Automated Detection of Buildings on Aero Images

Leonid Novotortsev
Keldysh Institute of Applied Math RAS
Moscow, Russia
torets13@gmail.com

Alexey Voloboy
Keldysh Institute of Applied Math RAS
Moscow, Russia
voloboy@gin.keldysh.ru

ABSTRACT

One of the challenging problems in photogrammetry is extracting of three-dimensional objects from aero images, in particular, extraction of different kinds of buildings. All methods that provide satisfactory results are rather time consuming and process data quite long. In the paper we propose a method that detects areas on aero images that might contain a building behind them. Our method allows reducing the amount of data which should be processed by more complex algorithms. This leads to reducing of the total time spent on extraction process.

Keywords
computer vision, line extraction, building extraction.

1. INTRODUCTION

Photogrammetry deals with making measurements from photographs, especially for recovering the exact positions of surface points and reconstruction of the whole scene or specific object types. At this point there are a lot of methods that provide satisfactory results in most cases. But still there are problems that cannot be solved by general methods with required accuracy or speed. One of them is the detection and reconstruction of buildings from aero images.

There are several approaches to that problem. There is an approach, which proposes to use special data for building recognition such as LIDAR [Rot02a], [Soh07a]. Results, provided by those methods, have high detection and low miss percentage, but in many cases LIDAR data is not available.

Another approach is to use structural, contextual, and spectral information from satellite photos [Jin05a] and aero images. There are several modern methods, which based on this approach that provide good results (recognition with probability ~85%) [Ok13a], [Gha14a], [Sin14a]. But there is a general limitation to this approach. In some cases it is impossible to detect buildings, which have the same colour and have no visible shadow.

To overcome this problem it is possible to use photos of the same scene, which have been shot from a bit different views. That way the object that is non distinct in one image should be clearly visible in the other. By using multiple images it is also possible to extract and utilize three-dimensional information about the scene. In order to process multiple images more efficiently special methods are needed.

The first approach is to build DEM (Digital Elevation Model) and process it in order to extract buildings. General methods of extracting buildings [Bru97a], [Cor97a] and [Gir98a] are rather rough. For that reason, methods that use prior information about building shape are applied. For example there is a method, which detects only flat roofs [Bre95a]. Such methods provide accurate results but are too limited.

The other approach based on extracting linear objects from images. The first step is to detect and extract lines, then match them and apply plane sweep strategy [Bai00a]. But this method rather complex, since it requires matching lines on several images (it is recommended to use 5 or 6 images). Processing such amount of data takes quite long. This problem could be solved by preprocessing each image and limiting the area to which this method is applied.

Since the main purpose of this paper is to propose preprocessing algorithm, we will refer to method, described above as “complex algorithm” or “complex method” (because of its comparable complexity to proposed method).

In order to get regions which might contain buildings it is possible to use methods, which were designed to detect buildings on single image [Jin05a], [Ok13a], [Gha14a], [Sin14a]. But these methods are rather complex themselves and they utilize completely different approaches. This way the advantage in computation time which we have got from preprocessing is decreased. In order to avoid this it is better to use more simple method for preprocessing. It is even better if some information acquired by preprocessing can be used by complex method.

The general idea of this paper is to detect straight lines on image, then unite them into contours and select those which might contain a building within.

It is important to notice, that our goal is not to detect building edges themselves but to find areas, which contain buildings. Then a complex algorithm will process much less data. Also that complex method can reuse straight lines and segments detected by proposed method. This way the amount of additional computations (compared to computations made by complex algorithm) on preprocessing step will be rather small.

2. GENERAL IDEA

In this paper we propose the usage of the fact that most buildings have linear edges. Therefore, it is convenient to use line extraction methods in order to achieve our goal, because such limitation allows to easily and accurately analyze the data. The whole algorithm can be divided into following steps:

1. Edge filter
2. Line extraction
3. Combine lines into contours
4. Analyze contours and leave only those, which might belong to the buildings.

In that scheme the first and the second steps are also done by complex algorithm and only the last two steps are additional. This way it is possible to pass lines, extracted on the second step to complex method in order to prevent repeated computations. It is also noticeable that the last two steps are rather simple since they process mostly vector data rather than raster data

3. EDGE FILTER

There are few methods that detect edges on images. Among them, Sobel filter [Sob73a] and Canny filter [Can86a] are noticeable.

Sobel Filter

The Sobel filter was designed for edge detection algorithms and creates an image which emphasizes edges. Technically, it is a discrete operator, computing an approximation of the gradient of image channel or intensity. The operator uses 3x3 kernels for horizontal and vertical directions, which are applied to each point of the image:

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \times A \qquad (1)$$

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \times A \qquad (2)$$

Where A is the source image, G_x and G_y are images which at each point contain the horizontal and vertical derivative approximations, $\times$ is 2-dimensional convolution operation.

From these values the gradient magnitude is calculated:

$$G = \sqrt{G_x^2 + G_y^2} \qquad (3)$$

The result of the Sobel operator is a 2-dimentional map of gradient at each point. The higher the gradient value the more likely there is an edge at this point of the image.

But in our task it is necessary to strictly determinate, if there is an edge for all points in image. Just using a threshold for values from Sobel filter is not enough.

Canny filter was proposed to overcome such problems as thinning edge and removal of false edges, created by image noise.

Canny Filter

Canny filter can be broken down to a four steps.

The first step is to apply a Gauss filter. On the second step the Sobel filter is applied and the edge direction angle is calculated from equations (1), (2) and rounded to one of the 4 values: 0, 45, 90, or 135 degrees.

$$\Theta = \operatorname{atan2}(G_y, G_x) \qquad (4)$$

On the third step non-maximum suppression is applied to “thin” the edge. The algorithm for each pixel in the gradient image is:

1. Compare the gradient magnitude of the current pixel with the edge strength of the pixel in the positive and negative gradient directions
2. If the edge strength of the current pixel is not the largest compared to the other pixels in the mask with the same direction the value will be suppressed.

After that, pixels with a gradient magnitude larger than certain threshold (often called high threshold) are marked as strong edge. A pixel is marked as weak edge if gradient magnitude is larger than the second threshold (low threshold) but smaller than high threshold. Pixel is suppressed if the gradient magnitude is lower than the low threshold.

On the last step all weak edge pixels that don’t have strong edge pixel in their neighborhood are suppressed.

In this paper we propose to use improved Canny edge detector, proposed in [Bao05a], since this algorithm provides even more precise results.

4. LINE EXTRACTION

The next step of our method is to extract lines and segments form edge map we received on previous step. And one of the most widely used techniques for locating straight lines is Hough transform [Hou62a]. The main idea of the Hough transform is to map image pixels, defined by their position (x, y) to the ρ–θ parameter space corresponding to all possible lines through the point, using the following formula:

$$r = x\cos\theta + y\sin\theta, \qquad (5)$$

where ρ is the perpendicular distance of the line to the origin, and θ is the angle between a normal to the line and the positive x axis. Usually the parameter space is partitioned into adjoining rectangular cells using a predefined resolution in both the ρ and θ dimensions.

The accuracy of the Hough transform received much attention due to the errors resulting from discretization of parameters [Ngu08a, O'Rp81a, Li86a].

An interesting approach is suggested in [Du10a]. In this paper it is proposed not to process straight line segments but rather their neighborhood and search for intersections of these neighborhoods in a voting process. That approach provides much better results than a plane voting approach.

In this paper we propose to use the following approach to straight segment detection. First we divide the image into equal square cells (except for those, which are placed at the border of the image). In each cell the Hough transform is applied. Finally, straight line segments from neighbouring cell are merged if they are located on the same line.

As the result of the described algorithm we will get a set of straight line segments, corresponding parts of linear objects. But at this point there are no relations between these segments, so it is necessary to unite them.

5. COMBINING STRAIGHT LINE SEGMENTS

First we will define two parameters, which are be used in the merging algorithm. The first parameter we will call "expected distance between buildings" or EDBB for short. The second parameter we will call "expected building size" or EBS. These parameters will define how lines will be merged. The first parameter has the following meaning: if distance between objects is shorter than EDBB value, then these objects don't belong to the same building or don't belong to any building at all. Technically, this is average linear size of buildings in scene. EBS is set in such a way that objects that are further than this value, then they either belong to different building or other type of objects. Technically, this is the size of the smallest object which is classified as building. In this paper we propose to set EDBB and EBS manually. The possibility of automated definition of these parameters will be studied in future researches.

There are 3 different types of segment layout, which will be analyzed separately:

1. Segments are positioned on the same line
2. Segments are positioned on intersecting lines
3. Segments are positioned on parallel lines

After segment analysis and merging closed contour finalizing is performed. On this phase all lines, that were not included into closed contours or located inside one of them, are marked as rejected. But they might be reused again in methods, which

Segments on the Same Line

Sometimes a single strait line edge is divided into two or more separate edges. This might occur because of low image quality, shadows or low contrast. In this case these segments must be united.

Usually the task is solved by using a threshold. That approach is not correct since a single threshold is not enough to accurately determinate if segments should be merged.

In this paper we suggest to use two thresholds. If the distance between the segments is lower than the first threshold (EDBB) and shorter than one of the segments, then these segments should be merged. If the distance is larger than the second threshold (EBS) or longer than both segments, then segments shouldn't be merged. Otherwise additional analysis should be done.

First we define a region of interest. Coordinates of closest ends of segments are interpreted as left, top, right and bottom border of a rectangle, representing that region. Then the rectangle is expanded for 2 pixels in each direction in order to detect edges at the border of the region.

After that, two thresholds are selected for the Canny filter based on the histogram of that rectangle (the using absolute minimum or maximum is not effective, because of random deviations produced by noise). Then the Canny filter is applied. After that, the number of pixels, which were marked as edges by the Canny filter, are counted in the neighborhood of the "lost" segment (the one between two involved segments). If this number is close to the high threshold then these two segments should be merged.

Segments on Intersecting Lines

This case is similar to the previous one. Just as in the previous subsection, there are three cases to consider, based on distance between the nearest ends of segments. Also the same methods are applied to determinate if segments should be merged.

One of the differences is the fact, that segments (one or both) might contain a point of intersection. In this case, the segment that has an intersection point is divided into two segments at that point.

In some cases false angles between segments might appear after merging. This problem is solved in the following way. If the angle between segments is close to 180 degrees (e.g. they are nearly positioned on the same line) an addition check is performed. Then, if the length of one completed (extended to point of intersection) segment is much shorter than the length of the other one, they can be considered as single straight line segment.

Segments on Parallel Lines

This type of layout can represent four kinds of building layouts.

The fist kind is parallelism of random edges that are not related in any way. This type is characterized by significant difference in edge length. In this case the edges are left as is.

The second and the third kinds of layout represent parallel edges of the same building or of different buildings accordingly. These two types are distinguished and processed with method used in previous subsections (two thresholds and the Canny filter).

The last one is duplication of building an edge (for example if there is a drainpipe at the roof of the building). If lines are positioned close to each other and only one of them is part of some contour, then they should be merged into one and the contour should be expanded accordingly.

6. FILTER CONTOURS

At this step of the described method we filter those closed contours which were created in previous steps. Also all straight line segments, which are not part of some contour, are ignored (but they will still be passed to complex algorithm).

In this method we propose to use two criteria for the filter. The first one is the limitation of closed contour area. It should be at least more, than EBS*EBS. A second criterion is the shape of close contours. They should have a convex n-gon shape.

These two criteria are quite simple, so there is a lot of space for further research here.

7. RESULTS

In Figure 1 and Figure 2 the source images are presented. These images are shot from airplane when it flew in different directions (strips). We applied the described method to those images. The results of region detection are shown in Figure 3 and Figure 4 accordingly.

Besides buildings the selected regions include shadows from these buildings and some other objects. Two regions are in fact just a parking lot. But that inaccuracy is acceptable, since the described method is supposed to decrease the amount of data passed to much more complex algorithms.

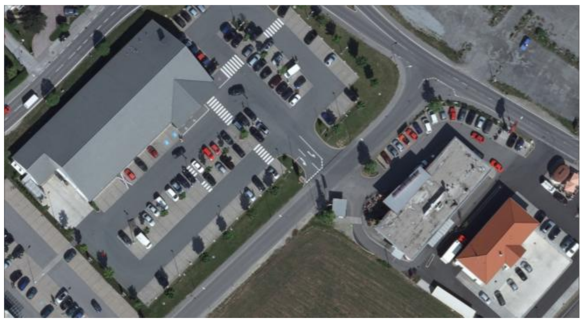

Figure 1. Source image, the first strip

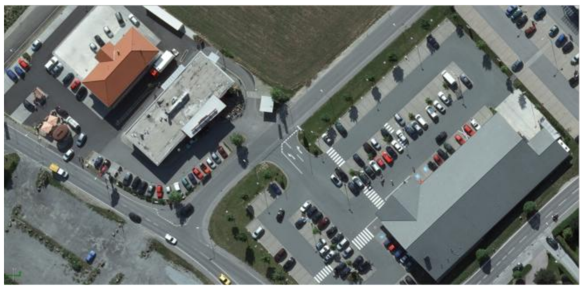

Figure 2. Source image, the second strip

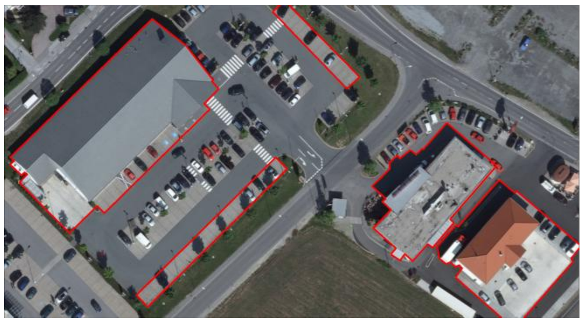

Figure 3. Contours produced by proposed method from Figure1.

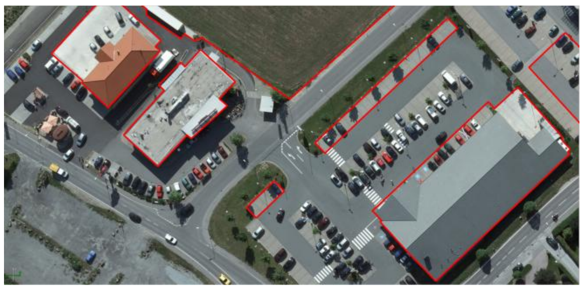

Figure 4. Contours produced by proposed method from Figure 2.

The average detection quality of described method is rather low: around 71%. But that is compensated by the fact that if the building which have not been detected in one image, would be detected in the other. Since complex method processes all images, detection of each building at least in one image is enough. That way the detection rate will be nearly the same as if only complex method [Bai00a] is applied to set of images, but the amount of data processed is much smaller.

8. CONCLUSION

The problem of detection and extraction of buildings is quite a challenging problem in computer vision and photogrammetry in particular. But existing methods are either not accurate enough or have high computational complexity. Of course the last type of algorithms is preferable. But in photogrammetry it is common to use aero images which may be larger than 10000 by 10000 pixels. Complex algorithms will process such amount of data too long. So we present a method that will decrease the volume of input data for them.

On presented example our method selected all buildings without missing a single one. Besides, given the fact that in aero images buildings occupy approximately 20-60% of images (depending on territory, where images were shot), our method will give around 40-80% boost, which is a very good result. The described algorithm is original.

9. REFERENCES

[Bao05a] Bao P., Zhang D., Wu X. Canny edge detection enhancement by scale multiplication. IEEE transactions on pattern analysis and machine intelligence, *27*(9), pp. 1485-1490, 2005.

[Du10a] Du S., van Wyk, B. J., Tu, C., & Zhang, X. An improved Hough transform neighborhood map for straight line segments. IEEE Transactions on Image Processing, 19(3), pp. 573-585, 2010.

[Hou62a] Hough P. V. C. Method and Means for Recognizing Complex Patterns, U.S. Patent 3069654, Dec. 18, 1962.

[Ngu08a] Nguyen T. T., Pham X. D, and Jeon J. An improvement of the Standard Hough Transform to detect line segments. IEEE International Conference on Industrial Technology, pp. 1-6, 2008.

[O’Ro81a] O'Rourke J. Dynamically Quantized Spaces for Focusing the Hough Transform. IJCAI, 81, pp. 24-28, 1981.

[Li86a] Li H., Lavin M. A., Le Master R. J. Fast Hough transform: A hierarchical approach. Computer Vision, Graphics, and Image Processing, 36(2), pp. 139-161, 1986.

[Can86a] Canny J. A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 6, pp. 679-698, 1986

[Cor97a] Cord M., Paparoditis N., Jordan M. Dense, reliable and depth discontinuity preserving DEM computation from HRV urban stereopairs. International Archives of Photogrammetry and Remote Sensing, 32, pp. 49-56, 1997.

[Bru97a] Brunn A., Weidner U. Extracting buildings from digital surface models. International Archives of Photogrammetry and Remote Sensing, 32(3) sect. 4W2, pp. 27-34, 1997.

[Gir98a] Girard S. et al. Building detection from high-resolution color images. Remote Sensing – International Society for Optics and Photonics, pp 278-289, 1998.

[Bre95a] Berthod M. et al. High-resolution stereo for the detection of buildings. Automatic Extraction of Man-Made Objects from Aerial and Space Images. – Birkhäuser Basel, pp. 135-144, 1995.

[Bai00a] Baillard C., Zisserman A. A plane-sweep strategy for the 3D reconstruction of buildings from multiple images. International Archives of Photogrammetry and Remote Sensing, 33(B2) part 2, pp 56-62, 2000.

[Rot02a] Rottensteiner F., Briese C. A new method for building extraction in urban areas from high-resolution LIDAR data. International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences, 34(3/A), pp 295-301, 2002.

[Soh07a] Sohn G., Dowman I. Data fusion of high-resolution satellite imagery and LIDAR data for automatic building extraction. ISPRS Journal of Photogrammetry and Remote Sensing, 62(1), pp. 43-63, 2007.

[Jin05a] Jin X., Davis C. H. Automated building extraction from high-resolution satellite imagery in urban areas using structural, contextual, and spectral information. EURASIP Journal on Applied Signal Processing, 2005, pp 2196-2206, 2005.

[Sob73a] Sobel I. and Feldman G. A 3 × 3 Isotropic Gradient Operator for Image Processing. R. Duda and P. Hart (Eds.), Pattern Classification and Scene Analysis, pp. 271–272, 1973

[Ok13a] Ok A. O.. Automated Extraction of Buildings and Roads in a Graph Partitioning Framework. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences II-3/W3, pp 79-84, 2013.

[Gha14a] Ghaffarian S., Ghaffarian S. Automatic building detection based on supervised classification using high resolution Google Earth images. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 40(3), pp 101-106, 2014.

[Sin14a] Singhal S., Radhika S. Automatic Detection of Buildings from Aerial Images Using Color Invariant Features and Canny Edge Detection. International Journal of Engineering Trends and Technology(IJETT) 11(8), pp 393-396, 2014.