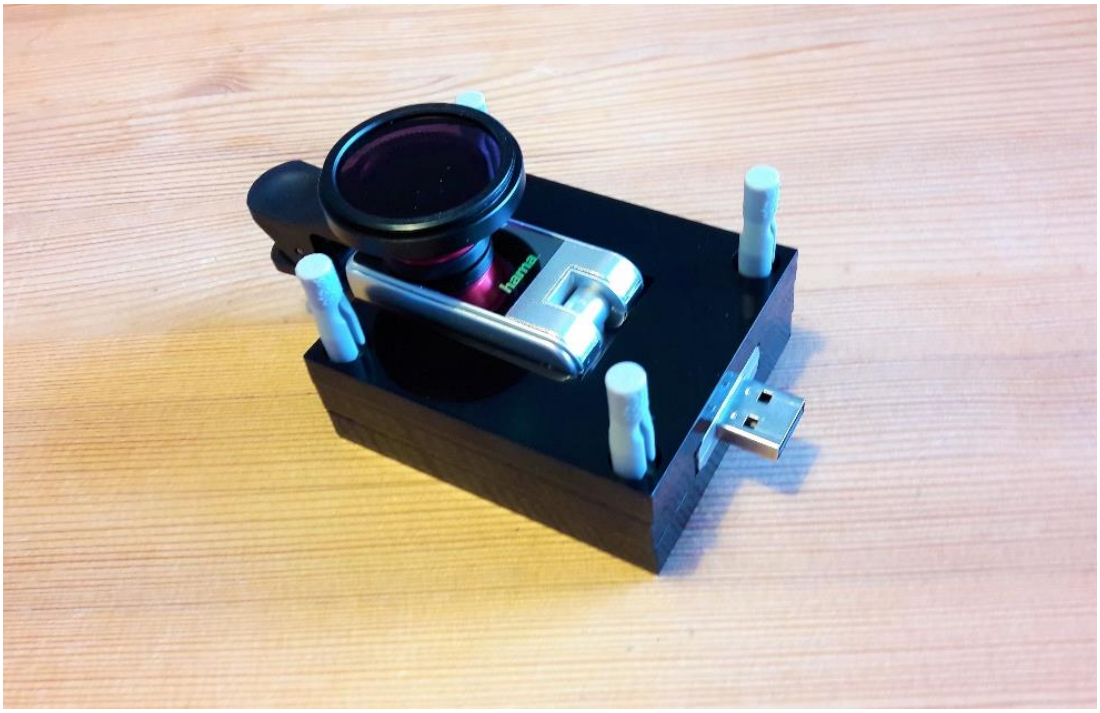


Figure 12: Prototype Design

For image acquisition, a 1.3 MP webcam was chosen. In order to enlarge its field of view, a clip-on fish-eye lens was added. To make sure that the webcam's image sensor does not get damaged by the direct sunlight, a ND-8 filter was places on top of the fish-eye lens. The ND Filter (Neutral Density Filter) is a type of optical filter appropriate for outdoor applications that reduces the intensity of light that reaches the lens without altering the natural colour [85], [86], making it possible for the camera to be pointed at the sun without causing damage to the image sensor (*Figure 12*).

The external casing of the prototype, also meant to keep the camera fixed in the same position during the testing period, is made out of three acrylic layers, each of 6 by 7 cm and a thickness of 8 cm (*Figure 13*), cut to fit the webcam's shape (*Appendix B*).

Figure 13: Layers for hardware design

Due to the fact that the acrylic sheet used is sturdy, it was chosen also for the base and top layer. The role of the first casing layer is mainly to provide a surface on which the camera can be placed. The second layer has two cut-outs, one of the size of the USB part of the webcam's body which ensures the camera does not change position in the horizontal direction and the other to make room for the clip-on fish-eye lens. The third layer has only one cut-out of the size of the lens part of the camera. Because of the camera shape, the third layer not only prevents the camera to change position horizontally, but also vertically.

In the corners of each layer, holes of a diameter of 8 mm were cut so that connectors can be used. Plastic nail anchors were chosen as connectors, because they keep a tight grip on all three layers, without allowing them to fall apart. The nail anchors used are meant for 3 mm screws, however, their outer diameter is what is important, which is of 8 mm.

The video feed camera recording is imported through USB connection to a computer which uses the *Python 3.6* programming language in combination with *Open Source Computer Vision 3.4* (*OpenCV 3.4*) libraries to analyse the data.

Python is a free open-source general-purpose programming language native to Linux that can be run on any operating system, being used for a wide variety of applications such as automation, web development and data science [87]. Its main advantages over other programming languages include the large amount of support libraries as well as the user-friendliness and implementation speed [88]. The areas of practical applications mentioned above allow the implementation of complex techniques, making *Python* a good fit for this project.

In this project, *Python 3.6* creates the link between the various components of the software infrastructure being responsible for the exchange of information between the camera and the *OpenCV 3.4* libraries. The mentioned libraries were designed to create a general infrastructure for practical computer vision applications, containing more than 2500 optimised algorithms that offer a wide variety of possibilities for software implementation such as motion tracking, filters, image segmentation and object recognition [89]. Based on these methods, the identification of the sun and clouds was achieved by using a *background reduction* algorithm in combination with *blob recognition* and *motion tracking* algorithms.

4.2. Data Processing

The data received from the camera is processed and analysed through the use of *Python 3.6* in combination with *OpenCV 3.4* libraries using a 64-bit Intel® Core™ i3-4010U CPU operating at a frequency of 1.70 GHz (*Appendix D*).

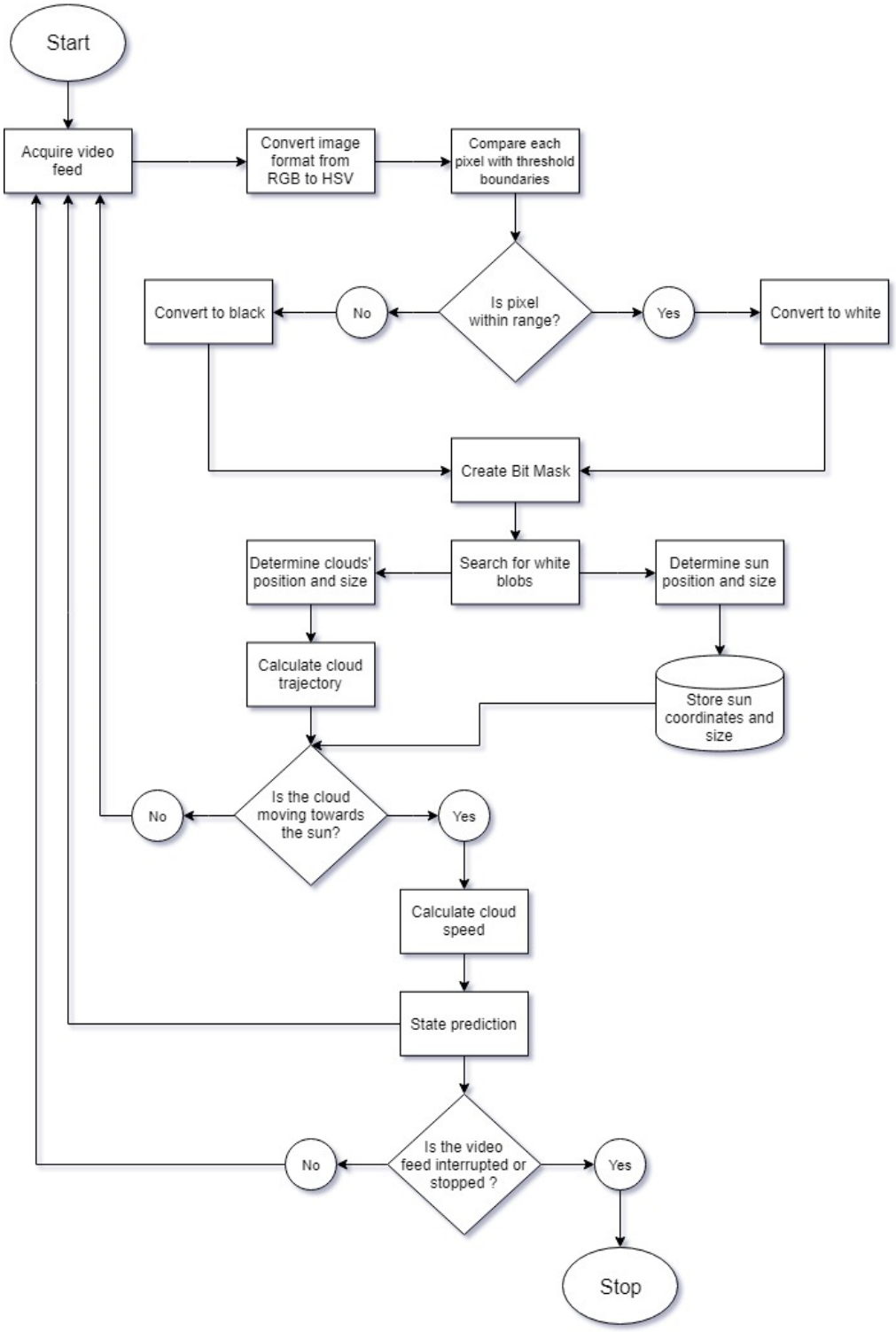


Figure 14: Solar Predictor flowchart

The video feed is processed so that a pure black and white mask from the real-time video feed is created (*Figure 14*). This is done by converting the RGB image format to the Hue-Saturation-Value (HSV) colour space – a cone-like space with its apex pointing downward (*Figure 15*) – with the Hue (H) being the dominant colour observed, the Saturation (S) the amount of white light present in the chosen Hue and the Value (V) the chromatic intensity [90].

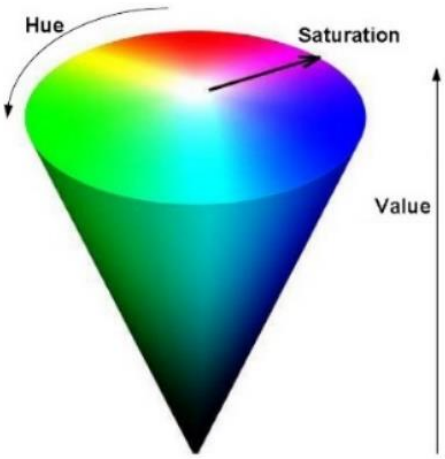


Figure 15: HSV colour space representation (Image source: [90]*)*

The reason for this conversion is that the HSV colour space allows for an easier implementation of the thresholds necessary to create the colour filters mainly because it represents the colours in a way that can be better understood by the human vision system [90]. Using this, the blue of the sky is discarded – being considered as background – leaving only the clouds and the sun which are detected as white colour of different intensities. In this stage, the difference between the sun and the clouds is made, with the sun being a white-yellow blob of high intensity and round shape and the clouds being white-grey blobs of amorphous shapes.

With the background removed, the only non-black pixels that remain in the filtered video feed are the ones from the sun and clouds. For this reason, another filter which converts the feed to a Bit Mask with all the remaining non-black pixels set to 1 (or white) and the background to 0 (or black) was implemented. In order to make it easier to distinguish between the white blob of the sun and the white blobs of the clouds, in the initial phase of the software, the sun coordinates and size are extracted and a yellow circle with the same size is drawn on the Bit Mask, at the same coordinates (*Figure 16*). Because for an intra-hour interval the motion of the sun is assessed as negligible [40], for the implementation of the software, the sun is considered stationary.

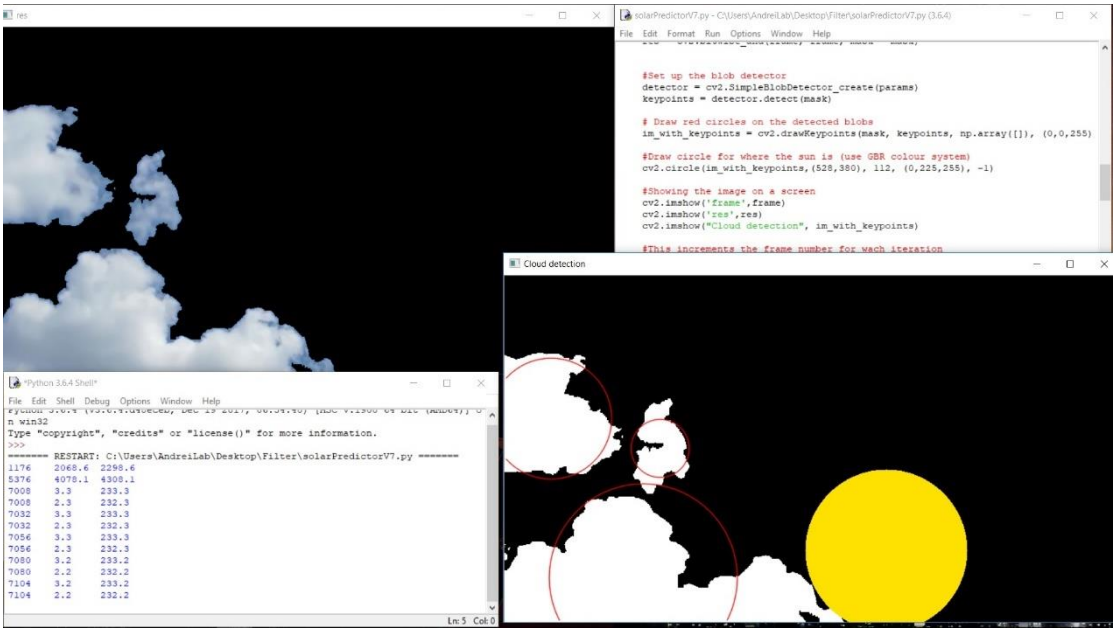


Figure 16: Software implementation with background reduction and blob detection

On the Bit Mask, blob detection algorithms are implemented for the different types of blobs to find the clouds' location and size based on colour and area. This allows the recognition part of the program to work with a pure black and white image. This part of the software is designed to detect only the clouds that can potentially cover the sun, discarding the clouds with a size lower than 1500 pixels that will not influence the solar irradiance. Clouds smaller than 1500 pixels do not influence the solar irradiance as it was determined visually that they are not large enough to cover the sun. For each of the detected blobs, their location and size are recorded so that they can be used in the prediction part of the software.

4.3. Data Prediction

In order to create the prediction algorithm, the velocity and the trajectory of each detected cloud had to be calculated. When a cloud is detected, it is recognized as a potential cover for the sun and calculation as performed within the algorithm as below:

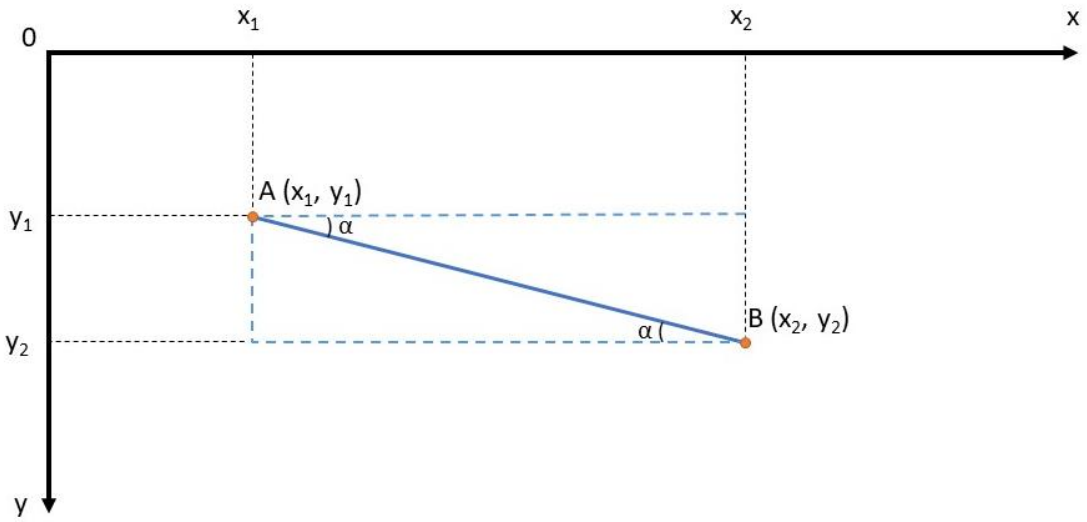


Figure 17: Representation of cloud movement from position A to B in the Python coordinate axis system

The distance from cloud-position A to cloud-position B is calculated according to Pythagorean theorem:

$$AB^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2 \Rightarrow AB = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (1)$$

Using the calculated distance (AB) and the time (t) necessary for the cloud to travel from position A to position B, the velocity is calculated using the velocity equation:

$$v = \frac{AB}{t} \qquad (2)$$

As the software uses frames as time unit, the time was converted to seconds based on the frame rate of 30 frames per second used by the camera.

From basic geometry, it is known that the alternate-interior angles formed by two parallel lines (situated in the same plane) with the transversal line that intersects both of them, are equivalent (see $\measuredangle\alpha$ in *Figure 17*). In order to calculate the trajectory, the angle made by AB with the horizontal axis was determined by applying the formula for the tangent of an angle, defined as the ratio between the length of the opposite side and the length of the adjacent side:

$$\tan(\measuredangle\alpha) = \frac{y_2 - y_1}{x_2 - x_1} \Rightarrow \measuredangle\alpha = \arctan\frac{y_2 - y_1}{x_2 - x_1} \qquad (3)$$

Using the angle at which the cloud is moving, it can be determined if its trajectory is one that will reach the known position of the sun. If the cloud is on a path that will reach the sun, its velocity and the distance from the current location of the cloud to the sun are calculated. The mentioned distance is calculated similarly with the distance calculation from *Equation 1*.

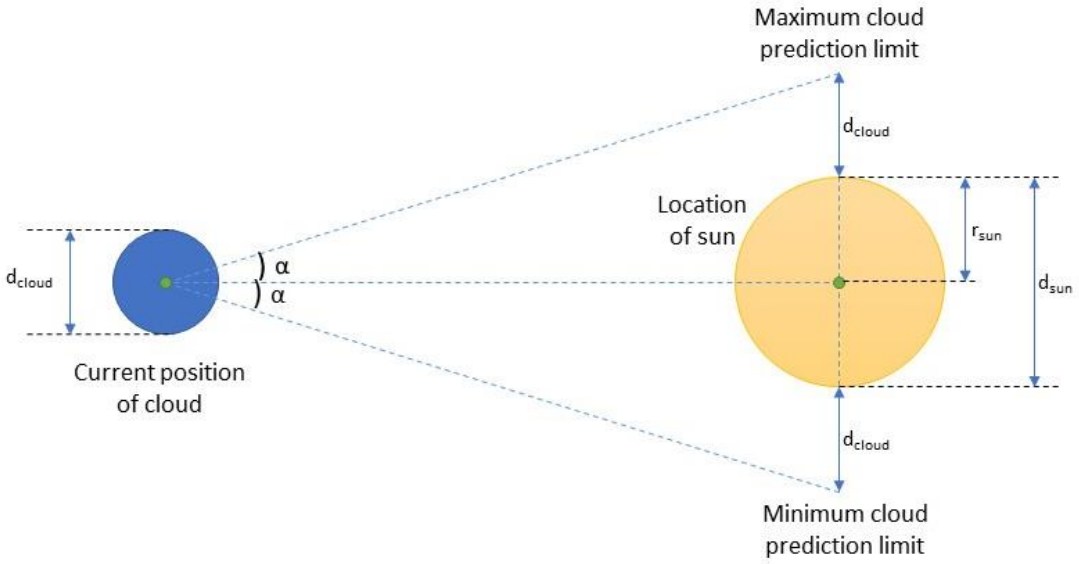


Figure 18: Schematic representation of the prediction range

By knowing the velocity and the distance, the time it takes for the detected cloud to reach the sun is calculated using *Equation 2*, which states the prediction. The maximum range for which the prediction is made is of ($2*d_{cloud} + d_{sun}$), as shown in *Figure 18*. Using the cloud speed and size, as well as the coordinates and size of the sun, it can be determined how long it will take for the mentioned cloud to exit its corona, thus making it possible to determine the time when the peak solar irradiance will return.

It is important to mention that the created Solar Predictor is most suitable for the detection and prediction of *Cumulus* clouds. This because they are a type of clouds that are normally detached from one another and separated by areas of blue sky, making it possible to detect their boundaries with a Digital Camera system [91].

Chapter 5 – Research Results

To determine the efficiency of the developed software used in the Solar Predictor to collect, process and predict the data gathered from the camera, tests were performed in a real environment. In order to do so, the webcam was placed on the roof of a building in such a way that it had a clear view on the whole sky (sun included), making sure that the field of view is not obstructed by trees or buildings.

During the testing phase (*Appendix E*), the sun was positioned in the lower half of the field of view and it was considered stationary for the duration of each test. The created software was run and through the implemented *background reduction* and *cloud motion tracking* algorithms, the background (the sky) was discarded and only the clouds remained (*Figure 19*). By checking the direction of cloud's movement, the predictions were stated.

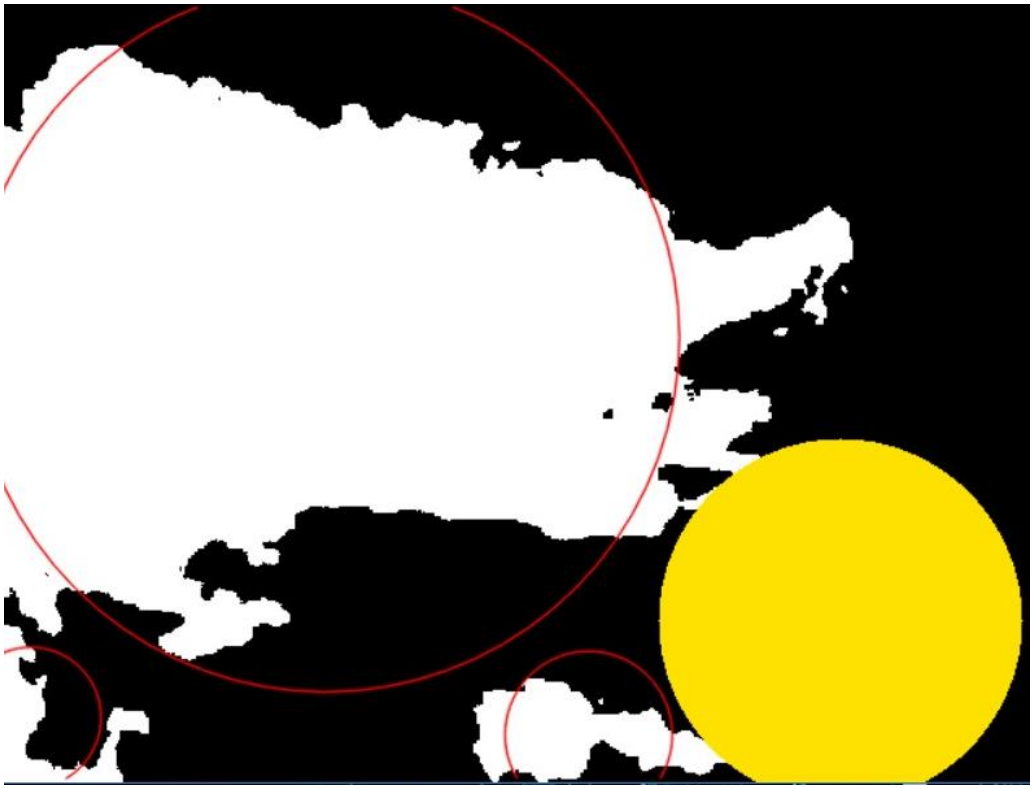


Figure 19: Blob detection

Due to the fact that – because of the emitted sunlight – the clouds' edges become less clear in the immediate vicinity of the sun, the software cannot detect the exact moment when the cloud reaches the sun. Therefore, the moment when the predicted cloud reached the sun was determined visually. In this way, the time interval from where the prediction started until it reached the sun was measured using a chronometer and it was compared to the prediction given by the software.

It is important to note that, because clouds are not objects with a constant shape, some of the samples had to be discarded as some clouds disappeared before reaching the sun, before exiting the sun's corona or split into smaller clouds. Also, there were cases where the general direction and/or the speed of the wind changed, carrying the clouds in another direction and/or at different speeds. As the collection of samples depended entirely on the weather and on the availability of clouds as well as the wind direction, four samples could be collected.

The tests allowed the implemented system to achieve predictions for time intervals up to one minute in advance. However, as the predictions are depending on the weather conditions as well as on the position of the sun, in case the clouds are moving towards the sun from a bigger distance, the system could achieve predictions for time intervals larger than one minute.

Comparing with results from scientific literature, such as the system created by *Chow et al.* in 2011 that reached a prediction time interval of maximum 5 minutes [92], a one-minute prediction time interval is

lower, but not by much. Nevertheless, it is important to highlight that *Chow et al.* did not have restrictions with regards to resources, allowing the usage of professional equipment such as a Total Sky Imager, as well as the creation of a more complex and resource-intensive programming technique. Furthermore, for *Chow et al.* the data gathering period alone was of seven months, while the entire duration of the Solar Predictor project was of four months, including research, design, software development, testing and validation.

In order to determine de variability of the collected data, statistical methods of analysis such as *Mean Bias Error* and *Root Mean Square Error* were used. Moreover, the *Paired t-Test* together with the calculation of the *Standard Error of the Statistic* were implemented for the four collected samples.

The overall systematic error of the model is given by the Mean Bias Error (MBE) as in *Equation 4*:

$$MBE = \frac{1}{n} * \sum_{i=1}^{n} \frac{x_i - x_{true}}{x_{true}} \qquad (4)$$

$where \quad n - sample\ size$

$x_i - predicted\ time$

$x_{true} - measured\ time$

As the calculated MBE is -0.5004 (-50.04%), it has a negative bias; it means that the model tends to underestimate the predictions with an average of 50%.

In order to observe the difference between the predicted time and the measured time, the Root Mean Square Error (RMSE) was calculated using *Equation 5*.

$$RMSE = \sqrt{\frac{1}{n} * \sum_{i=1}^{n} \left(\frac{x_i - x_{true}}{x_{true}}\right)^2} \qquad (5)$$

The RSME is the absolute measure of fit that determines the accuracy of the prediction [93]. Using *Equation 5*, the RMSE for the extracted samples is 0.52998 (52.998%), so the collected data fits the predictions at a rate of approximately 53%. This means that the differences between the samples of population values predicted by the Solar Predictor and the values observed, even though the different samples have different rates of error, in the root mean square they differ by about half in average.

In order to assess if there is a statistically significant effect between the predicted time and the measured time, for n samples, it is necessary to check if the mean values differ between the two data sets. Based on the configuration of the data (two data sets with a number of samples *n* lower than 30) and due to the fact that the observation was collected in pairs sets (the predicted time and the measured time), the *Paired t-Test* was used as it is the most appropriate to analyse the differences between the readings [94].

In order to perform the statistical test, the *R programming language* was used as being an open-source software for statistical analysis with thousands of available packages for various topics such as statistics, econometrics and bi-informatics with a large variety of available documentation [95], [96].

The first step of implementing the *Paired t-Test* is to define the Null Hypothesis (H_0) stating that there is a difference between the means and the Alternative Hypothesis (H_a) stating that there is no difference between the means.

The second step is the Paired t-Test statistic value is calculated using *Equation 6* [94]:

$$t_0 = \frac{\bar{d}}{\sigma / \sqrt{n}} \qquad (6)$$

$where \quad \bar{d} - mean\ of\ the\ differences$

$n - sample\ size$

$\sigma - standard\ deviation$

For a 95% confidence interval, the value of alpha refers to the significance level and is 0.05, calculated as (1 - 0.95). In case the calculated significance level, the p-value, is lower than 0.05, then reject the Null Hypothesis (H_0).

```
        Paired t-test

data:  predicted and measured
t = -3.0485, df = 3, p-value = 0.05549
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -1.99284398  0.04284398
sample estimates:
mean of the differences
                  -0.975
```

Figure 20: R output for Paired t-Test

Since the p-value is 0.05549, greater than 0.05 (or 5%), the Null Hypothesis (H_0) is accepted, meaning that there is a difference between the means.

In this situation is necessary to compute the estimate of the *Standard Error of the Statistic*, which indicates the precision of an estimate of the population [97]. It is defined as being the standard deviation divided by the square root of the sample size:

$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}} \qquad (7)$$

Due to the fact that two populations with different variances were used, it is necessary to distinguish between the two variances by rewriting *Equation 7* as below:

$$\sigma_{\overline{x_1}-\overline{x_2}} = \frac{\sigma_1 + \sigma_2}{\sqrt{n}} \qquad (8)$$

For the two data sets used – the predicted time and the measured time – the standard deviation for the predicted samples (σ_1) is of 0.3947573 and the standard deviation for the measured samples (σ_2) is of 1.804624.

$$\sigma_{\overline{x_1}-\overline{x_2}} = \frac{0.3947573 + 0.8504901}{\sqrt{4}}$$

The *Standard Error of the Statistic* calculated with the formula above is:

$$\sigma_{\overline{x_1}-\overline{x_2}} = 0.6226237$$

As the degree of precision represented by the *Standard Error of the Statistic* calculated based on the extracted samples from the data sets is of 62.3%, it means that the Solar Predictor has a maximum error of 62.3% for an entire population. For example, if the software predicts that a cloud will reach the sun in 30s, the cloud may reach it in a time interval within the 62.3% margin of error, which can be around 50s. Due to the fact that the prediction of the time necessary for the cloud to exit the sun's corona is based on the prediction of the time needed for the cloud to reach the sun, the same error is applicable also for the case when the cloud exits the sun's corona.

Even though the prediction was achieved for intervals lower than one minute, it showed the capabilities of the Solar Predictor to make quasi-continuous predictions.

Chapter 6 – Conclusions and Recommendations

The purpose of this project was to investigate different methods of predicting solar irradiance based on the motion of the clouds as well as to create a proof-of-concept for a Solar Predictor. Based on the project requirements and on the available scientific literature, the research revealed that Camera Detection in combination with *background reduction* and *motion tracking* algorithms are the most appropriate.

Given that for an intra-hour interval the sun can be considered motionless, the solar irradiance is mainly influenced by the motion of the clouds [40]. In accordance with the defined research question, the state-of-the-art solar prediction technology available in scientific literature revealed that Camera Detection exhibited the most advantages to predict cloud motion and solar irradiance, its main advantage being the fact that the hardware consists of a relatively simple device that allows enough flexibility for software implementation.

Regarding the accuracy of the prediction software, improvements were made by adjusting the cloud and sun recognition filters, as well as by implementing trajectory computing and speed determining software techniques available in the *OpenCV 3.4* libraries. The reached accuracy in the time interval allocated for this project prove the capabilities of the Solar Predictor to make quasi-continuous intra-hour predictions.

The most challenging part of the Solar Predictor was the requirement of creating a less resource-intensive method to achieve an accurate prediction of solar irradiance. In terms of software, *background reduction* and *motion tracking* algorithms were implemented to achieve the detection of clouds, as well as to determine and predict their movement. The overall size of the created software was of 6.62 KB (by considering also the size of the installed libraries it reaches approximately 300 MB), running on one single CPU, which achieved comparable results with other prediction systems from literature, such as the one created by *Chu et al.* in 2015 in which 10 computer cores were used [59].

An in-depth comparison with similar systems from the related scientific literature revealed the fact that the method implemented in this project yielded satisfying results, taking into account the project duration, its objectives and constraints, as well as the fact that the project was aimed for research purposes only. Other comparable systems were developed over a longer period of time using specialised hardware such as Total Sky Imagers and highly resource-intensive programming techniques [92], [98]. The Solar Predictor achieved similar results for a shorter period of time, by using a normal webcam, a fish-eye lens and an optical filter in combination with a more compact programming method of a total hardware cost of €54.81 (*Appendix A*).

The Solar Predictor Project has the potential to be upgraded in different ways, both in terms of software and hardware, depending on the needs of the area of implementation. For example, it can be used in a domestic environment as an aid for the management of household electrical systems connected to solar panels, making it possible to schedule appliances depending on the availability of solar energy. Furthermore, the solar predictor can be implemented on a much larger scale, such as an interconnected grid of devices that form a Smart Grid. A way to do that is by using multiple interconnected Solar Predictors in the form of a sensor network system able to communicate to each other at all times, so that the accuracy of solar irradiance prediction can increase.

As mentioned in Chapter 5, the created Solar Predictor system exhibited a maximum error of 62.3%. In order to reduce this error and improve the system's overall accuracy, a number of improvements can be undertaken.

The first possible improvement would be to increase the time interval for which the software records the displacement of the clouds based on their coordinates (*Figure 17*). At the moment the software calculates the displacement of a cloud in one second (or 30 frames). By increasing the time interval in that the software records the position of the clouds, the clouds' displacement can become better visible. Even though the precise time interval for the cloud position recording is still to be determined, prediction systems from scientific literature informs that the recording of cloud displacement is to be done every 20-30 seconds [63], [92], which in the case of the used webcam means every 600 to 900 frames. As the software approach for the Solar Predictor differs from the ones in the researched scientific literature, the displacement recording rate may differ as well. This means that, in the event of future improvements, the best approach would be to record the displacement and the stated prediction at various time intervals (ex. 5s, 10s, 15s etc.) for the same samples and compare them with the real time necessary for the cloud to cover and exit the sun. This would determine the most suitable time interval for the recording of the cloud displacement.

Another factor that can be further developed is the approximation of the clouds' shape. In the current version of the software, when a cloud is detected it is estimated as a circle, this being an embedded functionality in the *OpenCV 3.4* blob detection library. For further improvement of the shape approximation, a new method can be created so that the detected clouds can be estimated as ellipses. In this way, the errors will be reduced for clouds that have elongated shapes. Furthermore, another approach would be to implement an edge detection algorithm on the Bit Mask that would extract or approximate the outer edges of the cloud, offering the possibility to use a more accurate cloud detection. However, it is important to note that such algorithms is high resource-intensive and may require a high-performance CPU [99]–[103].

In order to keep in line with the project requirement to create a low resource-intensive system both from a hardware and software perspective, an Artificial Neural Network was considered redundant to be implemented in the Solar Predictor. However, a relatively new machine learning library, *TensorFlow*, represents a possible alternative for the creation of a complete ANN [104]. This is done by making use of a pre-trained Convolutional Neural Network model called *Inception*, created by *Google*, which was trained to apply the learning from a previous learning session to a new learning session [105]–[107]. *TensorFlow* makes it possible to train only the last layer of the network, making the model faster and lower powered comparing with running from CPUs [108]. Being an open-source library that focuses on Machine Learning and Deep Neural Networks, it presents numerous advantages such as the fact that it demonstrates fast compilation time in comparison to other similar libraries [109] and provides the Application Programming Interface (API) for *Python* for building and executing computational graphs [104]. Above all, the main advantage of *TensorFlow* is the fact that it does not need a large amount of computing power or time; this because it can be compiled on a separate device and then loaded and executed on devices that have a limited storage space [108].

The use of the *TensorFlow* libraries mentioned above would allow the addition of a cloud classifier that could recognise different types of clouds and adjust the prediction for each case, depending on their optical thickness. For example, an optically thin cloud that passes in front of the sun, still allows solar light to pass through, but at lower intensity. Depending on the type of cloud, the percentage of light that reaches the PV system can be added to the prediction, providing in this way more information to the end user.

The findings from the scientific literature highlighted the fact that the creation of similar systems required a longer time for data acquisition and validation. One relevant example is the forecasting system developed by *Chu et al.* that achieved prediction for intra-hour time intervals up to 20 minutes, having allocated six months only for data acquisition and another six months for software preparation [59], [61]. The testing of the created software for a larger number of samples would be another important factor in the further improvement of the Solar predictor. By allocating a longer time for data acquisition, the statistical tests can output the accuracy and errors based on a larger sample population.

Overall, the Solar Predictor project was very challenging and demonstrated that cloud motion and solar irradiance can be predicted at an intra-hour temporal resolution for time intervals up to one minute, even when lower resources and significantly less time for acquiring data and validation are allocated.

List of definitions and abbreviations

ANN	= Artificial Neural Network
API	= Application Programming Interface
Azimuth	= The angular distance between a celestial object and the observer
CCD	= Charge Coupled Device
CMOS	= Complementary Metal Oxide Semiconductor
CPU	= Central Processing Unit
HSV	= Hue, Saturation and Value
Intra-hour	= Within an hour; in the context of the report, it referrers to temporal resolution
IR	= Infrared
KB	= Kilobyte
MBE	= Mean Bias Error
MP	= Megapixel
ND Filter	= Neutral Density Filter
Overcast (sky)	= Meteorological condition of clouds obscuring at least 95% of the sky
PV	= Photovoltaic
RGB	= Red, Green and Blue
RMSE	= Root Mean Square Error
USB	= Universal Serial Bus

Referenced cited

[1] E. Kabir, P. Kumar, S. Kumar, A. A. Adelodun, and K.-H. Kim, 'Solar energy: Potential and future prospects', *Renew. Sustain. Energy Rev.*, vol. 82, no. 1, pp. 894–900, 2018.

[2] P. G. Vasconcelos Sampaio and M. O. Aguirre González, 'Photovoltaic solar energy: Conceptual framework', *Renew. Sustain. Energy Rev.*, vol. 74, pp. 590–601, 2017.

[3] N. Kannan and D. Vakeesan, 'Solar energy for future world: - A review', *Renew. Sustain. Energy Rev.*, vol. 62, pp. 1092–1105, 2016.

[4] F. P. Sioshansi, *Smart Grid: Integrating Renewable, Distributed & Efficient Energy*. Academic Press, 2012.

[5] National Renewable Energy Laboratory, 'Solar Energy and Capacity Value', 2013.

[6] A. Hahn, A. Ashok, S. Sridhar, and M. Govindarasu, 'Cyber-Physical Security Testbeds: Architecture, Application, and Evaluation for Smart Grid', *IEEE Trans. Smart Grid*, vol. 4, no. 2, pp. 847–855, 2013.

[7] Y. Mo *et al.*, 'Cyber–Physical Security of a Smart Grid Infrastructure', *Proc. IEEE*, vol. 100, no. 1, pp. 195–209, 2012.

[8] J. Joy, E. A. Jasmin, and V. R. John, 'Challenges of Smart Grid', *Int. J. Adv. Res. Electr. Electron. Instrum. Eng.*, vol. 2, no. 3, pp. 976–981, 2013.

[9] V. C. Gungor *et al.*, 'Smart Grid Technologies: Communication Technologies and Standards', *IEEE Trans. Ind. Informatics*, vol. 7, no. 4, pp. 529–539, 2011.

[10] E. Fadel *et al.*, 'A survey on wireless sensor networks for smart grid', *Comput. Commun.*, vol. 71, no. 1, pp. 22–33, 2015.

[11] Y. Zhang, W. Chen, and W. Gao, 'A survey on the development status and challenges of smart grids in main driver countries', *Renew. Sustain. Energy Rev.*, vol. 79, no. 1, pp. 137–147, 2017.

[12] Energy Storage Association, 'Distributed Grid-Connected PV Integration', 2017. [Online]. Available: http://energystorage.org/energy-storage/technology-applications/distributed-grid-connected-pv-integration. [Accessed: 24-Mar-2018].

[13] Energy Storage Association, 'Renewable Integration Benefits', 2017. [Online]. Available: http://energystorage.org/energy-storage/energy-storage-benefits/benefit-categories/renewable-integration-benefits. [Accessed: 24-Feb-2018].

[14] R. K. Breteler, 'Energy Consumption trends in UAE compared to the Netherlands', 2014.

[15] J. Malinowski and K. Kaderly, 'Peak Shaving - A Method to Reduce Utility Costs', in *IEEE Region 5 Conference: Annual Technical and Leadership Workshop*, 2004, pp. 41–44.

[16] M. Uddin, M. F. Romlie, M. F. Abdullah, S. A. Halim, A. H. A. Bakar, and T. C. Kwang, 'A review on peak load shaving strategies', *Renew. Sustain. Energy Rev.*, vol. 82, pp. 3323–3332, 2018.

[17] P. Yeung, 'Reducing Energy Costs with Peak Shaving in Industrial Environments', 2007.

[18] R. Perez, T. Hoff, C. Herig, and J. Shah, 'Maximizing PV peak shaving with solar load control: validation of a web-based economic evaluation tool', *Sol. Energy*, vol. 74, no. 5, pp. 409–415, 2003.

[19] G. Zakeri, D. Craigie, A. Philpott, and M. Todd, 'Optimization of demand response through peak shaving', *Oper. Res. Lett.*, vol. 42, no. 1, pp. 97–101, 2014.

[20] S. K. H. Chow, E. W. M. Lee, and D. H. W. Li, 'Short-term prediction of photovoltaic energy generation by intelligent approach', *Energy Build.*, vol. 55, pp. 660–667, 2012.

[21] I. Solorio and H. Jörgens, *A Guide to EU Renewable Energy Policy*. Edward Elgar Publishing, 2017.

[22] H. Geller, P. Harrington, A. H. Rosenfeld, S. Tanishima, and F. Unander, 'Polices for increasing energy efficiency: Thirty years of experience in OECD countries', *Energy Policy*, vol. 34, no. 5, p. Pages 556-573, 2006.

[23] K. Tanaka, 'Review of policies and measures for energy efficiency in industry sector', *Energy Policy*, vol. 39, no. 10, pp. 6532–6550, 2011.

[24] S. Bird and D. Hernández, 'Policy options for the split incentive: Increasing energy efficiency for low-income renters', *Energy Policy*, vol. 48, pp. 506–514, 2012.

[25] C. W. Gellings, *The Smart Grid: Enabling Energy Efficiency and Demand Response*. The Fairmont Press, Inc., 2009.

[26] S. M. Amin and B. F. Wollenberg, 'Toward a smart grid: power delivery for the 21st century', *IEEE Power Energy Mag.*, vol. 3, no. 5, pp. 34–41, 2005.

[27] H. Farhangi, 'The path of the smart grid', *IEEE Power Energy Mag.*, vol. 8, no. 1, pp. 18–28, 2010.

[28] X. Fang, S. Misra, G. Xue, and D. Yang, 'Smart Grid - The New and Improved Power Grid: A Survey', *IEEE Commun. Surv. Tutorials*, vol. 14, no. 4, pp. 944–980, 2012.

[29] J. H. Seinfeld and S. N. Pandis, *Atmospheric Chemistry and Physics: From Air Pollution to Climate Change*, 3rd ed. John Wiley & Sons, 2016.

[30] S. Rauf, S. Rasool, M. Rizwan, M. Yousaf, and N. Khan, 'Domestic Electrical Load Management Using Smart Grid', *Energy Procedia*, vol. 100, no. 1, pp. 253–260, 2016.

[31] B. Elsinga and W. G. J. H. M. van Sark, 'Analytic model for correlations of cloud induced fluctuations of clear-sky index', *Sol. Energy*, vol. 155, pp. 985–1001, 2017.

[32] A. Mahmood *et al.*, 'Home Appliances Coordination Scheme for Energy Management (HACS4EM) using Wireless Sensor Networks in Smart Grids', *Procedia Comput. Sci.*, vol. 32, no. 1, pp. 469–476, 2014.

[33] K. Ma, T. Yao, J. Yang, and X. Guan, 'Residential power scheduling for demand response in smart grid', *Int. J. Electr. Power Energy Syst.*, vol. 78, pp. 320–325, 2016.

[34] K. Ma, G. Hu, and C. J. Spanos, 'Energy consumption scheduling in smart grid: A non-cooperative game approach', in *9th Asian Control Conference (ASCC)*, 2013, pp. 1–6.

[35] T. Logenthiran, D. Srinivasan, and T. Z. Shun, 'Demand Side Management in Smart Grid Using Heuristic Optimization', *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1244–1252, 2012.

[36] M. Paulescu, E. Paulescu, P. Gravila, and V. Badescu, *Weather Modeling and Forecasting of PV Systems Operation*. Springer Science & Business Media, 2012.

[37] R. Marquez and C. F. M. Coimbra, 'Intra-hour DNI forecasting based on cloud tracking image analysis', *Sol. Energy*, vol. 91, no. 1, pp. 327–336, 2013.

[38] M. S. Ghonima, B. Urquhart, C. W. Chow, J. E. Shields, A. Cazorla, and J. Kleissl, 'A method for cloud detection and opacity classification based on ground based sky imagery', *Atmos. Meas. Tech.*, vol. 5, no. 11, pp. 2881–2892, 2012.

[39] F. Barbieri, S. Rajakaruna, and A. Ghosh, 'Very short-term photovoltaic power forecasting with cloud modeling: A review', *Renew. Sustain. Energy Rev.*, vol. 75, pp. 242–263, 2017.

[40] Z. Zhen, Z. Wang, F. Wang, Z. Mi, and K. Li, 'Research on a cloud image forecasting approach for solar power forecasting', *Energy Procedia*, vol. 142, pp. 362–368, 2017.

[41] D. R. Brooks, *Bringing the Sun Down to Earth: Designing Inexpensive Instruments for Monitoring the Atmosphere*. Springer Science & Business Media, 2008.

[42] V. Srikrishnan, G. S. Young, L. T. Witmer, and J. R. S. Brownson, 'Using multi-pyranometer arrays and neural networks to estimate direct normal irradiance', *Sol. Energy*, vol. 119, pp. 531–542, 2015.

[43] D. Y. Goswami, F. Kreith, and J. F. Kreider, *Principles of Solar Engineering*, 2nd ed. CRC Press, 2000.

[44] Omni instruments, 'CMP3 Pyranometer'. [Online]. Available: http://www.omniinstruments.co.uk/weather-stations-and-instruments/pyranometers-solar-irradiance/cmp3-pyranometer.html. [Accessed: 22-Feb-2018].

[45] K. Hornik, M. Stinchcombe, and H. White, 'Multilayer feedforward networks are universal approximators', *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[46] T. Muneer and S. Younes, 'The all-sky meteorological radiation model: proposed improvements', *Appl. Energy*, vol. 83, no. 5, pp. 436–450, 2006.

[47] J. C. Baltazar, Y. Sun, and J. Haberl, 'Improved methodology to measure normal incident solar radiation with a multi-pyranometer array', *Energy Procedia*, vol. 57, pp. 1211–1219, 2014.

[48] R. Perez, R. Seals, P. Ineichen, R. Stewart, and D. Menicucci, 'A new simplified version of the perez diffuse irradiance model for tilted surfaces', *Sol. Energy*, vol. 39, no. 3, pp. 221–231, 1987.

[49] D. Faiman, D. Feuermann, P. Ibbetson, and A. Zemel, 'A multipyranometer instrument for obtaining the solar beam and diffuse components, and the irradiance on inclined planes', *Sol. Energy*, vol. 48, no. 4, pp. 253–259, 1992.

[50] D. Faiman, D. Feuermann, and A. Zemel, 'Site-independent algorithm for obtaining the direct beam insolation from a multipyranometer instrument', *Sol. Energy*, vol. 50, no. 1, pp. 53–57, 1993.

[51] D. Faiman, A. Zemel, and A. Zangvil, 'A method for monitoring insolation in remote regions', *Sol. Energy*, vol. 38, no. 5, pp. 327–333, 1987.

[52] Omni instruments, 'Pyranometers - Solar Irradiance', 2015. [Online]. Available: http://www.omniinstruments.co.uk/weather-stations-and-instruments/pyranometers-solar-irradiance.html. [Accessed: 14-Jun-2018].

[53] R. Lukac, *Single-Sensor Imaging: Methods and Applications for Digital Cameras*. CRC Press, 2008.

[54] D. Litwiller, 'CMOS vs. CCD: Maturing Technologies, Maturing Markets - The factors determining which type of imager delivers better cost performance are becoming more refined', *Photonics Spectra*, vol. 39, no. 8, pp. 54–61, 2005.

[55] J. E. Shields, R. W. Johnson, and T. L. Koehler, 'Automated whole sky imaging systems for cloud field assessment', in *Fourth Symposium on Global Change Studies, American Meteorological Society*, 1993.

[56] A. Heinle, A. Macke, and A. Srivastav, 'Automatic cloud classification of whole sky images', *Atmos. Meas. Tech.*, vol. 3, no. 3, pp. 557–567, 2010.

[57] Q. Li, W. Lu, and J. Yang, 'A hybrid thresholding algorithm for cloud detection on ground-based color images', *J. Atmos. Ocean. Technol.*, vol. 28, no. 10, pp. 1286–1296, 2011.

[58] M. Yamashita, M. Yoshimura, and K. Iwao, 'Monitoring and Discrimination for Sky Conditions Using Multi-temporal Whole Sky Imageries', in *ACRS Proceedings, Hanoi*, 2015.

[59] Y. Chu, H. T. C. Pedro, M. Li, and C. F. M. Coimbra, 'Real-time forecasting of solar irradiance ramps with smart image processing', *Sol. Energy*, vol. 114, pp. 91–104, 2015.

[60] J. F. Kenney, *Mathematics of Statistics, Parts 1-2*. Literary Licensing, 2013.

[61] Y. Chu, H. T. C. Pedro, and C. F. M. Coimbra, 'Hybrid intra-hour DNI forecasts with sky image processing enhanced by stochastic learning', *Sol. Energy*, vol. 98, no. 3, pp. 592–603, 2013.

[62] D. N. Joanes and C. A. Gill, 'Comparing measures of sample skewness and kurtosis', *J. R. Stat. Soc. Ser. D Stat.*, vol. 47, no. 1, pp. 183–189, 1998.

[63] R. Chauvin, J. Nou, S. Thil, A. Traoré, and S. Grieu, 'Cloud detection methodology based on a sky-imaging system', *Energy Procedia*, vol. 69, pp. 1970–1980, 2015.

[64] N. Otsu, 'A threshold selection method from gray-level histograms', *IEEE Trans. Syst. Man Cybern.*, vol. 9, no. 1, pp. 62–66, 1979.

[65] M. Yabuki, M. Shiobara, K. Nishinaka, and M. Kuji, 'Development of a cloud detection method from whole-sky color images', *Polar Sci.*, vol. 8, no. 4, pp. 315–326, 2014.

[66] J. Sabburg and J. Wong, 'Evaluation of a Ground-Based Sky Camera System for Use in Surface Irradiance Measurement', *J. Atmos. Ocean. Technol.*, vol. 16, no. 6, pp. 752–759, 1999.

[67] M. C. Larciprete *et al.*, 'Temperature dependent emissivity of different stainless steel textiles in the infrared range', *Int. J. Therm. Sci.*, vol. 113, pp. 130–135, 2017.

[68] T. Williams, *Thermal Imaging Cameras: Characteristics and Performance*. CRC Press, 2009.

[69] Y. Gao and G. Y. Tian, 'Emissivity correction using spectrum correlation of infrared and visible images', *Sensors Actuators A Phys.*, vol. 270, pp. 8–17, 2018.

[70] Sapling Learning, 'Electromagnetic Spectrum', *Chempendix*. [Online]. Available: https://sites.google.com/site/chempendix/em-spectrum. [Accessed: 12-Mar-2018].

[71] J. Fraden, *Handbook of Modern Sensors: Physics, Designs, and Applications*, 4th ed. Springer Science & Business Media, 2010.

[72] P. Tzoumanikas, E. Nikitidou, A. F. Bais, and A. Kazantzidis, 'The effect of clouds on surface solar irradiance, based on data from an all-sky imaging system', *Renew. Energy*, vol. 95, no. 1, pp. 314–322, 2016.

[73] B. A. Wielicki, R. D. Cess, M. D. King, D. A. Randall, and E. F. Harrison, 'Mission to Planet Earth: Role of Clouds and Radiation in Climate', *Bull. Am. Meteorol. Soc.*, vol. 76, no. 11, pp. 2125–2153, 1995.

[74] M. Bombrun, D. Jessop, A. Harris, and V. Barra, 'An algorithm for the detection and characterisation of volcanic plumes using thermal camera imagery', *J. Volcanol. Geotherm. Res.*, vol. 352, pp. 26–37, 2018.

[75] A. J. L. Harris, D. D. Donne, J. Dehn, M. Ripepe, and A. K. Worden, 'Volcanic plume and bomb field masses from thermal infrared camera imagery', *Earth Planet. Sci. Lett.*, vol. 365, pp. 77–85, 2013.

[76] A. J. Prata and C. Bernardo, 'Retrieval of volcanic ash particle size, mass and optical depth from a ground-based thermal infrared camera', *J. Volcanol. Geotherm. Res.*, vol. 186, no. 1–2, pp. 91–107, 2009.

[77] T. Lopez, H. E. Thomas, A. J. Prata, A. Amigo, D. Fee, and D. Moriano, 'Volcanic plume characteristics determined using an infrared imaging camera', *J. Volcanol. Geotherm. Res.*, vol. 300, pp. 148–166, 2015.

[78] S. Smith and R. Toumi, 'Measuring Cloud Cover and Brightness Temperature with a Ground-Based Thermal Infrared Camera', *J. Appl. Meteorol. Climatol.*, vol. 47, no. 2, pp. 683–693, 2008.

[79] A. Hoyano, K. Asano, and T. Kanamaru, 'Analysis of the sensible heat flux from the exterior surface of buildings using time sequential thermography', *Atmos. Environ.*, vol. 33, no. 24–25, pp. 3941–3951, 1999.

[80] B. Lehmann, K. G. Wakili, T. Frank, B. V. Collado, and C. Tanner, 'Effects of individual climatic parameters on the infrared thermography of buildings', *Appl. Energy*, vol. 110, pp. 29–43, 2013.

[81] A. Sharma and A. Kakkar, 'Forecasting daily global solar irradiance generation using machine learning', *Renew. Sustain. Energy Rev.*, vol. 82, pp. 2254–2269, 2018.

[82] J. Li, J. K. Ward, J. Tong, L. Collins, and G. Platt, 'Machine learning for solar irradiance forecasting of photovoltaic system', *Renew. Energy*, vol. 90, pp. 542–553, 2016.

[83] V. Sharma, D. Yang, W. Walsh, and T. Reindl, 'Short term solar irradiance forecasting using a mixed wavelet neural network', *Renew. Energy*, vol. 90, pp. 481–492, 2016.

[84] Hama, 'Hama Pocket Webcam', *Datasheet 00053901*. [Online]. Available: https://www.hama.com/00053901/hama-webcam-pocket. [Accessed: 14-Jun-2018].

[85] F. McCollough, *Complete Guide to High Dynamic Range Digital Photography*. Lark Books, 2008.

[86] R. Hoddinott, *The Digital Photographer's Guide to Filters: The Complete Guide to Hardware and Software Filtration*. David & Charles, 2007.

[87] A. Martelli, A. Ravenscroft, and S. Holden, *Python in a Nutshell*, 3rd ed. O'Reilly Media, Inc., 2017.

[88] M. Lutz, *Learning Python: Powerful Object-Oriented Programming*, 5th ed. O'Reilly Media, Inc., 2013.

[89] J. Howse, P. Joshi, and M. Beyeler, *OpenCV: Computer Vision Projects with Python*. Packt Publishing Ltd, 2016.

[90] I. El-Feghi, H. Aboasha, M. A. Sid-Ahmed, and M. Ahmadi, 'Content-Based Image Retrieval Based on Efficient Fuzzy Color Signature', in *IEEE International Conference on Systems, Man and Cybernetics*, 2007, pp. 1118–1124.

[91] J. A. Day, *The Book of Clouds*. Sterling Publishing Company, Inc., 2005.

[92] C. W. Chow *et al.*, 'Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed', *Sol. Energy*, vol. 85, no. 11, pp. 2881–2893, 2011.

[93] B. A. Walther and J. L. Moore, 'The concepts of bias, precision and accuracy, and their use in testing the performance of species richness estimators, with a literature review of estimator performance', *Ecography (Cop.).*, vol. 28, no. 6, pp. 815–829, 2005.

[94] D. C. Montgomery and G. C. Runger, *Applied Statistics and Probability for Engineers*, 6th ed. John Wiley & Sons, 2014.

[95] M. J. Crawley, *The R Book*, 2nd ed. John Wiley & Sons, 2012.

[96] T. Hothorn and B. S. Everitt, *A Handbook of Statistical Analyses using R*, 3rd ed. CRC Press, 2014.

[97] M. L. McHugh, 'Standard error: meaning and interpretation', *Biochem. Medica*, vol. 18, no. 1, pp. 7–13, 2008.

[98] D. Bernecker, C. Riess, E. Angelopoulou, and J. Hornegger, 'Continuous short-term irradiance forecasts using sky images', *Sol. Energy*, vol. 110, pp. 303–315, 2014.

[99] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*. Springer Science & Business Media, 2008.

[100] R. Dyckerhoff and P. Mozharovskyi, 'Exact computation of the halfspace depth', *Comput. Stat. Data Anal.*, vol. 98, pp. 19–30, 2016.

[101] I. Ruts and P. J. Rousseeuw, 'Computing depth contours of bivariate point clouds', *Comput. Stat. Data Anal.*, vol. 23, no. 1, pp. 153–168, 1996.

[102] A. Ruano, H. R. Khosravani, and P. M. Ferreira, 'A Randomized Approximation Convex Hull Algorithm for High Dimensions', *IFAC-PapersOnLine*, vol. 48, no. 10, pp. 123–128, 2015.

[103] J.-F. Cayula and P. Cornillon, 'Edge Detection Algorithm for SST Images', *J. Atmos. Ocean. Technol.*, vol. 9, no. 1, pp. 64–80, 1992.

[104] T. Hope, Y. S. Resheff, and I. Lieder, *Learning TensorFlow: A Guide to Building Deep Learning Systems*. O'Reilly Media, Inc., 2017.

[105] R. Bonnin, *Machine Learning for Developers: Uplift your regular applications with the power of statistics, analytics, and machine learning*. Packt Publishing Ltd, 2017.

[106] M. Sewak, M. R. Karim, and P. Pujari, *Practical Convolutional Neural Networks: Implement advanced deep learning models using Python*. Packt Publishing Ltd, 2018.

[107] M. Abadi *et al.*, 'TensorFlow: A System for Large-Scale Machine Learning', in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, 2016, pp. 265–283.

[108] K. Hwang, *Cloud Computing for Machine Learning and Cognitive Applications: A Machine Learning Approach*. MIT Press, 2017.

[109] L. Rampasek and A. Goldenberg, 'TensorFlow: Biology's Gateway to Deep Learning?', *Cell Syst.*, vol. 2, no. 1, pp. 12–14, 2016.

Appendix A – Bill of Materials

Part nr.	Materials	Specification	Cost (€)
01	Camera	1.3 MP, 1280 x 1024 resolution, 30 fps	12.83
02	Lens	Clip-on fish-eye lens	8.49
03	Light Filter	ND-8 filter	18.95
02	Protective case	Acrylic one-sided satin, 180 x 70 mm	10.00
03	Data Cable	USB, 5 m	4.54
		Total	**54.81**

The Hama Pocket webcam shown in the picture has been found to satisfy all specifications of the Solar Predictor as it supports 1280 x 1024 real-time video recording.

Note: Several materials (such as the data cable) might require adjusted specifications depending on the location the system is implemented in. For example, the data cable might be required to have different lengths for different locations, thus changing the price. The bill of materials above is done for the location where the system was installed at approximately 5 m from the nearest computer.

Appendix B – Hardware Casing Design

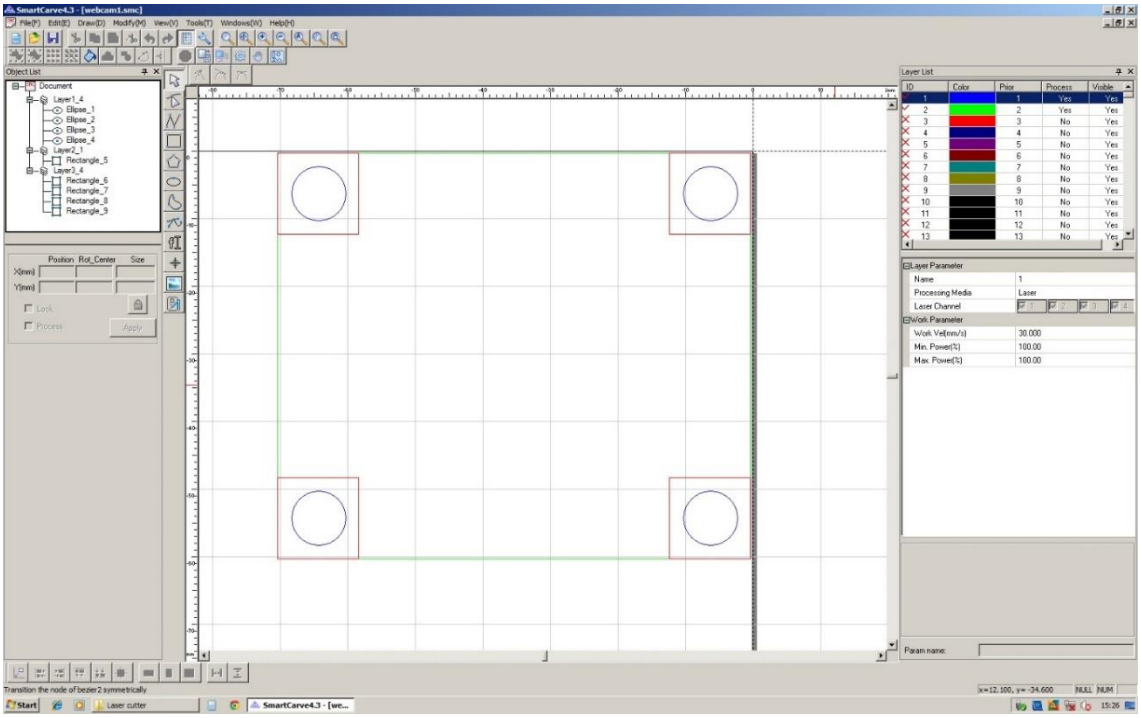
Colour code:

- Blue – Cut first
- Green – Cut second
- Red – Do not cut (only for measurement)

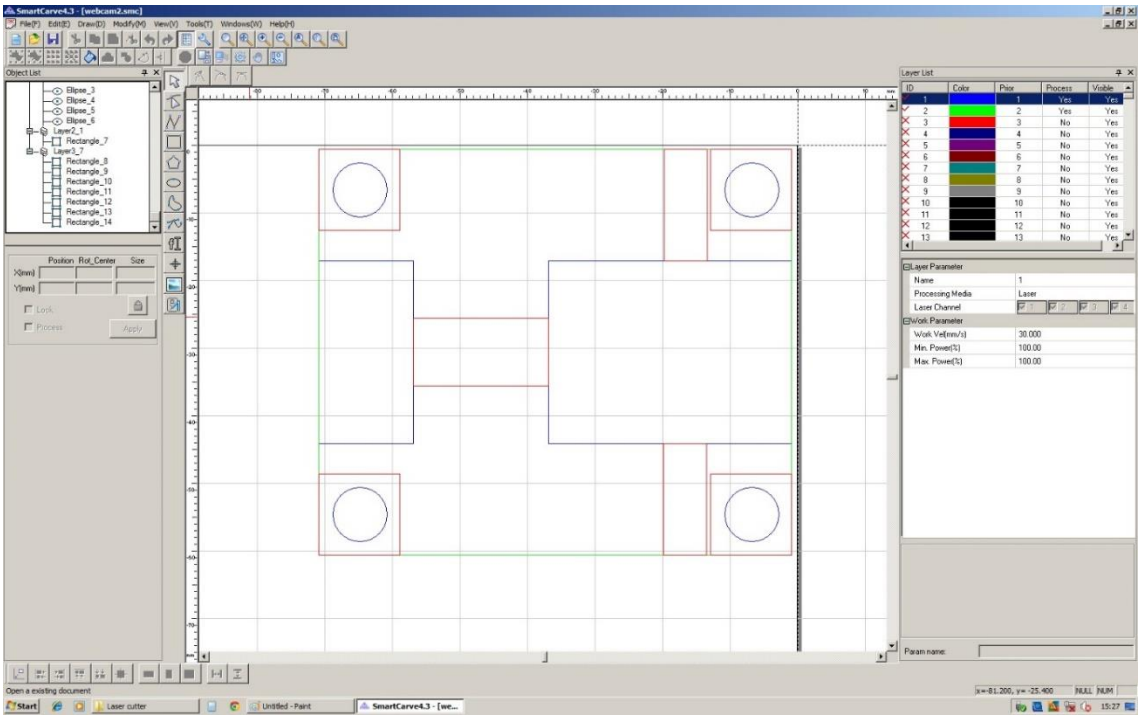
Thickness of layers = 8mm

Diameter of blue circles = 8 mm

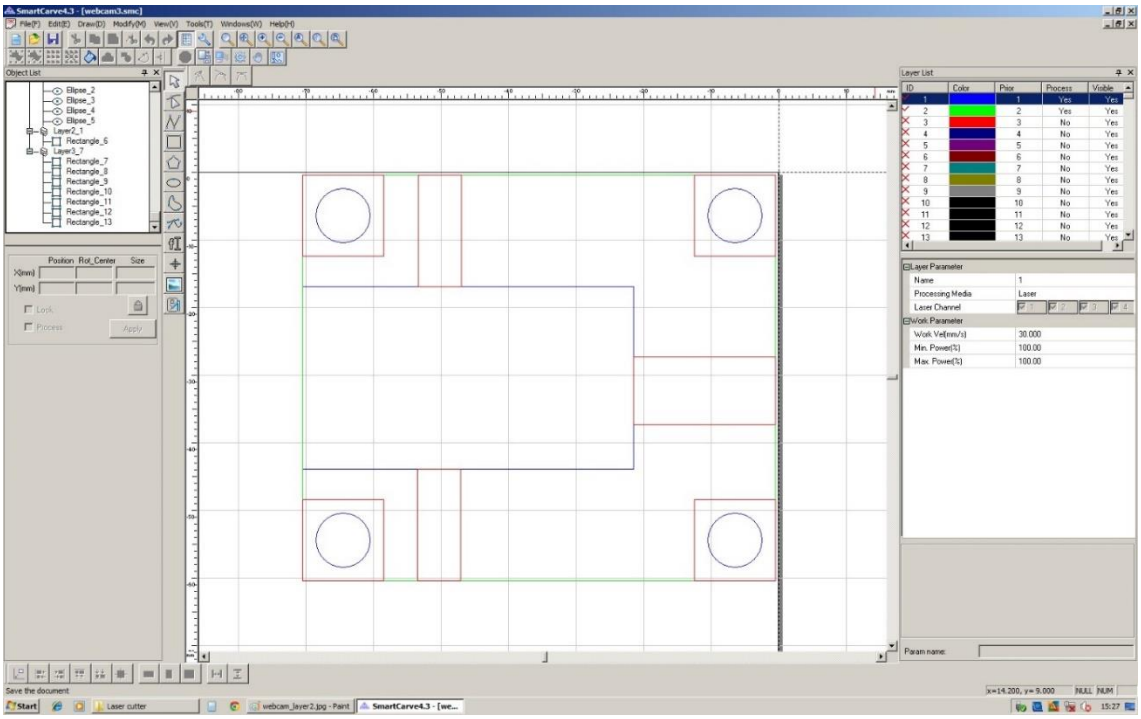
Layer 1:



Layer 2:



Layer 3:



Appendix C – Design FMEA

Item and Function/ Requirements	Potential Failure Mode	Potential Effect(s) of Failure	Severity	Potential Cause(s)/ Mechanism(s) of Failure	Occurrence	Current Controls - Prevention	Current Controls -Detection	Detection	RPN	Recommended Action
Camera	Change in position	The field-of-view is shifted	6	Connection to the surface	4	Specific instructions are given	Visually checking the placement	4	96	Follow the given instructions
	Obstructed field of view	The camera cannot see part of the sky	7	Spots on the lens	7	The lens is cleaned before each test	Visually checking the camera lens	3	147	Regularly check camera
	Internal circuit malfunction	Distorted data is recorded	8	Malfunction of the camera	2	Camera is tested before implementation	Test of the camera during maintenance check-ups	8	128	Create standardised tests for the different camera features
		No data is recorded	8	Malfunction of the camera	2	Camera is tested before implementation	Test of the camera during maintenance check-ups	7	112	Create standardised tests for the different camera features
Protecting case	The case breaks	The device is not protected anymore therefore it is open to outside factors	4	Too much force is applied on the case	2	Strong material is used for the outer case	The case is visually checked	3	24	Use stronger material (i.e. Aluminium or Titanium)
CPU	Internal circuit malfunction	Data cannot be acquired	6	Connection between camera and CPU is defective	3	All the connections are checked before implementation	Test of the camera-CPU connection during maintenance check-ups	5	90	Create standardised tests for the different CPU features
		Data cannot be analysed	6	Software malfunction	2	Software is tested before implementation	Run software diagnostics during maintenance check-ups	6	72	Create standardised tests for the different CPU features
Data and power cable	One or multiple connections fail	Signals cannot be transmitted	8	Poor hardware connection	3	All the connections are checked before implementation	Manually check the connections	6	144	Acquire durable signal wire

Appendix D – Implemented Software

```
##############################
# Name: solarPredictor.py
# Author: Andrei-Cristian STEFAN
# Date: 26.05.2018
##############################

# Standard imports
from math import*
import cv2
import numpy as np;
import time

# Read video
cap = cv2.VideoCapture(0)

# Open output files
file = open("writePrediction.txt", "w") # w=write

frameNr = 1
i = 0

# This sets the frame rate of the video to be used in calculations
fps = 30

#####################################################
#Setup SimpleBlobDetector parameters.
params = cv2.SimpleBlobDetector_Params()

#Change thresholds
params.minThreshold = 0;
params.maxThreshold = 255;
#minDistBetweenBlobs = 10;

#Filter by Color
params.filterByColor = True
params.blobColor = 255

#Filter by Area
params.filterByArea = True
params.minArea = 1500
params.maxArea = 1000000000

#Filter by Circularity
params.filterByCircularity = False
params.minCircularity = 5
params.maxCircularity = 20

#Filter by Convexity
params.filterByConvexity = False
params.minConvexity = 0.87
```

```
#Filter by Inertia
params.filterByInertia = False
params.minInertiaRatio = 0.01
params.maxInertiaRatio = 0.99

#Create a detector with the parameters
ver = (cv2.__version__).split('.')
if int(ver[0]) < 3 :
   detector = cv2.SimpleBlobDetector(params)
else :
   detector = cv2.SimpleBlobDetector_create(params)
####################################################

# Initialise parameters for velocity calculations
xpos = np.array([0])
ypos = np.array([0])
dis = 0
t = 1
vel = 0
cloudsize = 0

# This part runs in the first second to find the location of the sun
while (frameNr<30):

   #Capture frame by frame
   _, frame = cap.read()

   # Convert BGR to HSV
   hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

   # define range of white color in HSV
   sensitivitySun = 60
   lower_white_Sun = np.array([0, 0, 255 - sensitivitySun])
   upper_white_Sun = np.array([255, sensitivitySun, 255])

   # Threshold the HSV image to get only white colors
   mask = cv2.inRange(hsv, lower_white_Sun, upper_white_Sun)

   # Bitwise-AND mask and original image
   res = cv2.bitwise_and(frame, frame, mask = mask)

   #Set up the blob detector
   detector = cv2.SimpleBlobDetector_create(params)
   keypoints = detector.detect(mask)

   # Draw red circles on the detected blobs
   im_with_keypoints = cv2.drawKeypoints(mask, keypoints, np.array([]), (0,0,255),
cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)

   frameNr = frameNr + 1

   cv2.imshow("Sun detection", im_with_keypoints)

   i = 0
```

```
#This print function is made so that the user can easier see in the Control Panel the separate frames
    print ('----')

    for kp in keypoints:

        #This prints the values in the control panel
        print ('Sun:\t %d (%d, %d) size=%d' % (frameNr, kp.pt[0], kp.pt[1], kp.size))

        #This draws circles on the blobs
        cv2.circle(im_with_keypoints, (int(kp.pt[0]), int(kp.pt[1])), int(kp.size), (0, 0, 255))

        i = i + 1

# These parameters will be used for sun position and size
xsun = int(kp.pt[0])
ysun = int(kp.pt[1])
rsun = int(kp.size/2)

# This part performs the cloud detection
while (frameNr>=30):

    # Capture frame by frame
    _, frame = cap.read()

    # Convert BGR to HSV
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Define range of white color in HSV
    sensitivity = 120
    lower_white = np.array([0, 0, 255 - sensitivity])
    upper_white = np.array([255, sensitivity, 255])

    # Threshold the HSV image to get only white colors
    mask = cv2.inRange(hsv, lower_white, upper_white)

    # Bitwise-AND mask and original image
    res = cv2.bitwise_and(frame, frame, mask = mask)

    # Set up the blob detector
    detector = cv2.SimpleBlobDetector_create(params)
    keypoints = detector.detect(mask)

    # Draw red circles on the detected blobs
    im_with_keypoints = cv2.drawKeypoints(mask, keypoints, np.array([]), (0,0,255), cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)

    # Draw circle for where the sun is (GBR colour system)
    cv2.circle(im_with_keypoints,(xsun, ysun), rsun, (0,225,255), -1)

    # Show the image on screen
    cv2.imshow('frame',frame)
    cv2.imshow('res',res)
    cv2.imshow("Cloud detection", im_with_keypoints)
```

```
    # This increments the frame number for wach iteration
    frameNr = frameNr + 1

    i = 0
    # This print function is made so that the user can easier see in the Control Panel the separate
frames
    print ('----------')

    for kp in keypoints:

        xpos = np.append(xpos,kp.pt[0])
        ypos = np.append(ypos,kp.pt[1])

        print ('frame=%d ' % frameNr)

        #This draws circles on the blobs
        cv2.circle(im_with_keypoints, (int(kp.pt[0]), int(kp.pt[1])), int(kp.size), (0, 0, 255))

        i = i + 1

    # This part runs every second (30 frames)
    if frameNr%fps == 0:

        # This prints the values in the control panel
        print ('frame=%d (%d, %d) size=%.1f' % (frameNr, kp.pt[0], kp.pt[1], kp.size))

        for num in range(i):

            try:

                xdis = xpos[-1-num] - xpos[-2-2*num]
                ydis = ypos[-1-num] - ypos[-2-2*num]
                dis = (xdis**2 + ydis**2)**0.5

                # Velocity in pixels per second
                vel = dis/fps
                ra = atan(ydis/xdis)

                # Direction calculation using the angle (in degrees)
                ang = atan(ydis/xdis)/(2*pi)*360
                print ('frame=%d ; (%d, %d) ; size=%.1f ; velocity=%.2f ; angle=%.2f' % (frameNr,
kp.pt[0], kp.pt[1], kp.size, vel, ang))

                # Calculations for the sun
                cloudsize = kp.size/2
                totaldis = cloudsize + rsun
                dis2 = ((xsun - xpos[-1-num])**2 +(ysun - ypos[-1-num])**2)**0.5
                y2 = ysun - ypos[-1-num]
                angle2 = asin(y2/dis2)

                if (ra < angle2):
                    if  (ra > -angle2):

                        time = dis2/vel
                        print('Time to reach the sun: %.1f: ' %(time/24))
```

```
                print('Time to exit the sun: %.1f' %((time/24)+230))
                print('----------')

                # Write to file
                file.write('%d\t%.1f\t%.1f\n' %(frameNr, (time/fps), (time/fps)+230))

        except IndexError:
            None

    #Press "Esc" to close the video feed
    k = cv2.waitKey(5) & 0xFF
    if k == 27:
        break

#####################################################

#Terminating the camera feed and video windows
file.close()
cap.release()
cv2.destroyAllWindows()
```

Appendix E – Test Script

Step	Description	Expected results	Pass/Fail	Comments/Remarks
1	Set the webcam so it can have a clear view on the whole sky	The field of view is not obstructed by trees or buildings	Pass	The sun was positioned in the lower half of the field of view and it was considered stationary for the duration of each test
2	Run the created software to reduce the background track cloud motion	The sky-background was descarded and only the clouds remained	Pass	A Bit Mask with only pure white and pure black pixels was created
3	Filter the data for unwanted noise (smaller clouds)	All clouds with a size lower than 1500 pixels have been discarded	Pass	The filterring process was done automatically
4	Extract the coordinates of the clouds, as well as their size	For each cloud, their x and y coordinated will be extracted together with their relative size in pixels	Pass	The implemented blob detection method successfully extracted the coordinates and size
5	Use the extracted coordinates to calculate the moving direction	For each cloud, the moving direction will be determined based on their displacing and angle after each second (30 frames)	Pass	The created software methods for the calculation of angle and displacement showed satisfying results
6	State prediction (in seconds)	The prediction will be stated only if the clouds' moving direction is towards the sun	Pass	The implementation of the prediction method excludes the clouds that do not move towards the sun
7	Compare the predicted time with the time measured with the chronometer	The results with the implemented regions of interest will improve the detection and visibly eliminate the noise.	Pass	Two clouds were detected, but only the biggest one was predicted This because the other wasn't on a collision trajectory with the sun. It means that predictions are stated only for the clouds that have the potential of obstructuing the sunlight