Smartphone-based image analysis for calculating the solar potential of house roofs

Chris Grieger

September, 2012 - reviewed in June 2014

Contents

1 Project description **3**

2 Synopsis **4**

3 Sensors **5**
3.1 GPS 5
3.2 Compass 5
3.3 Accelerometer 5
3.4 Camera 7

4 Single image analysis **7**
4.1 OpenCV 7
4.2 Method 9
4.3 Taking the photo 9
4.4 Preprocessing 10
4.5 Edge & corner detection 11
4.6 Measuring pixel coordinates 11
4.7 Roof area calculation 11

5 Results **12**
5.1 Roof area 12
5.2 Solar potential 13

6 Appendix **15**
6.1 A 15

1 Project description

Smartphones are becoming pervasive and cheaply available for people around the world. While mainly developed for communication purposes, those mobile devices are equipped with sensors like cameras, GPS, compass, accelerometers and gyroscopes. This project aims to evaluate the suitability of the iPhone 4 for planning photovoltaic installations. The quality of the data produced by the iPhones sensor systems will be measured and evaluated in the context of acquiring positional data.

Through photogrammetric evaluation of single images, the area of a roof should be determined. This area will be fed through an already available mathematical model to calculate the possible solar potential of that roof. The developed software will use OpenCV. OpenCV is an open source framework for computer vision available for different systems. It provides bindings for various popular programming languages like C, C++, Java and Objective-C.

2 Synopsis

Study has shown that the iPhone 4's available sensor data has no practical use for acquiring accurate positional measurements. Using the accelerometer to measure distance, requires previous filtering and a model of restrictions (see section 3). Acquiring the positional data of a roof corner through the current GPS position is not accurate enough because of possible shielding, standard error and sensor accuracy.

As a part of this project a physical model of a house was constructed to test the single image evaluation in a lab environment. The next step was to evaluate the roof area of a real house. Single image evaluation has shown to be accurate enough for field use. Using the approach to be shown, it was possible to determine the roof area with an accuracy of $\pm$ 5%.

3 Sensors

This section aims to give a short overview of the iPhone 4's available sensors.

3.1 GPS

The iPhones Location API uses both the signal from the global positioning system (GPS) and trilateration[1] through known locations of radio masts and Wi-Fi access points. Accuracy[2] can vary between 1m and more than 10m.

3.2 Compass

For the compass the following measurements have been made by [3]

	absolute [°]	**relative** [%]
max. deviation	24,74	8,00
min. deviation	-10,26	-6,00
range	35,00	14,00
average	7,82	-0,09
std. deviation	12,20	3,11

Table 1: Compass accuracy

3.3 Accelerometer

The integrated accelerometer is normally used to detect gestures. Application can register actions when the phone is panned or tilted. The accelerometer can not be used to reliably determine the phone's travelled distance.

In a simple model, the phone's axial alignment must not change at all during a distance measurement. Even when resting in the same place, the sensors measure very small acceleration values in x, y and z direction. This leads to a very large absolute error when trying to calculate the distance traveled. Calculation of the absolute traveled distance is done by Integrating the measured acceleration values twice. The following figures show the results of an experiment gathering the sensor data for 37 seconds while the phone was standing still.

[1] `http://en.wikipedia.org/wiki/Trilateration`
[2] `http://www.cmtinc.com/gpsbook/index.htm#chap8`
[3] [Luh 2011]

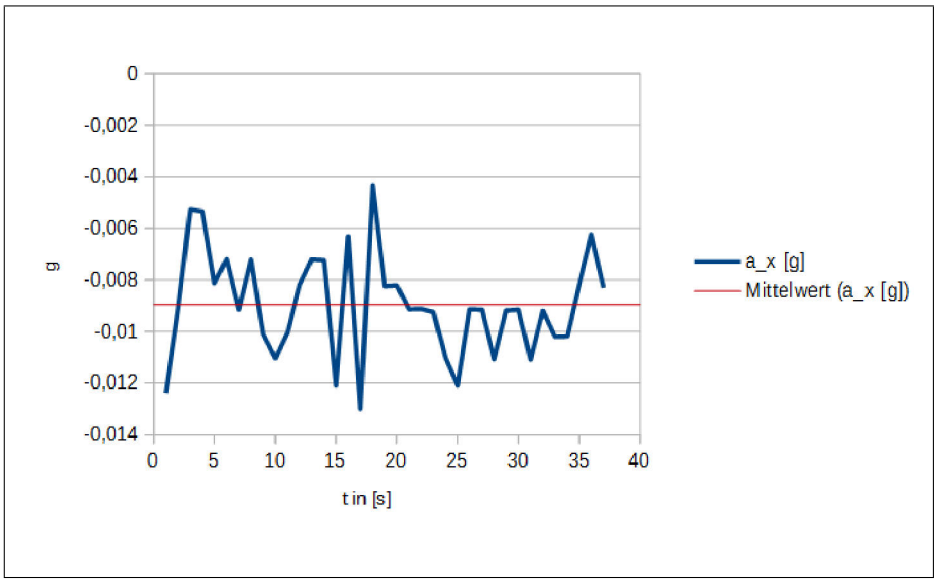

Figure 1: Measured acceleration in x direction (iPhone 4 resting)

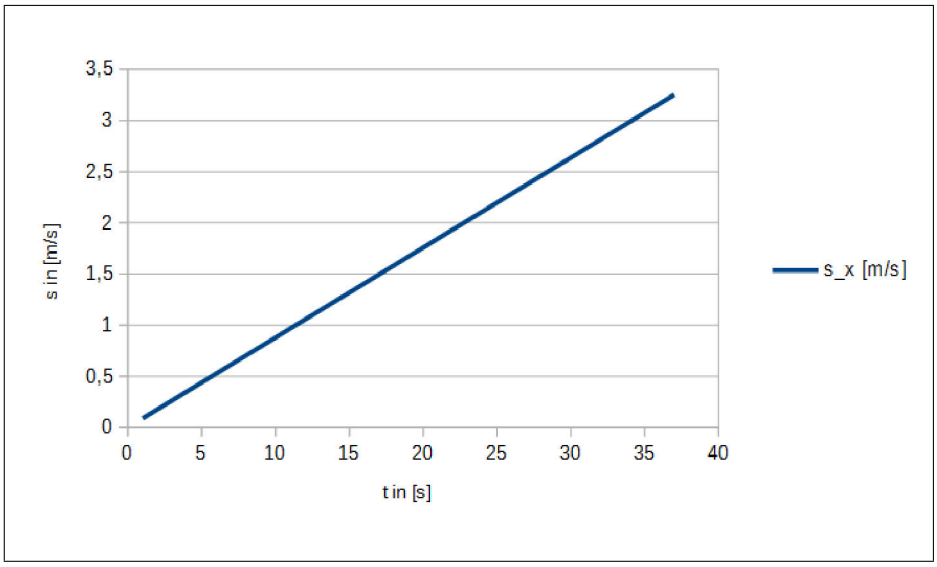

Figure 2: Velocity in x direction (iPhone 4 resting)

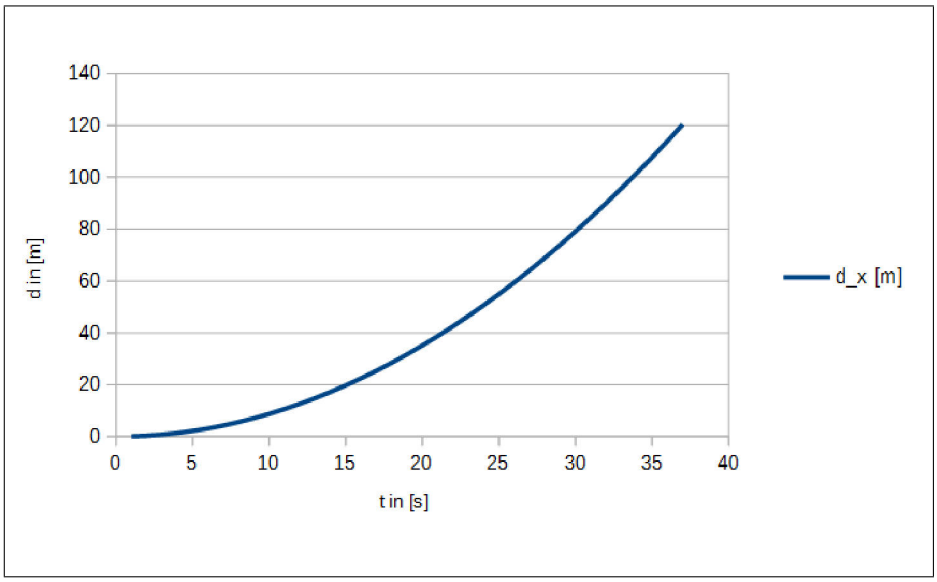

Figure 3: Travelled distance in x direction (iPhone 4 resting)

After 10 seconds a travelled distance of ∼8m in x direction has been registered by the acceleration sensor.

3.4 Camera

The technical facts for the iPhone 4 camera are as follows:

> The iPhone 4 features an additional front-facing VGA camera, and a backside-illuminated 5 megapixel rear-facing camera with a 3.85 mm f/2.8 lens and an LED flash. The rear-facing camera is capable of recording HD video in 720p at 30 frames per second. Both cameras make use of the tap to focus feature, part of iOS 4, for photo and video recording. The rear-facing camera has a 5 digital zoom.[4]

4 Single image analysis

Using photogrammetric single image analysis, the roof area of a house should be determined.

4.1 OpenCV

All required computer vision algorithms are implemented by using the OpenCV framework. OpenCV provides a comprehensive library containing various implementations for camera calibration, image rectification and pattern- / object recognition.

> OpenCV is released under a BSD license and hence its free for both academic and commercial use. It has C++, C, Python and

[4] `http://en.wikipedia.org/wiki/IPhone_4#Camera`

> Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing.[5]

Aptogo provides a package to include OpenCV inside of iOS applications. It contains a sample project and a version of OpenCV compiled and linked for use on iOS devices.

[5] `http://opencv.org/`

4.2 Method

The flow of the application will be as follows:

1. Take photo
2. Preprocessing
3. Edge filtering
4. User selects the window corners
   (a) Select two points for horizontal edge
   (b) Select two points for vertical edge
   (c) Input edge length in centimetres
5. User selects roof corners
   (a) Select two points for horizontal edge
   (b) Select two points for vertical edge
6. Calculate roof area

4.3 Taking the photo

A photo is taken with the integrated camera directly inside the application. All four corners of a roof's side have to be visible on the photo.

Figure 4: Sample roof photo

4.4 Preprocessing

The photo is scaled to a width of 256 pixels, retaining the original aspect ratio. Color depth is reduced to 8 bits and the image is converted to a grey scale image.

Figure 5: Sample roof photo - greyscale representation

In the next step a Gaussian blur filter with a 7x7 kernel is applied to the image.

4.5 Edge & corner detection

Edges are detected using a Canny filter[6]. The ratio of minimal and maximal threshold is 3:5. The kernel size of the underlying Sobel filter operation[7] is 3x3.

Figure 6: Sample roof photo - detected edges

4.6 Measuring pixel coordinates

Pixel coordinates of a corner are measured when the user touches the screen at the location of visible corner (see flow at page 4). The most distinctive corner around the touch location is detected by the OpenCV function `goodFeaturesToTrack();`. This function uses the Harris-Corner-Detection[8] algorithm.

4.7 Roof area calculation

The pixel / centimetre[9] ratio of both vertical and horizontal window edges is calculated. It is necessary to calculate the px/cm ratio for both edges, since transformation of 3D to 2D coordinates leads to unequal distortions. Internal camera parameters are not calculated. The rectification of lens errors is foregone in favour of simplicity of use of the application.

The ratio of pixels per centimetre is the quotient of the Euclid distance of two window edge image coordinates and the associated edge length.

$$verticalRatio = \frac{\sqrt{(Pv1_x - Pv2_x)^2 + (Pv1_y - Pv2_y)^2}}{l_{windowVertical}}$$

$$horizontallRatio = \frac{\sqrt{(Ph1_x - Ph2_x)^2 + (Ph1_y - Ph2_y)^2}}{l_{windowHorizontal}}$$

[6] `http://en.wikipedia.org/wiki/Canny_edge_detector`

[7] `http://en.wikipedia.org/wiki/Sobel_operator`

[8] `http://en.wikipedia.org/wiki/Corner_detection#The_Harris_.26_Stephens_.2F_Plessey_.2F_Shi.E2.80.93Tomasi_corner_detection_algorithm`

[9] $[px/cm]$

The length of a roof's edge is calculated from the product of the Euclid distance between two roof corners and the associated px/cm ratio.

$$l_{roofVertical} = \frac{\sqrt{(PvRoof1_x - PvRoof2_x)^2 + (PvRoof1_y - PvRoof2_y)^2}}{verticalRatio}$$

$$l_{roofHorizontal} = \frac{\sqrt{(PhRoof1_x - PhRoof2_x)^2 + (PhRoof1_y - PhRoof2_y)^2}}{horizontalRatio}$$

The roof's area $[cm^2]$ is the product of the vertical and horizontal roof edge lengths.

$$A = l_{roofVertical} \times l_{roofHorizontal}$$

Using the coordinates of long edges that are close to the camera produces good results.

5 Results

5.1 Roof area

The method was first tested on a physical model of a house.

Figure 7: Model of a house

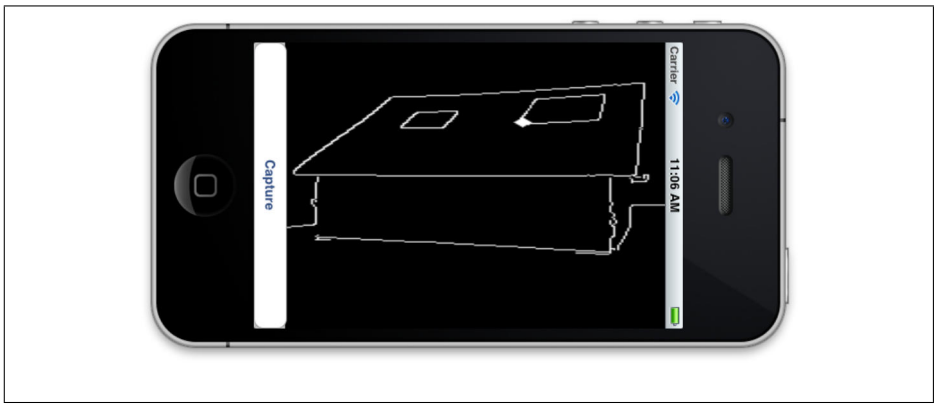

Figure 8: Model of a house - Edge detection result

All calculation results can be found in Appendix A. In the test results the calculated roof area differs by $1,38\%$ from the physical model. During tests with a real house (see page 4), a difference of $2,78\%$ was calculated.

5.2 Solar potential

A roofs solar potential can be calculated with the SMA Solarchecker App. It determines the theoretic solar potential by passing the average sun hours for the current GPS location, the compass orientation of the roof, tilt of the roof and the user-calculated roof area through a mathematical model. The application also calculates the possible return per year and the amout of time until amortisation. From an ecological point of view, the application also displays the possible savings of CO^2 [10].

[10] This is the product of the yearly energy output and $0,541\ kg[CO^2]$

The following images show the results for the real roof tested (21.07.2012):

Figure 9: Estimated output

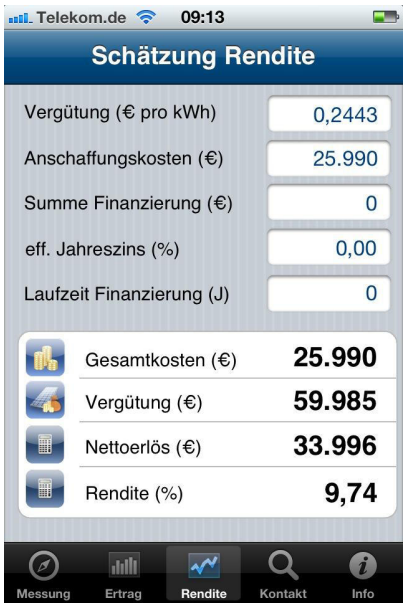

Figure 10: Estimated returns

6 Appendix

6.1 A

Mobilephone 4 Solarpotential - Ergebnisse

OHNE PERSPEKTIV TRANSFORMATION	P1_x [px]	P1_y [px]	P2_x [px]	P2_y [px]	Länge [cm]	Länge [px]	Verhältnis [px/cm]	Berechnete Länge [cm]	Differenz [cm]	Differenz %
Unterkante Fenster	71	71	67	34	8,5	37,2155881318568	4,3783044861008	8,5	0	0,00%
Seitenkante Fenster	71	71	52	63	9,5	20,6155281280883	2,17005559243035	9,5	0	0,00%
Unterkante Dach	116	184	118	14	34,5	170,011764298827	-	38,8305027296617	4,33050272966173	12,55%
Oberkante Dach	50	136	38	12	34,5	124,579292019179	-	28,453775294675	6,04622470532503	17,53%
Seitenkante Dach	118	14	38	12	30	80,0249960949702	-	36,876933648205	6,87693364820504	22,92%
						Dachfläche [cm^2]	Dachfläche berechnet [cm^2]	Abweichung [cm^2]	Abweichung %	
						1035	1049,28798358266	14,2879835826645	1,38%	
MIT PERSPEKTIV TRANSFORMATION	P1_x [px]	P1_y [px]	P2_x [px]	P2_y [px]	Länge [cm]	Länge [px]	Verhältnis [px/cm]	Berechnete Länge [cm]	Differenz [cm]	Differenz %
Unterkante Fenster	163	178	221	171	8,5	58,4208866759141	6,87304549128401	8,5	0	0,00%
Seitenkante Fenster	163	178	165	74	9,5	104,019228991567	10,9493925254281	9,5	0	0,00%
Unterkante Dach	13	332	253	331	34,5	240,002083324291	-	34,9193212279254	0,41932122792545	1,22%
Oberkante Dach	12	23	251	20	34,5	239,018827710287	-	34,7762615587801	0,27626155878012	0,80%
Seitenkante Dach	12	23	13	332	30	309,001618118741	-	28,2208914696535	1,77910853034654	5,93%
						Dachfläche [cm^2]	Dachfläche berechnet [cm^2]	Abweichung [cm^2]	Abweichung %	
						1035	981,417103170615	53,5828968293846	5,18%	
Echtes Haus	P1_x [px]	P1_y [px]	P2_x [px]	P2_y [px]	Länge [cm]	Länge [px]	Verhältnis [px/cm]	Berechnete Länge [cm]	Differenz [cm]	Differenz %
Unterkante Fenster	93	109	93	99	80	10	0,125	80	0	0,00%
Seitenkante Fenster	94	105	69	129	270	34,6554469023269	0,12835350704565	270	0	0,00%
Oberkante Dach	25	238	57	81	1490	160,227962603286	-	1281,82370082629	208,176299173712	13,97%
Seitenkante Dach	31	237	125	171	749	114,85643212289	-	894,844517820892	145,844517820892	19,47%
						Dachfläche [cm^2]	Dachfläche berechnet [cm^2]	Abweichung [cm^2]	Abweichung %	
						1116010	1147032,91149729	31022,9114972909	2,78%	