A novel shade analysis technique for solar photovoltaic systems

Simon Wakter
Fredrik Wikerman
Abstract

Solar power is a rapidly growing energy source. As efficiency increases and costs decrease, solar photovoltaic (PV) systems are becoming increasingly attractive for small and medium scale consumers to invest in. Despite advancements in photovoltaic technology, minor shading will still drastically reduce the performance of a solar cell module. Therefore, shade analysis should be performed before installation in order to assess the financial viability of an investment as well as to optimize the location and orientation of a photovoltaic module. The process of accurate shade analysis however, is something that is currently only carried out by professionals using expensive equipment.

This bachelor’s thesis aims to develop the concept of a smartphone based shade analysis technique, which by using MATLAB’s predefined image processing should produce a shading diagram together with data ready for use as input in PV simulation software. The results should be comparable to those of professional shading analysis products.

As a result, the concept of a shading analysis technique based on panoramic image capturing using a smartphone and MATLAB image processing has been shown to work. The final analysis technique uses a smartphone captured panoramic image as input and provides two forms of output:

1. A shading diagram for easy and quick overview, providing information about which obstacles may cause shading at a specific time of day and year
2. An excel file that can be used as input in the PV simulation software System Advisor Model (SAM)
Sammanfattning

Solenergi är en snabbt växande energikälla. Allteftersom effektiviteten ökar och kostnaden minskar, blir solcellssystem allt mer attraktivt för små och medelstora konsumenter att investera i. Trots framsteg inom solcellsteknik, fortsätter även mindre skuggning att ge upphov till stora effektbortfall i solcellsmoduler. Därför utförs ofta skugganalys före installation, för att bedöma investeringens ekonomiska lönsamhet, likväl som för att optimera placeringen av solcellsmodulen. För närvarande är dock precis skugganalys något som endast utförs av yrkesverksamma med hjälp av dyr specialutrustning.

Detta kandidatexamsarbete syftar till att utveckla ett koncept kring en smartphone-baserad skugganalyse som med MATLABs fördefinierade digitala bildbehandling ska producera ett skuggdiagram tillsammans med data som kan användas i PV-simuleringsverktyg. Resultaten ska vara jämförbara med resultat från professionella produkter för skugganalys

Efter projektets genomförande kan det konstateras att konceptet med en skugganalyse som baserad på MATLABs bildbehandling och en panoramabild som tagits med en smartphone fungerar. Den slutliga analysetekniken använder sig av en panoramabild som indata och ger utdata i två former

1. Ett skuggdiagram för enkel och snabb överblick, vilket ger information om vilka hinder som skulle kunna orsaka skugga vid specifik tid
2. En excel-fil som kan användas som indata i PV-simuleringsprogrammet System Advisor Model (SAM)
# Table of Contents

Abstract ..................................................................................................................... 2

Sammanfattning ....................................................................................................... 3

1 Introduction .......................................................................................................... 5
  1.1 Background ..................................................................................................... 5
  1.2 PV cells and the effects of shading ................................................................. 7
  1.3 The sun path diagram and shading diagrams ................................................ 8
  1.4 PV simulation software .................................................................................. 9
  1.5 Current methods and products for shade analysis ......................................... 9
  1.6 Digital image processing and MATLAB ...................................................... 13
    1.6.1 Current use of image processing in shade analysis ............................... 13
    1.6.2 Digital images ....................................................................................... 13
    1.6.3 Digital image processing ....................................................................... 14
    1.6.4 MATLAB and the Image Processing Toolbox ....................................... 14

2 Problem description and objective ................................................................. 15

3 Method ................................................................................................................. 16
  3.1 Model design ................................................................................................. 16
    3.1.1 Image acquisition ................................................................................. 16
    3.1.2 Image processing .................................................................................. 17
    3.1.3 Sun path diagram superimposition ....................................................... 17
    3.1.4 Data manipulation ................................................................................ 18
    3.1.5 Restrictions ......................................................................................... 18
  3.2 Sensitivity analysis ....................................................................................... 19

4 Results ............................................................................................................... 20

5 Conclusion ......................................................................................................... 21

6 Discussion, comparisons and recommendations ............................................. 22
  6.1 Discussion ..................................................................................................... 22
  6.2 Comparisons ................................................................................................. 23
  6.3 Recommendations ....................................................................................... 24

7 References ....................................................................................................... 25
  7.1 Software references ...................................................................................... 25

8 Appendix ........................................................................................................... 26
  8.1 Methods for mitigating loss/damage caused by shading ............................. 26
  8.2 Obstruction matrix ....................................................................................... 27
  8.3 MATLAB code ............................................................................................ 28
1 Introduction

Solar energy can be converted into electricity using photovoltaic cells, offering social, economic and environmental benefits over fossil fuel generated electricity. Solar photovoltaic (PV) systems are continuously increasing in efficiency and recent advancements in manufacturing have decreased the initial cost of investing in a solar PV system, thus making it economical for many regions. Despite these recent advancements, shading still constitutes a major problem since even partial shading (Figure 1) drastically reduces the performance of a solar cell array (Zheng et al., 2013). To assess the financial viability of an investment in a PV system, it is important to estimate the generation losses caused by shading.

Figure 1. A solar PV system on a residential building with several solar cell arrays affected by shading.¹

1.1 Background

Increasing environmental consciousness and recent advancements in PV technology is making solar power an increasingly attractive investment opportunity for small and medium scale consumers. A solar PV system provides several attractive benefits (Chakraborty et al., 2013):

- Highly reliable energy source that can match well with daytime loads, especially for people in warm/sunny regions, thus providing a chance of self-sustainability
- Modular and easy to install
- No moving parts and no noise
- Pollution-free
- Low maintenance
- Possibility of making money by selling electricity back to the grid

Although the initial cost of investing has decreased in recent years, it is still critical to be able to make precise return on investment (ROI) estimations to assess the viability of an investment. Solar PV systems, unlike solar thermal panels, are sensitive to partial shading and highly dependent on receiving uniform sunlight across the entire array to reliably generate electricity (Lorenzo et al., 1994; Gao et al., 2009). Thus, in order to perform precise ROI estimations, it is critical to approximate the generation losses caused by shading. Many different sources of shading can affect the effectiveness of a particular installation. Such sources are:

- The earth’s axis of rotation relative to the plane of the earth’s solar orbit causes a 46.9° maximum solar altitude angle difference, known as declination (Duffie and Beckman, 2006) (Figure 2 a))
- Latitude-specific differences between seasons caused by earth’s roundness (Figure 2 b))
- Far-object shading, e.g. mountains and/or clouds that depend on the geography and weather conditions of a location (Figure 2 c))
- Near-object shading, e.g. trees and/or buildings (Figure 2 d))

![Figure 2. a) Solar declination at both maxima, ± 23.45° from the equator. b) Difference in spread of incoming solar rays depending on latitude. c) Far-object shading, caused by clouds. d) Near-object shading caused primarily by trees.](image-source)

---

2 Image source: http://www.astronomy-ags.dk/as/rw/season.htm
3 Image source: http://jh402is2104.blogspot.it/2011/02/uneven-heat-distribution-and-planetary.html
4 Image source: http://3.bp.blogspot.com/_x6pflQrTPHo/TMVCZ4h9FtI/AAAAAAAAABs/ Vs4M5zqX5dI/s1600/IMG_1923.jpg
5 Image source: http://www.greenbeltsolar.com/images/articles_images/solar_panel_shaded_roof.jpg
Performing shade analysis at the specific location of the installation can help to understand and assess the effects that the first, second and fourth sources of shading have on PV output. Unfortunately the third source of shading cannot easily be avoided without completely changing the location of the installation which creates a need for long power lines and in turn increases the initial investment and maintenance cost. After the analysis, possible alterations to the surrounding area or the installation itself could be made to optimize the effectiveness of the PV system. Such alterations could involve trimming vegetation or improving the orientation of the solar PV system.

1.2 PV cells and the effects of shading

A solar PV system consists of one or several solar PV modules, each consisting of multiple PV cells. The PV cells generate electrical energy directly from sunlight by taking advantage of the PV effect (Lorenzo et al. 1994). The most commonly used PV cell is constructed from p-type and n-type silicon (Figure 3). When these two types of silicon are placed in close contact with each other this creates a semiconductor material. When the semiconductor material is exposed to sunlight, electrons are knocked loose from the n-type silicon. If the two sides are connected, forming an electrical circuit, the moving electrons can be used as electric current (Lorenzo et al. 1994).

![Figure 3. An illustration of photovoltaic electricity generation.](Image source: Zheng et al., 2013)

Under normal conditions, when each cell in a solar module receive the same amount of sunlight they generate roughly the same amount electricity (there is always some difference in output due to variation in cell quality, as is expected from the manufacturing process). Under conditions with uneven shading, a shaded cell will allow current from other unshaded cells to pass through its parallel resistance, resulting in a reverse voltage and potentially a major drop in power output (Chakraborty, 2013). The unshaded cells are still generating power, but the shaded cell(s) will now absorb power instead (Figure 4). The shaded cell will receive a (high) current from the accumulated generation of all unshaded cells, leading to a high reversed terminal voltage, and thus a high absorbing power, on the shaded cell. This can lead to damage to the shaded PV cell, causing permanent generation loss to the system (Chakraborty, 2013; Zheng et al., 2013).
1.3 The sun path diagram and shading diagrams

A sun path diagram is essentially a plot of the sun’s position, or path, in the sky at different times of the day and usually also for different months. The plot contains the solar altitude (the sun’s vertical position or elevation over the horizon) and the solar azimuth angle (the angle between the sun’s horizontal position and a reference direction, usually north or south).

Scanning the surrounding environment at the site of a PV system installation creates a horizon profile, showing the elevation of each surrounding object at its specific azimuth angle. Overlaying the horizon profile with the sun path diagram produces a shading diagram (Figure 5), where possible sources of near-object shading can easily be identified (UO Solar Radiation Monitoring Laboratory, 2008).

Sun path data is free and easily accessible from many different online sources (for example from the University of Oregon, www.suncalc.net, www.sunearthtools.com and the US Naval Observatory).

Figure 5. Shading diagram showing solar elevation and azimuth of the sun together with the horizon outline of an object.\(^7\)

---

\(^7\) Figure credit: Zheng et al., 2013

\(^8\) Image source: https://lh4.ggpht.com/pPW7A2Nv0LYj673pGLwve5OE- qmRyQ9MwBj7hNLdDD9AF66eMHCW784wV-X0Tr__Hxca=1900-rw
1.4 PV simulation software

PV simulation software uses weather data and specifications of a PV system to perform power generation estimates, usually over a one year period. Shading diagrams can also be input, which reduces available solar radiation at specified times. The software can then estimate generation potential and predict if the system will have a net-positive return on investment.

There are a number of alternatives in terms of professional PV simulation software available on the market. Some of the most commonly used are

- **TRNSYS** is a graphically based software with powerful tools for simulation of the behaviour of many transient systems, among them biological processes or traffic flow in addition to energy and PV related systems (TRNSYS, 2014)
- **System Advisor Model (SAM)** uses the PV model from TRNSYS. It is developed by the US National Renewable Energy Laboratory (NREL) and provides a free alternative with a lot of functionality (SAM, 2014)
- **PV*SOL** from Valentin Software that comes in a basic, advanced and expert version, each adding more capabilities (PV*SOL, 2014)
- **PV Designer Software** from Solmetric offers energy calculations accounting for partial shading based on user created 3D environments. (Solmetric, 2014)
- **PV Syst** offers powerful tools for quick analysis of a system's behaviour and allows comparisons between different configurations given the same site. (PV Syst, 2014)

1.5 Current methods and products for shade analysis

The process of accurate shade analysis involves making on-site measurements, which are then used to render the surrounding environment of the particular installation, either as a 2D or 3D model. There are a few different methods to create these 2D or 3D models, and although almost all methods involve mapping the horizon in some way and then combining it with sun path data, each method has its own approach and its own advantages and disadvantages. Most methods will be covered in detail here but all have been compiled into a comprehensive list (see Table 1 below)

One device, produced by the company SolMetric, utilizes a specialized camera with a fisheye lens to capture an image of the entire surrounding environment. A sun path diagram is then overlaid onto the image, to create an estimate of the amount of solar access in that particular spot at that specific site (Figure 6). The data can then be analysed further using additional software, creating a full 2D or 3D environment with more exact estimations of the actual system output.

![Figure 6. A professional using the Solmetric SunEye 210 and the resulting picture of the analysis showing the surrounding area combined with sun path data for the location and an estimation of solar access.](http://www.solmetric.com/buy210.html)
SunPathFinder is both the name of a company and the name of another device that is used for shade analysis. Their method consists of using a highly polished, transparent, convex plastic dome in combination with an analogue sun path diagram (SolarPathFinder, 2014). The resulting picture will show both the surrounding area in combination with the sun path diagram (Figure 7). This method provides a cheaper alternative but offers no direct analysis or estimation of the generation losses. The picture taken with the solar pathfinder equipment must be analysed manually by a person or afterwards by using PV analysis software.

There are also a number of smartphone apps that provide more or less accurate shade analysis. Two examples are the Sun Shades app for android and the Solmetric iSV app for iPhone. Both use vastly different approaches for mapping the horizon and also differ greatly in other aspects. The Sun Shades app requires the user to map the horizon, either by using a set of crosshairs overlain in camera view or by simply pointing the phone at the horizon and then tracing it around 360°. The app uses the phone’s accelerometer, GPS, and compass to then calculate the outline of the horizon following the user’s movements.

One, potentially large, weakness of this method is that all of these instruments have a considerable margin of error, especially when used together. This often results in a horizon outline that is jagged or chopped as a result from fluctuating readings or simply the user’s inability to perform the measurement with sufficiently stable hands. The app then combines the traced outline of the horizon with sun path data to, among other things, calculate a rough estimate of the percentage of lost sun hours per day.

The Solmetric iSV app instead uses the camera of the smartphone together with compass and tilt sensors of the iPhone to provide an “augmented reality” view on the phones screen (Figure 8). The app does not map the horizon but simply provides this augmented reality view with sun path data overlain directly to give quick estimates of the timing and impact of near-object shading. While the Sun Shades app does provide the possibility of rough estimations and the option to collect and save the data in many ways, the Solmetric iSV does not provide any way to save or collect data.

---

10 Image source: https://www.solarpathfinder.com/PF
There are also some ways of performing shade analysis with only a computer. A rough shade analysis can be done simply with Google Sketchup. By measuring distances either in reality or with Google Earth, a rough 3D-sketch of the site and the surrounding environment can be created. Google Sketchup then supports sun and shadow effects that can be easily and dynamically changed or even animated within the program to see the timing and effect of any near-object shading.

There are also other methods, Solar Site Selector and PanoramaMaster (Figure 9). Solar Site Selector basically provides a small round stand where transparent acetate foils can be placed. The transparent foils have sun path diagrams printed onto them and by using an included compass for direction it provides the same capabilities, although analogue, as the SolMetric iSV app. The PanoramaMaster on the other hand provides a special bracket mount for tripods, which helps the user to take a perfect panoramic image with their own camera by rotating the camera a fixed number of degrees between each picture. The pictures must then be loaded into a computer, where the images are stitched together into a panoramic picture using included software. The panoramic image can then be used in PV simulation software.

Finally, although it is not the object of this thesis, it is worth mentioning that a number of different techniques for mitigating the losses/damage caused by shaded or damaged solar cells exist. Some of these are covered in the Appendix, Section 8.1.

Image sources:
11 https://lh6.ggpht.com/Gs8_OoX7651bmYgN9uIE-TFGT9h8TZ1rSfke_zqfC9XMDY9eHrK17TFY0nu-AUQ3CHdLw-l6900-rw (left) and http://ep.yimg.com/ca/I/yhst-77580361692593_2263_3022602 (right)
12 Image sources: http://www.solardesign.co.uk/images/shading2-2.jpg (left) and http://www.solardesign.co.uk/images/panoramamaster.jpg (right)
<table>
<thead>
<tr>
<th>Image capturing method</th>
<th>SunEye 210</th>
<th>Solar Pathfinder</th>
<th>Solar shade app</th>
<th>ISV iPhone solar app</th>
<th>Solar site selector</th>
<th>PanoramaMaster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital camera equipped with fisheye lens. Potential live video feed</td>
<td>None, requires external digital camera</td>
<td>360° panoramic image captured with the smartphone camera</td>
<td>Single (non-panoramic) image captured with iPhone camera</td>
<td>None, meant for direct manual analysis</td>
<td>Creates a 360° panoramic picture by using a special tripod mount together with software. Requires a separate camera</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sun path diagram overlay method</th>
<th>Digital. Automatically, instantaneously and continuously overlaying sun path diagram</th>
<th>Analogue</th>
<th>Digital</th>
<th>Digital</th>
<th>Analogue</th>
<th>Digital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>2,195.00€</td>
<td>215.77€</td>
<td>13.00€</td>
<td>4.49€</td>
<td>64.71€</td>
<td>294.14€</td>
</tr>
</tbody>
</table>

| Generation losses estimation | Direct, more exact estimation possible with further analysis using additional software | No direct estimation, later loss estimation possible with further analysis using additional software | Direct, more exact estimation possible with further analysis using additional software | None | None | No direct estimation, later loss estimation possible with further analysis using additional software |

| Advantages | • Fast overview • Quick estimations • Accurate • Easy to use | • No need for external equipment • Cheap • Quick estimations | • No need for external equipment • Fast overview • Cheap | • Fast overview | • Good site overview |

| Disadvantages | • Very expensive • Need separate unit | • Expensive • Bulky | • Accuracy depends on the user | • No estimations • Does not save data | • Expensive • No estimations • Analog | • Expensive • No estimations • Bulky |


All information relating to cost is taken from the respective developers. All prices not in € were converted using www.xe.com currency converter on 2014-03-13 19:31 UTC.
1.6 Digital image processing and MATLAB

Digital image processing is the automatic manipulation of a digital image using predefined algorithms. It is important to make this definition to separate it from digital image manipulation, which is the process of manually manipulating an image using image editing software like Adobe Photoshop, Google Picasa etc. using brushes or similar tools. It is also useful to differentiate between image synthesis and analysis. Image synthesis is the process of synthesising/creating an image from numerical values, whereas image analysis is the inverse process, extracting numerical data relevant to the context from image data.

1.6.1 Current use of image processing in shade analysis

Industry leading methods rely on image processing for horizon mapping, something that no smartphone app currently does. As mentioned in section 1.5, the Sun Shades app relies greatly on the user accurately tracing the horizon, which along with dependence on many of the phones positioning instruments leaves a large risk of error. Meanwhile, the Solmetric iSV does not provide a horizon-mapping feature at all.

If image processing could be utilized in a smartphone app, it might provide better accuracy for horizon mapping which when combined with the agility and ease-of-use of a smartphone app, would allow users to perform accurate shade analysis themselves.

1.6.2 Digital images

To understand digital image processing it is vital to understand what a digital image is. A digital image is a 2D representation of 3D objects, stored in a digital format. There are a number of different ways to represent this visual 2D information in digital form. First, any digital image is made up of individual pixels or picture elements (pels) (Marques, 2011). The information for each pixel could either be mapped individually (bitmap format) or as a function of its location (vector format) within the entire image. In bitmap format, one or more two-dimensional matrices of pixels are used to represent each pixel’s information individually. For example, colour images will often have three matrices of identical size, each containing information about either the red, green or blue content of each pixel. Vector format on the other hand relies on different drawing commands and functions to represent an image. Each format has its own advantages and disadvantages. Bitmap provides better quality and displays faster, while requiring more memory for storage purposes. Vector graphics on the other hand need less memory and can be resized or manipulated geometrically without introducing artefacts, which the bitmap format cannot (Solomon and Breckon, 2011).

There exists a large number of image file formats, each using a different method and/or algorithm to store the image information. Some common examples of image file formats are JPEG (Joint Photographic Experts Group), Microsoft Windows bitmap (BMP or DIB, Device Independent bitmap), GIF (Graphics Interchange Format) and TIFF (Tagged Image File Format). Most image file formats that use the bitmap format contain a file header together with the actual pixel data. The file header contains general information about the image, usually some signature indicating the file type together with other relevant information such as image dimensions and number of bits per pixel. For image processing purposes, the bitmap format is preferable because of its simplicity. No decompressing or rasterizing has to be done in order to access a specific pixel (Solomon and Breckon, 2011).
1.6.3 Digital image processing

Digital image processing is often divided into three different categories to differentiate between different levels of image processing. The three categories are; low level, medium level and high level of image processing. Low level processing involves primitive operations, usually primitive operations with one function applied to each pixel only based on its location. Such operations could include contrast or brightness alterations. Medium level processing entails attribute extraction, i.e. recognition of patterns, edges or contours in an image. The breakdown and/or classification of an image or image elements into meaningful numerical data constitute high level processing.

Furthermore, digital image processing usually involves several steps in a specific order to obtain a desired solution (Figure 10). The steps are (Marques, 2011);

1. **Image Acquisition** – the acquisition of the actual digital image
2. **Preprocessing** – improvement of image quality might be needed before further processing is possible or to simplify it. May include improvements to contrast and brightness or noise reduction
3. **Segmentation** – separating the main or relevant components from an image, e.g. foreground/background
4. **Extraction** – identifying specific features within an image, e.g. letters or patterns
5. **Classification** – if several relevant features have been extracted during step 4, classifying and sorting/ordering these might be necessary or desirable

In modern applications of image processing, a knowledge base is often implemented to help or speed up the process at each step. For example, the knowledge base could contain information about the colour of the sky, something that could speed up or be necessary during the segmentation process if the sky was to be separated from the rest of the image.

1.6.4 MATLAB and the Image Processing Toolbox

MATLAB (or MATrix LABoratory) is powerful software used for data analysis and visualization. MATLAB’s basic data type is the matrix and it also has extensive built in functions for manipulation of matrices and matrix operations, both of which make MATLAB highly suitable for use image processing (MATLAB, 2014). MATLAB also offers the option to add many different toolboxes, sets with predefined functions and computations, such as the Image Processing Toolbox (IPT) for image processing.

---

13 Image source: Marques, 2011
The IPT offers powerful predefined functions for working with images. Functions like \texttt{imread}, \texttt{imsave} and \texttt{imwrite} provide easy import/export capabilities which allow the user to work with images of many different file formats without ever worrying about things like rasterizing or decompression, no matter the file format in question.

There are also functions for image enhancement (e.g. \texttt{imadjust}, \texttt{imcontrast} or \texttt{histeq}) and powerful region of interest based (roi-based) functions that allow selection of regions of interest based on color (or other parameters) as well as powerful segmentation functions such as \texttt{activecontour}.

\section{Problem description and objective}

Currently, accurate shade analysis is a complicated and expensive process, carried out only by professionals using expensive equipment and software. There are also (less accurate) products targeted at individual consumers that want to do the shade analysis by themselves. Consumers are forced to choose between either accurate or affordable/simple shade analysis. Installation companies could also save money by letting their installation professionals use the smartphone app instead of equipping each one with, for example, a \textit{SunEye 210}.

A smartphone based shade analysis method will allow consumers themselves to perform shade analysis. Using image processing, the shade analysis could potentially prove accurate enough to make a generation loss estimation that reliably can be used as a basis for an investment decision.

The objective of this thesis is to develop a proof of concept for a smartphone based shade analysis method for PV installations. The analysis method should use smartphone camera technology together with MATLAB, using built in image processing and computations, to produce a shading diagram together with data ready for use in PV simulation tools as final output. The concept of the shade analysis method should be clearly described in a report ready to use as a basis for a future smartphone application.
3 Method

This section will outline and discuss the method and model design used in this project to create shading diagram using MATLAB-based image processing.

3.1 Model design

In order to obtain the final output, a shading diagram with data that can be used in PV simulation software, a comprehensive model was created to help with overall overview of the work process. A flowchart is a good way to visualize this process and to display all the steps necessary to achieve the objective (Figure 11).

3.1.1 Image acquisition

Image acquisition refers to the actual process of capturing the image meant for use in the image processing procedure. For the purposes of this project, it is important that the image displays as much of the surrounding area as possible, especially in horizontal direction. To achieve this with a smartphone, creating a panorama is the easiest way (as opposed to using multiple pictures). Creating a panorama is a user friendly and easy process that is built into most modern smartphones.

The smartphone created panoramas come with one inherent weakness. The vertical field of view is severely limited, something that may become a problem at certain sites, e.g. sites located in major cities with skyscrapers or other tall buildings. The problem consists of two separate parts; first that the entire building or obstruction will not fit into the cameras view and therefore it won’t be accurately represented in the shading diagram or the data file. Second, the panoramic image will not fit a full sun path curve since too little of the hemisphere is captured in the image. The first part can unfortunately not be avoided or mitigated but the second can be avoided by padding the image to artificially create more sky above the horizon to fit the full extent of the sun path curve. This process is covered in detail below (see section 3.1.2).
To minimize the risk of any problems being introduced during the image acquisition process and in order to ensure that the sun path data is correctly superimposed it is of paramount importance to follow these three steps.

- The image should cover a full 360°, no more and no less.
- The image should have magnetic south in the centre of the image. It is therefore easiest to use the smartphone’s compass to find a landmark that is directly north. The image can then be captured with this landmark just in the left corner of the smartphone’s screen and the smartphone is then rotated clock-wise until the same landmark is just visible in the right corner of the smartphone’s screen.
- The true horizon (sea level horizon) should be at the bottom of the image and no part of the horizon may exceed the height of the image. This is easier if the phone is held vertically (as normal) instead of horizontally.

3.1.2 Image processing

After the panorama image has been captured, it is first imported into the MATLAB environment. When the image has been imported, the image’s dimensions are determined. To make use of MATLAB’s contour detection functions, the image is then converted to greyscale and image contrast is enhanced to bring out details of the horizon outline as well as to blend the sky together with any potential clouds to avoid making them part of the foreground during segmentation.

The image is then divided horizontally into segments that are processed individually with MATLAB’s contour detection function for segmentation purposes. The image is converted into a binary image where a pixel value of 0 represents elements of the horizon and a 1 represents elements of the sky.

As mentioned in the previous section, the panoramic image is not tall enough to fit a full sun path diagram and must therefore be extended. Fortunately, this is easily done by simply creating a new matrix of pixels with the value 1 and padding the segmented image’s matrix with the new matrix until the original image matches the vertical size of the sun path diagram.

3.1.3 Sun path diagram superimposition

To begin the process of sun path diagram superimposition, the relevant data must first be obtained. This is done by first determining the location of the site, and then acquiring the sun path data online from the website sunearthtools14. The data is then imported into MATLAB and dimensions corrected to the same size as the panorama image to be able to make a correct superimposition. After the sun path diagram has been superimposed onto the new image file created during image processing, corrections to colour and contrast are made to make the final shading diagram easier to read.

---

3.1.4 Data manipulation

In order to output the final shading diagram to PV simulation software, the format of the data must fit the import format of the simulation software. Using the shading diagram image, data is extracted into a 91-by-361 matrix, covering the full azimuth range from 0 to 360 degrees and the full elevation range from 0 to 90 degrees. Each element in the matrix is either assigned a value of zero or one, depending on if that specific part if part of the sky (background) or the horizon outline (foreground), see section 8.2 – Obstruction Matrix. A zero now represents background and a one now represents foreground, a necessary change from before (see section 3.1.2) to match the input format of SAM. Unfortunately, limitations in Microsoft Excel won’t allow MATLAB to write data matrices that exceed 256 columns or strings longer than 1024 characters (otherwise the following error code is displayed: 0x800A03EC. To work around this, only every second azimuth degree is exported to the excel file.

3.1.5 Restrictions

Since the goal of this thesis is not to provide a finished smartphone app but rather prove the concept of a smartphone based shade analysis method that uses image processing, a number of restrictions will be introduced. The restrictions introduced are:

- The panorama image will be captured using a smartphone
- The smartphone used will be a Samsung Galaxy S4
- MATLAB and the IPT will be used for all computations and all image processing
- No actual estimations of generation loss or ROI projections will be made
3.2 Sensitivity analysis

Since the method in this project is based on image processing, it is mainly sensitive to changes or variations in the input panoramic image. The method is also sensitive in regards to the direction of the centre of the panoramic image and it is therefore important to follow the three steps explained in Section 3.1.1, Image acquisition. Except for these orientation related problems, there are also a number of factors of the surrounding environment that could affect the quality of the output shading diagram.

Dark clouds could potentially trick the software into masking it as foreground and incorrectly making it part of the horizon outline. In the same way, brightly painted building, lakes/rivers, snow and other reflective surfaces may after image analysis be subtracted from the horizon outline.

Pictures taken in mountains pose two different challenges. First, there may be snow covering part or all of the mountains. Snow covered mountain tops are easily recognized as background (or sky) by the program because they are “connected” to the rest of the sky and the program uses contours in greyscale images to differentiate between background and foreground. Secondly, images taken at a height that is well above the true horizon (i.e., sea level horizon) may incorrectly show the height of the sun’s path and the elevation of obstructions after analysis if the image is not captured correctly, with the horizon at the bottom of the screen.

The figure below (Figure 12) shows a small section of a larger panoramic image. As can be seen in the first picture (a) the mountain tops are snow-covered. As the image is processed, the snow and the sky gradually look more and more the same and finally both become part of the background after segmentation is complete.

![Figure 12](image.png)

*Figure 12. a) Original image with snow-covered mountains. b) Grayscale image. Clouds and snow part of mountains still visible. c) Contrast-adjusted image. d) Final image after contour processing.*
4 Results

The concept of a shading analysis technique based on MATLAB image processing and a panoramic image captured using a smartphone has been shown to work. The final analysis technique uses a panoramic image as input and provides two forms of output:

1. A shading diagram for easy and quick overview, providing information about which obstacles may cause shading at a specific time of day and year
2. An excel file that can be used as input in the PV simulation software SAM

Below is an example of a panoramic image fully processed in the MATLAB program Shading_Analysis.m (see the Appendix, section 8.3) to provide a shading diagram (Figure 13) together with an excel file as final output. The panoramic image was taken in Turin, Italy (coordinates 45.065547, 7.698714) the 16th of March 2014.

![Panoramic image of Turin](image1)

![Image turned into grayscale](image2)

![Grayscale image after contour enhancement](image3)

![Processed image with superimposed sun path data](image4)

**Figure 13.** a) Panoramic image of Turin, b) Image turned into grayscale, c) Grayscale image after contour enhancement d) Processed image with superimposed sun path data. A red mark shows each full hour on the sun’s path. Azimuth 180° marks south.
5 Conclusion

The concept of using a smartphone based image acquisition method and MATLAB image processing to produce an input for PV simulation has been proved to work.

With the completion of this project it can be concluded that although the MATLAB programme and image acquisition method do not provide a viable alternative to current methods and products at this point, it does provide a very powerful platform for the development of a new and better alternative product. The MATLAB image processing proved to be a powerful tool for prototyping; new ideas could quickly be tested and/or modified to enhance the final product (shading diagram and obstruction matrix) and it should prove easy to develop the prototype into a future final product ready for end-consumer use.
6 Discussion, comparisons and recommendations

In this section, conclusions and comparisons relevant to the results will be presented together with some recommendations for future reference.

6.1 Discussion

MATLAB’s image processing capabilities are many and powerful. So powerful that proving that the concept itself worked proved easier than expected, although the speed of the program suffers greatly if image size is increased. Balancing the right image size and right amount of image enhancement turned out to be the biggest challenge. If the image was too heavily enhanced before segmentation, too much detail was lost while if the image was not enhanced enough, segmentation couldn’t be performed reliably and background and foreground were mixed together completely.

After the amount of enhancement had been balanced, another challenge became trying to make the program run faster, as it was very slow at first (taking a little over 40 minutes to run). A lot of improvements were made and the average run time of the final program is roughly 6 minutes.

There are still things that could be done to speed up the process (see section 6.3 below). Processing time also varies greatly from image to image, partly because of different image sizes and partly, it seems, because some of MATLAB’s predefined functions seem to struggle with certain images.

It is also clear that the analysis technique is highly dependent on the quality of the input image. Using an image where the true horizon not properly aligned and/or the centre of the image not aligned toward the south will cause the sun path diagram to be superimposed at a position offset to its correct position. This in turn will lead to inaccurate estimations of the effects of any shading.

There is also an inherent weakness to the whole process (mentioned earlier in section 3.1.1) which is based on the smartphone captured panoramic image. Although part of the problem could be resolved (and section 3.1.2), the main part remains unsolved. Any obstructions that are too high to fit into the smartphone camera’s vertical field of view will simply be “cut off” at the top (Figure 14).

![Shading diagram](image)

**Figure 14.** Final output shading diagram. Input panoramic image was captured with the phone held horizontally to illustrate the effect of tall obstructions on the output.
6.2 Comparisons

Unfortunately, it was not possible to compare the results of this analysis technique with any results obtained using the only image processing based shading analysis product on the market, the SunEye 210.

It was however possible to compare the results with the only shading analysis app on the market, Solar Shades. Both shading diagrams (Figure 15 & Figure 16) were created based on the same site in Turin, Italy (coordinates 45.0655, 7.698714). The Solar Shades app required several tries to get a horizon outline that looked acceptable. Two panoramic images had to be captured because the first one was not captured properly (too far below horizon). The horizon outline created with the Solar Shades app, which is based on readings from accelerometers in the smartphone, is far too jagged and uneven to make any estimation. It has one advantage though; since it is not based on image processing there is no risk of any part of the horizon being cut off.

Figure 15. Shading diagram created with the Solar Shades app.

![KEX test - Sun Path](image)

Figure 16. Shading diagram created with the MATLAB program Shading_Analysis.m
6.3 Recommendations

Because the concept could be proved to work and the program produced satisfactory results, the main recommendations will regard the speed of the program.

Any MATLAB code can be written in a number of ways. The focus of this project has been to create a working program to develop a proof of concept and see if that concept would work. This concept could however be further developed and improved upon in a number of ways:

- By rewriting the code using vectorization, the speed of the program could probably be increased. The program contains a lot of for-loops, some of which could probably be rewritten in a more efficient way.
- By utilizing MATLABs parallel computing toolbox, the program might be rewritten in a more efficient way. Using functions in the toolbox would, for example, allow MATLAB to process multiple steps in a for-loop simultaneously. We were not able to try this due to not having access to MATLABs parallel computing toolbox.
- Since MATLAB is based on interpreting commands, and thus eliminating the need for compilation, it’s an excellent development tool to prototype new ideas or techniques. Unfortunately, it also makes it rather slow. If the technique was implemented using another programming language or if the MATLAB code was compiled into C++ code it could greatly increase the speed.
- In order to avoid the problem with tall obstructions being “cut off”, a future smartphone application based on this concept should make sure to either use multiple panoramic images together, captured at different heights or use some lens attachment to widen the field of view of the smartphone camera. Both should solve the problem with “cut off” obstructions but both come with some drawbacks. Using several panoramic images will increase the image size and therefore increase processing time and using any lens attachment will introduce distortion that must be accounted for during image processing. Lens attachments could in a worst case scenario cause the smartphone’s built in panorama-generation to fail partially or completely to render a good panorama.
7 References


7.1 Software references

MATLAB (2014), by MATrix LABoratories, information and software available at www.matlab.com

PV Designer Software (2014), by Solmetric, information and software available at www.solmetric.com/

PV*SOL (2014), by Valentin Software, information and software available at www.valentin-software.com

PV Syst (2014), PV Syst, information and software available at www.pvsyst.com

SAM (2014), by NREL, information and software available at https://sam.nrel.gov/

TRNSYS (2014), by TRNSYS, information and software available at www.trnsys.com
8 Appendix

8.1 Methods for mitigating loss/damage caused by shading

The most commonly used and researched technique uses bypass diodes to keep the current from unshaded cells from passing through the parallel resistance of shaded and/or damaged cells. When a cell is shaded, the bypass diode is activated by the current from unshaded cells and “shuts off” the shaded cell. While this does drastically mitigate the generation loss and can prevent damage to the shaded cell, the shaded cell will still not produce any electricity. This technique is also expensive to use because it is very situation-dependent, meaning that the solar module needs to be designed knowing exactly what cells will be shaded, thus requiring lots of simulation/designing work using expensive professional software. However, if shading cannot be avoided, it can be used to avoid heat damage to one or more cells that might keep it/them from working at all (Zheng et al., 2013).
8.2 Obstruction matrix

A section of the obstruction matrix, with zeros coloured in blue and ones coloured in green for better distinction between the two.
8.3 MATLAB code

%% Shading_Analysis.m
% Created by Simon Wakter & Fredrik Wikerman on 2014-04-02
% Last changed on 2014-05-05

% Bachelor of Science Thesis EGI-2014
% A novel shade analysis technique for photovoltaic systems
% Students: Simon Wakter & Fredrik Wikerman
% Supervisor: Nelson Sommerfeldt

% This program was written in MATLAB version 2013a
% The program uses functions from the Image Processing Toolbox (IPT) and
% the export_fig function from the MATLA File Exchange.

% Matlab IPT available from http://www.mathworks.com/products/image/
% Export_fig available for free from
% http://www.mathworks.com/matlabcentral/fileexchange/23629-export-fig

%% DESCRIPTION

% This program uses a 360° panoramic colour image as input together with
% sun path data from http://www.sunearthtools.com/ in .xlsx file format.
% The program's final output is:
% 1) A shading diagram showing the horizon profile as a black
% foreground with the sky as an all white background with sun path data
% superimposed for each month of the year.
% 2) An .xls-file with a (rounded) obstruction matrix, using either a
% "1" (no obstruction) or "0" (obstruction) to indicate if there is an
% obstruction that may cause shading depending on the sun's position.
% The file will indicate this for every second azimuth value ranging
% from -180° to 180° and elevation values ranging from 0° to 90°.

%% Initialising workspace environment

clear all
close all
clc
tic

warning on verbose
% Turns on extra line to display MSGID of any displayed warning messages
warning off images:initSize:adjustingMag
% Turn of warning for image display when image resolution exceeds
% screen resolution
%% Load the image and manipulate it to bring out the foreground.

I = imread('Panorama_Turin_5.jpg');
I = imresize(I,0.1);
I_gray = rgb2gray(I);
I_a = imadjust(I_gray, [0.45 0.8],[],1);
I_a(end-10:end,:) = 0;

figure(10)
imshow(I)
title('Orginal image')
figure(11)
imshow(I_gray)
title('Gray image')
figure(12)
imshow(I_a)
title('Adjusted image')

%% Segmentation of adjusted image into foreground and background

step = round(length(I_a)/10);
bw = [];
for l = 1:step:length(I_a)
    if l >= length(I_a)
        mask = ones(size(I_a(:,l:end)));
        temp_bw = activecontour(I_a(:,l:end), mask, 2000,...
            'Chan-Vese', 0.15);
        if zeros(size(temp_bw(end,:))) == temp_bw(end,:)
            bw = [bw temp_bw];
        else
            temp_bw = imcomplement(temp_bw);
            bw = [bw temp_bw];
        end
    else
        mask = ones(size(I_a(:,1:l+step)));
        temp_bw = activecontour(I_a(:,1:l+step), mask, 2000,...
            'Chan-Vese', 0.15);
        if zeros(size(temp_bw(end,:))) == temp_bw(end,:)
            bw = [bw temp_bw];
        else
            temp_bw = imcomplement(temp_bw);
            bw = [bw temp_bw];
        end
    end
end

figure(13)
imshow(bw)
title('Manipulated image')
%% Importing and formatting sun path data

% Sun path data for every 5th minute
[NUM, TXT, RAW] = xlsread('Turin_sunpath_5.xlsx', 'B2:VE366');

Elevation = RAW(:,1:2:end);
Azimuth   = RAW(:,2:2:end);

TEST = Elevation(1,1);
for n = 1:365
    for k = 1:24
        TF_ele = strcmp(Elevation(n,k), TEST);
        TF_azi = strcmp(Azimuth(n,k), TEST);

        if TF_ele == 1
            Elevation(n,k) = {0};
        end

        if TF_azi == 1
            Azimuth(n,k) = {0};
        end
    end
end

Elevation = str2double(Elevation);
Azimuth   = str2double(Azimuth);

Max_Azi = max(max(Azimuth));
Min_Azi = min(min(Azimuth));
Max_Ele = max(max(Elevation));
Min_Ele = min(min(Elevation));

Div_Azi = Max_Azi - Min_Azi;
Div_Ele = Max_Ele - Min_Ele;

%% Plot sun path data and manipulate plot image to match image size.

figure(1000);
hold on
axis([Min_Azi Max_Azi 0 Max_Ele]);
for j = 15:30:365
    % Every 30th day, roughly in the middle of each month
    plot(Azimuth(j,:), Elevation(j,:), 'r', 'LineWidth', 0.2)
end
hold off

scale_width = round(size(bw,2)/360);
Total_width = Div_Azi*scale_width;

export_fig SunPathDiagram.png -transparent
close 1000

SPD = imread('SunPathDiagram.png', 'png');
Resize_SPD = imresize(SPD, [nan Total_width]);
Resize_SPD_D = im2double(Resize_SPD);
[A1, A2, A3] = size(Resize_SPD_D);

Resize_BWC = ones([A1 size(bw,2)],'double');
Resize_BWC(end-(size(bw,1)-1):end,:) = bw;

Azi_step = round(size(Resize_BWC, 2)/Div_Azi);
Ele_step = round(size(Resize_BWC, 1)/Div_Ele);
Resize_Azi = Azimuth*scale_width;
Resize_Ele = Elevation*Ele_step;

Reversed = flipud(Resize_BWC(:,:,1));

figure(14)
imshow(Reversed)
hold on, axis on, grid on

axis([0 size(Resize_BWC, 2) 0 size(Resize_BWC, 1)])
set(gca,'YDir','normal')

x_mark = linspace(0,size(Resize_BWC,2),19);
y_mark = linspace(0,size(Resize_BWC,1),9);

set(gca, 'XTick', x_mark), set(gca, 'YTick', y_mark)

x_scale = 360/length(x_mark); y_scale = Div_Ele/length(y_mark);

x_label = linspace(0,360,19);
y_label = linspace(0,Max_Ele,9);
set(gca, 'XTickLabel', x_label)
set(gca, 'YTickLabel', y_label)

for j = 15:30:365 \% Every 30th day, roughly in the middle of each month

    plot(Resize_Azi(j,end:-1:1), Resize_Ele(j,end:-1:1),'color',rand(1,3),'
\'LineWidth', 0.5)

end
legend('jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep',
\'okt', 'nov', 'dec')
title('Shading diagram')
xlabel('Azimuth (Degrees)'), ylabel('Elevation (Degrees)');

for j = 15:30:365
    plot(Resize_Azi(j,end:-12:1), Resize_Ele(j,end:-12:1),'r+')
end

hold off
%% Find pixel values of horizon outline, convert to azimuth and elevation
% degrees to export to Excel

Obstruct_pixel = []; 
H_step = round(size(Resize_BWC,2)/360); 
temp_h = 0; 
for h = 1:H_step:size(Resize_BWC,2) 
    for v = 1:1:size(Resize_BWC,1) 
        if Resize_BWC(v,h)==0 && size(Obstruct_pixel,2)~=h && h~=temp_h 
            temp_pixel = [v,h]; 
            Obstruct_pixel = [Obstruct_pixel temp_pixel]; 
            temp_h = h; 
        end 
    end 
end 
Obstruct_pixel(1,:) = size(Resize_BWC,1) - Obstruct_pixel(1,:); 

Ele_pixel = size(Resize_BWC,1)/90; 
Azi_pixel = size(Resize_BWC,2)/360; 
Elevation_obstruct = Obstruct_pixel(1,:)/Ele_pixel; 
Azimuth_obstruct = Obstruct_pixel(2,:)/Azi_pixel; 
Obstruction = [Elevation_obstruct; Azimuth_obstruct]; 
Obstruction_round = round(Obstruction); 
Obstruction_matrix = zeros(91,361); 
for l = 1:360 
    Obstruction_matrix(end:-1:end-(Obstruction_round(1,l) -1),l) = 1; 
end 

ty_Ele = (90:-1:0)'; 
xx_Azi = -180:2:180; 

xlswrite('Obstructions.xls', y_Ele, 1, 'A2:A93'); 
xlswrite('Obstructions.xls', xx_Azi, 1, 'B1'); 
xlswrite('Obstructions.xls', Obstruction_matrix(:,1:2:end), 1, 'B2'); 

toc