Improve 3D models from 2D images

T. Kostelijk
mailtjerk@gmail.com

October 4, 2011

1 Comments on this thesis

Dear Frans,
I write this in English so Isaac can read it also. This in an update of my thesis chapter. Isaac read it and gave me feedback. His feedback and all feedback you gave me are incorporated except for:
-the test on different datasets
(the dataset is not ready yet)
-references
(I'll do this on the end)
-parametervalues Houghline
(I'll do this on the end)
-large introduction
Explanation:I expanded the introduction a bit more then my previous version but I think you want more background information. I do this in a few chapters earlier which is not included in this document. Therefore some backreferences to sections also contain a questionmark.

Furthermore, I processed many small changes but a few big changes are:
-I explicitly defined all assumptions.
-I wrote a new subsection about alternative roof-types.
-I updated the way line segments are associated with building parts and illustrated it with Figure 10.

Because this new method of associating lines with walls is so much better and more intuitive then my old method (see section 2), I am considering to discard the old method.
However, I could also keep the old method and use it as a test against the new one. What is your advice about this?

Would you give me feedback about the level of detail, where do I go in to much and where is more detail desired?
Isaac told me to make the future work section smaller, do you agree with this and if yes would you tell me which part you think we should drop?
Furthermore my discussion section is very short, I think it is best to add more discussion when the other datasets and results are ready. What do you think?
I have tried to write down some mathematics but didn't do it everywhere. Could you give me feedback where to put (more) formulas or symbols and where not?
I think I need to put my work in context with the work of others. But on this specific part I can't find any work. How do I deal with that?
Do I remember it right that you are leaving for 1 month beginning at half september?
Would be nice if we can discuss your feedback before then, but that will depend on your schedule.
Please let me know when the feedback is done and lets make an appointment.
I think Isaacs presence will not be necesarry, right Isaac?

I am available most of the days. Please let me know when you are available.

Thanks in advance for reading my thesis chapter and giving me useful feedback.

Kind regards,
Tjerk Kostelijk
mailtjerk@gmail.com

2 Skyline detection

2.1 Introduction

If we take a regular image where both sky and earth are present, there is often a clear seperatation between them. This separation is called the skyline. The detection of this skyline has proven to be a very succesful computer vision application in a wide range of domains. In this domain it is used at urban images to provide a countour of a building. This contour is later used to provide 3D information about the scene.
The organisation of this chapter is as follows. First we give a summary of some related work on skyline detection. Next we explain how we developed a new robust skyline detection algorithm. Finally we present some results.

2.2 Related work

Skyline detection is used in a wide range of domains. In [8], Castano et all present a nice introduction of different skyline detection techniques, these are listed below.

2.2.1 Horizon detection for Unmaned Air Vehicles

In this domain, the horizon detector for Unmaned Air Vehicles takes can take advantage of the high altitude of the vehicle and therefor the horizon can be approximated to be a straigt line. In [], an algorithm is proposed which is based on the idea that this straight line spearates the image onto two regions that have a different appearence. They use color as a measure of appearance and generate two color distributions: one for the sky and one for the ground. The distributions are optimized using bisection search on a criterium that uses the covariance and the eigen values of the distributions. The line that best separates the two distributions is determined to be the horizon.
This work is not applicable for detecting a building countour as the straight line assumption doesn't work. But it needs to be mentioned that from this idea some inspiration on line fitting is done because the building has walls that have straight lines.

2.3 Method

2.3.1 Situation and assumptions

Before we present the method let's define the situation and make some assumptions.

definition: skyline in urban scene A skyline in an urban scene is a set op points of the length w (where w is the width of the size of the image) where each point describes the location of the transition from the sky to an object (e.g. a bulding) which is mounted on the earth.

The first question arises: how are we going to detect this transition. In general the color of the sky is very different then the buildingcolor. A colorbased: using an edge detector would be an intuitive decision.
However, the sky and building itself also contain edges (e.g. clouds and windows) so how do we determine the right edge? The number of possible edges could be decreased by thresholding the intensity of the edge but it would still be a difficult task to determine the right edge.

To solve this problem we draw an assumption that is inspired on. Instead of using the sharpest edge we take the most upper sharp edge and classify this edge as the skyline.
Top sharp edge assumption The first sharp edge (seen from top to bottom) in the image presents the skyline.

2.3.2 Related algorithm

To put our work in context, we first describe a related skyline detection algorithm as presented in [9].

To increase the difference between sharp and vague edges (and let sharp edges stand out more and vague edges disappear) the images are converted to Gaussian smoothed images. The smoothed image, which is a collection of w x h pixels, is first divided in $\#w$ columns. Next, each column produces a new column that stores its vertical derivatives. This is called the smoothed intensity gradient. The values of this column are high when a big change in color happens (e.g. an edge is detected) at that location in the image. The system walks through the values of a column starting from the top. When it detects a pixel with a gradient higher then a certain threshold it stores its y-value (the location of the heighest sharp edge of that column) and continues to the next column. The result is a set of y coordinates of length w, that represents the skyline.

2.3.3 Improved algorithm

Taking the smoothed intensity gradient is the most basic method of edge detection and has the disadvantage that is is not robust to more vague edges. It is not surprising that the algoritm in []was used in an interactive system where the user refines the result. Our aim is to develop a autonomous skyline detector, the only user interaction that we allow is to provide the system some parameters. We will now discuss the adaptations that we developed wrt the related algorithm.

The column based approach of the related algorithm seem te be very useful and is therefor unchanged. The related algorithm uses the smoothed intensity gradient as a method to detect edges. Because of the accuracy disadvantage of this method we took another approach in detecting edges. Because we used

Matlab as our implementation platform, we where able to test different edge detecting types. The following edge detecting types where tested.
The output of the different edge detection techniques was studied on an emprical basis and the Canny edge detector came with the most promising results. This is probably because Canny is the most 'intelligent' edge detector. It uses two thresholds, to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges. In table a Matlab buildin edge detectors are listed with their method explanation.

Edge detecting type	method
Sobel	The Sobel method finds edges using the Sobel approximation to the derivative. It ret
Prewitt	The Prewitt method finds edges using the Prewitt approximation to the derivative. It
Roberts	The Roberts method finds edges using the Roberts approximation to the derivative. I
Laplacian	The Laplacian of Gaussian method finds edges by looking for zero crossings after filte
zero-cross	The zero-cross method finds edges by looking for zero crossings after filtering I with a
Canny	The Canny method finds edges by looking for local maxima of the gradient of I. The

Because the optimal edge detector type can be scene dependend, it can be set by the user as a parameter in our functions.

The Canny edge detector outputs a binary image, therefor the column inlier threshold is set to 1, which means that it finds the first pixel that is white. This is, as in the related algorithm, done from top to bottom for every column in the image.

Because we know we are looking for sharp edges we improved the algorithm by introducing two preprocessing steps. First the contrast of the image is increased, this makes sharp edges stand out more. Secondly the image undertakes an extra Gaussian blur, this removes a large part of the nois. Note that dependend on the edge detector type this could mean that the image is blurred twice.

The system now has several parameters which has to be set manually by the user:

- contrast,
- intensity (window size) of Gaussian blur,
- Sobel edge detector threshold,

If the user introduces a new dataset these parameters needs to be configured as the image quality and lightning condition are scene dependend.

The sky and the earth are separated by a skyline in images. The detection of this skyline has proven to be a very succesful computer vision application in a wide range of domains. In this domain it is used to provide a countour of a building.
This contour is in a next step used to refine a 3D model of this building.
The organisation of this chapter is as follows. First related work on skyline detection is discussed, then a new algorithm of the skyline algorithm is described and finally some results are presented.

3.2 –Related work

A lot of related work on skyline detection is done and it is used in a wide range of domains. [8] yields a good introduction of different skyline detection techniques, these are listed below.

3.2.1 Cloud detection for Mars Exploration Rovers (MER)

Mars Exploration Rovers (MER) are used to detect clouds and dust devils on Mars. In [1] their approach is to first identify the sky (equivalentl, the skyline) and then determine if there are clouds in the region segmented as sky.

3.2.2 Horizon detection for Unmaned Air Vehicles (UAV)

In this domain, the horizon detector for UAVs can take advantage of the high altitude of the vehicle and therefor the horizon can be approximated to be a straigt line. This turns the detection problem into a line-fitting problem. Ofcourse this work is not applicable for detecting a building countour as the straight line assumption doesn't work. But it needs to be mentioned that from this idea some inspiration on line fitting is done because the building countour has straight line segments.

3.2.3 Planetary Rover localisation

In [9] they use the skyline detection in planetary rovers, their approach is to combine the detected skyline with a given map of the landscape (hills, roads) to detect its current location. The advantage of their technique is the simplicity and effectiveness of the algorithm which makes it suitable for this project. A

big drawback is that it is geared toward speed over extremely high accuracy because it is interactive system where an operator refines the skyline.
As mentioned in the introduction, in this project we use the skyline to extract the building contour to eventually update a 3D model which is a brand new purpose of skyline detection. There is no user interaction present, and the accuracy is a matter of high importance. This makes it different from existing skyline techniques and caution should be taken by using existing algorithms. From the related work the Planetary Rover localisation [9] seemed to fit most on this project. Therefor method [9] is used, but as a basis, and a custom algorithm with higher accuracy is developed. This is explained in the next section.

3.3 –The Algorithm

3.3.1 –The original algorithm

The skyline detection algorithm as described in [9] works as follows: The frames are first preprocessed by converting them to Gaussian smoothed images. The skyline of a frame is then detected by analysing the the image columns seperately. The smoothed intensity gradient is calculated from top to bottom. This is done by taking the derivative of the gaussian smoothed image.
The system takes the first pixel with gradient higher then a threshold to be classified as a skyline element. This is done for every column in the image. The result is a set of coordinates of length W, where W is the width of the image, that represent the skyline.
Taking the smoothed intensity gradient is the most basic method of edge detection and has the disadvantage that is is not robust to more vague edges. This is not surprising as it its purpose was a interactive system where the user refines the result. It is clear that an optimization is needed.

3.3.2 –The optimization

The column based approach seem te be very useful and is therefor unchanged. The effectiveness of the algorithm is totally depended of the method of edge detection and the preprocessing of the images. The original algorithm uses the smoothed intensity gradient as a way of detecting edges. This is a very basic method and more sophisticated edge detection algorithms are present.

To select a proper edge detector, a practical study is done on the different Matlab build in edge detection techniques. The output of the different edge detection techniques was studied and the Sobel edge detector came with the most promising results. The Sobel edge detector outputs a binary image, therefor the column inlier threshold method is replaced by finding the first white pixel. This is as the original algorithm done from top to bottom for every column in the image.
To make the algorithm more precise, two preprocessing steps are introduced.