Learning Windows

Ibraheem A. Alhashim
Department of Computing Science
Simon Fraser University
Burnaby, BC
iaa7@sfu.ca

Abstract

Detecting windows in urban environments is an essential task when reconstructing buildings from photographs. Recent work on city wide reconstruction has been focusing on capturing the details on building facades using both images and LIDAR data. However, segmenting and detecting windows in practice remains a manual and tedious task. In this report, we apply techniques used in face detection to the problem of detecting windows from a single photograph.

1 Introduction

Building facades differentiate in their appearance based on many factors including style, location, age, and many others. The most significant elements are the windows which can provide cues for other attributes such as building height, width, or sections. Being able to detect window structures from a photograph can help in many applications including building detection, location identification, reconstructions and other similar visualizations. A defining feature of windows that could be exploited is the fact that they typically align with the natural upright direction for humans. Unlike other challenging cases of object detection, a window's appearance might slightly change due to light conditions, distortions from input device (e.g. lens distortions), or obstruction from street level surroundings. Most methods exploit features such as collinear lines and grid structures in order to reconstruct building facades, however, it might be impractical to do so for large datasets. The ability to detect windows efficiently can also help in aligning data during acquisition. In this report, we explore the problem of window learning and detection by applying a well known method for efficient face detection that uses a boosting technique.

1.1 Related Work

Schindler and Bauer [7] proposed a model-based building reconstruction method from several photographs. The cameras are calibrated and the resulting images are aligned to construct a 3D point cloud for reconstruction of the planes in a building. The windows in their methods are detected by sweeping the extracted planes for density changing features and then apply a step of primitive fitting. They suggest using a trainable classifier to learn a distance function that improves their fitting procedure. However, they do not implement such classifier and relay on a heuristic derived from their experimentation.

In the area of fast object detection Lienhart et al. [5] extended the idea of a boosted cascade of simple features with two extensions. The first is a set of rotated haar-like features that complement the initial set presented in [8]. Their other contribution was demonstrating the performance of Gentle Adaboost with small classification trees that outperformed Discrete Adaboost and stumps. Gentle Adaboost is a robust flavor of adaptive boosting as it does not rely on the half-log ratio [2]. In this report, we will apply this method to learn a window classifier from different photographs.

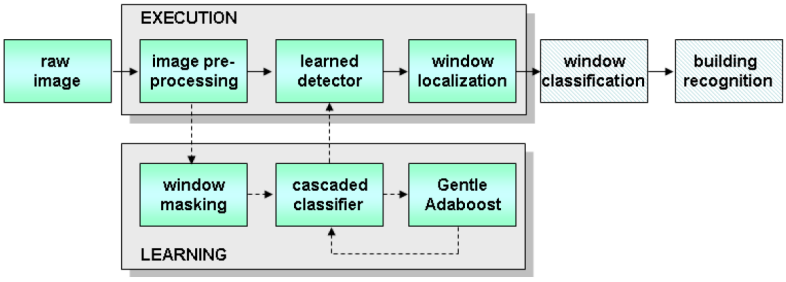

Figure 1: Schematic outline of the window detection system in [1].

Figure 2: The CityFit system requires a segmented, orthorectified photo as input [3]. However, they do not have an automatic window segmentation method. Manual window segmentation is done in their results.

Ali et al. [1] presented a method for window detection in urban environments. They rely on the method presented in [5] using a database of 1506 manually marked windows from 380 photographs of buildings in Zurich, Graz, and Vienna. They achieved detection rates of about 66% with respect to ground truth of 744 windows from 40 test images. We will use a subset of the database they used for our experiments in this report. With regard to practical use of their learned classier, the authors state that for building detection it is sufficient to detect a small number of windows. An outline of their system is shown in Figure 1.

A more recent work by Hohmann et al. [3] is an ambitious attempt at reconstructing many buildings at the same time (Figure 2). The goal of their CityFit project is to automatically reconstruct facades of buildings in the city of Graz. They use both photographs and LIDAR data and go over both to detect facades features for reconstruction. Other recent works try to find tiles on the facades [6], reconstruct its texture and structure [10], or work on window detection from LIDAR data [9]. However, all of these methods require the input to be orthorectified or assume that all windows lie on a plane. This demonstrate the importance of having a preprocessing stage of window identification early on for any building detection or reconstruction system. When windows are identified we are able to apply other steps to discover other building attributes such as height, width, or style.

2 Approach

In this report, we will follow the same algorithm used in [1] which depends on the object detection method presented in [8]. This method was motivated primarily by the problem of face detection, but it can be trained to detect different object classes. It uses Haar-like features similar to Haar wavelets used in signal analysis. Features in this algorithm are computed as the difference between the sum of pixel intensities in the different regions. The total number of features for a small region of 24×24 pixel can have a total of $117,941$ possible features. To be able to learn a classifier with this large number of features, a variant of AdaBoost called Gentle AdaBoost is used to select the best features and train the weak classifiers. The entire algorithm is widely used in object detection and is fully implemented in the OpenCV computer vision library [1].

[1] http://opencv.willowgarage.com/documentation/object_detection.html

(a) (b)

Figure 3: (a) we manually mark windows (blue) in our own test set. The testing set (gathered using Google images) has been taken using different cameras, light conditions, different perspectives, and at different parts of the world. (b) an example of images in the negative set.

2.1 Input Data

We use the TSG-60 database from Joanneum Research [4] which is part of the full set used in [1]. We implemented a simple interface [2] to manually mark 850 windows in 60 photographs (Figure 3a). The set is then converted to a gray-scale version in order to reduce computational time. A set of negative examples are also required for the learning process. We used a subset of a large source of images found on the web [3]. This negative set is then vetted to make sure no windows are present (Figure 3b).

2.2 Learning

The details of the parameters used for learning in [1] are not thoroughly discussed. Therefore, we experimented with different parameters and sets of input to achieve a desirable classifier. These parameters include: number of positive images, number of negatives, size of feature, and number of classifiers to be used. The total number of positive examples of windows is 850, however, the types of windows are limited since there are photographs of the same building but from a different perspective. The total number of negative images used is 2000. The range of window sizes, in pixels, used in the experiments is from 10×10 pixels to 50×50. The cascade classifier used consists of several simpler classifiers that are applied subsequently to regions of interest. The number of classifiers (stages) used in our experiments ranged from 10 to 20. Increasing the number of stages beyond 20 generally needs a computational time on the magnitude of days. We opt-out to use 15 stages in most of our experiments due to time constraints. Once we setup the framework with a small sub-set of our data, we are then able to learn the final classifier with more stages and using larger input.

3 Experiments

In our experiments, we used the Haar feature-based cascade classifier available in the OpenCV library. The code is written in C++ with support for the OpenMP API. We modified the performance tool available in OpenCV to visualize the outputs of the detection algorithm. All experiments were done on a quad core 3.0Ghz AMD Phenom II PC. The training time in our experiments ranged from 15 to 120 minutes, with the best classifier taking around 80 minutes to train.

In order to test our trained classifier, we compiled our own simple set of building photographs from different regions of the world including Europe, the US, and Vancouver. In each photograph

[2] http://code.google.com/p/ml-object-marker/

[3] http://tutorial-haartraining.googlecode.com/svn/trunk/data/negatives/

(a) (b)

(c) (d) (e)

Figure 4: (a) a city hall in Germany (b) the Dakota building in New York (c) the Fisher building in Chicago (d) the Dominion building in Vancouver (e) a number of glass buildings in downtown Vancouver. Marked with blue are the detected windows that correspond to the manually marked ground truth. Red rectangles are the false predictions, however, they tend to include a good number of non-marked windows.

we marked a representative portion of the windows on each main building. Since the task of the classifier is to identify window like objects it does detect windows on surrounding buildings (see bottom left of Figure 4a) and other smaller window like objects in the scene.

The size of our test set is rather small, consisting of only 12 photographs containing 193 marked windows. However, considering the large variation in each photograph the classifier was still able to correctly identify an average of 61% of marked ground truth. Some results of the detection process can be seen in Figure (4). Note that most windows in Figure (4c) were not marked as ground truth but a visual inspection of the results show that the majority of windows have been detected in the building. Our training set did not include any glass dominant buildings and so most windows in Figure (4e) were not detected.

The training relating to the different parameters of our experiments can be seen in Table 1. Since we do not mark the ground truth of the entire test set, the best results are obtained when the hit rate is high and the average classification uncertainty is low. An optimal classifier would have a hit rate of 1.0 and a conservative low number of uncertain detections per photograph. We found that the classifier with the feature scale 12×12 pixel and 20 stages of training gives the best result. It is worth mentioning that this classifier had a hit rate of about 0.92 on training data.

Size (px)	Num. stages	Hit rate	Avg. uncertainty
10×10	15	0.66	83
10×10	20	0.48	42
12×12	15	0.72	110
12×12	20	0.61	58
15×15	20	0.68	78
20×20	15	0.73	110
30×50	5	0.13	284

Table 1: Results of the different classifiers used in our experiments. The highlighted row represents the classifier with the best trade-off between detection rate and average amount of uncertainty per photograph.

3.1 Comparison

Due to time constraints we are only able to experiment with a small set of inputs compared to the set used in [1]. They claim detection rates of about 66% with respect to ground truth. In our simple experiment we were able to reach 73% detection rate with respect to a ground truth, however, it comes at the expense of a large factor of uncertainty or false detections elsewhere in the photograph. It is clear that in both our implementations we can apply further processing to improve detection rates. Perhaps a naive idea is to find pair-wise similarity between detected windows on the same image and then apply a clustering procedure to trim outliers (given some threshold).

4 Conclusion

For this report we have implemented a window detection classifier using a Haar feature-based cascade classifier. Given a larger data set (and more time) the classifier is expected to perform better than 61% of window detections per building. The classifier on its own may not be useful in practical applications. However, combined with further processing or included in a orthorectifying algorithm it would have potential in many applications that depend on building detection. The learning process is generally slow and data marking is a time consuming task. Having a simple algorithm that can assist a user in segmenting windows can be a straightforward application of this simple classifier.

Acknowledgments

Tutorials found online relating to face and object detection by Naotoshi Seo [4] and Aleksey Kodubets [5] were a great help in implementing the classifier. The OpenCV library has an excellent documentation, implementation, and machine learning API.

References

[1] Haider Ali, Christin Seifert, Nitin Jindal, Lucas Paletta, and Gerhard Paar. Window detection in facades. In *Proceedings of the 14th International Conference on Image Analysis and Processing*, ICIAP '07, pages 837–842, Washington, DC, USA, 2007. IEEE Computer Society.

[2] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

[3] Bernhard Hohmann, Ulrich Krispel, Sven Havemann, and Dieter Fellner. Cityfit high-quality urban reconstruction by fitting shape grammars to image and derived textured point clouds. In *In: Proceedings of the International Workshop 3D-ARCH 2009*, 2009.

[4] TSG-60: http://dib.joanneum.at/cape/TSG 60/.

[4] http://note.sonots.com/SciSoftware/haartraining.html

[5] http://www.computer-vision-software.com/blog/2009/06/opencv-haartraining-detect-objects-using-haar-like-features/

[5] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM-Symposium'03*, pages 297–304, 2003.

[6] Przemyslaw Musialski, Meinrad Recheis, Stefan Maierhofer, Peter Wonka, and Werner Purgathofer. Tiling of ortho-rectified facade images. In *Proceedings of the 26th Spring Conference on Computer Graphics*, SCCG '10, pages 117–126, New York, NY, USA, 2010. ACM.

[7] Konrad Schindler and Joachim Bauer. A model-based method for building reconstruction. In *Proceedings of the First IEEE International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*, pages 74–, Washington, DC, USA, 2003. IEEE Computer Society.

[8] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511 – I–518 vol.1, 2001.

[9] Ruisheng Wang, Jeff Bach, and Frank P. Ferrie. Window detection from mobile lidar data. In *Proceedings of the 2011 IEEE Workshop on Applications of Computer Vision (WACV)*, WACV '11, pages 58–65, Washington, DC, USA, 2011. IEEE Computer Society.

[10] A. A. Yakubenko, V. A. Kononov, I. S. Mizin, V. S. Konushin, and A. S. Konushin. Reconstruction of structure and texture of city building facades. *Program. Comput. Softw.*, 37:260–269, September 2011.